Dokumentacja wstępna UMA

Andrzej Tokajuk, Aleksander Szymczyk

21.11.2024r.

1 Treść zadania - Las turniej

Zmodyfikowany las losowy w zadaniu klasyfikacji. Podczas budowy każdego z drzew testy (z pełnego zbioru testów) wybieramy za pomocą turnieju: https://staff.elka.pw.edu.pl/ rbiedrzy/UMA/ turniejRuletka.pdf, czyli wybieramy losowo 2 testy, liczymy ich jakość, stosujemy ten z wyższą jakością. Do wstępnych testów polecam zbiór danych z zadaniem klasyfikacji grzybów: https://archive.ics.uci.edu/ dataset/73/mushroom, na którym powinno dać się uzyskać wyniki w okolicy 100%. Do dalszych testów należy znaleźć i pobrać inne zbiory danych (na wskazanej stronie lub w Kaggle). Przed rozpoczęciem realizacji projektu proszę zapoznać się z zawartością: https://staff.elka.pw.edu.pl/ rbiedrzy/UMA/ index.html.

2 Opis algorytmów

Random Forest to algorytm uczenia zespołowego (ensemble learning). Metoda ta polega na trenowaniu wielu słabych modeli bazowych (np. drzew). Wyniki predykcji są następnie agregowane poprzez głosowanie większościowe w klasyfikacji lub uśrednianie w regresji.

2.1 Las losowy (Random Forest)

2.2 Krok 1: Inicjalizacja lasu losowego

- Określamy liczbę drzew N, które mają być utworzone w lesie losowym (hiperparametr oznaczony `n_estimators`).

2.3 Krok 2: Generowanie próbki bootstrapowej dla każdego drzewa

- Dla każdego z N drzew:
  - Utwórz próbkę danych przez **próbkowanie bootstrapowe** z oryginalnego zbioru danych (losowanie próbek z powtórzeniami).
  - Każde drzewo ma nieco inne dane do treningu.

2.4 Krok 3: Budowanie drzew CART

CART (Classification and Regression Trees) to algorytm budowy drzew decyzyjnych, który służy zarówno do klasyfikacji, jak i regresji.
Poniżej opisany jest algorytm budowy pojedynczego drzewa:

2.4.1 Krok 1: Przygotowanie danych

- **Zbiór danych** powinien być w formie tabelarycznej. Zbiór posiada ustaloną liczbę cech. Wszystkie przykłady posiadają wartości dla każdej z cech w formie kategorycznej (etykieta klasy) lub w formie ciągłej (wartość liczbowa).
- Jeżeli w oryginalnym zbiorze danych występują brakujące wartości muszą być one odpowiednio **wstępnie przetworzone** by nie występowały w zbiorze przekazywanym do drzewa.

2.4.2 Krok 2: Inicjalizacja drzewa

- Tworzony jest **korzeń drzewa**, który wstępnie zawiera cały zbiór treningowy.

2.4.3 Krok 3: Sprawdzenie warunku stopu

Warunki stopu:

- **Liczba próbek** w węźle jest mniejsza niż ustalona wartość (hiperparametr oznaczony `min_samples_split`)
- Wszystkie próbki w węźle należą do **jednej klasy**.
- Drzewo osiągnęło **maksymalną głębokość** (hiperparametr oznaczony `max_depth`).

Jeśli warunek stopu jest spełniony, dany węzeł jest liściem; w przeciwnym przypadku przechodzimy do kolejnego kroku.

2.4.4 Krok 4: Wybór podziału

Dla cech **ciągłych**:

- Unikalne wartości danej cechy ciągłej sortowane są w rosnącej kolejności.
- Rozważany jest punkt podziału pomiędzy każdą parą sąsiednich wartości w zbiorze danych. (Dla każdej pary sąsiednich wartości v_i i v_{i+1} algorytm rozważa średnią $t = \frac{v_i + v_{i+1}}{2}$).

Dla cech **kategorycznych**:

- Rozważane są wszystkie możliwe podziały klas cechy na podzbiory (poza zbiorami pustym i całym).
- Dla cechy kategorycznej o k unikalnych wartościach, liczba możliwych podziałów wynosi $2^{k-1} - 1$.

Dla każdej cechy obliczane są wszystkie możliwości podziału, następnie:

- Spośród wszystkich punktów podziału losowo wybierane są **dwa**.
- Dla obydwu testów obliczana jest funkcja przystosowania, czyli wartość jakości podziału. Do jej zmierzenia wykorzystywany jest **wskaźnik nieczystości Gini** (Gini impurity).
- $$\text{Gini} = 1 - \sum_{i=1}^{C} p_i^2$$

  gdzie:

  - C to liczba klas docelowych (tych, które próbujemy przewidzieć)
  - p_i to proporcja próbek należących do klasy i w danym węźle
- Następnie obliczamy **średnią ważoną wartości Gini** na podstawie liczby próbek w węzłach po podziale.
  $$Gini_{\text{Podział}} = \left(\frac{N_{\text{lewy}}}{N_{\text{całkowity}}} \times Gini_{\text{lewy}} \right) + \left(\frac{N_{\text{prawy}}}{N_{\text{całkowity}}} \times Gini_{\text{prawy}} \right)$$

  gdzie:

  - N_{lewy} — liczba próbek w lewym węźle potomnym,
  - N_{prawy} — liczba próbek w prawym węźle potomnym,
  - $N_{\text{całkowity}}$ — łączna liczba próbek w oryginalnym węźle przed podziałem,
  - $Gini_{\text{lewy}}$ i $Gini_{\text{prawy}}$ — wartości wskaźnika Gini dla lewego i prawego węzła potomnego.
- Z dwóch testów, na zasadzie **turnieju**, wybierany jest ten o mniejszej wartości (mniejsza wartość nieczystości - lepszy podział).

2.4.5 Krok 5: Podział węzła

- Na podstawie ustalonego podziału tworzymy dwa węzły potomne:
  - **Lewy węzeł**: Zawiera próbki spełniające warunek podziału.
  - **Prawy węzeł**: Zawiera próbki niespełniające warunku podziału.

2.4.6 Krok 6: Rekurencyjne powtarzanie procesu

- Dla każdego węzła potomnego:
  - Powtarzane są kroki od 3 do 6 dla tego węzła
  - Proces kontynuuje rekurencyjnie, aż wszystkie warunki stopu zostaną spełnione w każdym liściu.

2.4.7 Krok 7: Przypisanie wartości liściom

- Na końcu budowy drzewa, każdemu węzłowi końcowemu (liściowi) przypisywana jest odpowiednia wartość etykiety docelowej.
- Przyjmowana jest **klasa większościowa** próbek w danym węźle.

2.5 Krok 4: Powtarzanie procesu dla każdego drzewa

- Powtarzamy kroki 2 oraz 3 aż do utworzenia N drzew, czyli zakończenia budowy całego lasu losowego.

2.6 Krok 5: Predykcja

- Po zbudowaniu lasu losowego może on zostać użyty do **predykcji** dla nowych danych.
- Próbka przechodzi przez kolejne węzły drzewa, podejmując decyzje w każdym z nich na podstawie wartości swoich cech, aż dotrze do liścia, który przypisuje jej odpowiednią klasę.
- Proces ten jest powtarzany dla wszystkich drzew w lesie.

2.7 Krok 6: Agregacja wyników - głosowanie większościowe

- Wyniki poszczególnych drzew są **agregowane**, aby uzyskać końcową predykcję.
- Wynik klasyfikacji jest określany poprzez **głosowanie większościowe** (wybierana jest klasa, która została przypisana próbce przez najwięcej drzew).

2.8 Przykładowe obliczenia

Na podstawie powyższego algorytmu przeprowadzono przykładowe (skrócone) obliczenia. Celem modelu jest przewidzenie czy użytkownik kliknie w reklamę.

Wiek	**Płeć**	**Czas spędzony na stronie (min)**	**Urządzenie**	**Kliknięto w reklamę**
21	m	13	komputer	tak
35	k	8	smartfon	nie
18	m	20	komputer	tak
26	m	14	komputer	tak
61	m	8	komputer	nie
14	k	10	tablet	nie
29	k	19	smartfon	tak

Tabela 1: Tabela z przykładowymi danymi

2.8.1 Inicjalizacja lasu

- Określono liczbę drzew w lesie losowym na $N = 100$.
- Minimalna liczba próbek w węźle została ustalona na `min_samples_split=1`.
- Maksymalna głębokość drzewa została ustalona na `max_depth = 10`.

2.8.2 Próbkowanie bootstrapowe

- Został wybrany podzbiór danych:

Wiek	Płeć	Czas spędzony na stronie (min)	Urządzenie	Kliknięto w reklamę
21	m	13	komputer	tak
21	m	13	komputer	tak
26	m	14	komputer	tak
14	k	10	tablet	nie
14	k	10	tablet	nie
29	k	19	smartfon	tak

Tabela 2: Dane dla pierwszego drzewa spróbkowane bootstrapowo

2.8.3 Budowanie pierwszego drzewa

- Zbiór danych przekazany do drzewa jest w odpowiedniej formie.
- Inicjalizowany jest korzeń drzewa zawierający cały zbiór danych.
- Nie spełniono żadnego z warunków stopu.
- Unikalne wartości cech ciągłych zostały posortowane rosnąco: `Wiek = [14, 21, 26, 29]`, `Czas_na_stronie=[10, 13, 14, 19]`.
- Wyłonione punkty podziału to `Wiek = [17,5; 23,5; 27,5]`, `Czas_na_stronie=[11,5; 13,5; 16,5]`.
- Możliwe podziały klas kategorycznych to `Urządzenie = [{komputer}, {tablet, smartfon}; {tablet}, {komputer, smartfon}, {smartfon}, {komputer, tablet}]`, `Płeć = [{m}, {k}]`.
- Spośród wszystkich punktów podziału wylosowana dwa: `Czas_na_stronie = 16,5`, `Płeć = [m, k]`.
- Dla wszystkich podziałów obliczono wskaźnik nieczystości Gini

Wiek	Płeć	Czas spędzony na stronie (min)	Urządzenie	Kliknięto w reklamę
26	m	14	komputer	tak
21	m	13	komputer	tak
21	m	13	komputer	tak
14	k	10	tablet	nie
14	k	10	tablet	nie

Tabela 3: Czas spędzony na stronie (min) $< 16,5$

$$G = 1 - p_{\text{tak}}^2 - p_{\text{nie}}^2$$

$$p_{\text{tak}} = \frac{3}{5}, \quad p_{\text{nie}} = \frac{2}{5}$$

$$G = 1 - (\frac{3}{5})^2 - (\frac{2}{5})^2 = 1 - \frac{9}{25} - \frac{4}{25} = 0,48$$

Wiek	Płeć	Czas spędzony na stronie (min)	Urządzenie	Kliknięto w reklamę
29	k	19	smartfon	tak

Tabela 4: Czas spędzony na stronie (min) $\geq 16,5$

$$G = 1 - p_{\text{tak}}^2 - p_{\text{nie}}^2$$

$$p_{\text{tak}} = 1, \quad p_{\text{nie}} = 0$$

$$G = 1 - 1^2 - 0^2 = 0$$

- Następnie obliczono średnią ważoną wartości Gini dla podziału:
  Czas spędzony na stronie (min) < 15.

$$N_{lewy} = 5, \quad N_{prawy} = 1, \quad N_{cakowity} = 6, \quad Gini_{lewy} = 0,5, \quad Gini_{prawy} = 0$$

$$Gini_{\text{Podział}} = \left(\frac{5}{6} \times 0,48\right) + \left(\frac{1}{6} \times 0\right) = 0,4$$

Wiek	Płeć	Czas spędzony na stronie (min)	Urządzenie	Kliknięto w reklamę
21	m	13	komputer	tak
21	m	13	komputer	tak
26	m	14	komputer	tak

Tabela 5: `Płeć = m`

$$G = 1 - p_{\text{tak}}^2 - p_{\text{nie}}^2$$

$$p_{\text{tak}} = 1, \quad p_{\text{nie}} = 0$$

$$G = 1 - 1^2 - 0^2 = 0$$

Wiek	Płeć	Czas spędzony na stronie (min)	Urządzenie	Kliknięto w reklamę
14	k	10	tablet	nie
14	k	10	tablet	nie
29	k	19	smartfon	tak

Tabela 6: `Płeć = k`

$$G = 1 - p_{\text{tak}}^2 - p_{\text{nie}}^2$$

$$p_{\text{tak}} = \frac{1}{3}, \quad p_{\text{nie}} = \frac{2}{3}$$

$$G = 1 - (\frac{1}{3})^2 - (\frac{2}{3})^2 = 1 - \frac{1}{9} - \frac{4}{9} \approx 0,444$$

- Oraz średnią ważoną wartości Gini dla podziału `Płeć = m`.

$$N_{lewy} = 3, \quad N_{prawy} = 3, \quad N_{cakowity} = 6, \quad Gini_{lewy} = 0, \quad Gini_{prawy} = 0,444$$

$$Gini_{\text{Podział}} = \left(\frac{3}{6} \times 0\right) + \left(\frac{3}{6} \times 0,444\right) = 0,222$$

- $$Gini_{\text{Płeć}} = 0,222 < Gini_{\text{Czas}} = 0,4$$

  więc na zasadzie turnieju jako punkt podziału zostaje wybrany test `Płeć = m`.

- Dane zostają podzielone na dwa węzły według wybranego testu.

Wiek	Płeć	Czas spędzony na stronie (min)	Urządzenie	Kliknięto w reklamę
21	m	13	komputer	tak
21	m	13	komputer	tak
26	m	14	komputer	tak

Tabela 7: Lewy węzeł

- Lewy węzeł spełnia warunek stopu: Wszystkie próbki w węźle należą do jednej klasy. Zostaje więc on liściem, który predykuje `Kliknięto w reklamę = tak`.

Wiek	Płeć	Czas spędzony na stronie (min)	Urządzenie	Kliknięto w reklamę
14	k	10	tablet	nie
14	k	10	tablet	nie
29	k	19	smartfon	tak

Tabela 8: Prawy węzeł

- Prawy węzeł nie spełnia warunku stopu, więc cały proces wyboru punktu podziału jest dla niego powtarzany.
- Gdy wszystkie wywołania węzłów natrafią na warunek stopu, drzewo jest ukończone.
- Proces próbkowania bootsrapowego oraz budowania drzew jest powtarzany dla kolejnych 99 drzew.
- Predykcja:

Wiek	Płeć	Czas spędzony na stronie (min)	Urządzenie
24	m	7	smartfon

Tabela 9: Próbka, dla której chcemy określić czy kliknie w reklamę

- Dla każdego ze 100 drzew, próbka przechodzi przez kolejne węzły drzewa podejmując decyzje na podstawie swoich cech, aż dotrze do liścia, który przypisze jej predykowaną klasę.

Liczba drzew "głosujących"na tak	Liczba drzew "głosujących"na nie
78	22

Tabela 10: Agregowane wyniki predykcji drzew

- Na podstawie zagregowanych wyników predykcji drzew, przez głosowanie większościowe, wybrana została klasa `Kliknięto w reklamę = tak`.

3 Plan eksperymentów

3.1 Technologie

Program zostanie zaimplementowany w języku Python.
Biblioteki:

- Obliczenia: NumPy
- Zarządzanie danymi: Pandas
- Wizualizacje: Matplotlib, Seaborn
- Bibliotek z algorytmem do porównania wyników: scikit-learn

3.2 Brakujące wartości

W przypadku zbiorów posiadających brakujące wartości przed przekazaniem ich do modelu zostaną zastosowane metody imputacji danych. Na podstawie analizy zbioru danych i w zależności od charakteru, liczby etc. brakujących wartości zostanie wybrana jedna z poniższych technik:

- zastąpienie średnią dla danej cechy
- ustawienie flagi (wartości spoza zakresu) oznaczającej brak danych
- zastąpienie medianą danej cechy
- użycie regresji liniowej by przewidzieć brakujące wartości

3.3 Cechy eksperymentów

W celu obiektywnego porównania wyników oraz by móc wyciągnąć lepsze wnioski, eksperymenty będą przeprowadzane na modelach:

- Lasu losowego z biblioteki scikit-learn.
- Własnej implementacji lasu losowego z selekcją turniejową.

W ten sposób będziemy w stanie ocenić jak dobry jest nasz model w zestawieniu z popularnym modelem bibliotecznym.

Wnioski zostaną wyciągnięte na podstawie danych zagregowanych z 25 uruchomień. Analiza obejmie średnią, odchylenie standardowe, a także najlepszy i najgorszy uzyskany wynik.

Początkowa optymalizacja hiperparametrów rozpocznie się od zastosowania metody Random Search, aby w szybki sposób zidentyfikować wartości bliskie optymalnym. Następnie, w celu dokładniejszego zbadania otoczenia najlepszych wyników, zostanie wykorzystana metoda Grid Search.

3.4 Miary jakości

Do porównywania klasyfikatorów i hiperparametrów zostaną użyte poniższe miary jakości:

- dokładność (accuracy)
- F1 score
- tabela pomyłek (confusion matrix)
- krzywa ROC
- czas trenowania

3.5 Cel eksperymentów

Celem eksperymentów jest sprawdzenie jakości modelu lasu losowego z selekcją turniejową, w porównaniu do klasycznego algorytmu. W szczególności zamierzamy sprawdzić, czy nasz zmodyfikowany model osiąga lepszą wydajność na zbiorach danych, na których standardowy algorytm nie radzi sobie najlepiej, tzn, na zbiorach niezbalansowanych oraz z dużą liczbą cech.

4 Opis zbiorów danych

4.1 Wstępny zbiór danych - Mushroom

Zbiór klasyfikacji binarnej dzielący grzyby na jadalne i trujące. Ma za zadanie posłużyć wstępnej weryfikacji poprawności implementacji modelu. Po osiągnięciu wyniku około 100%, dalsze eksperymenty będą przeprowadzane na bardziej złożonych zbiorach danych.

https://archive.ics.uci.edu/dataset/73/mushroom

- Rodzaj cech: dyskretne
- Liczba wszystkich przykładów: 8124
- Liczba wszystkich cech: 22
- Brakujące wartości: posiada

4.1.1 Klasy etykiet

`edible(jadalne) - e`
`poisonous(trujące) - p`
Liczba przykładów poszczególnych klas:

- e: 4208
- p: 3916

Wyniki metody referencyjnej:

- dokładność - 100%
- precyzja - 100%

W zbiorze danych nie został wyróżniony zbiór testowy, dlatego zastosowana zostanie walidacja krzyżowa.

4.2 Niezbalansowany zbiór danych - Rain in Australia

Zbiór Rain in Australia zawiera informacje meteorologiczne z różnych miejsc w Australii. Został wybrany, aby sprawdzić działanie algorytmu na niezbalansowanym zbiorze danych.
https://www.kaggle.com/datasets/jsphyg/weather-dataset-rattle-package/data

- Rodzaj cech: dyskretne i ciągłe
- Liczba wszystkich przykładów: 145460
- Liczba wszystkich cech: 22
- Brakujące wartości: posiada

4.2.1 Klasy etykiet

Celem klasyfikacji jest przewidzenie czy jutro będzie padać, dla każdej daty.
Liczba przykładów poszczególnych klas:

- `No: 110316`
- `Yes: 31877`

Wyniki metody referencyjnej:

- dokładność - 84.38%
- precyzja - 57,7%

W zbiorze danych nie został wyróżniony zbiór testowy, dlatego zastosowana zostanie walidacja krzyżowa.

4.3 Wiele klas - Yeast

Zbiór danych opisuje właściwości białek w komórkach. Został wybrany, by zbadać działanie algorytmu przy klasyfikacji wieloklasowej.
https://archive.ics.uci.edu/dataset/110/yeast

- Rodzaj cech: ciągłe
- Liczba wszystkich przykładów: 1484
- Liczba wszystkich cech: 8
- Brakujące wartości: nie posiada

4.3.1 Klasy etykiet

Celem klasyfikacji jest przewidzenie lokalizacji białka w komórce. `CYT (cytosolic or cytoskeletal): 463`
`NUC (nuclear): 429`
`MIT (mitochondrial): 244`
`ME3 (membrane protein, no N-terminal signal): 163`
`ME2 (membrane protein, uncleaved signal): 51`
`ME1 (membrane protein, cleaved signal): 44`
`EXC (extracellular): 37`
`VAC (vacuolar): 30`
`POX (peroxisomal): 20`
`ERL (endoplasmic reticulum lumen): 5`

Wyniki metody referencyjnej:

- dokładność - 62.8%
- precyzja - 57,7%

W zbiorze danych nie został wyróżniony zbiór testowy, dlatego zastosowana zostanie walidacja krzyżowa.

4.4 Duża liczba cech - Online News Popularity

Zbiór danych przedstawia statystyki oraz strukturę artykułów na stronach internetowych. Został wybrany, by zbadać działanie algorytmu na danych z dużą liczbą cech.
https://archive.ics.uci.edu/dataset/332/online+news+popularity

- Rodzaj cech: ciągłe, dyskretne
- Liczba wszystkich przykładów: 39797
- Liczba wszystkich cech: 58
- Brakujące wartości: nie posiada

4.4.1 Klasy etykiet

Celem klasyfikacji jest przewidzenie ilości udostępnień artykułu.
Zadanie zostało przekształcone w zadanie binarne względem progu 1400
`Niepopularne (< 1400): 18490`
`Popularne (>= 1400): 21154`

Wyniki metody referencyjnej:

- brak - działanie algorytmu zostanie sprawdzone ręcznie na niewielkim zbiorze danych

W zbiorze danych nie został wyróżniony zbiór testowy, dlatego zastosowana zostanie walidacja krzyżowa.