

Article Review

Improving Imbalanced Classification by Anomaly Detection

증강지능 연구실 황승현

2023-09-13

논문 소개

PPSN 2020: Parallel Problem Solving
from Nature

Jiawen Kong 1

Wojtek Kowalczyk 1

Stefan Menzel 2

Thomas Bäck 1

[1] Leiden University, Leiden, The
Netherlands

[2] Honda Research Institute Europe
GmbH, Offenbach, Germany

Improving Imbalanced Classification by Anomaly Detection

Jiawen Kong^{1(✉)}, Wojtek Kowalczyk¹, Stefan Menzel², and Thomas Bäck¹

¹ Leiden University, Leiden, The Netherlands

{j.kong,w.j.kowalczyk,t.h.w.baeck}@liacs.leidenuniv.nl

² Honda Research Institute Europe GmbH, Offenbach, Germany
stefan.menzel@honda-ri.de

Abstract. Although the anomaly detection problem can be considered as an extreme case of class imbalance problem, very few studies consider improving class imbalance classification with anomaly detection ideas. Most data-level approaches in the imbalanced learning domain aim to introduce more information to the original dataset by generating synthetic samples. However, in this paper, we gain additional information in another way, by introducing additional attributes. We propose to introduce the outlier score and four types of samples (safe, borderline, rare, outlier) as additional attributes in order to gain more information on the data characteristics and improve the classification performance. According to our experimental results, introducing additional attributes can improve the imbalanced classification performance in most cases (6 out of 7 datasets). Further study shows that this performance improvement is mainly contributed by a more accurate classification in the overlapping region of the two classes (majority and minority classes). The proposed idea of introducing additional attributes is simple to implement and can be combined with resampling techniques and other algorithmic-level approaches in the imbalanced learning domain.

Keywords: Class imbalance · Anomaly detection · Borderline samples

목차

- Imbalanced data란?
- 새로운 접근법 – anomaly score, 4type
- 실험 분석
- 마무리

Imbalanced data란?

Imbalanced data의 정의와 기존 접근법 소개

Imbalanced classification

불균형 분류

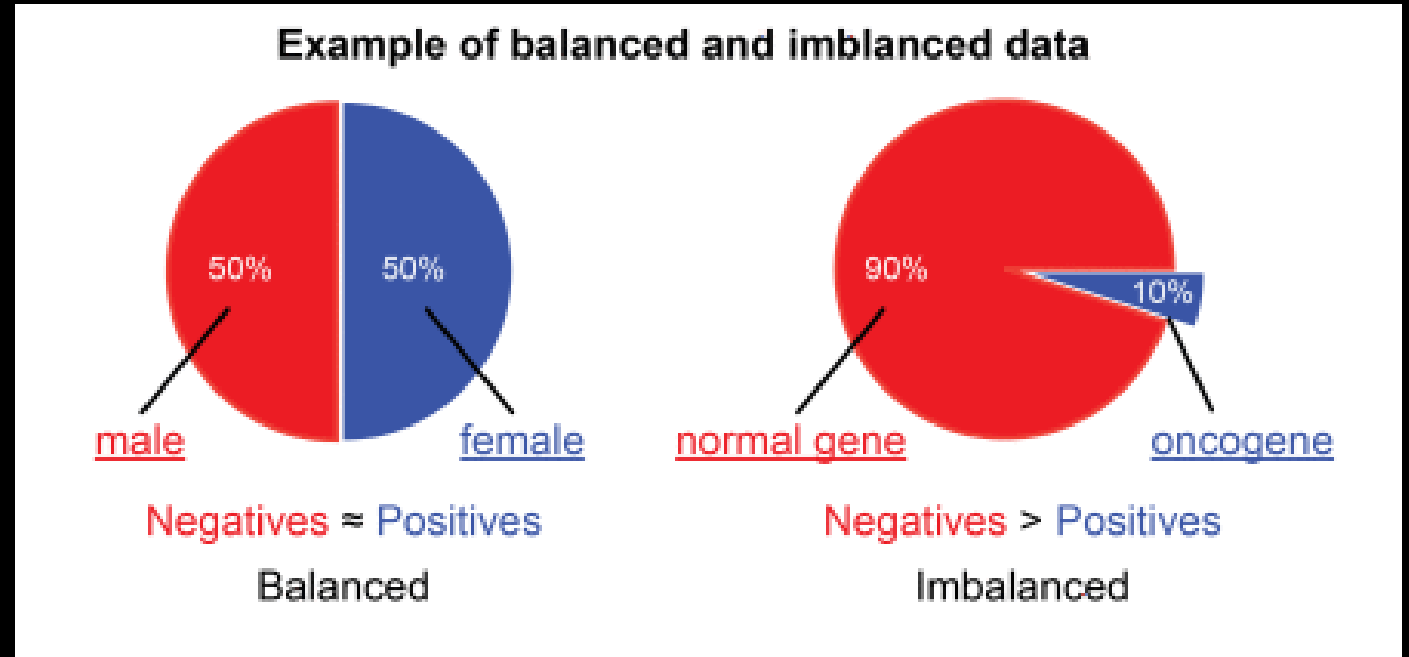
각 클래스 비율 크게 차이

고혈압 분류: 정상 95명 비정상 5명

골다공증 분류: 정상 80명 비정상 20명

Rare class : 소수 (환자)

Abundant class : 다수 (정상인)



Imbalanced classification 문제점

- 대충 만들어도 Accuracy 가 높게 나온다
 - 모델이 모두 정상이라고 판단하면 된다.
 - 정상인 90명 환자 10명을 분류할 때, 모두 정상인이라고 분류하면?
 - 정확도 80%

$$(Accuracy) = \frac{TP + TN}{TP + FN + FP + TN}$$

접근법 : Resampling

Oversampling

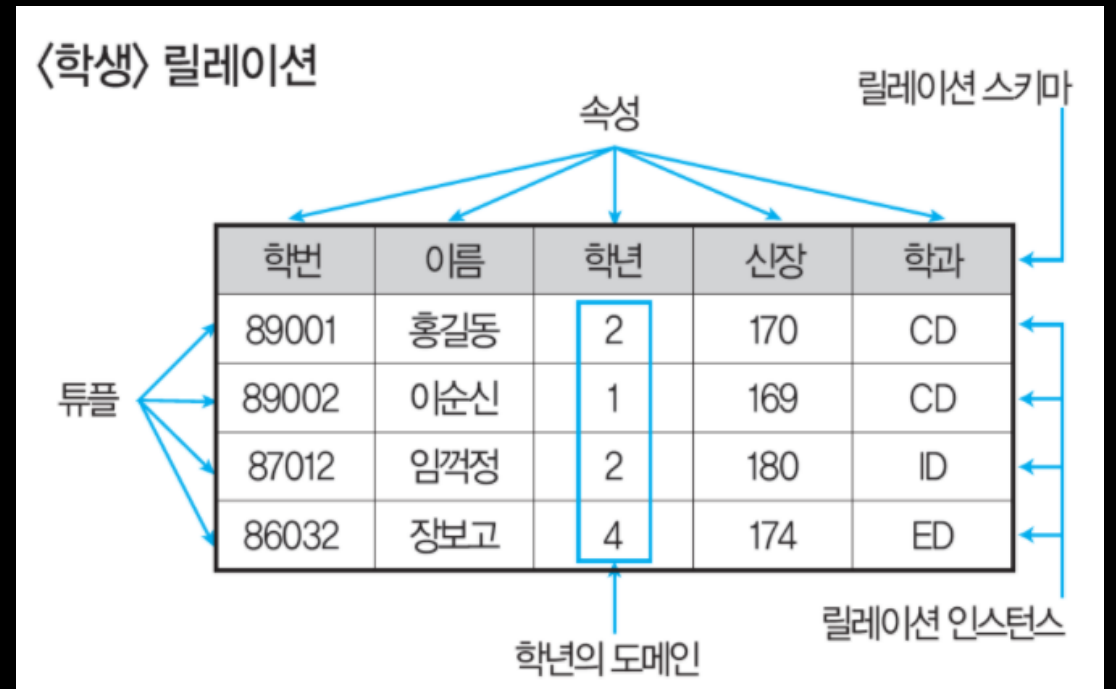
- Rare class 늘리기
- 소수를 복제하여 다수에 맞춤
- SMOTE, ADASYN 등

Undersampling

- Abundant class 줄이기
- 다수를 소수에 맞게 자름
- NCL, OSS 등

기존 접근법의 문제

- Oversampling : Overfitting
- Undersampling : 데이터 수 ↓
 - 학습 효율 떨어짐
- "합성"된 데이터
 - 원본 데이터 보장 x



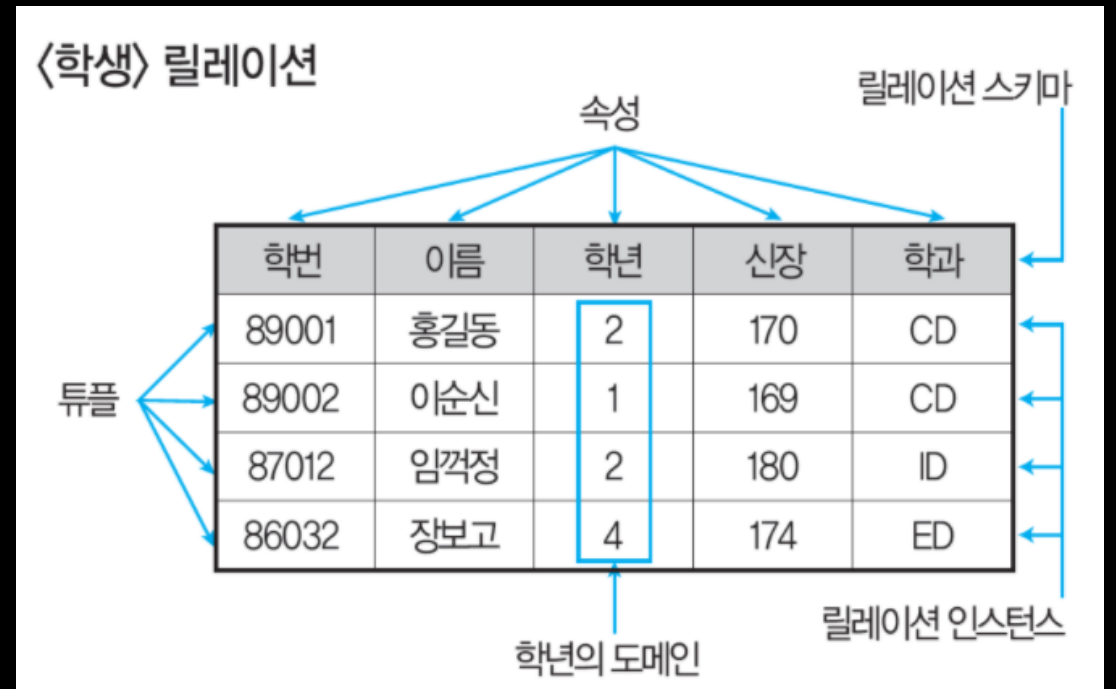
새로운 접근법

논문에서 소개하는 접근법

Anomaly Detection을 이용한 Imbalanced Classification

새로운 접근법

- 데이터셋의 튜플을 늘리지 말고
 - Rows
- 데이터셋의 속성을 늘리자
 - Columns
- 추가 속성
 1. outlier score
 2. safe, borderline, rare, outlier



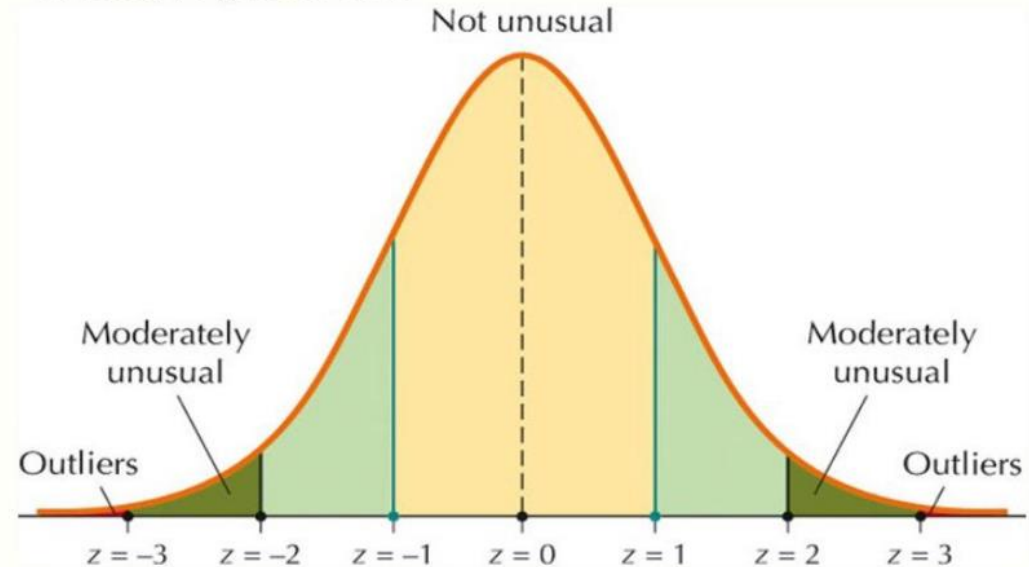
outlier score

- 이상치 점수
- 샘플이 이상치인지 아닌지 판단하는 지표
- Local Outlier Factor(LOF)

Detecting Outliers with z-Scores

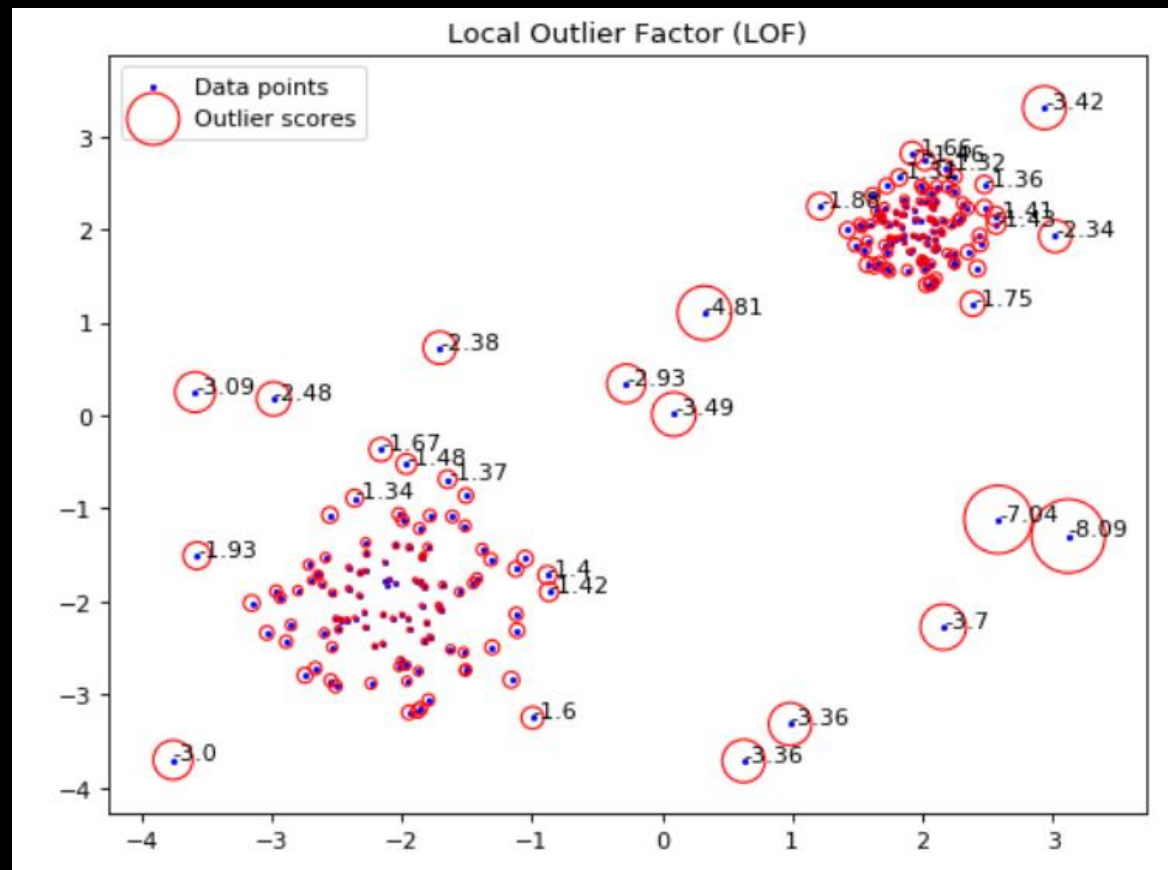
28

An **outlier** is an extremely large or extremely small data value relative to the rest of the data set. It may represent a data entry error, or it may be genuine data.



LOF

주변 데이터에서 크게 벗어난 데이터 식별
이웃에서 크게 떨어져 있다면?
이상치



LOF 구하는 과정

Local Density Estimation

각 데이터 포인트 주변의 로컬 밀도 추정

데이터 포인트와 K-Nearest Neighbors 사이의
거리 계산

Reachability Distance

도달 가능 거리

한 포인트가 이웃 포인트와 얼마나 멀리 떨어져 있는
지

가장 가까운 k번째 이웃까지의 거리

LOF 계산

가장 가까운 k개 이웃의 평균 도달 가능 거리

자신의 도달 가능 거리의 비율

Algorithm 1: Local Outlier Factor (LOF) algorithm [2]

Input : \mathbf{X} - input data $\mathbf{X} = (X_1, \dots, X_n)$
 n - the number of input examples
 k - the number of neighbours

Output: LOF score of every X_i

```
1 initialization;
2 calculate the distance  $d(\cdot)$  between every two data points;
3 for  $i = 1$  to  $n$  do
4   calculate  $k\text{-distance}(X_i)$ : the distance between  $X_i$  and its  $k$ th neighbour;
5   find out  $k$ -distance neighbourhood  $N_k(X_i)$ : the set of data points whose
   distance from  $X_i$  is not greater than  $k\text{-distance}(X_i)$ ;
6   for  $j = 1$  to  $n$  do
7     calculate reachability distance:

$$\text{reach-dist}_k(X_i, X_j) = \max\{k\text{-distance}(X_j), d(X_i, X_j)\};$$

8     calculate local reachability density:

$$\text{lr}_d(X_i) = 1/\text{avg-reach-dist}_k(X_i)$$


$$= 1/\left(\frac{\sum_{o \in N_k(X_i)} \text{reach-dist}_k(X_i, X_j)}{|N_k(X_i)|}\right);$$

   intuitively, the local reachability density of  $X_i$  is the inverse of the
   average reachability distance based on the  $k$ -nearest neighbours of  $X_i$ ;
9   calculate LOF:

$$\text{LOF}_k(X_i) = \frac{\sum_{o \in N_k(X_i)} \text{lr}_d(X_j)}{|N_k(X_i)| \cdot \text{lr}_d(X_i)}$$


$$= \frac{\sum_{o \in N_k(X_i)} \frac{\text{lr}_d(X_j)}{\text{lr}_d(X_i)}}{|N_k(X_i)|}$$

   the LOF of  $X_i$  is the average local reachability density of  $X_i$ 's  $k$ -nearest
   neighbours divided by the local reachability density of  $X_i$ .
10  end
11 end
```

safe, borderline, rare, outlier

Napierala and Stefanowski

k -neighbourhood.

K: KNN 파라미터

$R_{\min/\text{all}}$

희귀한 이웃의 수 / 전체 이웃의 수

Table 1. Rules to assign the four types of minority examples.

From: [Improving Imbalanced Classification by Anomaly Detection](#)

Type	Safe (S)	Borderline (B)	Rare (R)	Outlier (O)
Rule	$\frac{k+1}{2k} < R_{\min/\text{all}} \leq 1$	$\frac{k-1}{2k} \leq R_{\min/\text{all}} \leq \frac{k+1}{2k}$	$0 < R_{\min/\text{all}} < \frac{k-1}{2k}$	$R_{\min/\text{all}} = 0$
E.G. given the neighbourhood of a fixed size $k = 5$				
Rule	$\frac{3}{5} < R_{\min/\text{all}} \leq 1$	$\frac{2}{5} \leq R_{\min/\text{all}} \leq \frac{3}{5}$	$0 < R_{\min/\text{all}} < \frac{2}{5}$	$R_{\min/\text{all}} = 0$

실험

메소드

- 시나리오
 1. 아무 것도 안 한 원본
 2. Resampling
 3. Resampling + 추가 속성
 4. 추가 속성
- t-tests로 데이터셋 유사하게 조정
- K-fold 5

데이터셋

Table 2. Information on benchmark datasets [1].

From: [Improving Imbalanced Classification by Anomaly Detection](#)

Datasets	#Attributes	#Samples	Imbalance ratio (IR)
<i>glass1</i>	9	214	1.82
<i>ecoli4</i>	7	336	15.8
<i>vehicle1</i>	18	846	2.9
<i>yeast4</i>	8	1484	28.1
<i>wine quality</i>	11	1599	29.17
<i>page block</i>	10	5472	8.79

결과

The experiment with the two additional attributes outperforms the experiment with the classical resampling technique SMOTE.

두 가지 속성 추가 >>>> SMOTE

2D chess dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.8482	0.5743	0.6992	0.6208	0.8047	0.8285	---	---	---	---
	YES	0.9771	0.9557	0.9070	0.9226	0.9469	0.9859	0.9846	0.9485	0.9643	0.9723
SMOTE	NO	0.8584	0.6422	0.7102	0.6546	0.8183	0.5921	0.1636	0.5004	0.2437	0.5855
	YES	0.9704	0.9191	0.9061	0.9064	0.9453	0.9933	0.9833	0.9667	0.9622	0.9801
ADASYN	NO	0.8482	0.5743	0.6992	0.6208	0.8047	0.6172	0.1434	0.5904	0.2299	0.5892
	YES	0.9771	0.9557	0.9070	0.9226	0.9469	0.9925	0.8546	0.9667	0.8999	0.9721
NCL	NO	0.5786	0.1245	0.6652	0.3992	0.5541	0.5290	0.1076	0.4212	0.1693	0.4802
	YES	0.9715	0.8542	0.9667	0.8988	0.9716	0.9946	0.9119	0.9667	0.9337	0.9766
OSS	NO	0.7869	0.4197	0.4354	0.4354	0.4354	0.6262	0.3050	0.0205	0.0535	0.0958
	YES	0.9743	0.9321	0.9391	0.9318	0.9689	0.9937	0.9532	0.9564	0.9524	0.9745
glass1 dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.7029	0.6099	0.6235	0.6044	0.6309	0.6779	0.6394	0.5533	0.5828	0.6033
	YES	0.7328	0.6283	0.6344	0.6227	0.6956	0.7779	0.6506	0.65917	0.6430	0.7089
SMOTE	NO	0.7008	0.5750	0.6561	0.6000	0.6782	0.7140	0.5125	0.7256	0.5785	0.6111
	YES	0.7595	0.6393	0.6988	0.6589	0.7273	0.8288	0.6537	0.8802	0.7369	0.7760
ADASYN	NO	0.7095	0.5922	0.6728	0.6187	0.6842	0.7338	0.5159	0.7982	0.6103	0.6271
	YES	0.7799	0.6614	0.7106	0.6780	0.7419	0.8388	0.6545	0.8996	0.7456	0.7845
NCL	NO	0.5926	0.4401	0.9302	0.5843	0.3761	0.6750	0.4124	1.0000	0.5765	0.2177
	YES	0.5897	0.3976	0.9239	0.5527	0.3806	0.7790	0.4299	1.0000	0.5948	0.3403
OSS	NO	0.7010	0.5688	0.6841	0.6132	0.6804	0.6810	0.5850	0.5837	0.5883	0.6444
	YES	0.7611	0.6342	0.7136	0.6637	0.7295	0.7784	0.6085	0.7382	0.6543	0.7128
ecoli1 dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.8448	0.7241	0.6433	0.6432	0.7694	0.9919	0.8889	0.8000	0.7993	0.8797
	YES	0.8525	0.6435	0.6017	0.5734	0.6920	0.9889	0.9143	0.7500	0.7835	0.8512
SMOTE	NO	0.8824	0.7938	0.7233	0.7102	0.8328	0.9894	0.8290	0.8000	0.7268	0.8457
	YES	0.8629	0.8315	0.7300	0.7262	0.8303	0.9931	0.8824	0.9500	0.8881	0.9639
ADASYN	NO	0.8719	0.8407	0.7083	0.7221	0.8236	0.9903	0.7813	0.8000	0.7034	0.8389
	YES	0.8747	0.7833	0.6717	0.6623	0.7822	0.9934	0.8800	0.9500	0.8857	0.9634
NCL	NO	0.8007	0.6980	0.6333	0.5651	0.7380	0.9869	0.8258	0.9000	0.7886	0.8978
	YES	0.8523	0.7297	0.7550	0.6499	0.7982	0.9914	0.8533	0.9500	0.8556	0.9549
OSS	NO	0.8398	0.6284	0.7250	0.5958	0.7872	0.9877	0.8458	0.8133	0.7580	0.8668
	YES	0.9115	0.6858	0.8350	0.6787	0.8586	0.9890	0.8830	0.9117	0.8626	0.9408
vehicle1 dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.6699	0.5018	0.4301	0.4575	0.6004	0.8673	0.7074	0.3593	0.4747	0.5824
	YES	0.7385	0.5855	0.5329	0.5379	0.6081	0.6873	0.6296	0.6536	0.6536	0.7500
SMOTE	NO	0.7241	0.5398	0.5357	0.5196	0.5935	0.8945	0.9237	0.6913	0.8264	0.8882
	YES	0.7403	0.5825	0.5629	0.5704	0.6938	0.9204	0.9808	0.9745	0.7272	0.8882
ADASYN	NO	0.7211	0.5359	0.5570	0.5446	0.6791	0.8995	0.5485	0.9465	0.6537	0.8303
	YES	0.7481	0.5842	0.5789	0.5797	0.7025	0.9206	0.9800	0.9809	0.7284	0.8597
NCL	NO	0.7411	0.4153	0.9506	0.5769	0.7093	0.8411	0.4108	0.9768	0.5776	0.7059
	YES	0.7781	0.4560	0.9392	0.6118	0.7529	0.8752	0.5076	1.0000	0.6728	0.8139
OSS	NO	0.7125	0.4857	0.6060	0.5370	0.6837	0.8702	0.5745	0.7014	0.6293	0.7560
	YES	0.7531	0.5524	0.6286	0.5859	0.7174	0.9062	0.6088	0.9117	0.7290	0.8515
yeast4 dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.6736	0.3619	0.2217	0.2653	0.4482	0.8469	---	---	---	---
	YES	0.8647	0.8320	0.6708	0.7260	0.8132	0.9910	0.8628	0.8036	0.8270	0.8920
SMOTE	NO	0.7320	0.2632	0.6029	0.3082	0.6082	0.9052	0.2112	0.6769	0.3160	0.7773
	YES	0.9115	0.7665	0.6892	0.7171	0.8235	0.9922	0.7096	0.9442	0.8079	0.9639
ADASYN	NO	0.7226	0.2494	0.3958	0.2963	0.6041	0.9011	0.2061	0.6902	0.3104	0.7815
	YES	0.9114	0.7531	0.6553	0.6906	0.8096	0.9923	0.6951	0.9618	0.8051	0.9727
NCL	NO	0.8176	0.1929	0.6819	0.2992	0.7772	0.9063	0.2552	0.5745	0.3516	0.7256
	YES	0.9785	0.6733	0.9772	0.7928	0.9791	0.9917	0.7512	0.9436	0.8337	0.9649
OSS	NO	0.7066	0.2809	0.3561	0.3020	0.5713	0.8488	0.2094	0.0258	0.0447	0.0781
	YES	0.9130	0.7637	0.7699	0.7532	0.8708	0.9892	0.8312	0.8390	0.8310	0.9121
wine quality dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.5844	0.1180	0.1275	0.1132	0.2817	0.9790	0.9653	0.9113	0.9333	0.9525
	YES	0.9790	0.9653	0.9113	0.9333	0.9525	0.9944	0.9636	0.8274	0.8761	0.9031
SMOTE	NO	0.5597	0.0648	0.1801	0.0930	0.3704	0.6935	0.1065	0.4223	0.1680	0.5941
	YES	0.9685	0.9715	0.8630	0.9031	0.9239	0.9942	0.8809	0.9055	0.8890	0.9488
ADASYN	NO	0.5601	0.0654	0.1909	0.0963	0.3800	0.6920	0.1039	0.4231	0.1650	0.5933
	YES	0.9859	0.9709	0.8467	0.8917	0.9141	0.9944	0.8805	0.9055	0.8888	0.9488
NCL	NO	0.5922	0.1057	0.2563	0.1421	0.4817	0.7207	0.2582	0.1891	0.1818	0.3755
	YES	0.9845	0.8567	0.9492	0.8949	0.9703	0.9939	0.9359	0.8818	0.8890	0.9308
OSS	NO	0.5733	0.0729	0.2158	0.1054	0.4135	0.5078	---	---	---	---
	YES	0.9859	0.9636	0.9818	0.9723	0.9901	0.9941	0.9282	0.9424	0.9307	0.9690
page block dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.9083	0.8108	0.7442	0.7687	0.8519	0.9723	0.8743	0.7046	0.7603	0.8304
	YES	0.9369	0.8535	0.8289	0.8360	0.9014	0.9880	0.8481	0.8460	0.8379	0.9091
SMOTE	NO	0.9122	0.7485	0.7910	0.7620	0.8735	0.9646	0.6815	0.8792	0.7536	0.8699
	YES	0.9300	0.8216	0.8404	0.8245	0.9051	0.9847	0.7404	0.9496	0.8251	0.9533
ADASYN	NO	0.9130	0.7302	0.7990	0.7558	0.8763	0.9613	0.5716	0.9277	0.6983	0.9194
	YES	0.9328	0.8452	0.8321	0.8366	0.9032	0.9843	0.7529	0.9726	0.8435	0.9661
NCL	NO	0.9338	0.6528	0.9091	0.7502	0.9223	0.9669	0.6628	0.8690	0.7412	0.9127
	YES	0.9663	0.7318	0.9400	0.8186	0.9474	0.9844	0.7355	0.9606	0.8255	0.9577
OSS	NO	0.9071	0.7297	0.7036	0.7473	0.8711	0.9555	0.8375	0.8755	0.7310	0.8107
	YES	0.9248	0.7820	0.8349	0.7967	0.8972	0.9808	0.7845	0.8655	0.8111	0.9137

결과 - 자세히

- 2D chess
 - None + yes : 0.9771
 - SMOTE + no : 0.8584
 - SMOTE + yes : 0.9704
- Yeast4
 - None + yes : 0.8647
 - SMOTE + no : 0.7320
 - SMOTE + yes : 0.9115
- Wine quality
 - None + yes : 0.9790
 - SMOTE + no : 0.5597
 - SMOTE + yes : 0.9685

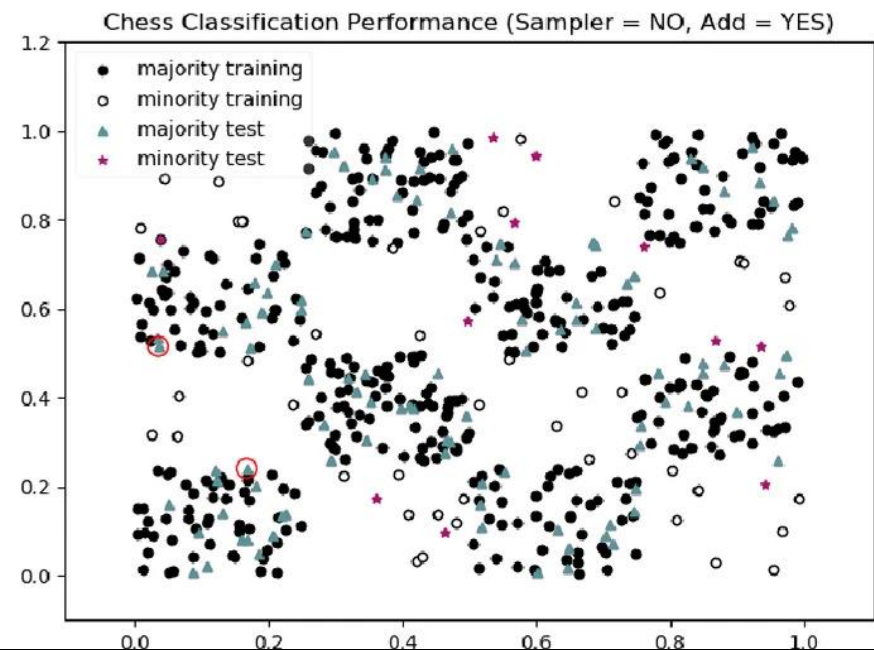
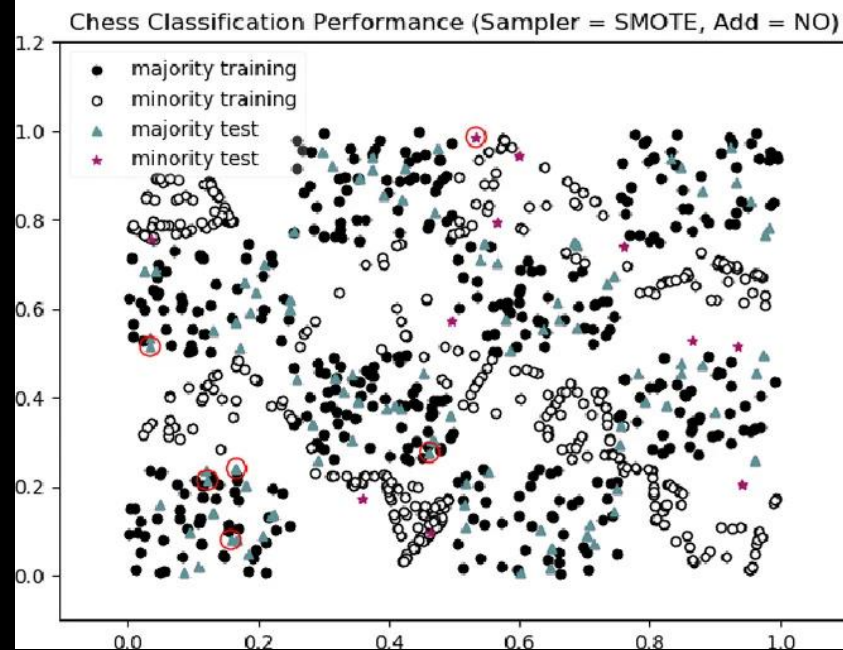
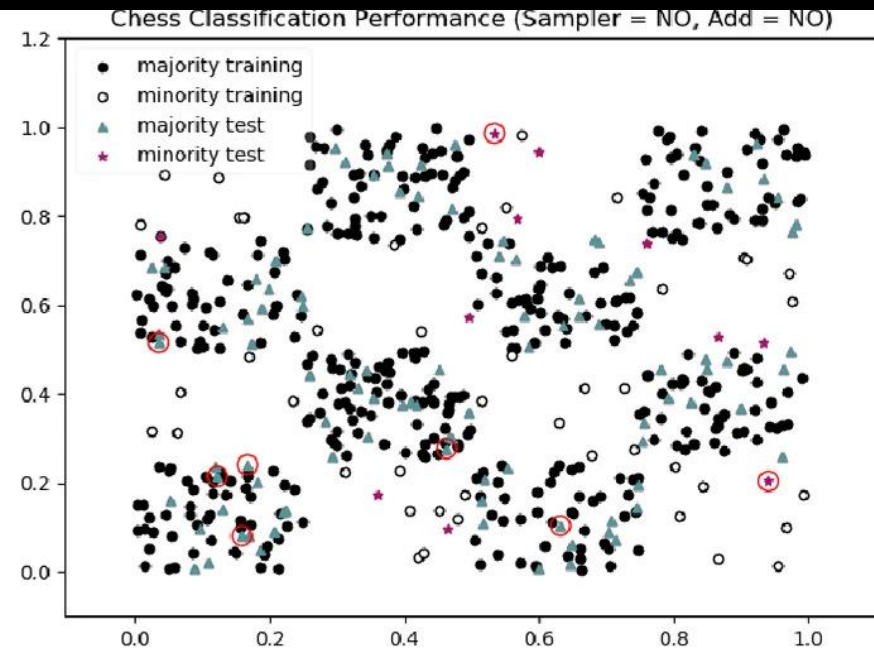
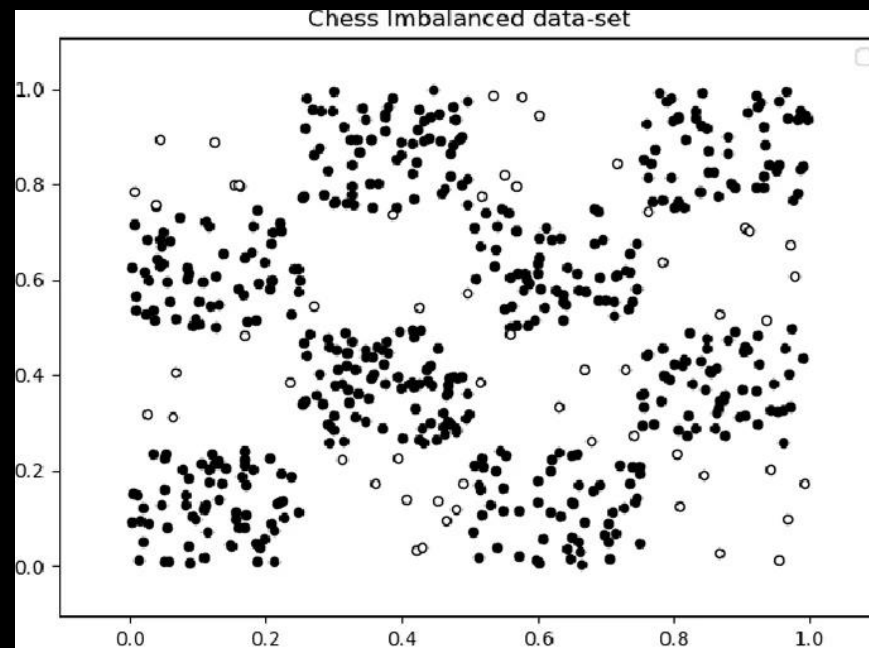
2D chess dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.8482	0.5743	0.6992	0.6208	0.8047	0.8285	---	---	---	---
	YES	0.9771	0.9557	0.9070	0.9226	0.9469	0.9859	0.9846	0.9485	0.9643	0.9723
SMOTE	NO	0.8584	0.6422	0.7102	0.6546	0.8183	0.5921	0.1636	0.5004	0.2437	0.5855
	YES	0.9704	0.9191	0.9061	0.9064	0.9453	0.9933	0.9833	0.9667	0.9622	0.9801
ADASYN	NO	0.8482	0.5743	0.6992	0.6208	0.8047	0.6172	0.1434	0.5904	0.2299	0.5892
	YES	0.9771	0.9557	0.9070	0.9226	0.9469	0.9925	0.8546	0.9667	0.8999	0.9721
NCL	NO	0.5786	0.1245	0.6652	0.3992	0.5541	0.5290	0.1076	0.4212	0.1693	0.4802
	YES	0.9715	0.8542	0.9667	0.8988	0.9716	0.9946	0.9119	0.9667	0.9337	0.9766
OSS	NO	0.7869	0.4197	0.4354	0.4354	0.4354	0.6262	0.3050	0.0205	0.0535	0.0958
	YES	0.9743	0.9321	0.9391	0.9318	0.9689	0.9937	0.9532	0.9564	0.9524	0.9745
glass1 dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.7029	0.6099	0.6235	0.6044	0.6309	0.6779	0.6394	0.5533	0.5828	0.5933
	YES	0.7328	0.6283	0.6344	0.6227	0.6956	0.7779	0.6506	0.65917	0.6430	0.7089
SMOTE	NO	0.7008	0.5750	0.6561	0.6000	0.6782	0.7140	0.5125	0.7256	0.5785	0.6111
	YES	0.7595	0.6393	0.6988	0.6589	0.7273	0.8288	0.6537	0.8802	0.7369	0.7760
ADASYN	NO	0.7095	0.5922	0.6728	0.6187	0.6842	0.7338	0.5159	0.7982	0.6103	0.6271
	YES	0.7799	0.6614	0.7106	0.6780	0.7419	0.8388	0.6545	0.8996	0.7456	0.7845
NCL	NO	0.5926	0.4401	0.9302	0.5843	0.3761	0.6750	0.4124	1.0000	0.5765	0.2177
	YES	0.5897	0.3976	0.9239	0.5527	0.3806	0.7790	0.4299	1.0000	0.5948	0.3403
OSS	NO	0.7010	0.5688	0.6841	0.6132	0.6804	0.6810	0.5850	0.5837	0.5883	0.6444
	YES	0.7611	0.6342	0.7136	0.6637	0.7295	0.7784	0.6085	0.7382	0.6543	0.7128
ecoli4 dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.8448	0.7241	0.6433	0.6432	0.7094	0.9919	0.8889	0.8000	0.7993	0.8797
	YES	0.8525	0.6435	0.6017	0.5734	0.6920	0.9889	0.9143	0.7500	0.7835	0.8512
SMOTE	NO	0.8824	0.7938	0.7233	0.7102	0.8328	0.9894	0.8290	0.8000	0.7268	0.8457
	YES	0.8629	0.8315	0.7300	0.7262	0.8303	0.9931	0.8824	0.9500	0.8881	0.9639
ADASYN	NO	0.8719	0.8407	0.7083	0.7221	0.8236	0.9903	0.7813	0.8000	0.7034	0.8389
	YES	0.8747	0.7833	0.6717	0.6623	0.7822	0.9934	0.8800	0.9500	0.8857	0.9634
NCL	NO	0.8007	0.6980	0.6333	0.5651	0.7380	0.9869	0.8258	0.9000	0.7886	0.8978
	YES	0.8523	0.7297	0.7550	0.6499	0.7982	0.9914	0.8533	0.9500	0.8556	0.9549
OSS	NO	0.8398	0.6284	0.7250	0.5958	0.7872	0.9877	0.8458	0.8133	0.7580	0.8668
	YES	0.9115	0.6858	0.8350	0.6787	0.8586	0.9890	0.8830	0.9117	0.8626	0.9408
vehicle1 dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.6699	0.5018	0.4301	0.4575	0.6004	0.8673	0.7074	0.3593	0.4747	0.5821
	YES	0.7385	0.5855	0.5329	0.5379	0.6794	0.9081	0.6873	0.6266	0.6536	0.7500
SMOTE	NO	0.7241	0.5398	0.5357	0.5357	0.5796	0.8935	0.5358	0.3227	0.5913	0.8264
	YES	0.7403	0.5825	0.5629	0.5704	0.6938	0.9204	0.5808	0.3745	0.7272	0.8882
ADASYN	NO	0.7211	0.5359	0.5570	0.5446	0.6791	0.8995	0.5485	0.3465	0.5937	0.8303
	YES	0.7481	0.5842	0.5789	0.5797	0.7025	0.9206	0.5800	0.3809	0.7284	0.8597
NCL	NO	0.7411	0.4153	0.9506	0.5709	0.7093	0.8411	0.4108	0.9768	0.5776	0.7059
	YES	0.7781	0.4560	0.9392	0.6118	0.7529	0.8752	0.5076	1.0000	0.6728	0.8139
OSS	NO	0.7125	0.4857	0.6060	0.5370	0.6837	0.8702	0.5745	0.7014	0.6293	0.7560
	YES	0.7531	0.5524	0.6286	0.5859	0.7174	0.9062	0.6088	0.9117	0.7290	0.8515
yeast4 dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.6736	0.3619	0.2217	0.2653	0.4482	0.8469	---	---	---	---
	YES	0.8647	0.8320	0.6708	0.7260	0.8132	0.9910	0.8628	0.8036	0.8270	0.8920
SMOTE	NO	0.7320	0.2632	0.6029	0.3082	0.6082	0.9052	0.2112	0.6769	0.3160	0.7773
	YES	0.9115	0.7665	0.6892	0.7171	0.8235	0.9922	0.7096	0.9442	0.8079	0.9639
ADASYN	NO	0.7226	0.2494	0.3958	0.2963	0.6041	0.9011	0.2061	0.6902	0.3104	0.7815
	YES	0.9114	0.7531	0.6553	0.6906	0.8090	0.9923	0.6951	0.9618	0.8051	0.9727
NCL	NO	0.8176	0.1929	0.6819	0.2992	0.7772	0.9063	0.2552	0.5745	0.3516	0.7256
	YES	0.9785	0.6733	0.9772	0.7928	0.9791	0.9917	0.7512	0.9436	0.8337	0.9649
OSS	NO	0.7066	0.2809	0.3561	0.3020	0.5713	0.8488	0.2094	0.0258	0.0447	0.0781
	YES	0.9130	0.7637	0.7699	0.7532	0.8708	0.9892	0.8312	0.8390	0.8310	0.9121
wine quality dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.5844	0.1180	0.1275	0.1132	0.2817	0.9790	0.9653	0.9113	0.9333	0.9525
	YES	0.9790	0.9653	0.9113	0.9333	0.9525	0.9944	0.9636	0.8274	0.8761	0.9031
SMOTE	NO	0.5597	0.0648	0.1801	0.0930	0.3704	0.6935	0.1065	0.4223	0.1680	0.5941
	YES	0.9685	0.9715	0.8630	0.9031	0.9239	0.9942	0.8809	0.9055	0.8890	0.9488
ADASYN	NO	0.5601	0.0654	0.1909	0.0953	0.3800	0.6920	0.1039	0.4231	0.1650	0.5933
	YES	0.9859	0.9709	0.8467	0.8917	0.9141	0.9944	0.8805	0.9055	0.8888	0.9488
NCL	NO	0.5922	0.1057	0.2563	0.1421	0.4817	0.7207	0.2582	0.1891	0.1818	0.3755
	YES	0.9845	0.8567	0.9492	0.8949	0.9703	0.9939	0.9359	0.8818	0.8890	0.9308
OSS	NO	0.5733	0.0729	0.2158	0.1054	0.4135	0.5078	---	---	---	---
	YES	0.9859	0.9636	0.9818	0.9723	0.9901	0.9941	0.9282	0.9424	0.9307	0.9690
page block dataset											
Methods	Add	Decision Tree					SVM				
		AUC	Precision	Recall	F1	Gmean	AUC	Precision	Recall	F1	Gmean
NONE	NO	0.9083	0.8108	0.7442	0.7687	0.8519	0.9723	0.8743	0.7046	0.7603	0.8304
	YES	0.9369	0.8535	0.8289	0.8360	0.9014	0.9880	0.8481	0.8460	0.8379	0.9091
SMOTE	NO	0.9122	0.7485	0.7910	0.7620	0.8735	0.9646	0.6815	0.8792	0.7536	0.8699
	YES	0.9300	0.8216	0.8404	0.8245	0.9051	0.9847	0.7404	0.9496	0.8251	0.9533
ADASYN	NO	0.9130	0.7302	0.7990	0.7558	0.8763	0.9613	0.5716	0.9277	0.6983	0.9194
	YES	0.9328	0.8452	0.8321	0.8366	0.9032	0.9843	0.7529	0.9726	0.8435	0.9661
NCL	NO	0.9338	0.6528	0.9091	0.7502	0.9223	0.9669	0.6628	0.8690	0.7412	0.9127
	YES	0.9663	0.7318	0.9400	0.8186	0.9474	0.9844	0.7355	0.9606	0.8255	0.9577
OSS	NO	0.9071	0.7297	0.7036	0.7473	0.8711	0.9555	0.8375	0.8755	0.7510	0.8107
	YES	0.9248	0.7820	0.8349	0.7967	0.8972	0.9808	0.7845	0.8655	0.8111	0.9137

결과

시각화

빨간색 동그라미:
잘못 분류된 것

아래 2개 비교



결과

Feature Importance

추가된 속성이 분류할 때 유의미하게 쓰임

2D chess dataset (2 original & 2 added attributes)													
Score Attr Add	org1	org2	add1	add2	—	—	—	—	—	—	—	—	—
NO	0.4636	0.5364	—	—	—	—	—	—	—	—	—	—	—
YES	0.0101	0.0097	0.8152	0.1636	—	—	—	—	—	—	—	—	—
glass1 dataset (9 original & 2 added attributes)													
Score Attr Add	org1	org2	org3	org4	org5	org6	org7	org8	org9	add1	add2	—	—
NO	0.2063	0.0213	0.2354	0.1291	0.0302	0.0418	0.2634	0.0000	0.0726	—	—	—	—
YES	0.1770	0.0056	0.1527	0.1099	0.0000	0.0110	0.1892	0.0000	0.0056	0.2413	0.1077	—	—
ecoli4 dataset (7 original & 2 added attributes)													
Score Attr Add	org1	org2	org3	org4	org5	org6	org7	add1	add2	—	—	—	—
NO	0.1693	0.0587	0.0000	0.0000	0.6591	0.1729	0.0000	—	—	—	—	—	—
YES	0.0000	0.0337	0.0000	0.0000	0.6119	0.0000	0.0808	0.1742	0.0994	—	—	—	—
vehicle1 dataset (18 original & 2 added attributes)													
Score Attr Add	org1	org2	org3	org4	org5	org6	org7	org8	org9	org10	org11	org12	org13
NO	0.1304	0.0654	0.0892	0.0403	0.0563	0.0233	0.0028	0.0707	0.0000	0.0635	0.0172	0.0416	0.0438
YES	0.0248	0.0216	0.1317	0.0426	0.0227	0.0205	0.0179	0.0024	0.0000	0.0338	0.0290	0.1291	0.0828
Score Attr Add	org14	org15	org16	org17	org18	add1	add2	—	—	—	—	—	—
NO	0.0862	0.0414	0.0498	0.0516	0.1265	—	—	—	—	—	—	—	—
YES	0.0146	0.0310	0.0325	0.0291	0.0471	0.2413	0.0485	—	—	—	—	—	—
yeast4 dataset (8 original & 2 added attributes)													
Score Attr Add	org1	org2	org3	org4	org5	org6	org7	org8	add1	add2	—	—	—
NO	0.3301	0.2446	0.1839	0.0720	0.0106	0.0000	0.1253	0.0355	—	—	—	—	—
YES	0.0385	0.0297	0.0483	0.0116	0.0000	0.0000	0.0248	0.0053	0.7771	0.0646	—	—	—
wine quality dataset (11 original & 2 added attributes)													
Score Attr Add	org1	org2	org3	org4	org5	org6	org7	org8	org9	org10	org11	add1	add2
NO	0.0466	0.1802	0.1215	0.1194	0.0806	0.0635	0.0428	0.1287	0.0483	0.0841	0.1244	—	—
YES	0.0000	0.0098	0.0000	0.0000	0.0000	0.0000	0.0263	0.0000	0.0000	0.0000	0.0000	0.9629	0.0000
page block dataset (10 original & 2 added attributes)													
Score Attr Add	org1	org2	org3	org4	org5	org6	org7	org8	org9	org10	add1	add2	—
NO	0.5452	0.0096	0.0117	0.1899	0.0530	0.0285	0.0983	0.0382	0.0122	0.0134	—	—	—
YES	0.5282	0.0006	0.0036	0.1745	0.0205	0.0223	0.0833	0.0129	0.0007	0.0082	0.1288	0.0164	—

결론

1. In most cases, introducing these two additional attributes can improve the class imbalance classification performance. For some datasets, only introducing additional attributes gives better classification results than only performing resampling techniques.
2. An analysis of the experimental results also illustrates that the proposed method has a better ability to handle samples in the overlapping region

결론 3줄 요약

1. 추가 속성 2개(Outlier score, 4 Type) 넣으면
Resampling보다 더 성능 향상한다.
2. overlapping region(중복 영역)에서도 더 우수하다.
3. 꼭 넣어라

의문점

실험의 의문점. 이상치를 어떻게 구하나?

소스코드 분석

<https://github.com/FayKong/PPSN2020>

논문에서 사용한 소스코드 공개

Product Solutions Open Source Pricing

FayKong / PPSN2020 Public

<> Code Issues 3 Pull requests Actions Projects Security Insights

master 1 branch 0 tags Go to file Code

FayKong Merge branch 'master' into master	549f26f on Feb 22, 2022	48 commits
2D chess	Update chess_LOF_plot.py	last year
Imbalanced Benchmark	Update and rename benchmark_four_types.py to Benchmark_four_types....	2 years ago
data	Delete ecoli	2 years ago
results	Delete test.txt	2 years ago
LICENSE	Create LICENSE	2 years ago
README.md	Merge branch 'master' into master	last year
requirements.txt	Create requirements.txt	last year

README.md

Improving Imbalanced Classification by Anomaly Detection

This repository contains the code for introducing two additional attributes to imbalanced datasets and thus improving the imbalanced classification

Outlier score

전체 데이터를 기준으로 한다.

```
Benchmark_LOF.py x
Package requirements 'scikit-learn==0.23.1', 'numpy==1.21.0', 'matplotlib==3.4.2' are not satisfied Install requirements Ignore requirements
1 import ...
16
17 # %% create chess dataset
18
19 glass1_data = pd.read_csv('~data/glass1.csv')
20 X = glass1_data.iloc[:, 0:9]
21
22
23 y = glass1_data['Class']
24 X = StandardScaler().fit_transform(X)
25 X = np.c_[X]
26
27 ir = len(y[y == 1]) / len(y)
28 clf = LocalOutlierFactor(n_neighbors=10, contamination=ir) # highly rely on the IR in training set
29 y_pred = clf.fit_predict(X)
30 X_scores = clf.negative_outlier_factor_
31 X_scores_1 = np.array([X_scores[x:x + 1] for x in range(0, len(X_scores), 1)]) # change shape (n,) to (n, 1)
32 X_all['LOF'] = X_scores_1
33
34 # # experiment with LOF score
35 X_new = X_all.drop(['Class'], axis=1)
36 X_add = StandardScaler().fit_transform(X_new)
37 X_add = np.c_[X_add]
38
```

전체데이터를 기준으로 한다.

전체데이터를 기준으로 한다.

```

chess_four_types.py
Package requirements 'scikit-learn==0.23.1', 'numpy==1.21.0', 'matplotlib==3.4.2' are not installed. Install requirements. Ignore requirements.
1 25 A 2
2
3 1 usage
4
5 def npr_diff(x, y):
6     scale = 4 * np.nanstd(X, axis=0)
7     result = np.sum(np.square(abs(x - y) / scale))
8     return result
9
10 die_metric = np.zeros((X.shape[0], X.shape[0]))
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1
```


메소드


Decision tree, SVM 사용


Train, test 나누지 않고 모델 학습


Imbalanced Benchmark

 Baseline_DT_resampling.py

 Benchmark_DT.py


 Benchmark_DT_added.py

 Benchmark_DT_added_resampling.py

 Benchmark_four_types.py

 Benchmark_LOF.py

 Benchmark_SVM.py

 Benchmark_SVM_added.py

 Benchmark_SVM_added_resampling.py

 Benchmark_SVM_resampling.py

일반적인 머신러닝 모델 학습 과정

전체 데이터 100

- Train: 80
 - Train: 64
 - Validation: 16
- Test: 20

전체 데이터

- 모델 학습할 때 쓰는 데이터
 - 모델 학습
 - 모델 검증 (과적합 방지 등)
- 학습완료한 모델의 성능 테스트

모델 학습 과정 중에 이상치는 어떻게..?

전체 데이터 100

- Train: 80
 - Train: 64
 - Validation: 16
- Test: 20

전체 데이터의 이상치?

- Train의 이상치?
 - Train의 이상치?
 - Validation의 이상치?
- Test의 이상치?

전체 데이터를 기준으로 했을 때의 문제점

- 과적합
 - 학습한 모델에 새로운 데이터가 들어오면 잘 판단할 수 있을까?
- 의미있는 실험인가?
 - 애초에 Outlier score, type을 제공하고 학습
 - 정답을 보여주는 것과 다름 없지 않나?

마무리

결론, 향후 계획

결론

- 논문을 잘못 잡았다.
- 이 논문
 - 이상치 점수를 요소로 추가하여 머신러닝
- 내가 원했던 논문
 - 기존 머신러닝 알고리즘을 안 씀
 - 이상치 탐지 알고리즘을 이용하여 데이터 분석

향후 계획

- 전통적인 Outlier detection method 연구
 - Tukey's IQR method
 - Standard deviation method
 - Z-score method
 - Isolation Forest
 - DBSCAN
- 딥러닝 Outlier detection method 연구
 - Anomaly transformer 등등
- 현재 보유중인 데이터셋 활용 방안 탐구
 - 실험에 활용된 데이터셋 분석
 - 유사한 데이터셋에 적용