Dokumentacja ćwiczenia 5.

WSI

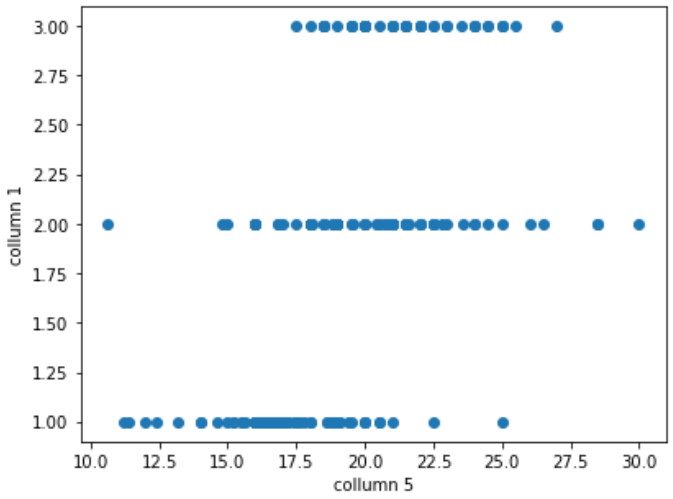
Łukasz Topolski

1. Wstęp

Ćwiczenie te polegało na implementacji lasu losowego oraz przeprowadzenia klasyfikacji metodą k-krotnej walidacji krzyżowej dla zbioru 3 odmian wina rosnącego w tym samym rejonie Włoch na podstawie ich parametrów wejściowych.

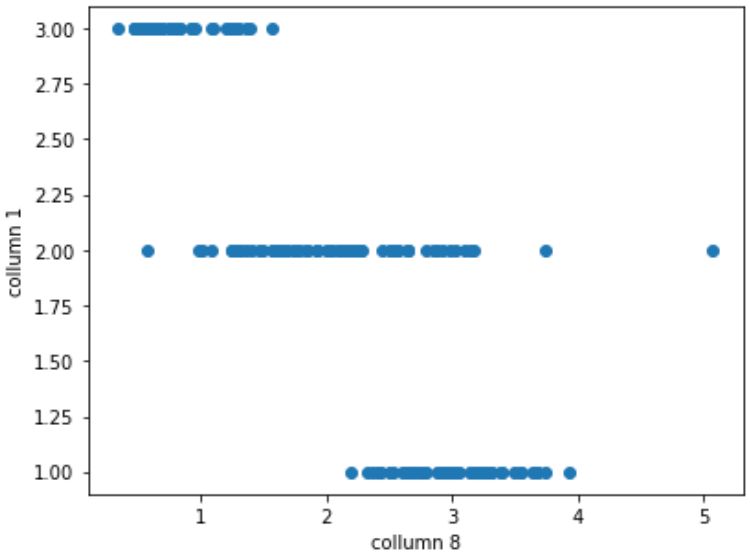
2. Analiza danych wejściowych

Dane, na podstawie których zostały przeprowadzone badania przedstawiają 178 próbek, w skład których wchodzą trzy grupy win (pierwsza – 59 próbek, druga – 71 próbek i trzecia – 48 próbek). Pojedynczy element badania posiada aż 13 parametrów chemicznych. Po narysowaniu wykresów zależności każdego z parametrów od kategorii i przeanalizowaniu ich dochodzimy do wniosku, że nie ma wyraźnego separatora dzielącego ziarna na kategorie pod względem któregokolwiek z parametrów, ale w przypadku parametru 5. i 8. możemy wyznaczyć pewne przedziały i na podstawie ich potwierdzić istnienie owych klas (rys 1. i 2.)



Rys1.: zależność 5. parametru od przypisanej klasy

Na powyższym rysunku (rys1.) większa część wartości z kategorii 1. jest mniejsza niż z pozostałych klas, natomiast z grupy 2. i 3. te wartości są zbliżone do siebie. Na poniższym wykresie (rys2.) możemy zauważyć, że wartości z grupy 3. są najmniejsze, z 2. są pośrednie, a z 1. – największe (ale zbliżone do tych z klasy 2.). Z obu wykresów wynika, że istnieją trzy klasy win, z których pierwsza ma najmniejsze wartości parametru 5. oraz największe wartości parametru 8., druga ma pośrednie, a nawet największe wartości parametru 5. oraz pośrednie wartości parametru 8., natomiast trzecia ma wartości parametru 5. zbliżone do kategorii 2., ale wartości parametru 8. są najmniejsze.



Rys2.: zależność 8. parametru od przypisanej klasy

3. Opis implementacji

Cały program został napisany w Pythonie (wersja 3.8.1) i dzieli się na trzy pliki:

- *decision_tree.py*
- *random_forest.py*
- *test_forest.py*

Pierwszy z nich jest implementacją drzewa decyzyjnego, drugi zawiera implementację lasu losowego, a trzeci z nich jest odpowiedzialny za wczytanie danych i na ich podstawie dokonanie k-krotnej walidacji. Bibliotekami wykorzystanymi w rozwiązaniu zadania są matplotlib.pyplot, random i math.

3.1 desision_tree.py

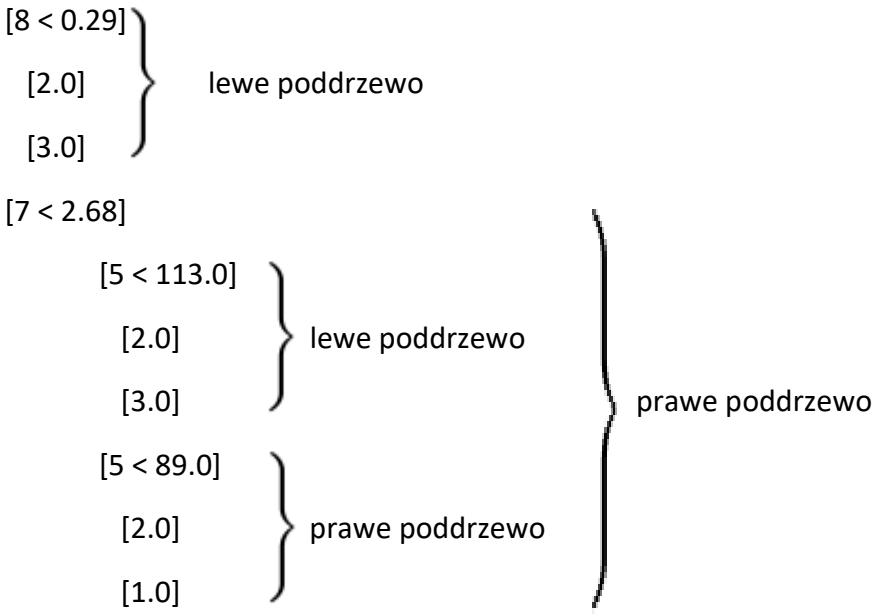
W tym pliku jest zawarta klasa reprezentująca drzewo decyzyjne. Posiada następujące parametry: zbiór danych uczących, maksymalną głębokość drzewa, minimalną ilość elementów, na podstawie których jest określany podział elementów oraz numery kolumn, na podstawie których jest tworzone drzewo. W tej klasie są zawarte metody wykorzystywane do tworzenia drzewa, metoda dokonująca klasyfikacji oraz metoda wypisująca drzewo.

Drzewo składa się z węzłów, które są reprezentowane jako słowniki zawierające informacje o indeksie kolumny dzielącej, wartość dzieląca oraz o węzłach lub liściach będących dziećmi tego węzła.

Tworzenie drzewa decyzyjnego odbywa się w sposób rekurencyjny. Najpierw jest wyznaczana (na postawie indeksu Ginniego) kolumna oraz wartość z tej kolumny, które najlepiej podzielą zbiór uczący. I w ten sposób powstaje korzeń drzewa decyzyjnego. Następnie dla każdego z podzbiorów kolejnego węzła (oczywiście zaczynając od korzenia) jest dokonywany podział i stworzenie kolejnego węzła na tej samej zasadzie, co korzeń. I tak do momentu, w którym: węzeł posiada tylko jedno dziecko (podzbiór zbioru uczącego), węzeł jest z ostatniego piętra drzewa, któryś z podzbiorów posiada zbyt małą ilość elementów albo wszystkie elementy z jednego z zbiorów są tej samej klasy.
Oprócz sytuacji, gdy w podzbiorze są elementy tej samej kategorii, to wartość liścia jest określana poprzez wybranie najczęściej występującej klasy w tym podzbiorze.

Wypisanie drzewa także odbywa się rekurencyjnie. Algorytm wypisujący dopóki nie dojdzie do liścia to wypisuje kolejne poddrzewa. Tak wygląda przykładowe drzewo decyzyjne:

[7 < 1.32] < ----- korzeń



Klasyfikacja jednego elementu również odbywa się rekurencyjnie. Badając odpowiednie parametry przechodzimy po drzewie (jeśli warunek jest spełniony, to wybieramy lewy węzeł, w przeciwnym wypadku – prawy) dopóki nie dotrzemy do liścia.

3.2 random_forest.py

Ten plik zawiera klasę reprezentującą las losowy. Składa się z listy drzew decyzyjnych, zbioru uczącego, parametrów potrzebnych do stworzenia drzewa oraz ilości tych drzew. Metodami tej klasy są train_forest (wywoływana w konstruktorze) oraz classify_item. Pierwsza z nich wybiera (ze zwracaniem) spośród danych uczących tyle elementów, ile ma być drzew, potem wybiera (bez zwracania) odpowiednią ilość indeksów kolumn (część całkowitą z pierwiastka z ilości wszystkich parametrów) i na podstawie wybranych danych tworzy odpowiednią ilość drzew. Natomiast druga metoda klasyfikuje element podany jako argument poprzez wybór najczęściej występującej klasy spośród wyborów wszystkich drzew lasu.

3.3 test_forest.py

W tym pliku są zawarte funkcje odpowiedzialne za wczytanie danych, podział danych na k części oraz na dane przeznaczone do testowania wszystkich lasów, dokonanie klasyfikacji z wykorzystaniem k-krotnej walidacji krzyżowej oraz obliczenie statystycznych parametrów.

W procesie podziału danych i klasyfikacji są wykonywane następujące czynności: najpierw wybierany jest spośród wszystkich danych tzw. zbiór walidacyjny, następnie pozostała część danych jest dzielona na k części, potem są tworzone lasy na podstawie k-1 części i jest testowany na podstawie niewykorzystanej części oraz na zbiorze walidacyjnym.

Podczas przypisywania próbek do grup w ramach testowania lasu następuje uzupełnianie macierzy błędów dla wszystkich kategorii. Wygląda ona następująco:

$$\begin{matrix} P_{11} & P_{12} & P_{13} \\ P_{21} & P_{22} & P_{23} \\ P_{31} & P_{32} & P_{33} \end{matrix}$$

gdzie P_{nm} to ilość elementów kategorii n przypisanych do kategorii m. Na podstawie tej macierzy określane są pozostałe parametry statystyczne przydatne do oceny modelu, czyli: precyzyjność, czułość i specyficzność dla każdej z kategorii oraz dokładność modelu.

4. Otrzymane rezultaty

Po uruchomieniu pliku *test_forest.py* przy następujących parametrach: ilość drzew – 50, maksymalna głębokość – 4, minimalna ilość elementu podzbioru – 10 oraz k - 3 otrzymałem następujące rezultaty:

Las 1.:

Macierz pomyłek:

[17, 0, 0]

[0, 16, 0]

[0, 0, 11]

Dokładność: 1.0

precyzja kategorii 1.: 1.0

czułość kategorii 1.: 1.0

precyzja kategorii 2.: 1.0

czułość kategorii 2.: 1.0

precyzja kategorii 3.: 1.0

czułość kategorii 3.: 1.0

Dokładność dla zbioru walidacyjnego: 0.93

Las 2.:

Macierz pomyłek:

[10, 0, 0]

[0, 20, 2]

[0, 0, 12]

Dokładność: 0.95

precyzja kategorii 1.: 1.0

czułość kategorii 1.: 1.0

precyzja kategorii 2.: 0.91

czułość kategorii 2.: 1.0

precyzja kategorii 3.: 1.0

czułość kategorii 3.: 0.86

Dokładność dla zbioru walidacyjnego: 0.98

Las 3.:

Macierz pomyłek:

[15, 0, 0]

[1, 16, 0]

[0, 0, 12]

Dokładność: 0.98

precyzja kategorii 1.: 1.0

czułość kategorii 1.: 0.94

precyzja kategorii 2.: 0.94

czułość kategorii 2.: 1.0

precyzja kategorii 3.: 1.0

czułość kategorii 3.: 1.0

Dokładność dla zbioru walidacyjnego: 0.98

średnia dokładność wszystkich modeli dla zbioru walidacyjnego: 0.96

średnia dokładność wszystkich modeli: 0.98

Przy zmniejszeniu ilości drzew do 25 rezultaty były bardzo zbliżone do siebie(średnia dokładność wszystkich modeli: 0.96, a dla zbioru walidacyjnego – 0.95)

Dopiero zmniejszenie ilości drzew do wartości mniejszej niż 25 (np. 15) wykazały gorsze rezultaty:

Las 1.:

Macierz pomyłek:

[12, 5, 0]

[0, 15, 0]

[0, 2, 10]

Dokładność: 0.84

precyzja kategorii 1.: 0.71

czułość kategorii 1.: 1.0

precyzja kategorii 2.: 1.0

czułość kategorii 2.: 0.68

precyzja kategorii 3.: 0.83

czułość kategorii 3.: 1.0

Dokładność dla zbioru walidacyjnego: 0.84

Las 2.:

Macierz pomyłek:

[13, 1, 0]

[2, 14, 2]

[0, 0, 12]

Dokładność: 0.89

precyzja kategorii 1.: 0.93

czułość kategorii 1.: 0.87

precyzja kategorii 2.: 0.78

czułość kategorii 2.: 0.93

precyzja kategorii 3.: 1.0

czułość kategorii 3.: 0.86

Dokładność dla zbioru walidacyjnego: 0.89

Las 3.:

Macierz pomyłek:

[8, 2, 0]

[1, 19, 0]

[0, 5, 9]

Dokładność: 0.82

precyzja kategorii 1.: 0.8

czułość kategorii 1.: 0.89

precyzja kategorii 2.: 0.95

czułość kategorii 2.: 0.73

precyzja kategorii 3.: 0.64

czułość kategorii 3.: 1.0

Dokładność dla zbioru walidacyjnego: 0.82

średnia dokładność wszystkich modeli: 0.85

średnia dokładność wszystkich modeli dla zbioru walidacyjnego: 0.85

W przypadku zmniejszenia głębokości drzew do 1 otrzymane rezultaty są gorsze:

Las 1.:

Macierz pomyłek:

[13, 2, 0]

[0, 15, 0]

[0, 0, 14]

Dokładność: 0.95

precyzja kategorii 1.: 0.87

czułość kategorii 1.: 1.0

precyzja kategorii 2.: 1.0

czułość kategorii 2.: 0.88

precyzja kategorii 3.: 1.0

czułość kategorii 3.: 1.0

Dokładność dla zbioru walidacyjnego: 0.93

Las 2.:

Macierz pomyłek:

[11, 0, 0]

[17, 3, 3]

[0, 0, 10]

Dokładność: 0.55

precyzja kategorii 1.: 1.0

czułość kategorii 1.: 0.39

precyzja kategorii 2.: 0.13

czułość kategorii 2.: 1.0

precyzja kategorii 3.: 1.0

czułość kategorii 3.: 0.77

Dokładność dla zbioru walidacyjnego: 0.57

Las 3.:

Macierz pomyłek:

[14, 3, 0]

[0, 12, 0]

[0, 2, 13]

Dokładność: 0.89

precyzja kategorii 1.: 0.82

czułość kategorii 1.: 1.0

precyzja kategorii 2.: 1.0

czułość kategorii 2.: 0.71

precyzja kategorii 3.: 0.87

czułość kategorii 3.: 1.0

Dokładność dla zbioru walidacyjnego: 0.89

średnia dokładność wszystkich modeli: 0.8

średnia dokładność wszystkich modeli dla zbioru walidacyjnego: 0.8

Zwiększenie głębokości drzew nie wpłynęło znacząco na otrzymane wyniki.

W przypadku zwiększenia k do 5 otrzymałem następujący rezultat:

Las 1.:

Dokładność: 0.96

Dokładność dla zbioru walidacyjnego: 0.95

Las 2.:

Dokładność: 1.0

Dokładność dla zbioru walidacyjnego: 0.93

Las 3.:

Dokładność: 0.92

Dokładność dla zbioru walidacyjnego: 0.95

Las 4.:

Dokładność: 1.0

Dokładność dla zbioru walidacyjnego: 0.93

Las 5.:

Dokładność: 0.92

Dokładność dla zbioru walidacyjnego: 0.95

średnia dokładność wszystkich modeli: 0.96

średnia dokładność wszystkich modeli dla zbioru walidacyjnego: 0.94

Każde kolejne zwiększenie k nie wpływało znacząco na rezultaty (np. dla k = 8 średnia dokładność wszystkich modeli: 0.98, a dla zbioru walidacyjnego - 0.94)

5. Wnioski

- Przy dobraniu odpowiednich parametrów model osiąga przyzwoitą dokładność, precyzję dla poszczególnych klas, itp.
- Przy zbyt małej ilości drzew możemy zaobserwować zjawisko niedouczenia modelu
- Zbyt mała głębokość drzew oraz zbyt duża minimalna ilość elementu podzbioru również powoduje niedouczenie modelu
- Zwiększanie parametru k nie wpływa znacząco na dokładność modelu, chociaż możemy zaobserwować skłonności do przeuczenia się modelu