Politechnika Warszawska

WYDZIAŁ ELEKTRONIKI
I TECHNIK INFORMACYJNYCH

Uczenie Maszynowe - Projekt

Połączenie Lasu Losowego ID3 z Maszyną Wektorów Nośnych (SVM) w zadaniu klasyfikacji binarnej

Projekt wstępny

Bartosz Cywiński, Łukasz Staniszewski
304025, 304098

prowadzący
dr Rafał Biedrzycki

WARSZAWA 11 kwietnia 2022

Spis treści

1. Temat projektu

Celem projektu jest implementacja połączenia lasu losowego z SVM w zadaniu klasyfikacji binarnej. Wykonana zostanie własna implementacja obu algorytmów na podstawie źródeł [1], [2] oraz [3]. Najistotniejszą modyfikacją w stosunku do klasycznego algorytmu lasu losowego będzie to, że klasyfikatorem w lesie będzie zarówno drzewo decyzyjne ID3 jak i SVM, zakładając że będzie taka sama liczność obu klasyfikatorów. Proces inferencji w lesie losowym będzie niezmienny, a więc ostateczną predykcją dla danego przykładu będzie najliczniej przewidywana klasa przez wszystkie klasyfikatory.

2. Drzewo decyzyjne

Pierwszym z klasyfikatorów użytych w zmodyfikowanym lesie losowym będzie drzewo decyzyjne. Algorytmem uczenia drzewa będzie algorytm ID3. Drzewo decyzyjne zaimplementowane będzie w sposób uniwersalny na tyle, żeby poprawnie działało zarówno dla klasyfikacji binarnej jak i wieloklasowej.

2.1. Opis algorytmu

Algorithm 1 ID3

Input: S: zbiór par uczących, Y: zbiór klas, D: zbiór atrybutów wejściowych, d: obecna głębokość drzewa

if $S = \emptyset$ **then return**
end if

if wszystkie przykłady należą do klasy y **then**
return Liść z klasą y
end if

$(j, t) = \arg\min_{j,t} H(S)$
`Root` = węzeł zbudowany na zbiorze S
if !stop **then**
    `Root.left = ID3(`S_-, Y, D, $d+1$`)`
    `Root.right = ID3(`S_+, Y, D, $d+1$`)`
end if
return `Root`

Drzewo decyzyjne ma strukturę drzewa binarnego. W każdym węźle, podczas konstruowania drzewa, wyszukiwany jest atrybut w zbiorze wszystkich atrybutów zbioru danych oraz jego konkretna wartość, która podzieli zbiór danych na dwa podzbiory w taki sposób, aby suma entropii podzbiorów była jak najmniejsza. Na podstawie powstałych w wyniku operacji podziału podzbiorów rekurencyjnie tworzone są kolejne węzły drzewa decyzyjnego.

Drzewo decyzyjne ma następujące atrybuty:

1. maksymalna głębokość drzewa
2. minimalna różnica między entropią po podziale, a entropią przed podziałem
3. minimalna liczba przykładów w węźle drzewa

Oznaczając zbiór przykładów z etykietami jako S, na początku procesu uczenia drzewa decyzyjnego, drzewo ma tylko jeden węzeł (korzeń), który zawiera wszystkie przykłady ze zbioru S. Kolejno wskutek wywołania algorytmu ID3 rozpoczyna się właściwy proces uczenia drzewa, opisany przedstawionym powyżej pseudokodem.

Linie 1-2: Jeśli w zbiorze S, który był argumentem algorytmu ID3, nie ma już żadnej pary uczącej, proces dalszego uczenia drzewa należy zatrzymać.

Linie 4-6: Jeśli w zbiorze S znajdują się przykłady należące tylko do jednej klasy, to znaczy że zbiór jest już idealnie podzielony i nie należy kontynuować procesu uczenia, więc zwracany jest liść drzewa, którego predykcja jest jedyną klasą w zbiorze S.

Linia 8: Oznaczając zbiór wszystkich atrybutów w zbiorze danych jako D, dla każdego indeksu atrybutu $j = 0, ..., D-1$ oraz dla każdej wartości atrybutu występującej w zbiorze danych t wykonywane są następujące kroki:

1. Zbiór wszystkich par uczących S dzielony jest na dwa podzbiory: $S_- = \{(x, y) | (x, y) \in S, x^{(j)} < t\}, S_+ = \{(x, y) | (x, y) \in S, x^{(j)} \geq t\}$.
2. Oceniana jest jakość podziału. W tym celu liczona jest entropia podziału jako entropia ważona dwóch zbiorów: $H(S_-, S_+) = \frac{|S_-|}{|S|} H(S_-) + \frac{|S_+|}{|S|} H(S_+)$, przy czym wartość entropii H dla zbioru S definiuje się jako: $H(S) = \sum_{c \in C} -p(c) \log_2 p(c)$, gdzie C to zbiór wszystkich klas obecnych w zbiorze S, a $p(c)$ to stosunek liczby przykładów klasy c w zbiorze S do liczby wszystkich przykładów w zbiorze S.

Wykonując powyższe kroki znajdywany jest taki atrybut j oraz taka jego wartość t dla których entropia jest najniższa.

Linia 10: Aby dokonać podziału węzła spełnione muszą być następujące warunki:

1. Nie może być przekroczona maksymalna głębokość drzewa.
2. Różnica między entropią $H(S)$, a entropią ważoną $H(S_-, S_+)$ musi być większa od minimalnej dopuszczalnej różnicy między wartościami entropii.
3. Liczność, zarówno zbioru S_-, jak i zbioru S_+ musi być większa od minimalnej dopuszczalnej liczby przykładów w węźle drzewa.

W przypadku niespełnienia jakiegokolwiek z powyższych warunków dany węzeł drzewa nie zostanie dalej podzielony.

Linie 11-12: Do obecnego korzenia (węzła Root), przypisywane są węzły dzieci. Węzły te tworzone są przez rekursywne wywołanie algorytmu ID3, kolejno dla podzbiorów S_-, jak i S_+, jednocześnie zwiększając obecną głębokość drzewa o 1.

2.1.1. Przykładowe obliczenia przy podziale zbioru danych:

Zakładając, że:

$$S = \{([0,2,5],0),([0,4,6],1),([0,-1,7],0)\}$$

Analizując ten prosty zbiór danych można zauważyć, że atrybut o indeksie $j = 1$ oraz jego wartość $t = 4$ podzieli zbiór S tworząc podzbiory S_- oraz S_+ w sposób następujący:

$$S_- = \{(x,y)|(x,y) \in S, x^{(1)} < 4\} = \{([0,2,5],0),([0,-1,7],0)\},$$
$$S_+ = \{(x,y)|(x,y) \in S, x^{(1)} \geq 4\} = \{([0,4,6],1)\}$$

Zatem licząc entropie poszczególnych zbiorów:

$$H(S_-) = -\frac{2}{2}\log_2\frac{2}{2} + (-\frac{0}{2}\log_2\frac{0}{2}) = 0 + 0 = 0$$
$$H(S_+) = -\frac{0}{1}\log_2\frac{0}{2} + (-\frac{1}{1}\log_2\frac{1}{1}) = 0 + 0 = 0$$
$$H(S_-,S_+) = \frac{2}{3}H(S_-) + \frac{1}{3}H(S_+) = \frac{2}{3}\cdot 0 + \frac{1}{3}\cdot 0 = 0$$

Widać na tym przykładzie, że entropia podziału zbioru S jest równa jej minimalnej możliwej wartości, bo podzbiory S_- i S_+ idealnie podzieliły zbiór S pod względem klas. Dla odróżnienia, gdyby został wybrany ten sam atrybut o indeksie $j = 1$, ale o wartości $t = 2$, podział wyglądałby następująco:

$$S_- = \{(x,y)|(x,y) \in S, x^{(1)} < 2\} = \{([0,-1,7],0)\},$$
$$S_+ = \{(x,y)|(x,y) \in S, x^{(1)} \geq 2\} = \{([0,2,5],0),([0,4,6],1)\}$$

Ponownie licząc entropie poszczególnych zbiorów:

$$H(S_-) = -\frac{0}{1}\log_2\frac{0}{1} + (-\frac{1}{1}\log_2\frac{1}{1}) = 0 + 0 = 0$$
$$H(S_+) = -\frac{1}{2}\log_2\frac{1}{2} + (-\frac{1}{2}\log_2\frac{1}{2}) = \frac{1}{2} + \frac{1}{2} = 1$$
$$H(S_-,S_+) = \frac{1}{3}H(S_-) + \frac{2}{3}H(S_+) = \frac{1}{3}\cdot 0 + \frac{2}{3}\cdot 1 = \frac{2}{3}$$

Co pokazuje, że gdy zbiory byłyby tak podzielone, entropia miałaby większą wartość, dlatego też podział nie zostałby wybrany jako najlepszy możliwy.

2.2. Predykcje algorytmu

Po całkowitym wykonaniu się algorytmu uczącego drzewo ID3, drzewo jest w pełni zbudowane i można na nim wykonywać predykcje. Dla pojedynczej próbki danych algorytm przechodzi od korzenia do liścia, w każdym węźle wybierając dziecko do którego powinien następnie przejść na podstawie wartości atrybutu podziału danego węzła dla próbki danych. Mianowicie, przyjmując za j indeks atrybutu podziału danego węzła, a za t przyjmując wartość tego atrybutu, jeśli dla próbki danych x: $x^{(j)} < t$, to algorytm przechodzi do lewego dziecka. W przeciwnym przypadku, algorytm przejdzie do prawego dziecka. Po dotarciu do liścia drzewa decyzyjnego, jako predykcja zwracana jest klasa większościowa danego liścia.

3. Algorytm SVM

Drugim z klasyfikatorów użytych w implementacji będzie Maszyna Wektorów Nośnych (SVM) dopuszczająca pomyłki. SVM implementowany jest dla przypadku binarnego ze zbiorem klas $Y = \{-1, 1\}$, a także, dla ułatwienia implementacji, zakładana jest wersja algorytmu bez przekształcenia jądrowego (bazowe jądro liniowe).

3.1. Opis algorytmu

Zadanie polega na znalezieniu funkcji rozgraniczającej $f(x) = x \cdot w - b$, która tworzy hiperpłaszczyznę zapewniającą klasyfikację. Otrzymana funkcja powinna zapewniać jak najmniejszą liczbę pomyłek przy klasyfikowaniu elementów zbioru wejściowego do odpowiedniej klasy.

Klasyfikacja odbywa się poprzez zwrócenie dla danego zestawu cech x klasy $y(x) = -1$ lub $y(x) = 1$, do której przynależność wynika z następującej zależności:

$$y(x) = \begin{cases} -1 & , f(x) \leq 0 \\ 1 & , f(x) > 0 \end{cases} \tag{1}$$

Na końcu procesu inferencji, otrzymane predykcje są mapowane z powrotem do odpowiednich klas ze zbioru X, przykładowo: $-1 \rightarrow 0$.
Ze względu na trenowanie dopuszczające pomyłki, aby otrzymać wyżej wymienioną funkcję f, należy znaleźć parametry (w, b), minimalizujące funkcję straty J:

$$(w, b) = \arg\min_{w,b} J(w, b) \tag{2}$$

$$J(w, b) = \Sigma_i \cdot \xi_i + \lambda \cdot ||w||^2 \tag{3}$$

Przy czym, istotne są odpowiednie ograniczenia (gdzie ξ_i oznacza stratę dla i-tego przykładu trenującego, jeśli klasyfikacja jest błędna; a λ decyduje o istotności szerokości regionu separującego):

$$\lambda > 0 \tag{4}$$

$$\forall_i \, [\xi_i \geq 0 \, \wedge \, y_i \cdot (x_i \cdot w - b) \geq 1 - \xi_i] \tag{5}$$

Wymienione wyżej warunki (4) oraz (5) w połączeniu z postacią funkcji straty (3) implikują ostateczną postać funkcji J:

$$J(w,b) = \Sigma_i \max(1 - f(x_i) \cdot y_i, 0) + \lambda \cdot ||w||^2 \tag{6}$$

Powyższy opis algorytmu można podsumować w postaci algorytmu uczenia (Algorithm 2) oraz algorytmu predykcji (Algorithm 3). W algorytmie predykcji zastosowane zostało wyrażenie logiczne, zwracające 1 gdy jest prawdziwe, a 0 gdy fałszywe, co pozwala mu zwracać numery klas zamiast liczb rzeczywistych. Na końcu, zwracane klasy mapowane są do odpowiednich wartości, przykładowo: $-1 \rightarrow 0$. W algorytmie uczenia, początkowo poprawiane są wejściowe etykiety tak, aby były ze zbioru $\{-1, 1\}$, następnie inicjalizowane są parametry modelu i przekazywane do metody optymalizacyjnej, której opis znajduje się w następnym podrozdziale. Na końcu zwracane są wytrenowane parametry modelu.

Algorithm 2 Uczenie SVM

Input: X: zestaw przykładów dla zbioru trenującego, Y: zestaw etykiet dla zbioru trenującego, λ: parametr funkcji straty, V: wektor parametrów dla optymalizatora

1: $Y' =$ `correct_targets`(Y)
2: `initialize`: $w_0 = [0\,0\,...\,0]^T$, $b_0 = 0$
3: $(w,b) =$ `gradient_descent`$(w_0, b_0, X, Y', V, \lambda)$
4: **return** (w,b)

Algorithm 3 Predykcja SVM

Input: X: zestaw przykładów dla zbioru ewaluacyjnego, (w,b): parametry modelu, λ: parametr funkcji straty

1: **return** `repair_targets`$(2 \cdot ((X \cdot w - b) > 0) - 1)$

3.2. Optymalizacja

Istotnym elementem uczenia modelu SVM, jest wyznaczenie jego parametrów poprzez minimalizację funkcji straty J. Zaimplementowany został algorytm Stochastycznego Spadku Gradientowego (Stochastic Gradient Descent / SGD), do którego działania wymagane jest obliczenie gradientu funkcji straty $J(w,b)$ po parametrach modelu w oraz b.

$$\nabla J = \begin{bmatrix} \frac{\partial J}{\partial w_1} & \cdots & \frac{\partial J}{\partial w_n} & \frac{\partial J}{\partial b} \end{bmatrix}^T \tag{7}$$

Gradient ten jest wektorem pochodnych cząstkowych wyliczanych po kolejnych parametrach modelu (gdzie $x_{k[i]}$ oznacza i-ty atrybut k-tego przykładu).

$$\frac{\partial J}{\partial w_i} = \lambda \cdot 2 \cdot w_i + \Sigma_k (1 \cdot \begin{cases} 0 & , 1 - f(x_k \cdot) y_k \leq 0 \\ -y_k \cdot x_{k[i]} & , 1 - f(x_k) \cdot y_k > 0 \end{cases}) \tag{8}$$

$$\frac{\partial J}{\partial b} = \Sigma_k (1 \cdot \begin{cases} 0 & , 1 - f(x_k) \cdot y_k \leq 0 \\ y_k & , 1 - f(x_k) \cdot y_k > 0 \end{cases}) \tag{9}$$

Zaimplementowany optymalizator SGD można przedstawić w formie algorytmu (Algorithm 4). Metoda zakłada jednokrotne przetworzenie całego zbioru trenującego w ramach jednego kroku i rozpoczyna się od ustalenia parametrów optymalizatora, gdzie max_steps oznacza maksymalną liczbę kroków optymalizacji, min_steps oznacza najmniejszą możliwą normę z parametrów modelu, natomiast β to tzw. learning rate.

Algorithm 4 SGD

Input: (w_0, b_0): zainicjowane parametry modelu, X: zbiór przykładów (wejść), Y: zbiór etykiet (wyjść), V: parametry optymalizatora, λ: współczynnik λ modelu

1: $(max\_steps, \beta, min\_eps) = V$
2: $(w, b) = (w_0, b_0)$
3: $step = 0$
4: **while** $step \leq max\_steps$ **do**
5: $\quad \nabla w = \lambda \cdot 2 \cdot w_i + \Sigma_k (1 \cdot \begin{cases} 0 & , 1 - f(x_k) \cdot y_k \leq 0 \\ -y_k \cdot x_{k[i]} & , 1 - f(x_k) \cdot y_k > 0 \end{cases})$ `for each` $i = 1...M$
6: $\quad \nabla b = \Sigma_k (1 \cdot \begin{cases} 0 & , 1 - f(x_k) \cdot y_k \leq 0 \\ y_k & , 1 - f(x_k) \cdot y_k > 0 \end{cases})$
7: $\quad$ **if** $|w| < min\_eps$ **then return** (w, b)
8: $\quad$ **end if**
9: $\quad w = w - \beta \cdot \nabla w$
10: $\quad b = w - \beta \cdot \nabla b$
11: $\quad step = step + 1$
12: **end while**
13: **return** (w, b)

3.3. Przykładowe obliczenia

Zakładając, że z każdym przykładem związane są 3 atrybuty i istnieje następujący zbiór treningowy T składający się z 3 przykładów oraz zbiór ewaluacyjny E składający się z 2 przykładów:

$$X_T = \begin{vmatrix} 1 & 2 & 3 \\ 6 & 5 & 4 \\ 7 & 8 & 7 \end{vmatrix}, Y_T = \begin{vmatrix} 1 \\ 0 \\ 1 \end{vmatrix}, X_E = \begin{vmatrix} -3 & 4 & 1 \\ 4 & 2 & 12 \end{vmatrix}$$

Na początku konieczne jest przerobienie etykiet klas dla przykładów trenujących oraz inicjalizacja modelu:

$$Y'_T = \begin{vmatrix} 1 \\ \mathbf{-1} \\ 1 \end{vmatrix}, w = \begin{vmatrix} 1 \\ -1 \\ 1 \end{vmatrix}, b = 3$$

Przy założeniu, że $max\_steps = 1$, $\lambda = 0.5$ oraz $\beta = 0.2$, wykonywany jest jeden krok SGD:

$$1 - f(X_T) \cdot Y'_T = \begin{vmatrix} 2 \\ 3 \\ -2 \end{vmatrix} \implies \nabla w = \begin{vmatrix} 0.5 \cdot 2 \cdot 1 + \mathtt{sum}(\begin{vmatrix} -1 & 6 & 0 \end{vmatrix}^T) \\ 0.5 \cdot 2 \cdot -1 + \mathtt{sum}(\begin{vmatrix} -2 & 5 & 0 \end{vmatrix}^T) \\ 0.5 \cdot 2 \cdot 1 + \mathtt{sum}(\begin{vmatrix} -3 & 4 & 0 \end{vmatrix}^T) \end{vmatrix} = \begin{vmatrix} 6 \\ 2 \\ 2 \end{vmatrix}$$

$$1 - f(X_T) \cdot Y'_T = \begin{vmatrix} 2 & 3 & -2 \end{vmatrix}^T \implies \nabla b = \mathtt{sum}(\begin{vmatrix} 1 & -1 & 0 \end{vmatrix}^T) = 0$$

Na końcu uczenia ustalane są ostateczne parametry modelu:

$$w = \begin{vmatrix} 1 \\ -1 \\ 1 \end{vmatrix} - 0.2 \cdot \begin{vmatrix} 6 \\ 2 \\ 2 \end{vmatrix} = \begin{vmatrix} -0.2 \\ -1.4 \\ 0.6 \end{vmatrix}, b = 3 - 0.2 \cdot 0 = 3$$

Po ustaleniu parametrów modelu, można przejść do predykcji:

$$Y_E = 2 \cdot ((\begin{vmatrix} -3 & 4 & 1 \\ 4 & 2 & 12 \end{vmatrix} \cdot \begin{vmatrix} -0.2 \\ -1.4 \\ 0.6 \end{vmatrix} - 3) > 0) - 1 = 2 \cdot (\begin{vmatrix} -7.4 \\ 0.6 \end{vmatrix} > 0) - 1 = \begin{vmatrix} -1 \\ 1 \end{vmatrix}$$

4. Las losowy

Las losowy będzie wykorzystywał oba algorytmy opisane powyżej. Za klasycznym algorytmem lasu losowego stoi idea baggingu. W algorytmie konstruuje się wiele nieskorelowanych drzew losowych, a następnie się je uśrednia. W przypadku modyfikacji algorytmu, wykonywanego w ramach projektu, zamiast samych drzew decyzyjnych, występował będzie zbiór klasyfikatorów - drzew decyzyjnych oraz SVM. Każdy klasyfikator wytrenowany będzie na zbiorze próbek danych losowanych z oryginalnego zbioru ze zwracaniem. Ponadto w procesie uczenia drzewa losowego przed każdym podziałem węzła losowany będzie bez zwracania podzbiór indeksów atrybutów uwzględnianych w procesie podziału. Natomiast w procesie uczenia SVM, podzbiór indeksów atrybutów losowany będzie raz, przed rozpoczęciem procedury treningowej. Zasadniczą ideą baggingu jest uśrednienie wielu modeli z dość małym obciążeniem, w skutek czego zmniejszana jest ich wariancja.

4.1. Opis algorytmu

Algorithm 5 Las losowy

Input: S: zbiór par uczących, Y: zbiór klas, D: zbiór atrybutów wejściowych, B: liczba klasyfikatorów

1: **for** b = 1 to B: **do**
2: Wylosuj ze zwracaniem $|S|$ próbek danych ze zbioru S.
3: **if** b mod 2 == 0 **then**
4: Wytrenuj drzewo decyzyjne algorytmem ID3 na wylosowanym zbiorze, przed każdym podziałem węzła losując bez zwracania podzbiór $m \leq |D|$ atrybutów uwzględnianych przy jego podziale.
5: **else**
6: Wylosuj bez zwracania podzbiór podzbiór $m \leq |D|$ atrybutów. Wytrenuj SVM na wylosowanym zbiorze, uwzględniając tylko wylosowane atrybuty.
7: **end if**
8: Dodaj wytrenowany klasyfikator do zbioru klasyfikatorów lasu losowego.

Linia 1: Liczba wszystkich klasyfikatorów w lesie losowym jest hiperparametrem algorytmu lasu losowego, z zastrzeżeniem że musi być ona podzielna przez 2, ponieważ zakładamy że las składa się z drzew decyzyjnych oraz SVM na przemian.

Linia 2: Trenując każdy model w lesie losowym na różnych podzbiorach oryginalnego zbioru danych (z założeniem, że próbki danych w podzbiorze mogą się powtarzać) redukowana jest wariancja modelu, w skutek czego zmniejszane jest przeuczenie lasu.

Linie 4-6: Przed każdym podziałem węzła w drzewie decyzyjnym losowany jest bez zwracania podzbiór atrybutów rozważanych podczas podziału. Ta modyfikacja klasycznego algorytmu uczenia drzewa zmniejsza korelacje między poszczególnymi drzewami w lesie. Gdyby ta modyfikacja nie była zastosowana, atrybutami podziału węzłów w większości drzew w lesie byłyby takie, które najskuteczniej dzielą zbiór danych. Wskutek tego las składałby się ze skorelowanych drzew, co nie zwiększyłoby skuteczności modelu, ponieważ słabe klasyfikatory w lesie byłyby zgodne co do złych predykcji, a to by skutkowało błędnymi ostatecznymi predykcjami lasu. Podobnie przed uczeniem SVM losowany jest bez zwracania podzbiór atrybutów uwzględnianych w procesie uczenia - model uczy się tylko na tych atrybutach.

4.2. Predykcje algorytmu

Po wytrenowaniu wszystkich klasyfikatorów, predykcją lasu losowego jest klasa większościowa w zbiorze predykcji każdego z klasyfikatorów w drzewie.

5. Plan eksperymentów

5.1. Analiza skuteczności hybrydy lasu losowego z SVM

Wykonane zostaną symulacje mające na celu porównanie zaimplementowanej hybrydy Lasu Losowego i SVM z klasycznym Lasem Losowym w zadaniu klasyfikacji binarnej pod względem metryk dokładności (accuracy), odzysku (recall) oraz precyzji (precision), a także macierzy pomyłek (confusion matrix). Wnioski i analizy przeprowadzone zostaną na zagregowanych wynikach z 25 uruchomień, na których wyliczona zostanie średnia arytmetyczna, mediana, odchylenie standardowe czy wartości minimalne i maksymalne. Badania wykonane zostaną na 3 zbiorach danych opisanych niżej.

5.2. Analiza wpływu hiperparametrów na skuteczność

Odrębnym eksperymentem będzie analiza wpływu hiperparametrów wszystkich modeli użytych w projekcie na skuteczność architektury. Przeszukanie przestrzeni hiperparametrów będzie wykonane metodą grid search. Analizowanymi hiperparametrami będzie dla SVM: λ; dla drzew decyzyjnych: maksymalna głębokość, minimalna różnica między entropią po podziale a entropią przed podziałem w węźle, minimalna liczba przykładów w węźle, maksymalna liczba atrybutów uwzględnianych przy podziale węzła; a także dla lasu losowego: liczba klasyfikatorów.

5.3. Zbiory danych

Eksperymenty zostaną wykonane na 3 zbiorach danych opisanych w poniższej tabeli. Dla każdego ze zbiorów zostanie wykonana walidacja krzyżowa, gdzie ostateczny wynik na zbiorze testowym będzie estymowany na zasadzie makro-uśredniania, tzn. jako średnia wartość wskaźników jakości uzyskanych na wszystkich podziałach.

Nazwa zbioru	Opis	Liczba przykładów pozytywnych	Liczba przykładów negatywnych
Breast Cancer [4]	Zbiór danych składający się z 569 przykładów posiadających 30 atrybutów ciągłych.	212	357
Ionosphere [5]	Zbiór danych składający się z 351 przykładów posiadających 34 atrybuty ciągłe.	225	126
QSAR biodegradation [6]	Zbiór danych składający się z 1055 przykładów posiadających 41 atrybuty ciągłe.	356	699

Tabela 5.1. Opis zbiorów danych, które zostaną wykorzystane do eksperymentów numerycznych.

Bibliografia

[1] Paweł Zawistowski, "Wykłady z przedmiotu Wprowadzenie do Sztucznej Inteligencji (WSI)", *Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska*, 2020.

[2] Paweł Cichosz, "Wykłady z przedmiotu Uczenie Maszynowe (UMA)", *Wydział Elektroniki i Technik Informacyjnych, Politechnika Warszawska*, 2022.

[3] R. T. Trevor Hastie i J. Friedman, *The Elements of Statistical Learning*. Stanford, California: Springer, 2008.

[4] University of Wisconsin, "Breast Cancer Wisconsin (Diagnostic) Data Set", 1995.

[5] Space Physics Group, Johns Hopkins University, "Ionosphere Data Set", 1989.

[6] Milano Chemometrics and QSAR Research Group, Università degli Studi di Milano Bicocca), "QSAR biodegradation Data Set", 2013.