Projekt: Ewaluacja Modelu Klasyfikacyjnego

Cel projektu

Celem projektu jest ewaluacja modeli klasyfikacyjnych przy użyciu dedykowanej klasy **Evaluator**. Należy wykorzystać wcześniej przygotowane klasy **Perceptron** oraz **Teacher** (własne lub gotowe) i rozwinąć projekt poprzez:

- Wczytywanie zbioru testowego z mapowaniem etykiet (atrybutów decyzyjnych),
- Implementację metody `evaluate`, która zliczy przypadki poprawnych/niepoprawnych klasyfikacji,
- Obliczenie miar ewaluacyjnych: dokładności, precyzji, pełności oraz F-miary,
- Prezentację wyników na konsoli z zaokrągleniem do 2 miejsc po przecinku.

Zadania obowiązkowe

1. **Nowy projekt i import istniejących klas**
    - Utwórz nowy projekt w wybranym środowisku programistycznym
    - Zaimportuj do projektu klasy **Perceptron** oraz **Teacher**
2. **Utworzenie klasy Evaluator**
    - Utwórz nową klasę o nazwie `Evaluator`
3. **Wczytywanie zbioru testowego**
    - Zaimplementuj w klasie `Evaluator` mechanizm wczytywania zbioru testowego z pliku CSV
    - Pamiętaj o odpowiednim mapowaniu etykiet decyzyjnych – przyjmij, że pierwsza napotkana wartość w zbiorze decyzyjnym jest mapowana na `1`, a wszystkie kolejne na `0`
4. **Implementacja metody evaluate**
    - Zaimplementuj metodę `evaluate`, która przechodzi przez wczytany zbiór testowy i zlicza:
        - **True Positives (TP):** poprawnie zwrócono wartość `1`
        - **False Positives (FP):** niepoprawnie zwrócono wartość `1`
        - **True Negatives (TN):** poprawnie zwrócono wartość `0`
        - **False Negatives (FN):** niepoprawnie zwrócono wartość `0`
5. **Obliczenie miar ewaluacyjnych**
    - Na podstawie zliczonych wartości oblicz:
        - Dokładność (Accuracy)
        - Precyzję (Precision)
        - Pełność (Recall)
        - F-miarę
    - Wyniki zaprezentuj na konsoli, zaokrąglając wartości do 2 miejsc po przecinku (podpowiedź: użyj `String.format`).

Zadania do domu (rozszerzenie)

1. **Rozszerzenie klasy Evaluator do obsługi wieloklasowych atrybutów decyzyjnych**
    - Dla wywołań metody `evaluate` z dwoma wartościami atrybutów decyzyjnych – ewaluator powinien obliczać i wyświetlać miary ewaluacyjne (dokładność, precyzję, pełność, F-miara).
    - Dla wywołań metody `evaluate` z więcej niż dwoma atrybutami decyzyjnymi – klasa powinna:
        - Zliczyć dane potrzebne do utworzenia macierzy omyłek
        - Wyświetlić prostą, graficzną reprezentację macierzy omyłek na konsoli (np. przy użyciu znaków ASCII)
2. **Prezentacja działania rozszerzonego ewaluatora**
    - Zaprezentuj działanie rozszerzonej klasy `Evaluator` na przynajmniej dwóch mini projektach, np.:
        - Jeden z wykorzystaniem modelu kNN
        - Jeden z wykorzystaniem modelu Perceptron (przy czym Perceptron może być oparty na gotowej implementacji)

Wskazówki

- **Mapowanie etykiet:** Spójność mapowania etykiet między zbiorem treningowym a testowym jest kluczowa – pierwsza napotkana etykieta musi być mapowana na `1`, a kolejne na `0`. Upewnij się, że dla obu zbiorów odpowiednia (ta sama) klasa jest reprezentowana przez wyjście perceptronu równe 1
- **Formatowanie wyników:** Do zaokrąglania wyników do 2 miejsc po przecinku możesz użyć metody `String.format` lub innego mechanizmu dostępnego w Javie

Podsumowanie

W ramach projektu studenci powinni:

- Utworzyć nowy projekt i zaimportować klasy **Perceptron** oraz **Teacher**.
- Zaimplementować klasę **Evaluator**, która:
    - Wczytuje zbiór testowy z odpowiednim mapowaniem etykiet decyzyjnych,
    - Zlicza przypadki TP, FP, TN i FN,
    - Oblicza miary ewaluacyjne (dokładność, precyzję, pełność, F-miara) i prezentuje wyniki na konsoli.

- (Do domu) Rozszerzyć funkcjonalność klasy **Evaluator** o obsługę wieloklasowych atrybutów decyzyjnych i prezentację macierzy omyłek.
- (Do domu) Zaprezentować działanie rozszerzonego ewaluatora na co najmniej dwóch mini projektach (np. kNN oraz Perceptron).