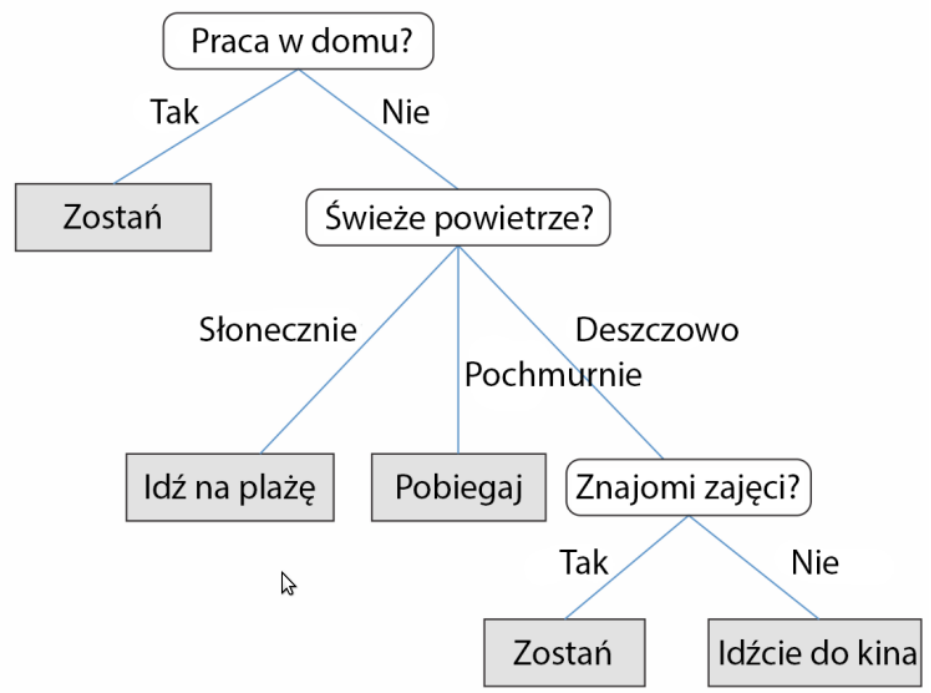
Metody Inżynierii Wiedzy
Łączenie różnych modeli w celu działania zespołowego - wykład 5

Adam Szmigielski
aszmigie@pjwstk.edu.pl
materiały: $ftp(public) : //aszmigie/MIW$

Drzewa decyzyjne



- Ten model decyzyjny jest klasyfikatorem, dokonującym klasyfikacji danych poprzez podejmowanie decyzji na podstawie szeregu odpowiedzi,
- Na podstawie cech zestawu uczącego model drzewa decyzyjnego wykorzystuje szereg pytań do określania etykiet klas próbek.

Konstrukcja drzew decyzyjnych

- Tworzymy korzeń drzewa i rozdzielamy dane wobec cechy mającej największy przyrost informacji (ang. information gain — IG)
- Poprzez wielokrotne iteracje możemy powtarzać procedurę rozdzielania danych w każdym potomnym węźle, aż uzyskamy same liście.
- Wszystkie próbki w danym węźle przynależą do tej samej klasy.
- Rozwiązanie to często skutkuje powstawaniem dużych, wielowęzłowych drzew, co może z łatwością prowadzić do przetrenowania (należy ograniczyć wysokość drzewa)

Maksymalizowanie przyrostu informacji

- Aby móc rozdzielać węzły zawierające najbardziej informatywne cechy, musimy zdefiniować *funkcję celu*,
- *Funkcją celu* jest maksymalizacja przyrostu informacji w każdym rozgałęzieniu, co możemy sformułować następująco:

$$IG(D_p, f) = I(D_p) - \sum_{j=1}^{m} \frac{N_j}{N_p} I(D_j)$$

  gdzie f to cecha użyta do rozgałęzianie, a D_p i D_j są zestawami danych odpowiednio: nadrzędnego węzła oraz j-tego węzła potomnego, I stanowi miarę zanieczyszczenia, N_p definiuje całkowitą liczbę próbek w węźle nadrzędnym, a N_j — w j-tym węźle potomnym.
- Funkcję celu będziemy optymalizować za pomocą algorytmu uczenia drzewa.

Określenie przyrostu informacji

- Przyrost informacji to różnica pomiędzy zanieczyszczeniem węzła nadrzędnego a sumą zanieczyszczeń węzłów potomnych — im niższe zanieczyszczenie tych drugich, tym większy przyrost informacji.
- Dla uproszczenia oraz w celu ograniczenia przestrzeni przeszukiwania jest stosowana implementacja **binarnych drzew**.
- Dla **drzew binarnych** węzeł nadrzędny rozgałęzia się na dwa węzły potomne: D_{lewy} i D_{prawy}

$$IG(D_p, f) = I(D_p) - \frac{N_{lewy}}{N_p} I(D_{lewy}) - \frac{N_{prawy}}{N_p} I(D_{prawy})$$

Miary przyrostu informacji

W **binarnych drzewach decyzyjnych** wyróżnia się miary zanieczyszczeń (kryteria rozgałęzień)

- **Entropia (I_H)**,
- **Wskaźnik Giniego** (ang. Gini impurity; I_G),
- **Błąd klasyfikacji (I_E)**.

Entropia

Dla wszystkich niepustych klas $p(i|t) \neq 0$:

$$I_H(t) = -\sum_{i=1}^{c} p(i|t) \log_2 p(i|t)$$

- Wyrażenie $p(i|t)$ oznacza proporcję pomiędzy próbkami należącymi do klasy i w danym węźle t,
- Entropia będzie wynosiła 0, jeśli wszystkie próbki w węźle będą należały do tej samej klasy,
- Maksymalną wartość osiągnie wtedy, gdy będziemy mieli do czynienia z jednorodnym rozkładem klas,
- Poprzez kryterium entropii próbujemy zmaksymalizować wzajemne informacje w drzewie.

Wskaźnik Giniego

Wskaźnik Giniego możemy interpretować jako kryterium służące do minimalizowania prawdopodobieństwa nieprawidłowej klasyfikacji:

$$I_G(t) = \sum_{i=1}^{c} p(i|t)(1 - p(i|t)) = 1 - \sum_{i=1}^{c} p(i|t)^2$$

- Podobnie jak w przypadku entropii, wskaźnik Giniego uzyskuje największą wartość, gdy klasy są między sobą idealnie wymieszane; np. dla binarnej konfiguracji klas (c = 2):

$$I_G(t) = 1 - \sum_{i=1}^{c} p(i|t)^2 = 0,5$$

- Wskaźnik Giniego i entropia generują zazwyczaj podobne wyniki,
- Zamiast różnych kryteriów zanieczyszczeń, lepiej jest eksperymentować z różnymi wartościami granicy przycinania.

Błąd klasyfikacji

Błąd klasyfikacji możemy określić jako:

$$I_E(t) = 1 - \max\{p(i|t)\}$$

- Jest to kryterium przydatne do przycinania,
- Nie jest zalecane do rozwijania drzewa, ponieważ wykazuje mniejszą czułość na zmiany w rozkładzie prawdopodobieństwa klas wewnątrz węzła.

Obliczanie wskaźników - przykład

ENTROPIA

$$I_H(D_p) = -\left(0{,}5\log_2(0{,}5) + 0{,}5\log_2(0{,}5)\right) = 1$$

$$A: I_H(D_{lewy}) = -\left(\frac{3}{4}\log_2\left(\frac{3}{4}\right) + \frac{1}{4}\log_2\left(\frac{1}{4}\right)\right) = 0{,}81$$

$$A: I_H(D_{prawy}) = -\left(\frac{1}{4}\log_2\left(\frac{1}{4}\right) + \frac{3}{4}\log_2\left(\frac{3}{4}\right)\right) = 0{,}81$$

$$A: IG_H = 1 - \frac{4}{8}0{,}81 - \frac{4}{8}0{,}81 = 0{,}19$$

$$B: I_H(D_{lewy}) = -\left(\frac{2}{6}\log_2\left(\frac{2}{6}\right) + \frac{4}{6}\log_2\left(\frac{4}{6}\right)\right) = 0{,}92$$

$$B: I_H(D_{prawy}) = 0$$

$$B: IG_H = 1 - \frac{6}{8}0{,}92 - 0 = 0{,}31$$

WSKAŹNIK GINIEGO

$$I_G(D_p) = 1 - \left(0{,}5^2 + 0{,}5^2\right) = 0{,}5$$

$$A: I_G(D_{lewy}) = 1 - \left(\left(\frac{3}{4}\right)^2 + \left(\frac{1}{4}\right)^2\right) = \frac{3}{8} = 0{,}375$$

$$A: I_G(D_{prawy}) = 1 - \left(\left(\frac{1}{4}\right)^2 + \left(\frac{3}{4}\right)^2\right) = \frac{3}{8} = 0{,}375$$

$$A: IG_G = 0{,}5 - \frac{4}{8}0{,}375 - \frac{4}{8}0{,}375 = 0{,}125$$

$$B: I_G(D_{lewy}) = 1 - \left(\left(\frac{2}{6}\right)^2 + \left(\frac{4}{6}\right)^2\right) = \frac{4}{9} = 0{,}\overline{4}$$

$$B: I_G(D_{prawy}) = 1 - \left(1^2 + 0^2\right) = 0$$

$$B: IG_G = 0{,}5 - \frac{6}{8}0{,}\overline{4} - 0 = 0{,}\overline{16}$$

BŁĄD KLASYFIKACJI

$$I_E(D_p) = 1 - 0{,}5 = 0{,}5$$

$$A: I_E(D_{lewy}) = 1 - \frac{3}{4} = 0{,}25$$

$$A: I_E(D_{prawy}) = 1 - \frac{3}{4} = 0{,}25$$

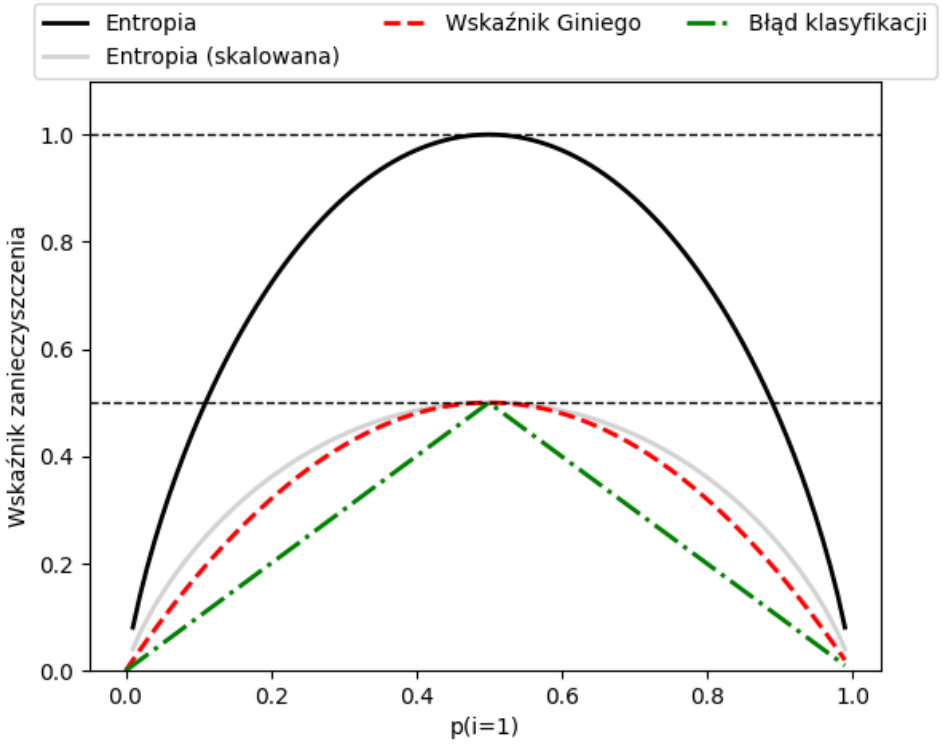
$$A: IG_E = 0{,}5 - \frac{4}{8}0{,}25 - \frac{4}{8}0{,}25 = 0{,}25$$

$$B: I_E(D_{lewy}) = 1 - \frac{2}{3} = \frac{1}{3}$$

$$B: I_E(D_{prawy}) = 1 - 1 = 0$$

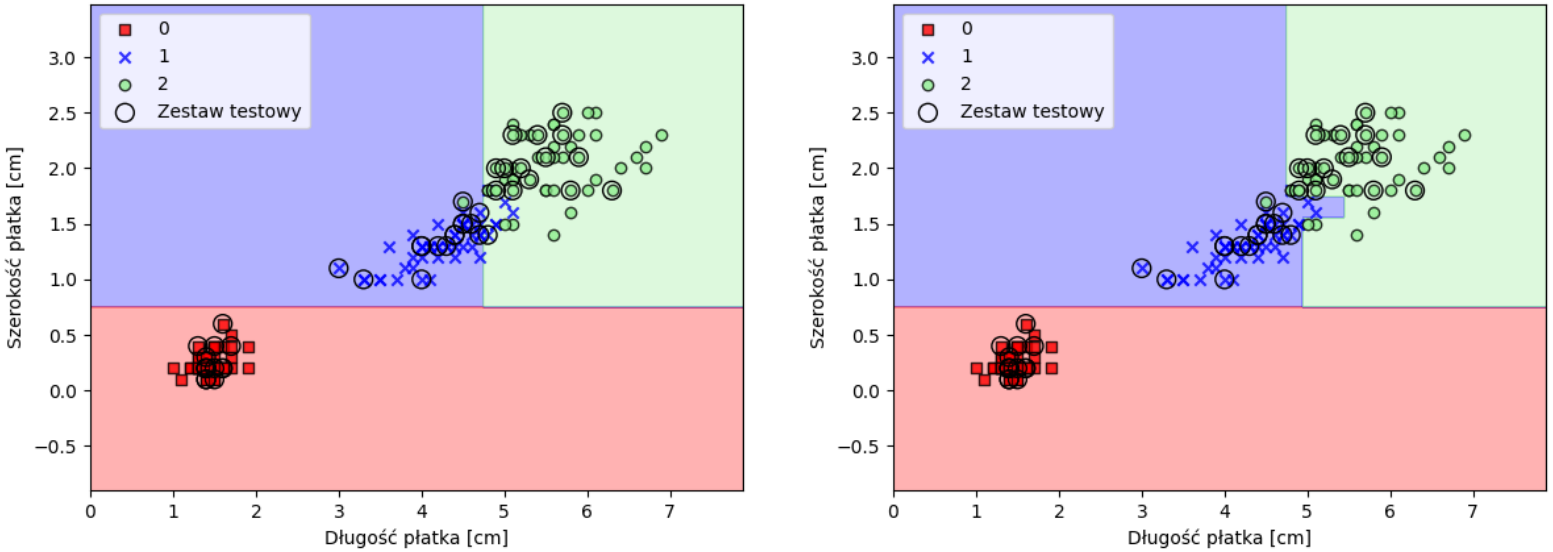
$$B: IG_E = 0{,}5 - \frac{6}{8} \times \frac{1}{3} - 0 = 0{,}25$$

Porównanie wskaźników zanieczyszczeń



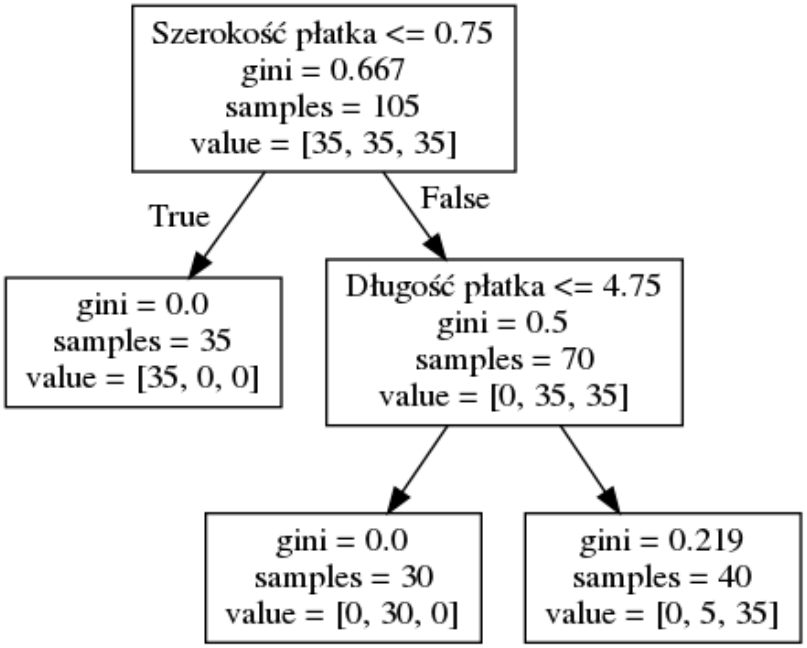
Wskaźnik Giniego daje wartości pośrednie pomiędzy entropią a błędem klasyfikacji.

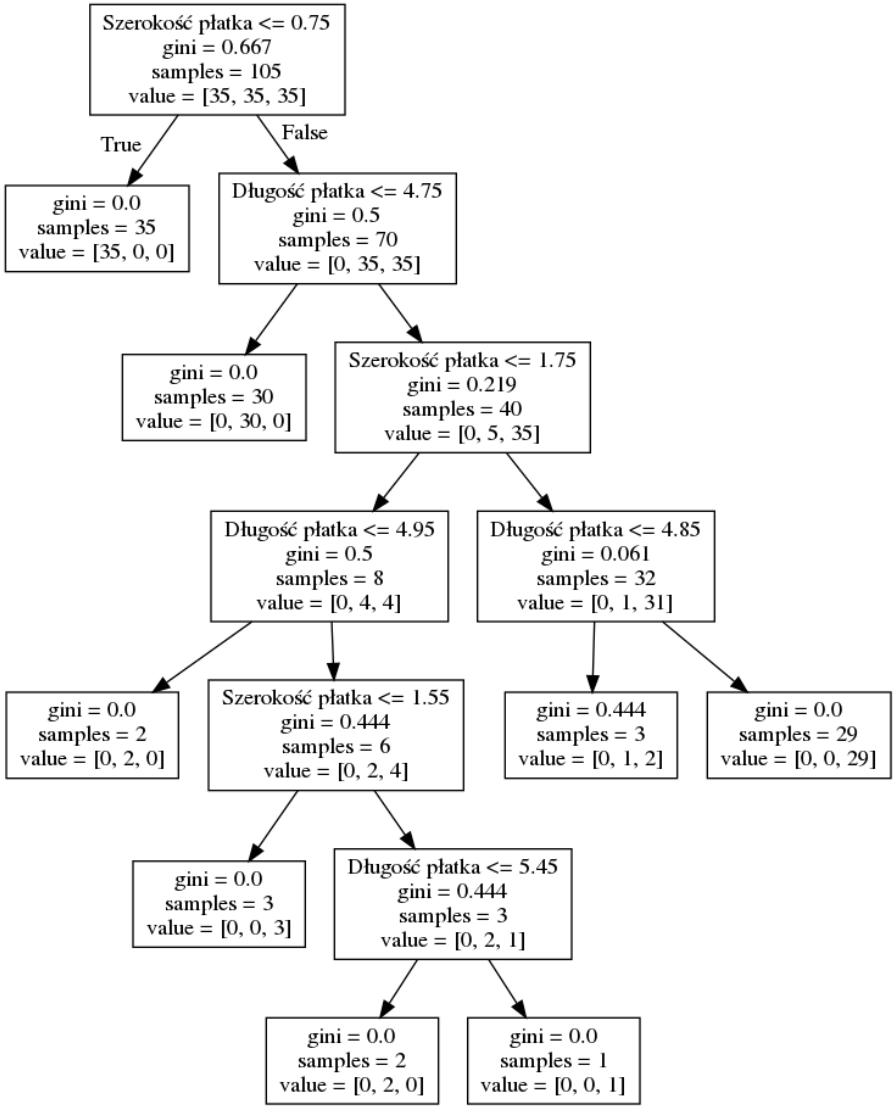
Interfejs scikit-learn - drzewa decyzyjne



Wykres granic decyzyjnych wygenerowanych za pomocą algorytmu drzewa decyzyjnego dla drzew o głębokościach 2 i 6

Szczegółowy podział drzew - (aplikacja Graphviz)





Głębokość drzew 2 i 6

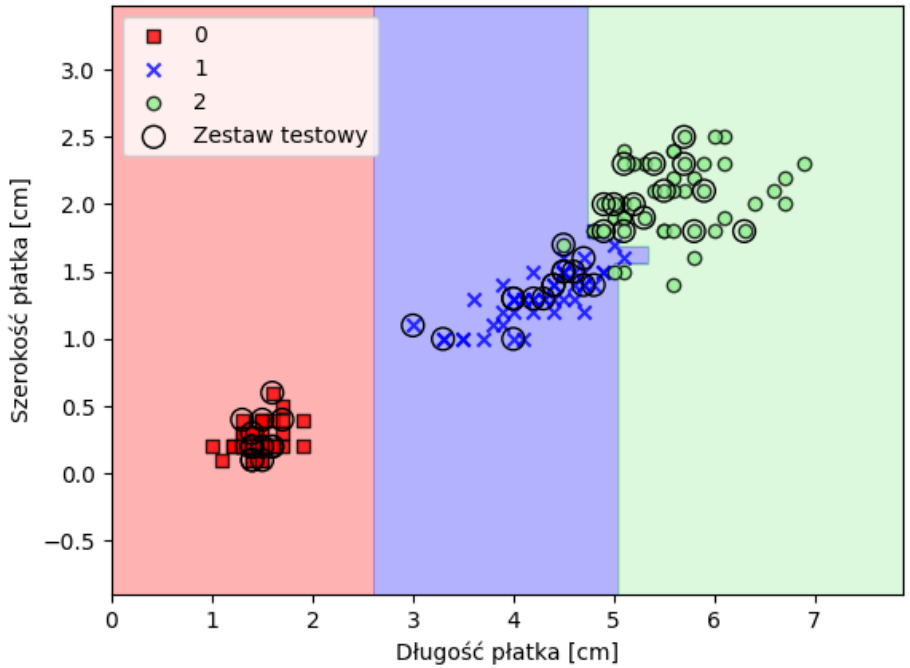
Metoda losowych lasów (ang. random forest)

- Koncepcja kryjąca się za uczeniem zespołów polega na łączeniu słabych klasyfikatorów (ang. weak learners) w jeden skuteczniejszy model — silny klasyfika tor (ang. strong learner),
- Przez **losowy las** możemy rozumieć zespół drzew decyzyjnych.
- Odznacza się dobrą skutecznością klasyfikacji, skalowalnością i łatwością stosowania.
- Klasyfikatory tego typu mają mniejszy błąd uogólniania oraz wykazujący niższą wrażliwość na przetrenowanie.

Algorytm losowego lasu

1. Wprowadź losowanie n próbek początkowych (ang. bootstrap; losowo dobierz n próbek z zestawu uczącego i wstaw za nie próbki zastępcze).
2. Wygeneruj drzewo decyzyjne na podstawie próbek początkowych. W każdym węźle:
   - Dobierz losowo d cech i nie zastępuj ich innymi.
   - Rozdziel węzeł pod kątem maksymalizacji funkcji celu (np. maksymalizując przyrost informacji).
3. Powtórz kroki 2. i 3. k-krotnie.
4. Zbierz prognozy otrzymane z każdego drzewa i przydzielaj próbkom etykiety klas poprzez większościowe głosowanie.

Wynik działania losowego lasu



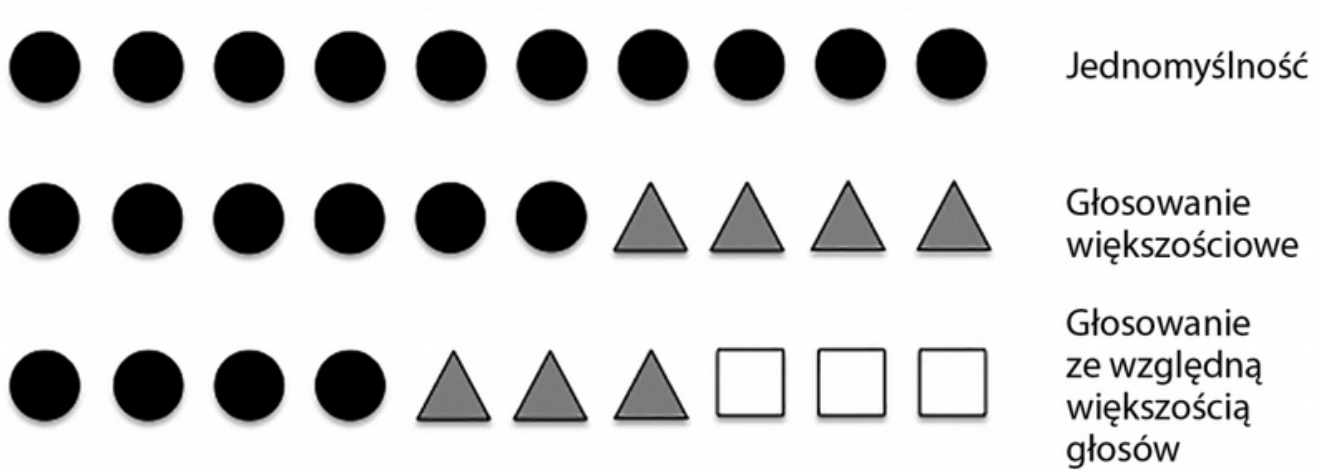
Losowy las składający się z 15 drzew ze wskaźnikiem Giniego jako kryterium zanieczyszczenia do tworzenia rozgałęzień.

Uczenie zespołów

Celem metod zespołowych (ang. ensemble methods) jest łączenie różnych klasyfikatorów w jeden metaklasyfikator wykazujący większą skuteczność uogólniania niż każdy ze składowych algorytmów.

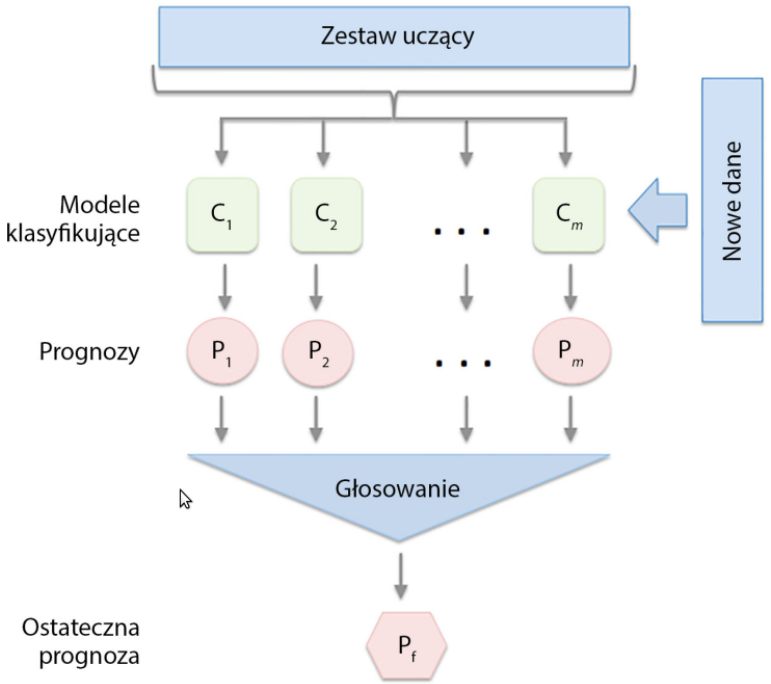
- Korzystając z zestawu uczącego, najpierw uczymy m klasyfikatorów $(C_1, \ldots, C_m)$. W zależności od używanej techniki zespół może być skonstruowany z różnych algorytmów klasyfikacji,
- Możemy również wykorzystać ten sam bazowy algorytm uczenia dopasowujący się do różnych podzbiorów zestawu uczącego.

Sposoby ustalania wyniku w głosowaniu



- **Głosowanie większościowe** (ang. majority voting) - polega na wyborze etykiety klas przewidzianej przez większość klasyfikatorów, tj. takiej, która uzyskała ponad 50% głosów.
- **Głosowanie ze względną większością głosów** (ang. plurality voting) -wybieramy etykietę klas, która otrzymała najwięcej głosów (dominantę)

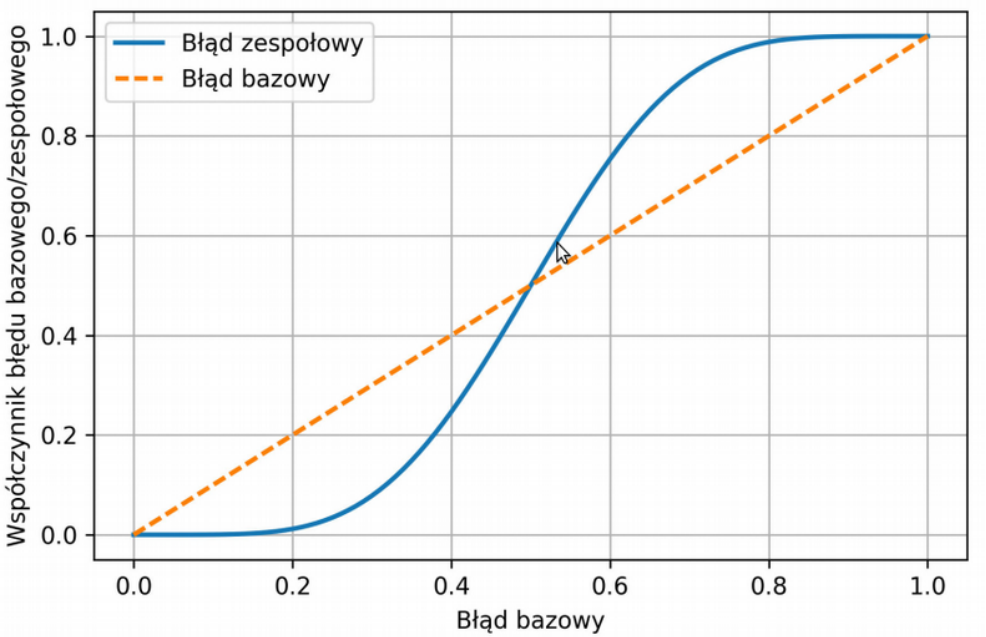
Głosowanie większościowe



Aby przewidzieć etykietę klasy za pomocą **głosowania większościowego** łączymy prognozy etykiet klas pochodzące z każdego pojedynczego klasyfikatora C_j, a następnie wybieramy etykietę $\hat{y}$, która otrzymała najwięcej głosów:

$$\hat{y} = moda\{C_1(x), C_2(x), \ldots, C_m(x)\},$$

Błąd zespołowy a pojedynczego klasyfikatora bazowego



- Prawdopodobieństwo wystąpienia błędu w zespole mniejsze od błędu pojedynczego klasyfikatora bazowego,
- W przypadku, gdy te poszczególne algorytmy są skuteczniejsze od losowego zgadywania ($\epsilon < 0,5$).
- Oś y reprezentuje zarówno błąd bazowy (linia przerywana), jak i błąd zespołowy (linia ciągła).

Prosty klasyfikator większościowy

- Ważone głosowanie większościowe można sformułować następująco:

$$\hat{y} = \arg\max_i \sum_{j=1}^{m} w_j \chi_A(C_j(x) = i)$$

  gdzie w_j jest wagą klasyfikatora bazowego C_j, $\hat{y}$ prognozą zespołu etykiety klas, χ_A funkcją charakterystyczną $[C_j(x) = i \in A]$ a A zbiorem unikatowych etykiet klas.

- W przypadku równych wag otrzymujemy:

$$\hat{y} = moda\{C_1(x), C_2(x), \ldots, C_m(x)\},$$

Prosty klasyfikator większościowy

Załóżmy,że mamy 3 binarne klasyfikatory bazowe C_1, C_2 i C_3, które dla próbki x generują etykiety:

$$C_1(x) \to 0, C_2(x) \to 0, C_3(x) \to 1.$$

- Dla klasyfikatora większościowego wynik głosowania wynosi 0:

$$\hat{y} = moda\{0, 0, 1\} = 0.$$

- W przypadku gdy wagi klasyfikatorów wynoszą $w_1 = 0,2$, $w_2 = 0,2$ i $w_3 = 0,6$ wówczas wynik wynosi 1:

$$\hat{y} = moda\{0, 0, 1, 1, 1\} = 1.$$

W bibliotece *numpy* dostępna jest funkcja $np.argmax(np.bincount([0, 0, 1], weights = [0.2, 0.2, 0.6]))$.

Łączenie klasyfikatorów klasyfikujących z prawdopodobieństwem

Załóżmy,że mamy 3 binarne klasyfikatory bazowe C_1, C_2 i C_3, które dla próbki x generują prawdopodobieństwa etykiety 0 lub 1 dla x-a:

$$C_1(x) \to [0,9; 0,1], C_2(x) \to [0,8; 0,2], C_3(x) \to [0,4; 0,6].$$

Obliczmy teraz pojedyncze prawdopodobieństwa prognoz:

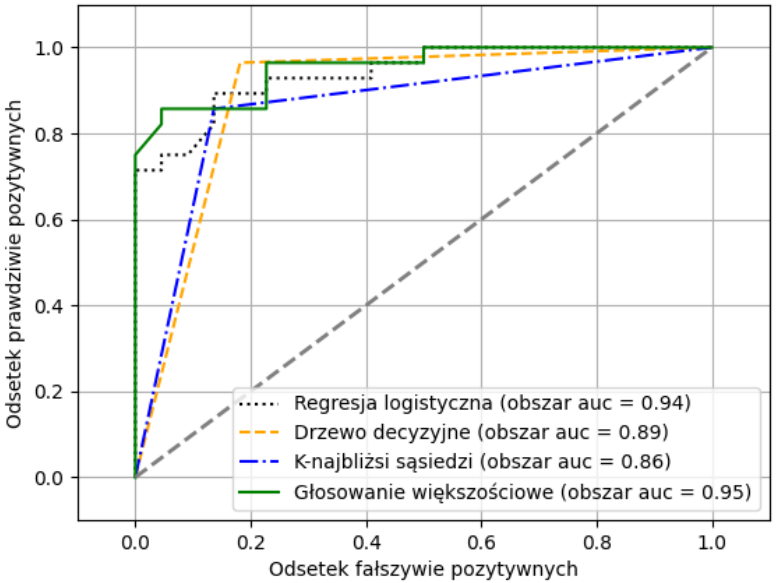
$$p(1|x) = 0,2 \cdot 0,9 + 0,2 \cdot 0,8 + 0,6 \cdot 0,4 = 0,58$$

$$p(0|x) = 0,2 \cdot 0,1 + 0,2 \cdot 0,2 + 0,6 \cdot 0,6 = 0,42$$

$$\hat{y} = \arg\max_i \{p(1|x), p(0|x)\} = 1$$

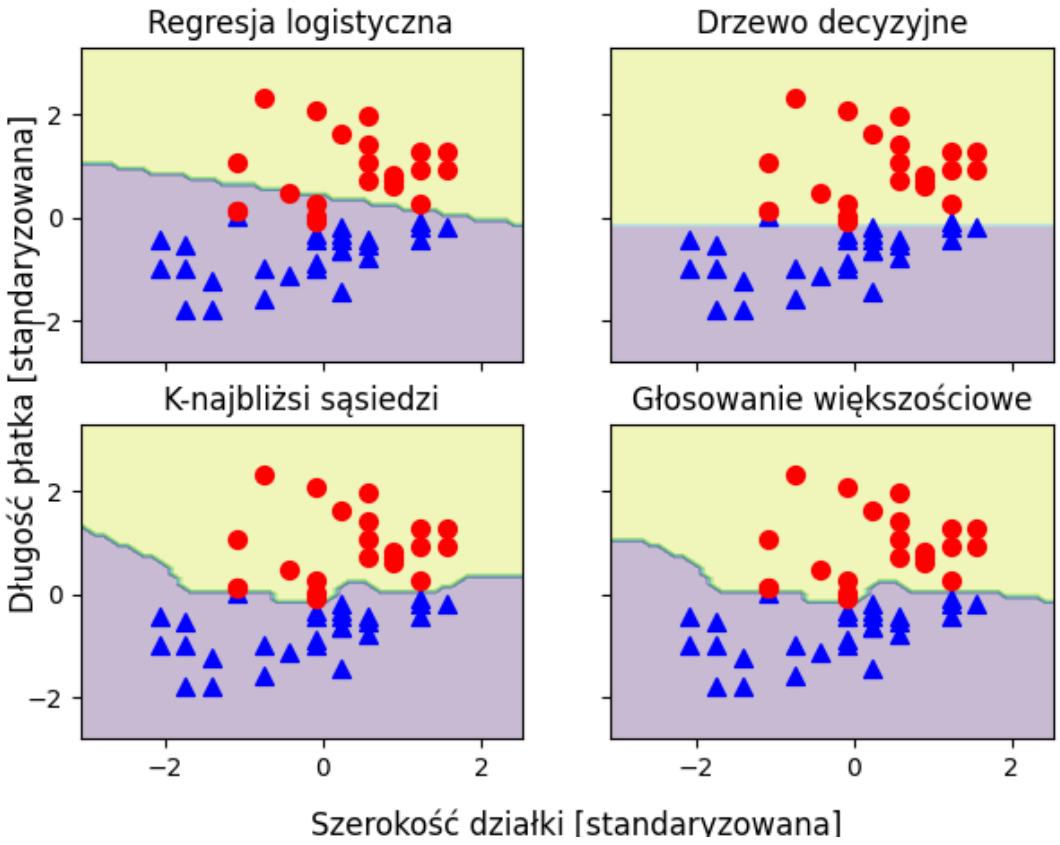
W bibliotece *numpy* dostępna jest funkcja
$np.average(ex, axis = 0, weights = [0.2, 0.2, 0.6])$.

Jakość klasyfikacii różnych klasyfikatorów



- Przekątną krzywej ROC (ang.*receiver operating characteristic*) możemy interpretować jako losowe zgadywanie,
- Obszar pod krzywą ROC (ang. *area under the curve — AUC*) opisujący skuteczność modelu klasyfikatora.
- Idealny klasyfikator znajdowałby się w lewym górnym rogu wykresu ($OPP = 1$, $OFP = 0$).

Porównanie klasyfikatorów

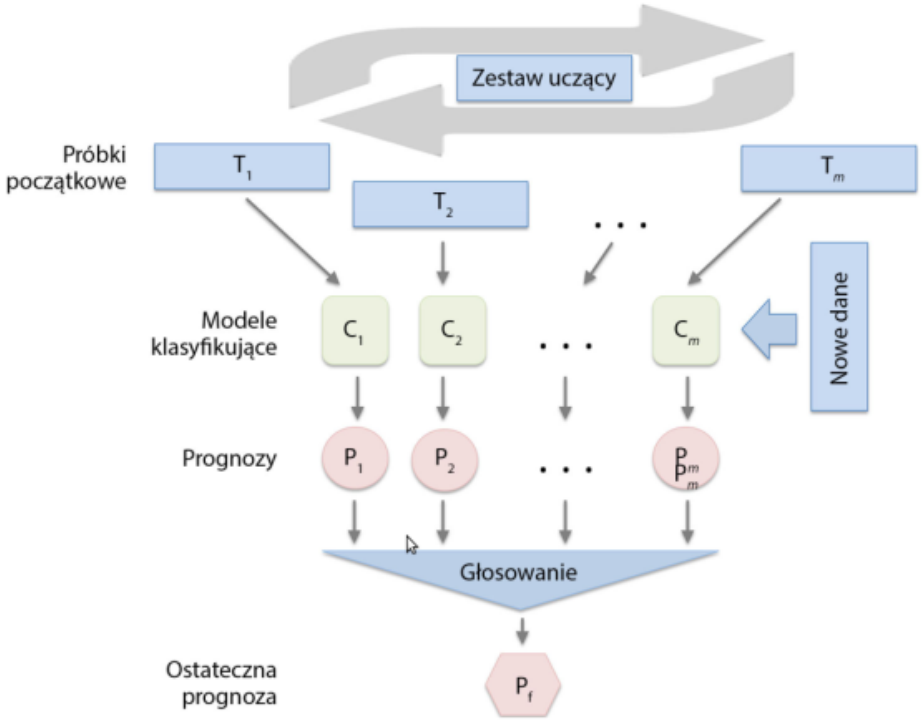


- Obszary klasyfikacji.

Tworzenie zespołu klasyfikatorów za pomocą próbek początkowych

Indeksy próbek	Agregacja: tura 1.	Agregacja: tura 2.	...
1	2	7	...
2	2	3	...
3	1	2	...
4	3	1	...
5	7	1	...
6	2	7	...
7	4	7	...

C_1 C_2 C_m



- Tworzymy próbki początkowe (podzbiór losowych próbek ze zwracaniem) z pierwotnego zestawu danych uczących,
- Próbki używa się do trenowania klasyfikatorów C_1, C_2, C_3, etc (najczęściej lasów losowych).

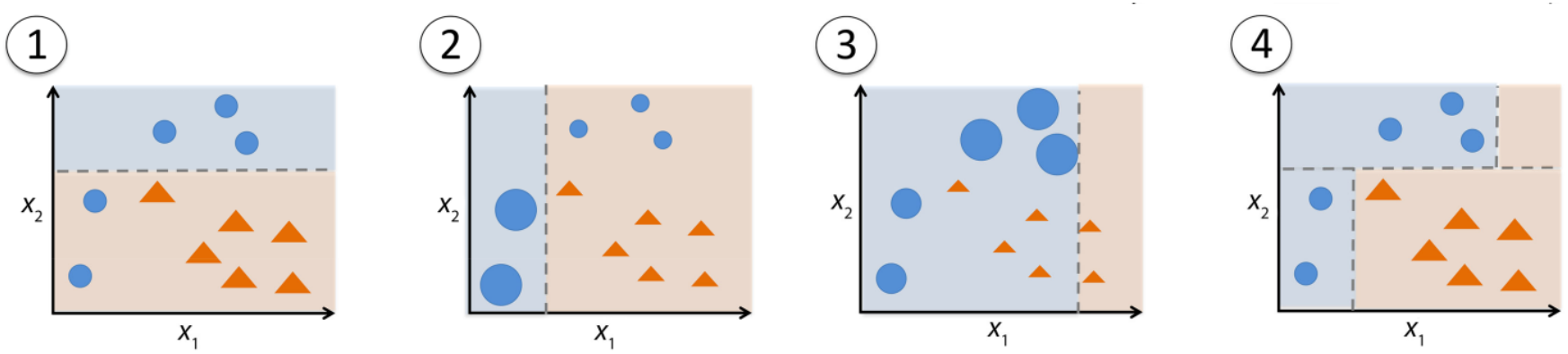
Technika wzmacniania adaptacyjnego

- Zespół składa się z bardzo prostych klasyfikatorów, często nazywanych słabymi klasyfikatorami (ang. weak learner), których skuteczność przewidywania jest tylko nieznacznie lepsza od losowego zgadywania.
- Podstawową koncepcją wzmacniania jest koncentracja na trudnych do klasyfikowania próbkach uczących w celu poprawy skuteczności całego zespołu.
- W przeciwieństwie do agregacji algorytm wzmocnienia wykorzystuje losowe podzbiory danych uczących, które są pobierane z zestawu uczącego bez zwracania.

Procedura wzmacniania

1. Stworzenie losowego podzbioru próbek uczących d_1 bez zwracania danych ze zbioru uczącego D i uczenie słabego klasyfikatora C_1.
2. Stworzenie drugiego losowego podzbioru d_2 bez zwracania danych z zestawu uczącego i dodanie 50% nieprawidłowo sklasyfikowanych próbek w kroku 1. w celu trenowania słabego klasyfikatora C_2.
3. Określenie podzbioru próbek uczących d_3 w zestawie treningowym D, wobec których klasyfikatory C_1 i C_2 są nieskuteczne, i trenowanie trzeciego słabego klasyfikatora C_3.
4. Połączenie klasyfikatorów: C_1 , C_2 i C_3 za pomocą głosowania większościowego.

Schemat działania algorytmu AdaBoost



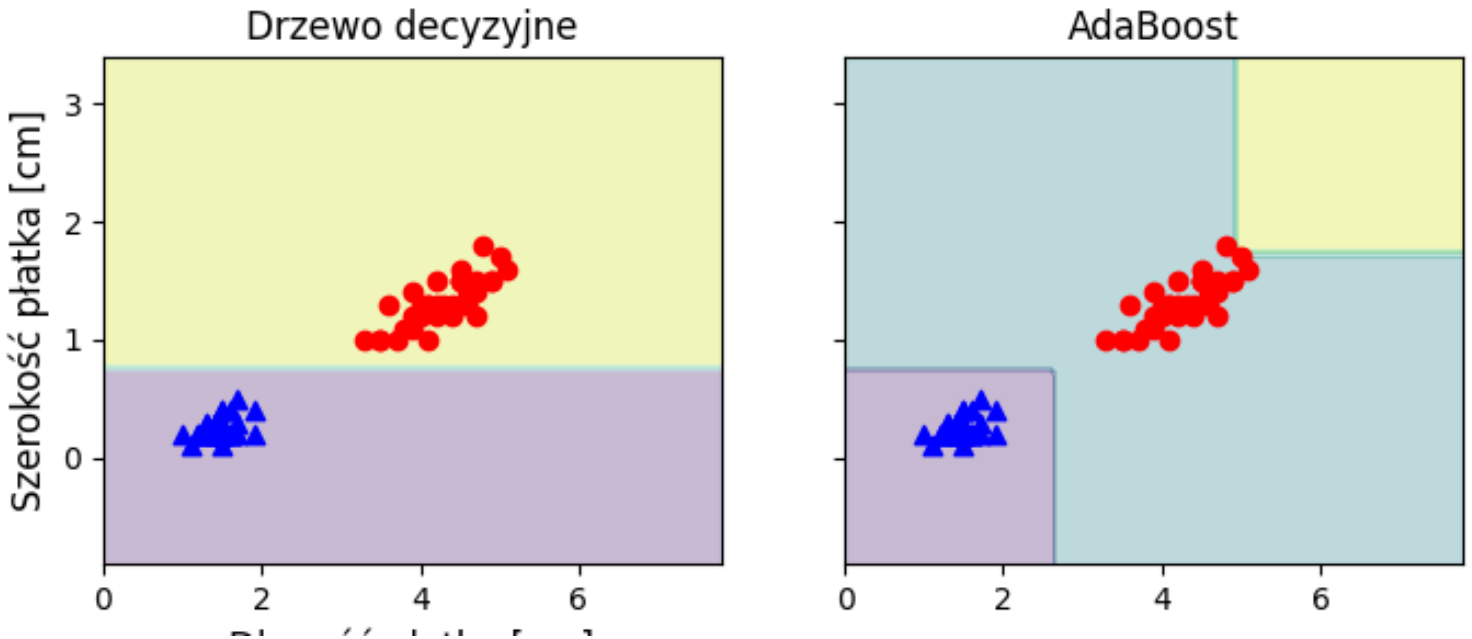
1. Punkt startowy dla klasyfikacji binarnej — wszystkie próbki mają tu jednakowe wagi.
2. Wyznaczamy większe wagi dwóm uprzednio niewłaściwie sklasyfikowanym próbkom (kółkom).
3. Słaby klasyfikator ukazany na wykresie 2. nieprawidłowo klasyfikuje trzy różne próbki z klasy oznaczonej kółkami, co oznacza, że w kolejnej iteracji (wykres 3.) uzyskują nowe, większe wagi.
4. Po głosowaniu większościowym ważonym otrzymujemy końcowy rezultat.

Poszczególne etapu algorytmu AdaBoost

1. Wyznacz wektor w zawierający jednakowe wagi, gdzie $\sum_i w_i = 1$
2. W j-tej turze z m iteracji wykonaj następujące czynności:
3. Wyucz ważony, słaby klasyfikator: $C_j = ucz(X, y, w)$.
4. Prognozuj etykiety klas: $\hat{y} = prognozuj(C_j, X)$.
5. Oblicz ważoną stopę błędu: $\varepsilon = w \cdot (\hat{y} \neq y)$. [a]
6. Wylicz współczynnik: $\alpha_j = 0,5 \log \frac{1-\varepsilon}{\varepsilon}$.
7. Zaktualizuj wagi: $w \leftarrow w \cdot \exp^{-\alpha_j \cdot \hat{y} \cdot y}$.
8. Znormalizuj wagi tak, aby ich suma dawała wartość 1: $w \leftarrow \frac{w}{\sum_i w_i}$.
9. Oblicz ostateczną prognozę: $\hat{y} = (\sum_j (\alpha_j \cdot prognozuj(C_j, X)) > 0)$.

[a] 1 gdy zła prognoza, 0 gdy prawidłowa

Porównanie klasyfikatorów drzewa decyzyjnego i AdaBoost



- Dokładność drzewa decyzyjnego dla danych uczących/testowych 0.667/0.667
- Dokładność algorytmu AdaBoost dla danych uczących/testowych 0.933/0.933

Zadania na ćwiczenia

Zadania wykonaj z wykorzystaniem pakietu *scikit-learn*

1. Stwórz zbiór danych za pomocą funkcji $make\_moons(n\_samples = 10000, noise = 0.4)$.
2. Rozdziel uzyskany zestaw danych na podzbiory uczący i testowy przy użyciu metody $train\_test\_split()$.
3. Wykorzystaj drzewo jako klasyfikator (*DecisionTreeClassifier*). Zbadaj działanie drzewa dla entropii i współczynnika Giniego oraz różnych głębokości drzewa.
4. Jako klasyfikator użyj lasów losowych (*RandomForestClassifier*). Zbadaj działanie klasyfikatora dla różnej liczby drzew decyzyjnych.
5. Wytrenuj klasyfikator regresji logistycznej (*LogisticRegression*) oraz SVM,
6. W celu poprawy klasyfikacji połączyć klasyfikatory *SVM*,

LogisticRegression oraz *RandomForestClassifier* w jeden zespół (*VotingClassifier*).

7. Oceń osiągnięte rezultaty.