WSI - ćwiczenie 4.
Klasyfikacja

Jakub Romankiewicz 325063

1. Testy algorytmu z gotowych bibliotek

Pracę rozpoczęto od wczytania zbioru danych do Pandas DataFrame. W celu lepszego zrozumienia działania algorytmu, zapoznano się z implementacją klasy RandomForestClassifier z pakietu sklearn.ensemble. Algorytm ten korzysta z tych samych metod, tj. fit i predict, jak te, które zostały zaimplementowane w ramach tego ćwiczenia. Aby zastosować algorytm do zbioru danych, konieczne było przekształcenie tekstowych etykiet, takich jak np. płeć (M lub F), na numeryczne etykiety (np. 0 i 1). W tym celu stworzono funkcję label_encode.

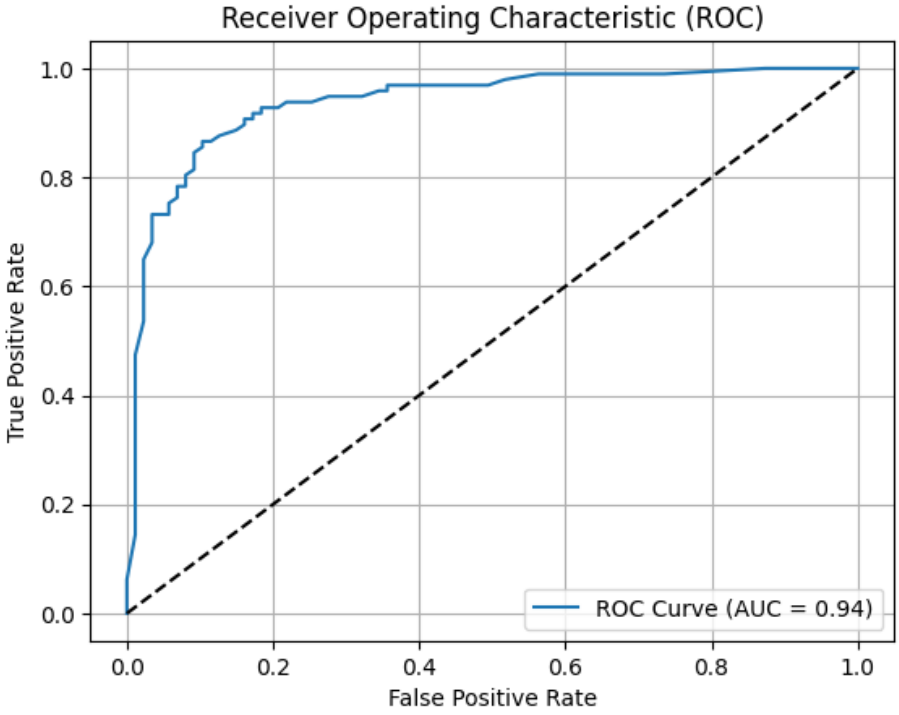
Następnie użyto funkcji train_test_split z biblioteki sklearn.model_selection, aby podzielić zbiór danych na część treningową i testową w klasycznych proporcjach 80% dla danych treningowych i 20% dla danych testowych. Funkcja ta losowo wybiera próbki ze zbioru danych, jednak dzięki parametrze random_state, zapewnia powtarzalność eksperymentów.

Na tym etapie stworzone zostały również funkcje get_metrics, roc_curve oraz rank_model, umożliwiające obliczenie metryk: dokładności (accuracy), precyzji (precision), czułości (recall) oraz krzywej ROC (ROC curve). Wartości tych metryk zostały następnie porównane z wynikami uzyskanymi za pomocą gotowych rozwiązań z biblioteki sklearn.metrics. Z tej samej biblioteki wykorzystano również funkcję auc(), do obliczenia pola pod krzywą ROC.

Wyniki algorytmu

Pojedyńczy pomiar dla liczby drzew = 100:

```
{'accuracy': 0.8695652173913043, 'precision': 0.8476190476190476, 'recall':
0.9175257731958762}
ROC AUC = 0.93796658371845
```



Rys. 1 Krzywa ROC dla algorytmu z bibioteki sklearn

Następnie przeprowadzono eksperyment polegający na uśrednianiu wyników klasyfikatora (z 10 różnych zestawów próbek) przy zmiennej n_estimators. Zmienność liczby drzew w lesie losowym będzie przydatna w kolejnych badaniach.

```
With 10 trees: accuracy: 0.8385869565217391, precision: 0.8530516514248495
With 50 trees: accuracy: 0.8456521739130434, precision: 0.8477415371443492
With 100 trees: accuracy: 0.8521739130434783, precision: 0.8500919609069733
With 200 trees: accuracy: 0.8516304347826086, precision: 0.850249863577855
With 500 trees: accuracy: 0.8543478260869565, precision: 0.849933873848633
```

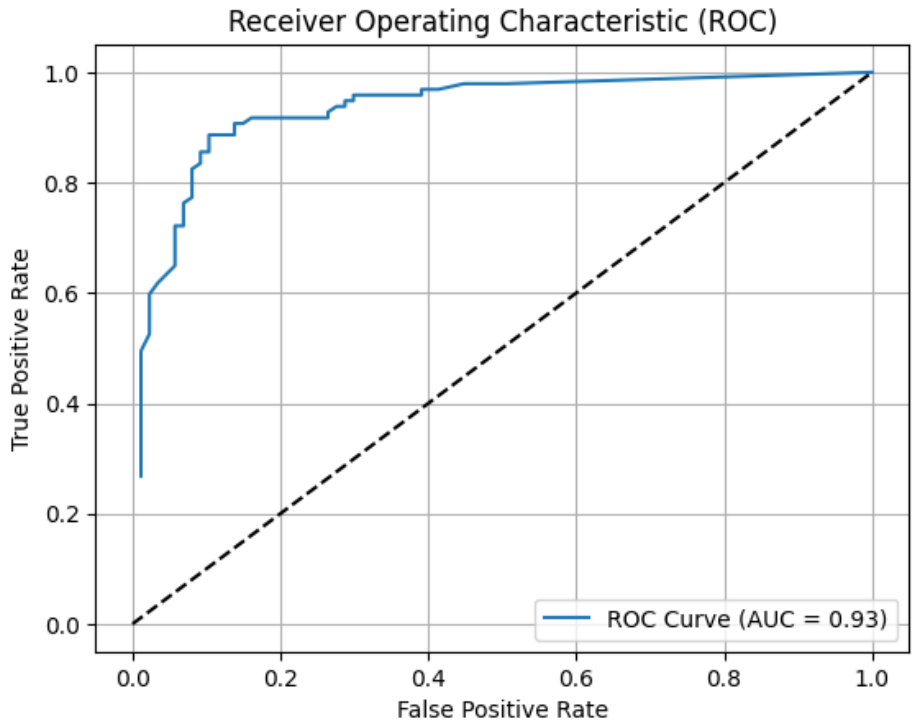
2. Implementacja algorytmu lasu losowego

Zaimplementowano klasę RandomForestSolver, która wiernie implementuje podany interfejs Solver. Algorytm ten zawiera funkcję fit(), która przyjmuje dane w formie DataFrame Pandas dla X i y oraz generuje las składający się z n_estimators drzew, z których każde jest trenowane na losowej próbce danych. Drzewa są budowane za pomocą algorytmu ID3, który wykorzystuje miarę informacji (entropię) do wyboru najlepszych cech i wartości podziału. Klasa oferuje również metodę predict(), która przewiduje klasy dla nowych próbek, oraz funkcję predict_proba(), która oblicza prawdopodobieństwa klasyfikacji.

Wyniki algorytmu

Pojedyńczy pomiar dla liczby drzew = 100:

```
{'accuracy': 0.8586956521739131, 'precision': 0.8317757009345794, 'recall':
0.9175257731958762}
ROC AUC = 0.9322194572816684
```



Rys. 1 Krzywa ROC dla własnej implementacji algorytmu

Ponownie przeprowadzono eksperyment polegający na uśrednianiu wyników klasyfikatora (z 10 różnych zestawów próbek) przy zmiennej n_estimators. Wyniki:

```
With 10 trees: accuracy: 0.8684782608695653, precision: 0.8694935212109292
With 50 trees: accuracy: 0.8679347826086957, precision: 0.8675294512519806
With 100 trees: accuracy: 0.865217391304348, precision: 0.8603995529997135
```

3. Próba poprawy wartości metryk za pomocą zmiany hiperparametru liczby drzew

Jak wynika z poprzednich eksperymentów, zmiana liczby drzew (n_estimators) i uśrednianie wyników nie prowadziły do zadowalających popraw w jakości klasyfikacji. Aby zwiększyć skuteczność modelu, konieczna byłaby zmiana samego algorytmu, na przykład poprzez losowanie części cech branych pod uwagę w algorytmie ID3, co mogłoby zwiększyć różnorodność drzew decyzyjnych.

4. Gdyby podany model był używany w rzeczywistości (np. w ochronie zdrowia) do przewidywania chorób serca, która z metryk powinna być maksymalizowana i dlaczego?

W kontekście przewidywania chorób serca, metryką, którą należy maksymalizować, jest precyzja (ang. precision). Precyzja mierzy, jaki procent przypadków, które model sklasyfikował jako pozytywne (np. pacjenci z chorobą serca), rzeczywiście jest pozytywny (czyli pacjent rzeczywiście ma chorobę). Jest to istotne w sytuacjach, gdzie błędne klasyfikowanie pacjentów jako chorych (fałszywe pozytywy) może prowadzić do niepotrzebnych działań medycznych, takich jak dodatkowe badania, leczenie czy niepokój u pacjentów.

5. Podsumowanie

W ćwiczeniu przeprowadzono testy algorytmu RandomForestClassifier z biblioteki sklearn oraz zaimplementowano własne rozwiązanie oparte na algorytmie ID3. Eksperymenty z różnymi liczbami drzew (n_estimators) nie wykazały znaczącej poprawy w jakości klasyfikacji, co sugeruje potrzebę wprowadzenia zmian w algorytmie, np. losowania cech w trakcie budowy drzew. Zastosowanie modelu w ochronie zdrowia wymaga maksymalizacji precyzji, aby uniknąć fałszywych pozytywów i niepotrzebnych działań medycznych.