1. Simulation

1.1 Simulation Datasets

We generated twenty simulation datasets with a thousand observations in the two-dimensional space. The majority class instances were generated from the bivariate normal distribution, where the mean vector was set to $(0, 0)$ and the covariance matrix was the identity matrix. The minority class instances were generated using GMM with two or three components while changing the imbalance ratio (IR), and adjusting the mean vector for each component to control the degree of class overlap. IR is the ratio between the number of majority and minority class instances. Table 1 summarizes the IR, which denotes the ratio of the size of the majority class to that of the minority class, along with the sample size of each class. The datasets have IRs of 2, 5, 10, and 20. The scatter plots of these datasets are represented in Figure 1.

1.2 Simulation set-up

In these experiments, we evaluated the performance of the oversampling methods on simulation datasets. The data sets were randomly split into training and test datasets using ten-fold cross validation. And in order to identify the effect of applying the β of GOMCD, the experiment was conducted by setting the β in 0.25 units from 0 to 1. The cutoff value α was set to 0.05.

Table 1 Different mean vector and variance settings for generating simulation datasets

[Mean vector, Variance]	Two components			
	IR and (Sample size of the Majority, Minority class)			
	2, (667, 333)	5, (833, 167)	10, (909, 91)	20, (952, 48)
[(−1, −1), 0.4], [(2, 2), 0.4]	A1	A2	A3	A4
[(−1.5, −1.5), 0.4], [(2, 2), 0.4]	B1	B2	B3	B4
[(−1.5, −1.5), 0.4], [(2.5, 2.5), 0.4]	C1	C2	C3	C4
[Mean vector, Variance]	Three components			
	IR and (Sample size of the Majority, Minority class)			
	2, (1334, 666)	5, (1667, 333)	10, (1818, 182)	20, (1904, 96)
[(−1, −1), 0.4], [(1, 1), 0.4], [(−2.7, −2.7), 0.2]	D1	D2	D3	D4
[(−1.5, −1.5), 0.4], [(1.5, 1.5), 0.4], [(−2.7, −2.7), 0.2]	E1	E2	E3	E4

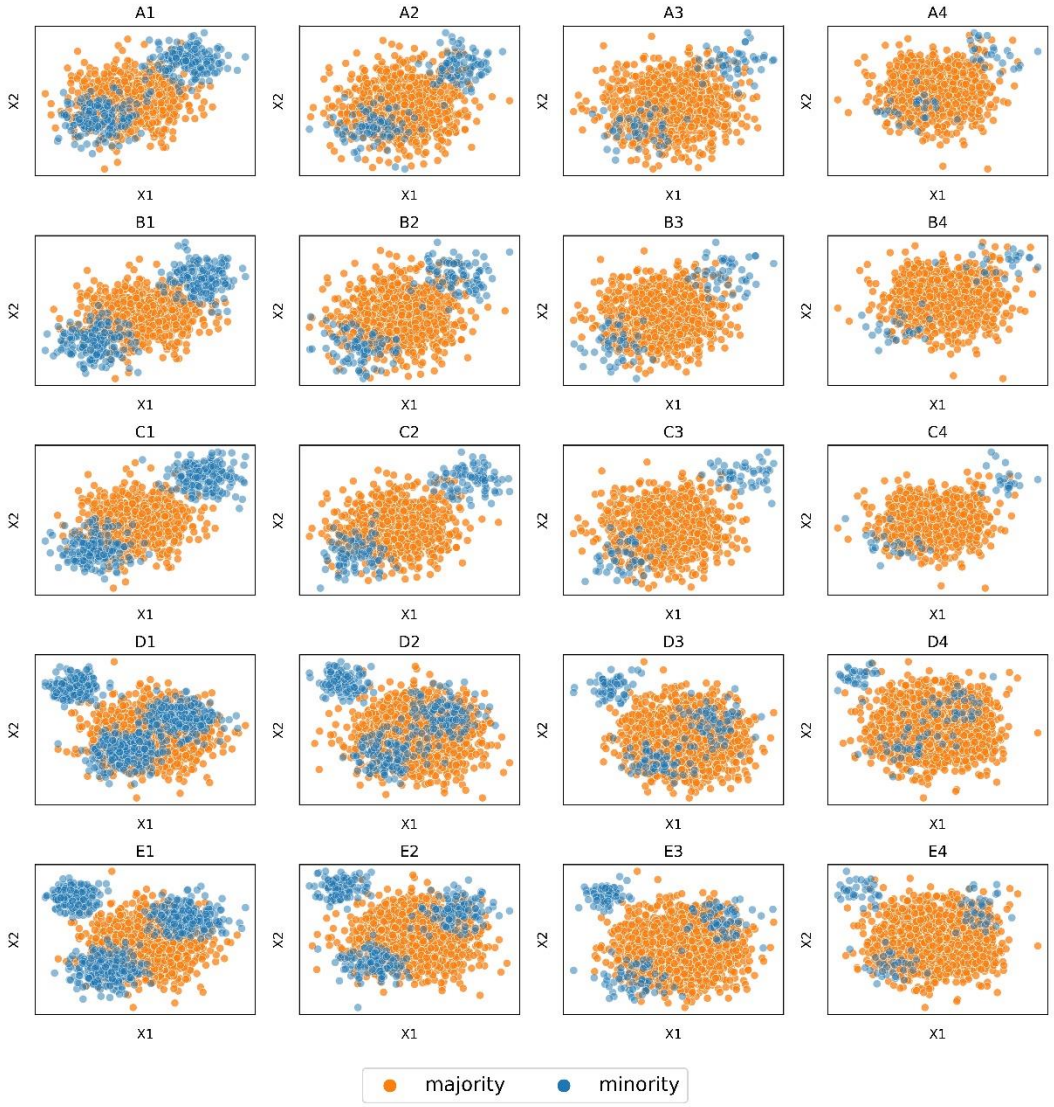

Figure 1 Scatter plots of simulation datasets

The effective of the GOMCD algorithm was verified through comparison with other methods, including the raw data (Default), SMOTE, BLSMOTE, SLSMOTE, Cluster-SMOTE, Gaussian-SMOTE, PDFOS, and MDO. The experiments were implemented using the *Python* package *smote-variants* version 0.7.1 (Kovács, 2019). The parameters of oversampling methods were set to default values in *smote-variants*. We used the *R* package *mclust* version 6.0.0 (Scrucca et al., 2016) for GMM. The MCD estimation of GOMCD is implemented using the *R* package *robustbase* version 0.95 (Maechler et al., 2021).

1.3 Simulation set-up

In these experiments, we evaluated the performance of the oversampling methods on simulation datasets. The data sets were randomly split into training and test datasets using ten-fold cross validation. And in order to identify the effect of applying the β of GOMCD, the experiment was conducted by setting the β in 0.25 units from 0 to 1. The cutoff value α was set to 0.05.

The effective of the GOMCD algorithm was verified through comparison with other methods, including the raw data (Default), SMOTE, BLSMOTE, SLSMOTE, Cluster-SMOTE, Gaussian-SMOTE, PDFOS, and MDO. The experiments were implemented using the *Python* package *smote-variants* version 0.7.1 (Kovács, 2019). The parameters of oversampling methods were set to default values in *smote-variants*. We used the *R* package *mclust* version 6.0.0 (Scrucca et al., 2016) for GMM. The MCD estimation of GOMCD is implemented using the *R* package *robustbase* version 0.95 (Maechler et al., 2021).

As classifiers, we chose k-nearest neighbor (kNN) (Cover and Hart, 1967), random forest (RF) (Breiman, 2001) and support vector machine (SVM) (Cortes and Vapnik, 1995), as base classifiers to evaluate the effectiveness of the oversampling methods, for the following reason. They are chosen to ensure that the classification results can be generalized and are not constrained to the usage of a specific classifier. The classification algorithms provided in the *Python* packages *scikit-learn* version 1.0 (Pedregosa et al., 2011) was implemented together with their default parameters except SVM. SVM needs the kernel, gamma, and cost. We used the radial basis function kernel, which is the default kernel in *scikit-learn*. The appropriate value of gamma and cost were determined through grid search. The ranges of gamma and cost were

{0.01, 0.1, 1, 10, 100} and {0.001, 0.01, 0.1, 1}, respectively.

1.4 Simulation results

We generated the minority class instances using all oversampling methods to equalize the minority and majority class sizes. The classification performances of 10 test datasets were averaged. The results for the effect of applying the correction parameter β are represented in Table 2. Table 2 lists the average classification rankings for all simulation datasets. The boldface and underlined values represent the highest and second-highest performance metrics, respectively.

Table 2 Average rankings of performance metric with varying β of GOMCD

Classifier	Metrics	$\beta = 0.00$	$\beta = 0.25$	$\beta = 0.50$	$\beta = 0.75$	$\beta = 1.00$
kNN	Recall	**2.210**	<u>2.558</u>	2.890	3.295	4.047
	Precision	3.730	3.455	3.315	<u>2.590</u>	**1.910**
	F1-score	3.240	3.112	3.095	<u>2.805</u>	**2.747**
	G-mean	**2.590**	<u>2.760</u>	2.918	3.088	3.645
RF	Recall	**2.260**	<u>2.495</u>	2.905	3.407	3.932
	Precision	3.805	3.415	3.215	<u>2.642</u>	**1.922**
	F1-score	3.325	3.105	3.112	<u>2.812</u>	**2.645**
	G-mean	**2.640**	<u>2.692</u>	2.920	3.180	3.568
SVM	Recall	**2.060**	<u>2.358</u>	2.925	3.485	4.172
	Precision	3.765	3.405	3.208	<u>2.612</u>	**2.010**
	F1-score	3.132	<u>2.900</u>	2.990	**2.898**	3.080
	G-mean	**2.412**	<u>2.450</u>	2.927	3.252	3.957

The boldface and underlined values represent the highest and second-highest performance metrics, respectively.

Figures 2-4 illustrate recall, precision, F1-score and G-mean, when combined with

kNN, RF and SVM, respectively. In Figures 2-4, the color of line changes from blue to red as the beta increases from 0 to 1. We obtained similar results from all classifiers for each classification performance metric. Regardless of the class overlap and classifier, it can be seen that both precision and F1-score decrease as IR increases. Increasing β leads to a tradeoff with lower recall and higher precision, regardless of the IR. Thus, we should determine β based on whether the classification objective will maximize recall or precision. For the comparison, GOMCD with $\beta = 0.25$ was chosen as it demonstrated high recall and F1-score, while considering the class overlap.

Table 3 lists the average classification rankings. In Table 3, the boldface and underlined values represent the highest and second-highest performance metrics, respectively. Table 3 indicates that GOMCD achieves high recall and G-mean across all classifiers. The average performance metric of each oversampling method for each simulation dataset is presented in Figures 5-7. Unlike most other oversampling methods, GOMCD achieves stable recall and G-mean despite increased IR, regardless of classifiers. In the case of datasets A and D, which have mean vectors close to the majority class, results in lower classification performance compared to other datasets. However, as can be seen from Figures 5-7, GOMCD shows good performance in terms with recall and G-mean on datasets A and D. The results of GOMCD with higher recall are similar to those of Gaussian-SMOTE and PDFOS, while Gaussian-SMOTE and PDFOS show lower precision and F1-score when combined with kNN and RF. Table 3 indicates that GOMCD with SVM performs well on all four performance metrics.

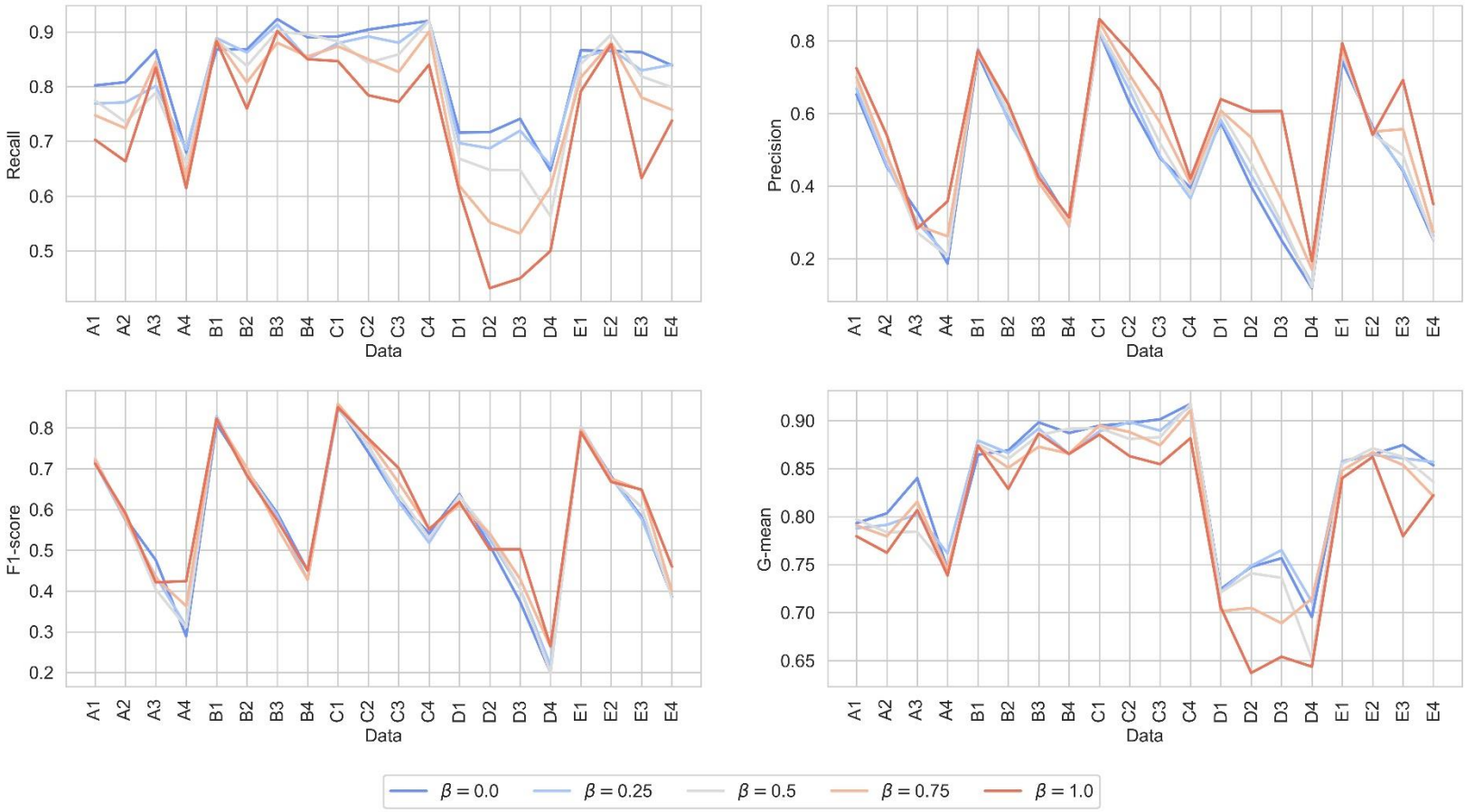

Figure 2 Influence of varying β on classification performance using kNN

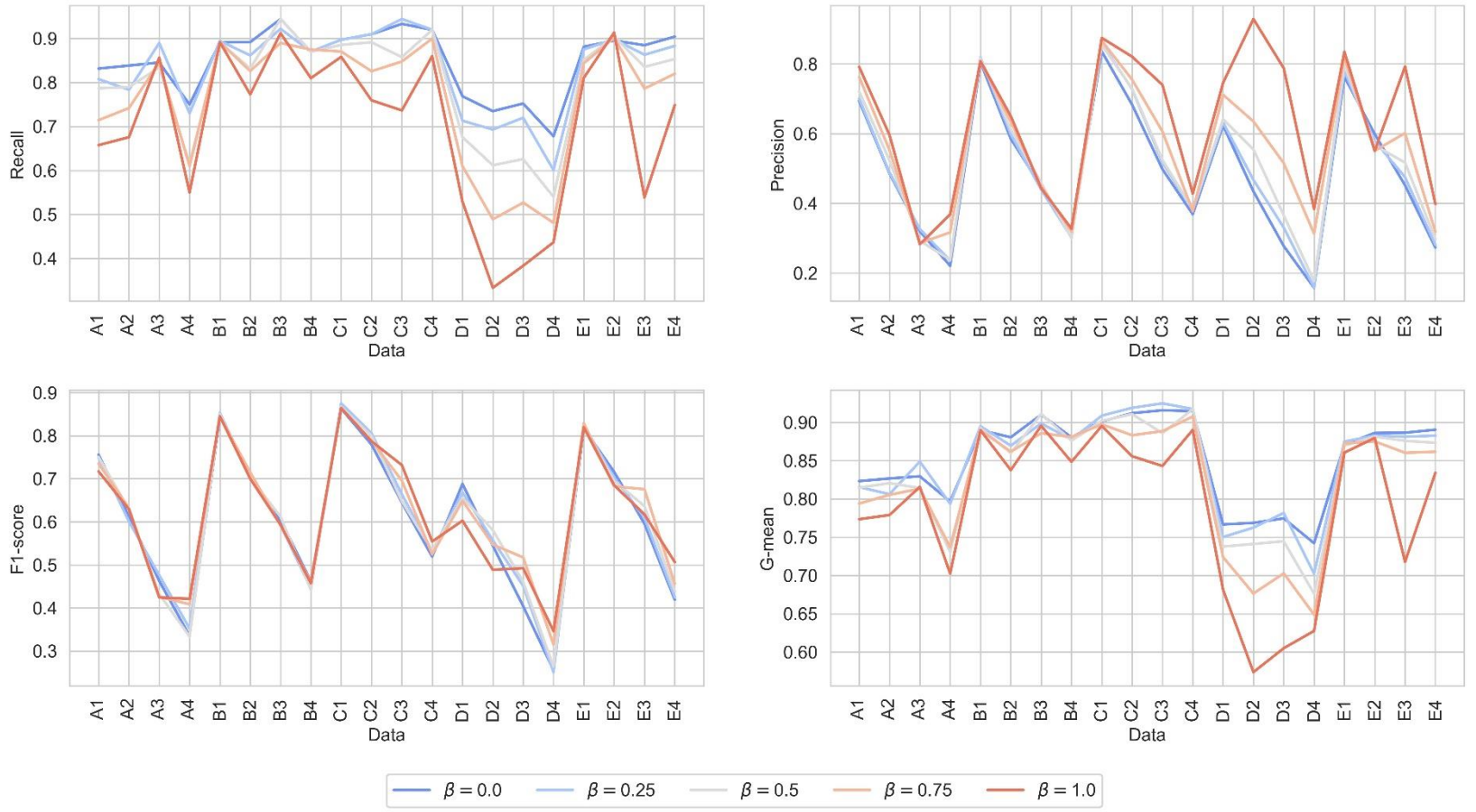

Figure 3 Influence of varying β on classification performance using RF

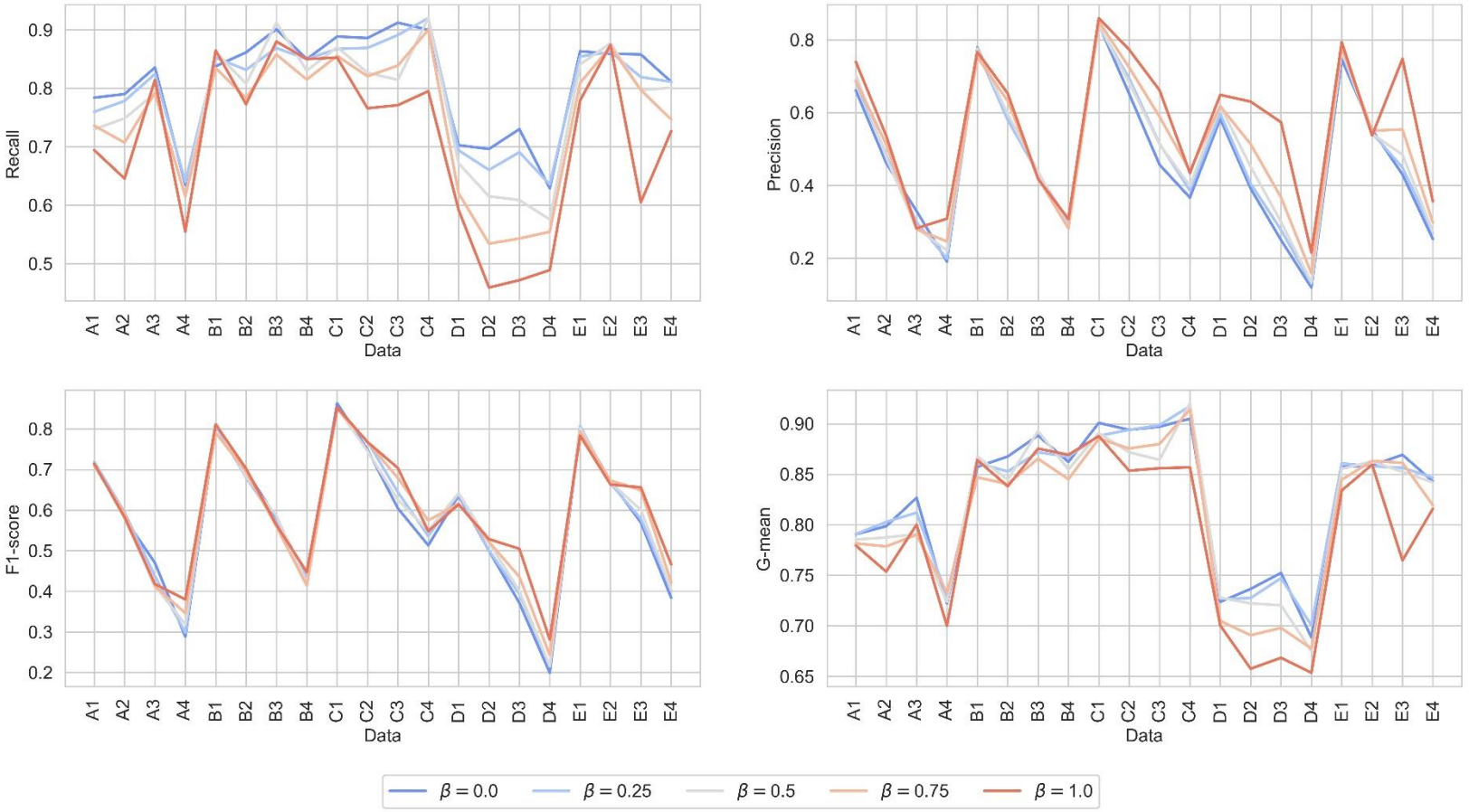

Figure 4 Influence of varying β on classification performance using SVM

Table 3 Average rankings of performance metric on simulation datasets with comparison methods

Classifier	Metrics	GOMCD	Default	SMOTE	BLSMOTE	SLSMOTE	Cluster-SMOTE	Gaussian-SMOTE	PDFOS	MDO
kNN	Recall	**3.357**	8.657	4.677	5.128	4.023	5.010	4.400	<u>3.792</u>	5.955
	Precision	4.310	**1.182**	4.825	<u>3.698</u>	5.025	4.748	6.618	6.967	7.627
	F1-score	3.762	**2.935**	4.682	<u>3.402</u>	4.625	4.710	6.560	6.575	7.748
	G-mean	**2.860**	6.972	4.340	4.778	<u>3.895</u>	4.638	4.967	4.873	7.678
RF	Recall	**2.960**	8.552	4.602	5.915	4.710	5.865	3.728	<u>3.405</u>	5.262
	Precision	5.150	**1.313**	4.528	<u>3.312</u>	4.035	4.045	7.200	7.415	8.003
	F1-score	4.375	<u>3.573</u>	4.152	**3.527**	3.788	4.435	6.660	6.705	7.785
	G-mean	**2.848**	7.088	4.068	5.103	<u>3.910</u>	5.055	4.820	4.860	7.250
SVM	Recall	**3.700**	8.632	4.052	4.020	<u>3.915</u>	4.315	5.245	4.785	6.335
	Precision	<u>4.380</u>	**1.260**	5.515	6.152	5.555	5.605	5.065	4.585	6.882
	F1-score	**3.770**	<u>4.093</u>	4.765	5.530	4.762	4.978	5.300	4.672	7.130
	G-mean	**3.170**	7.795	4.028	5.313	<u>3.983</u>	4.310	5.097	4.300	7.005

The boldface and underlined values represent the highest and second-highest performance metrics, respectively.

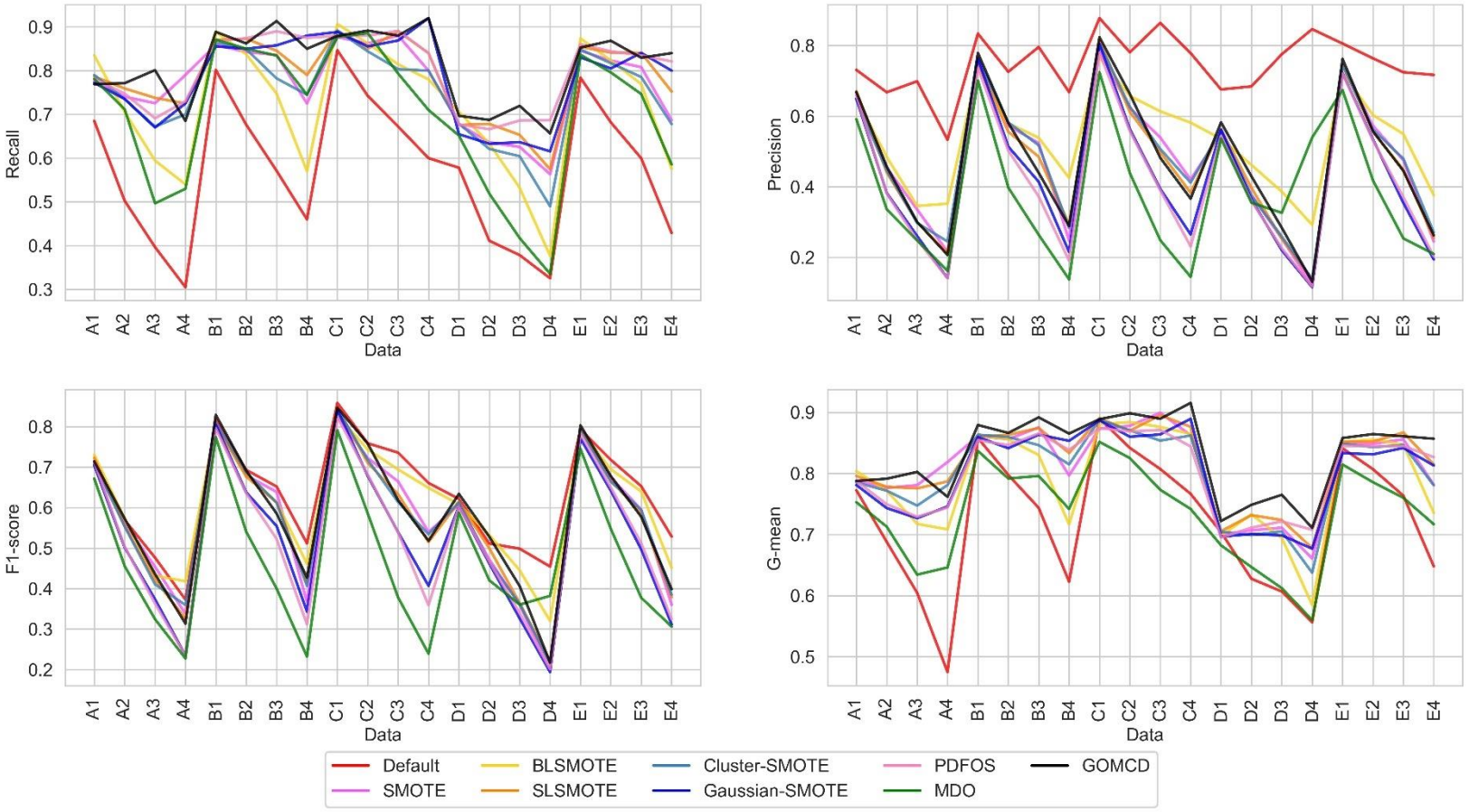

Figure 5 Experimental comparison results for the simulation datasets with kNN

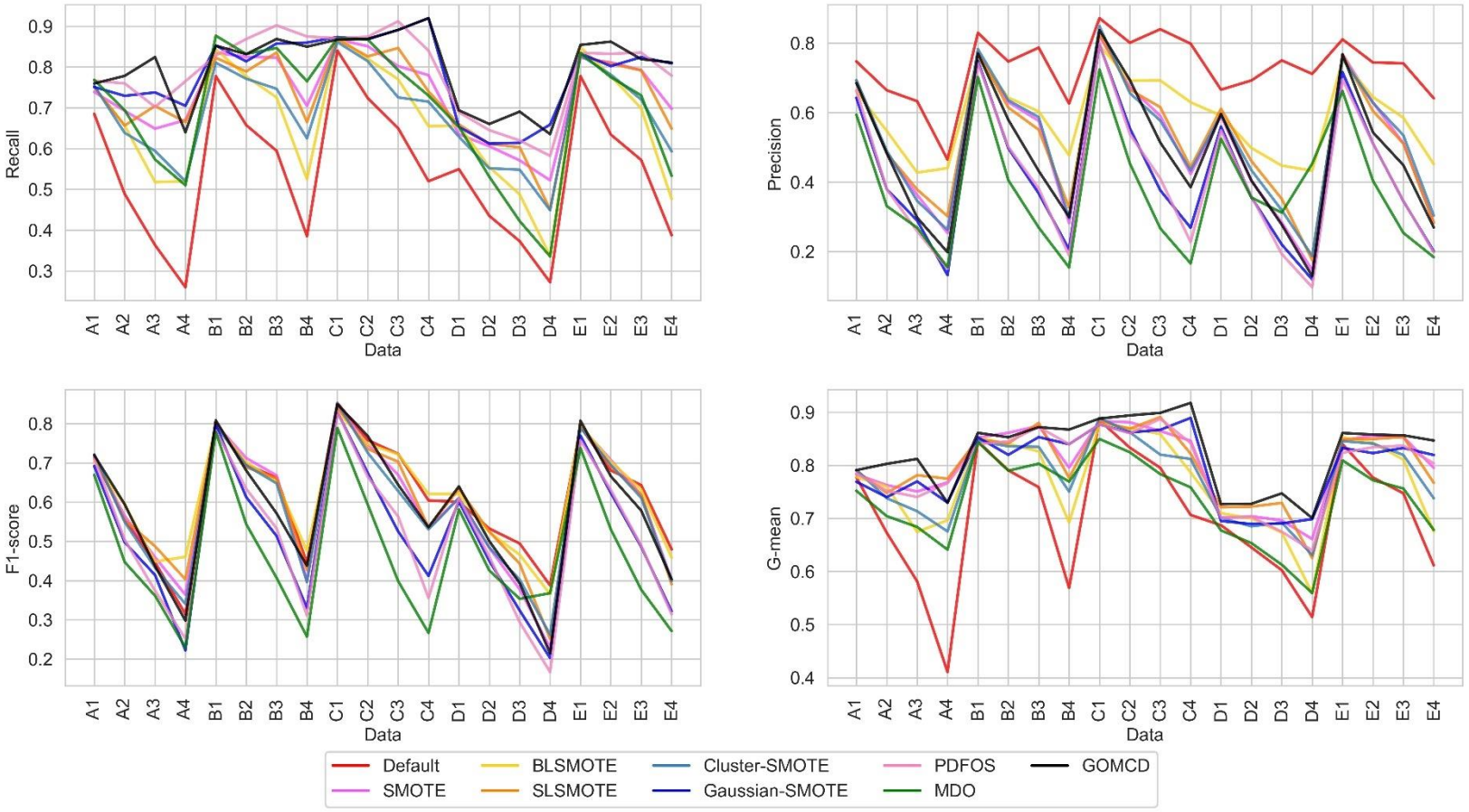

Figure 6 Experimental comparison results for the simulation datasets with RF

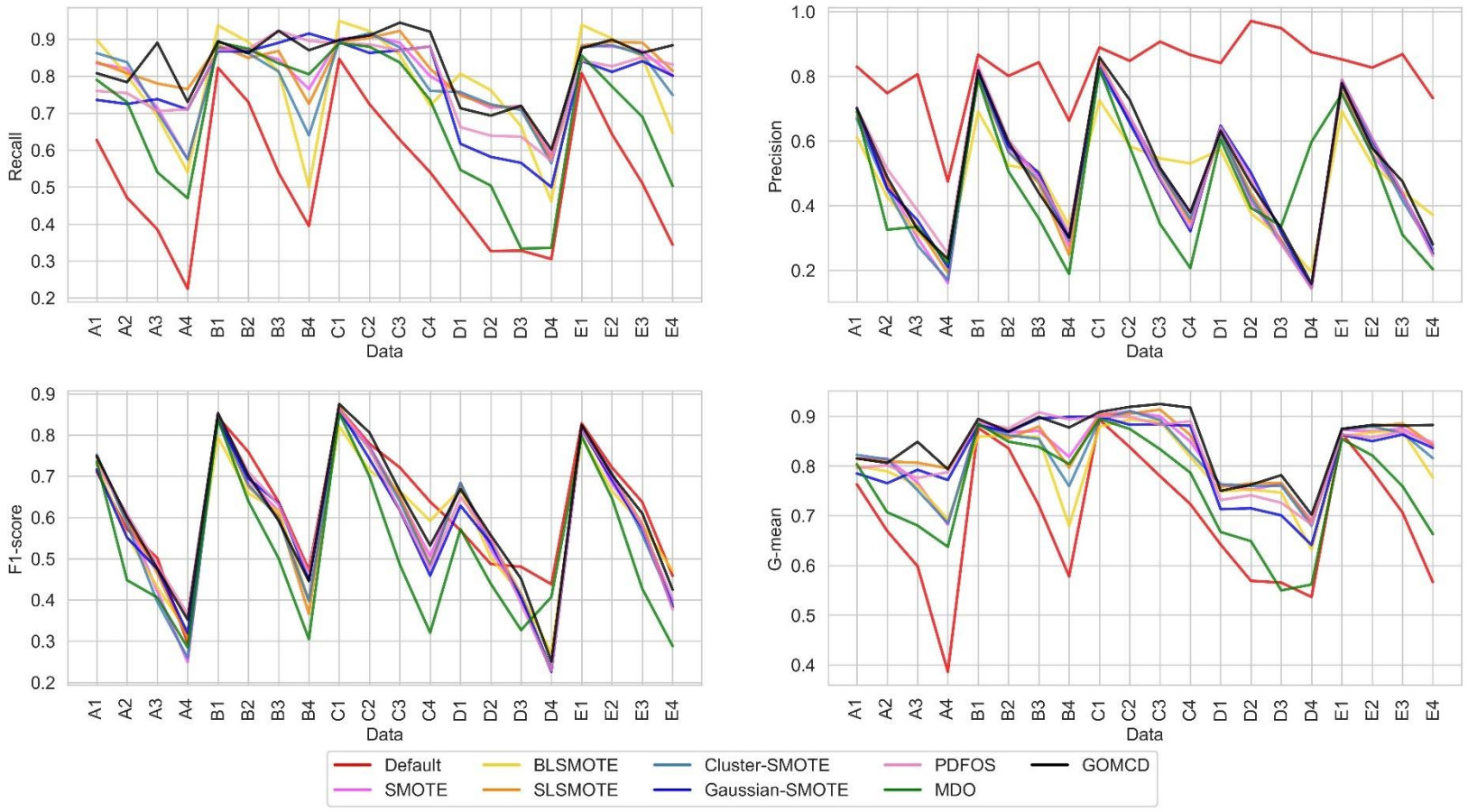

Figure 7 Experimental comparison results for the simulation datasets with SVM

2. Data Analysis

2.1 Data description

The experimental binary class datasets, including 24 sets, are from KEEL repository. Table 4 gives the detailed description of the 24 selected KEEL datasets. It shows the number of features, the sample size of each class, total sample size, and IR for each dataset. These datasets exhibit a considerable range of IRs, ranging from 2.9 to 72.69.

Table 4 Description of benchmark datasets from KEEL repository

ID	Dataset	Features	Sample size		IR
			Minority class	Majority class	
KEEL1	vehicle1	18	217	629	2.90
KEEL2	vehicle0	18	199	647	3.25
KEEL3	ecoli1	7	77	259	3.36
KEEL4	new-thyroid1	5	35	180	5.14
KEEL5	new-thyroid2	5	35	180	5.14
KEEL6	ecoli-0234vs5	7	20	182	9.10
KEEL7	yeast-0256vs3789	8	99	905	9.14
KEEL8	yeast-02579vs368	8	99	905	9.14
KEEL9	ecoli-046vs5	6	20	183	9.15
KEEL10	ecoli-01vs235	7	24	220	9.17
KEEL11	ecoli-0346vs5	7	20	185	9.25
KEEL12	ecoli-0347vs56	7	25	232	9.28
KEEL13	ecoli-067vs5	6	20	200	10.00
KEEL14	ecoli-0147vs56	7	29	307	10.59
KEEL15	led7digit-02456789vs1	7	37	406	10.97
KEEL16	ecoli-01vs5	6	20	220	11.00
KEEL17	ecoli-0146vs5	6	20	260	13.00
KEEL18	abalone-9vs18	8	42	689	16.40
KEEL19	winequality-red-4	11	53	1546	29.17
KEEL20	abalone-17vs7-8-9-10	8	58	2280	39.31
KEEL21	yeast6	8	35	1449	41.40
KEEL22	abalone-19vs10-11-12-13	8	32	1590	49.69
KEEL23	winequality-white-3-9vs5	11	25	1457	58.28
KEEL24	abalone-20vs8910	8	26	1890	72.69

2.2 Results

The comparison methods, classifiers, and validation process are the same as in Chapter 1. The cutoff value α and correction parameter β in GOMCD were set to 0.05 and 0.25, respectively. Table 5, which lists the average classification rankings for all datasets, indicates that GOMCD combined with random forest achieves the top rankings. When using kNN and SVM, GOMCD obtains high recall and G-mean.

In Figures 8-10, the average performance metric of each oversampling method for each KEEL dataset is presented. As increased IR, the classification performance metrics show a decrease in Figures 8-10. For KEEL 1, 7, 15, 19, and 22, all classifiers obtain lower classification performance metrics compared to datasets with similar IRs. Recall and precision exhibit a similar pattern to G-mean and F1-score, respectively, which aligns with the results discussed in Section 1.4.

In this analysis, GOMCD combined with random forest achieved good performance. GOMCD, Gaussian-SMOTE, and PDFOS, which expand the training sample area of the minority class (Gao et al., 2014; Lee et al., 2017), have higher average recall than precision. However, the ability of GOMCD to generate dense instances inside minority class areas might explain the better precision performance compared to Gaussian-SMOTE and PDFOS. In addition, by expanding the area while increasing the density at the center of the minority group, the potential for misclassification of the majority class was restricted, leading to a good performance in G-mean.

Table 5 Average rankings of performance metric on KEEL datasets with comparison methods

Classifier	Metrics	GOMCD	Default	SMOTE	BLSMOTE	SLSMOTE	Cluster-SMOTE	Gaussian-SMOTE	PDFOS	MDO
kNN	Recall	**4.075**	6.675	4.083	5.260	4.348	4.396	5.775	4.906	5.481
	Precision	5.115	4.406	5.637	5.283	5.994	5.577	4.275	4.521	**4.192**
	F1-score	4.604	5.298	5.202	5.256	5.583	5.196	4.721	4.646	**4.494**
	G-mean	**4.238**	5.896	4.673	5.379	5.167	4.821	5.208	4.773	4.846
RF	Recall	4.256	6.090	4.852	5.687	4.715	5.217	4.762	**4.138**	5.283
	Precision	**4.613**	4.719	4.956	4.958	5.129	4.844	5.179	5.340	5.262
	F1-score	**4.340**	5.310	4.808	5.360	4.833	4.925	5.077	5.096	5.250
	G-mean	**4.192**	5.529	4.892	5.477	4.850	5.069	5.062	4.581	5.348
SVM	Recall	**3.775**	5.850	5.308	5.935	4.952	5.804	4.594	3.865	4.917
	Precision	5.252	**3.990**	4.971	4.896	5.623	4.771	5.512	5.390	4.596
	F1-score	4.583	4.619	5.008	5.390	5.448	5.206	5.192	4.988	**4.567**
	G-mean	**4.133**	5.004	5.087	5.681	5.392	5.431	5.123	4.473	4.675
XGBoost	Recall	4.748	6.188	4.590	5.094	**4.269**	5.075	5.065	4.721	5.252
	Precision	4.998	**4.483**	5.106	4.956	5.275	5.035	5.052	5.131	4.962
	F1-score	4.804	5.246	4.833	4.938	**4.794**	5.129	5.125	5.017	5.115
	G-mean	4.810	5.492	4.677	5.025	**4.562**	5.171	5.215	4.894	5.154

The boldface and underlined values represent the highest and second-highest performance metrics, respectively.

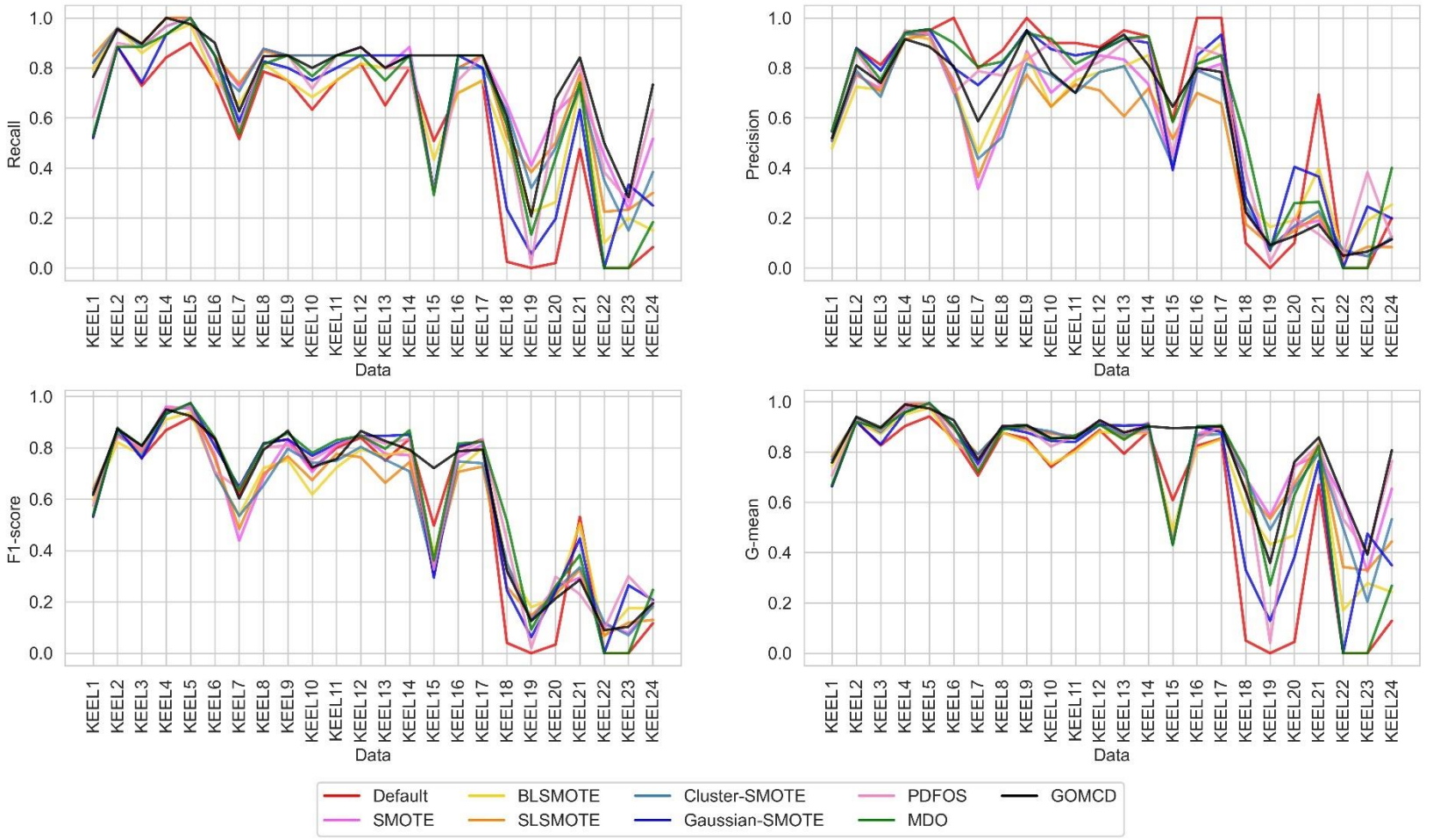

Figure 8 Experimental comparison results for the KEEL datasets with kNN

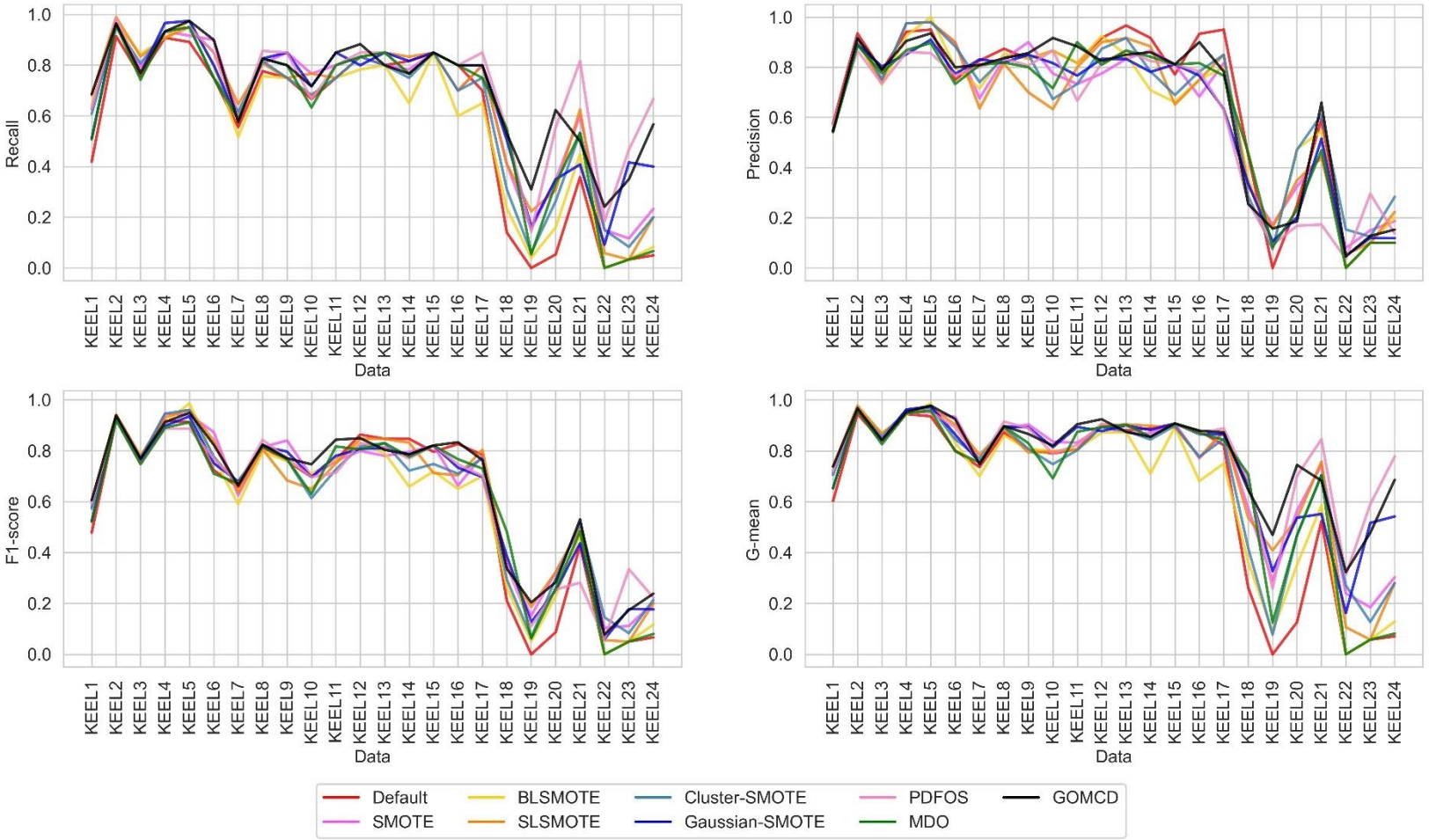

Figure 9 Experimental comparison results for the KEEL datasets with RF

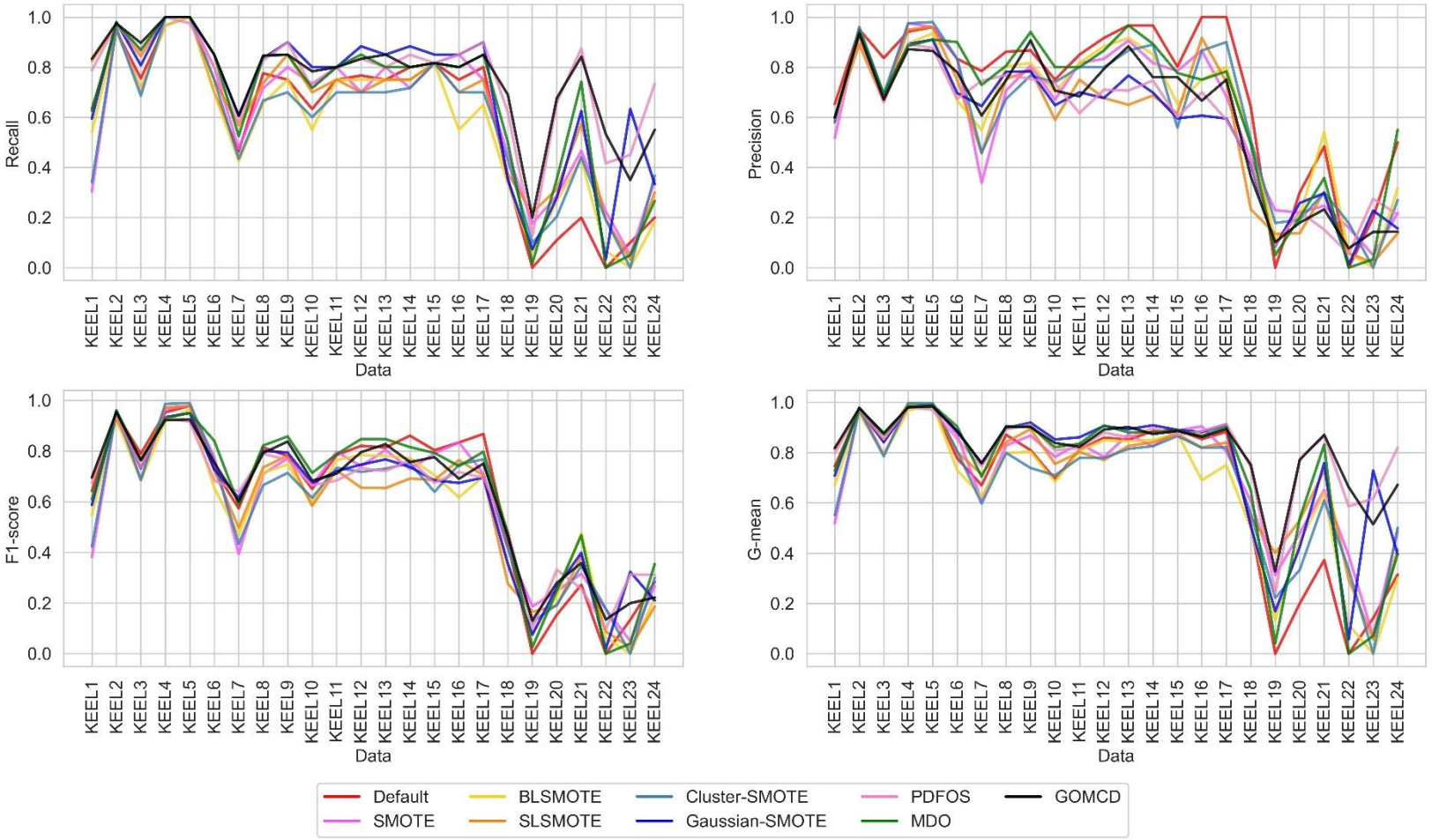

Figure 10 Experimental comparison results for the KEEL datasets with SVM