Politechnika
Wrocławska

Politechnika Wrocławska
Wydział Informatyki i Telekomunikacji
Kierunek Teleinformatyka

Metody Sztucznej Inteligencji - Projekt

Implementacja algorytmu oversamplingu ADASYN

Autorzy:
Kałwa Weronika 263876
Postawa Sandra 263826
Zych Zuzanna 263882

1 Wstęp i przegląd literatury

1.1 Omówienie działania algorytmu oversamplingu ADASYN

Algorytm ADASYN (Adaptive Synthetic Sampling - Adaptacyjne Syntetyczne Próbkowanie) jest zaawansowaną techniką oversamplingu, mającą na celu zwiększenie dokładności klasyfikatorów w problemach związanych z niezbalansowanymi zbiorami danych. Główna idea algorytmu opiera się na generowaniu syntetycznych przykładów dla mniejszościowej klasy w taki sposób, aby równoważyć zbiór danych, skupiając się przede wszystkim na tych obszarach, gdzie granica decyzyjna między klasami jest niejasna. W porównaniu do innych metod, takich jak SMOTE, ADASYN dąży do bardziej zróżnicowanego i adaptacyjnego generowania próbek, poprawiając w ten sposób klasyfikację w trudniejszych obszarach [1].

1.2 Cel projektu

Celem projektu jest implementacja algorytmu ADASYN w kontekście balansowania zbioru danych w problemach klasyfikacji z niezrównoważonymi klasami. Projekt skupi się na zrozumieniu działania algorytmu, jego implementacji praktycznej oraz ocenie efektywności w porównaniu z innymi technikami oversamplingu.

1.3 Przegląd literatury

1.3.1 Omówienie istniejących metod

Oprócz ADASYN istnieją także inne metody oversampligu, takie jak:

- SMOTE (Synthetic Minority Over-sampling Technique) [2]: Jest to technika statystyczna zwiększająca liczbę przypadków w zestawie danych w zrównoważony sposób. Składnik działa przez wygenerowanie nowych wystąpień z istniejących przypadków mniejszości, które są dostarczane jako dane wejściowe. Ta implementacja programu SMOTE nie zmienia liczby przypadków większościowych.
- BorderlineSMOTE [3]: Jest to modyfikacja metody SMOTE, ogranicza tworzenie nowych obiektów jedynie do granicy między przykładami z obydwu klas, co ma na celu zmniejszenie ryzyka przeuczenia.
- SVM-SMOTE [4]: Jest to metoda nadpróbkowania danych, która używa wektorów nośnych z SVM do generowania syntetycznych próbek klasy mniejszościowej przez interpolację między wektorami nośnymi a ich najbliższymi sąsiadami tej samej klasy, zwiększając liczbę danych na granicy decyzyjnej.
- BAGGING [5] polega na wykorzystaniu zbioru trenującego algorytmu klasyfikacji. Tworzony jest zbiór klasyfikatorów, z których każdy wykorzystuje algorytm i trenowany jest na zbiorze trenującym. Zbiór trenujący powstaje poprzez wylosowanie przykładów (ze zwracaniem) ze zbioru. Losowanie odbywa się zgodnie z rozkładem jednostajnym. Liczności zbioru. Każdy klasyfikator przydziela kategorię przykładowi. Ostateczna kategoria jest tą, która najczęściej była proponowana przez klasyfikatory
- NearMiss [6] to technika undersamplingu. Jej celem jest zrównoważenie rozkładu klas poprzez losowe eliminowanie przykładów z klasy większościowej. Gdy instancje dwóch różnych klas są bardzo blisko siebie, usuwamy instancje z klasy większościowej, aby zwiększyć odstępy między tymi klasami. Aby zapobiec problemowi utraty informacji w większości technik undersamplingu, szeroko stosuje się metody bliskich sąsiadów.
- K-Nearest Neighbors [7] tworzy wyimaginowaną granicę, aby sklasyfikować dane. Gdy do przewidywania dodawane są nowe punkty danych, algorytm dodaje ten punkt do najbliższego punktu granicy.

1.3.2 Metoda referencyjna

Głównym celem projektu jest zaimplementowanie algorytmu oversamplingu ADASYN za pomocą istniejących funkcji i implementacji. Do badania zostaną wykorzystane dane wygenerowane dane syntetyczne. W projekcie metodą, do której porównywane będą wyniki, będzie zaimportowany ADASYN, SMOTE oraz BorderlineSMOTE.

2 Projekt eksperymentów:

Eksperymenty zakładają porównanie klasyfikacji zbiorów, dla wybranych metod. Początkowo opracowano własny algorytm, który został nazwany FutureADASYN, opierający się na algorytmie ADASYN. Następnie zarówno algorytm FutureAdasyn, jak i ADASYN został porównany z algorytmami SMOTE oraz BorderlineSMOTE. Na wszystkich algorytmach zostaną wykonane eksperymenty na danych syntetycznych oraz rzeczywistych.
Do wykonania eksperymentów zostaną użyte biblioteki: numpy, matplotlib, imblearn. Zostanie użyta technika walidacji krzyżowej do stabilizacji informacji o uzyskanych rezultatach.

Idea walidacji krzyżowej polega na podzieleniu dostępnych danych na kilka podzbiorów, zwanych częściami (ang. folds), a następnie przeprowadzeniu wielu iteracji treningu i walidacji modelu. W każdej iteracji jeden z podzbiorów zostaje wybrany jako zbiór walidacyjny, a pozostałe służą jako zbiór treningowy. Proces ten jest powtarzany wielokrotnie, aby każdy z podzbiorów został użyty przynajmniej raz jako zbiór walidacyjny. Ostatecznie, wyniki z poszczególnych iteracji są uśredniane, co daje ogólny wynik skuteczności modelu.

2.1 Metryki oceny wyników

Zostaną wykorzystane metryki oceny takie jak:

- Dokładność (Accuracy): mierzy procent poprawnie sklasyfikowanych próbek,
- Precyzja (Precision): określa stosunek poprawnie pozytywnie sklasyfikowanych przypadków do wszystkich pozytywnych przypadków,
- Czułość (Recall): określa stosunek poprawnie sklasyfikowanych pozytywnych przypadków do wszystkich rzeczywiście pozytywnych przypadków,
- F1-Score: średnia harmoniczna precyzji i czułości.

2.2 Testy statystyczne

Zostaną przeprowadzone testy statystyczne opierające się na teście t-Studenta, aby ocenić różnice między wynikami uzyskanymi dla różnych modeli lub różnych zestawów danych, oraz skuteczność różnych algorytmów.

2.3 Opis poszczególnych eksperymentów wraz z ich celami

Cel: Celem każdego eksperymentu jest zaobserwowanie działania algorytmów: ADASYN, FutureAdasyn, SMOTE, BorderlineSMOTE oraz porównania jego pracy w oparciu o algorytm ADASYN.
Opis: Wszystkie eksperymenty zostaną przeprowadzone na danych rzeczywistych oraz syntetycznych. A wyniki zostaną porównane za pomocą metryk: Accuracy, Precision, Recall, F1-Score.

2.4 Opis rzeczywistych zbiorów danych wraz z metodami ich pozyskania i/ lub sposobów generowania danych syntetycznych

2.4.1 Dane rzeczywiste:

Do testowania algorytmów została użyta rzeczywista baza danych *QSAR biodegradation* [8]. Jest to niezbalansowany zbiór danych zawierający wartości 41 atrybutów (deskryptorów molekularnych) zastosowanych do klasyfikacji 1055 substancji chemicznych na 2 klasy (gotowe i niełatwe do biodegradacji).

- Klasa 0: *gotowe do degradacji*, klasa mniejszościowa, 356 obiektów.
- Klasa 1: *niełatwe do degradacji*, klasa większościowa, 699 obiektów.

2.4.2 Dane syntetyczne:

Dane syntetyczne zostały wygenerowane za pomocą funkcji *make_classification* biblioteki *scikit-learn*. Zbiór niezbalansowany o wadze 0.1 dla klasy 0 oraz 0.9 dla klasy 1. Przykład kodu:

```
X, y = make_classification(n_samples=200,
                           n_features=2,
                           n_informative=2,
                           n_repeated=0,
                           n_redundant=0,
                           random_state=2137,
                           weights=[0.1, 0.9])
```

3 Bibliografia

Literatura

[1] https://medium.com/@ruinian/an-introduction-to-adasyn-with-code-1383a5ece7aa [maj 2024]

[2] https://learn.microsoft.com/pl-pl/azure/machine-learning/component-reference/smote?view=azureml-api-2

[3] https://pb.edu.pl/oficyna-wydawnicza/wp-content/uploads/sites/4/2021/12/Modelowanie-i-optymalizacja-1.pdf [maj 2024]

[4] https://www.blog.trainindata.com/oversampling-techniques-for-imbalanced-data/ [maj 2024]

[5] https://repo.pw.edu.pl/docstore/download/WEiTI-ab4d82b3-a859-462f-a20b-2823dec969b1/pandrusz_Metauczenie+a+mo%C5%BCliwo%C5%9B%C4%87+poprawy+skuteczno%C5%9Bci+klasyfikacji.pdf [marzec 2024]

[6] https://www.geeksforgeeks.org/ml-handling-imbalanced-data-with-smote-and-near-miss-algorithm-in-python/ [marzec 2024]

[7] https://www.geeksforgeeks.org/regression-using-k-nearest-neighbors-in-r-programming/?ref=lbp [maj 2024]

[8] https://archive.ics.uci.edu/dataset/254/qsar+biodegradation [kwiecień 2024]