Drzewa decyzji są modelami predykcyjnymi, które strukturę modelu reprezentują w postaci drzewa. W trakcie budowy wykorzystywana jest metodologia dziel i rządź, w efekcie czego na każdym etapie budowy modelu dany problem predykcyjny (zbiór danych) dzielony jest na pod problemy, które w kolejnej iteracji algorytm próbuje rozwiązać dzieląc ja na kolejne pod problemy tworząc przy tym strukturę drzewa. W klasycznych drzewach decyzji dekompozycja problemu na pod problemy odbywa się poprzez stworzenie nowego węzła, na który składa się atrybut decyzyjne, natomiast gałęzie odpowiadają warunkom logicznym. Drzewo budowane jest do momentu, w którym osiągnięte jest kryterium stopu i wówczas taki węzeł zostaje zamieniony na liść czyli węzeł w którym podejmowana jest decyzja.

Do budowy drzew opracowano różne kryteria oceniające jakość danego podziału. Najpopularniejszymi kryteriami są indeks Gini opisany jako:

$$Q_G(q) = 1 - \sum_{i=1}^{c} p(y = s_i|q)^2$$

gdzie $p(y = s_i|q)$ jest prawdopodobieństwem warunkowym wskazującym liczbę obiektów z klasy s_i znajduje się w węźle q. Do wyznaczenia testu logicznego wykorzystuje się jednak Zysk Giniego

$$Q_G(q, f) = Q_G(q) - \sum_{i \in U(f)} \frac{|q_v|}{|q|} Q_G(q_v)$$

gdzie $U(f)$ jest zbiorem wartości atrybutu f. Dla drzewa binarnego, które dzieli obiekty na *lewo* i *prawo* mamy $U(f_L)$ czyli zbiór indeksów wektorów znajdujących się po lewej stronie i $U(f_P)$ czyli zbiór indeksów wektorów znajdujących się po prawej stronie. Budując drzewo maksymalizujemy zysk. Innym popularnym kryterium jest względny zysk informacyjny (IGR, ang. Information Gain Ratio), który stanowi rozszerzenie kryterium zysku informacyjnego Q_IG opisanego jako

$$Q_E(q) = -\sum_{i=1}^{c} p(y = s_i|q) log_2 p(y = s_i|q)$$

$$Q_{IG} = Q_E(q) - \sum_{v \in U(f)} \frac{|q_v|}{q} Q_E(q_v)$$

gdzie $Q_E(q)$ jest entropią Shanona, przy czym IGR wyznaczamy jako:

$$Q_{IGR}(q, f) = \frac{Q_{IG}(q, f)}{Q_E(q)}$$

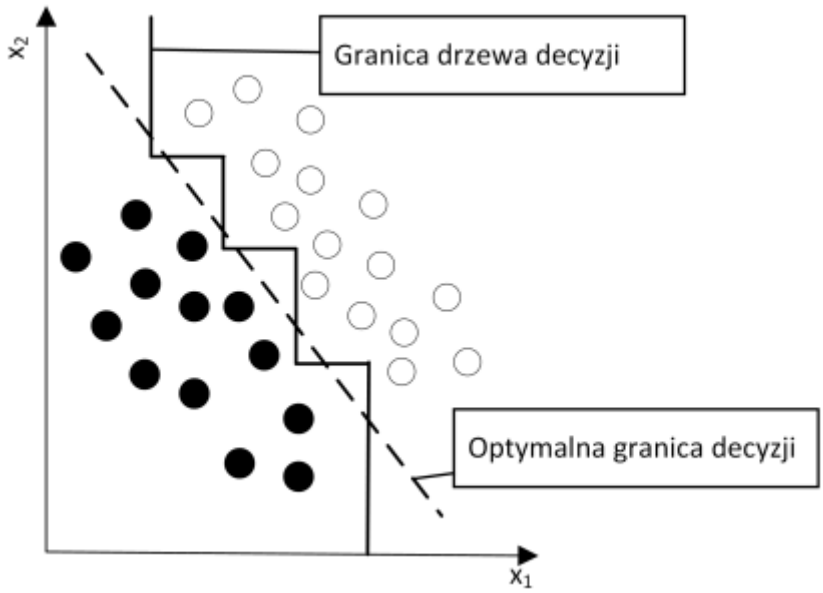
W przypadku drzew regresyjnych do oceny kryterium podziału stosuje się zwykle minimalizację wariancji

$$Q_V(q) = Var(q) - \sum_{v \in U(f)} Var(q_v)$$

gdzie $Var(\cdot)$ jest estymatorem wariancji dla obiektów znajdujących się w węźle q. Drzewa decyzji występują zwykle w dwóch postaciach tj. drzew binarnych jak np. CART (drzewo to wykorzystuje indeks Gini'eago) [46], które na każdym etapie budowy w każdym węźle decyzyjnym dokonują podziału danych zawsze na dwa podzbiory lub też drzew w który możliwy jest podział na więcej niż dwa podzbiory. Przykładem takiego drzewa jest algorytm C4.5 (drzewo to wykorzystuje indeks IGR) [194], gdzie atrybuty symboliczne dzielone są na s podzbiorów, gdzie s oznacza liczbę symboli kodowanych występujących w danym atrybucie. Atrybuty ciągłe dzielone są na dwa. W celu wyznaczenia atrybutu warunkowego każdą cechę w zbiorze treningowym ocenia się iteracyjnie szukając rozwiązania minimalizującego / maksymalizującego dane kryterium. W celu poprawy właściwości drzewa zwykle stosuje się przeszukiwanie wiązką, które pozwala na analizę nie tylko pierwszego

najlepszego podziału ale również *k* kolejnych, najlepszych podziałów, co zapewnia uzyskanie bardziej dokładnego drzewa.Drzewa decyzji rosną aż do osiągnięcia pełnej, porawnej klasyfikacji na zbiorze treningowym lub też do osiągnięcia kryterium stopu. Kryterium stopu definiowane jest zwykle jako maksymalna głębokość drzewa, lub też jako minimalna liczba rekordów należąca do danego węzła. Tak zbudowane drzewa decyzji mają jednak tendencję do przeuczania się więc poprawę generalizacji uzyskuje się poprzez ich przycięcie. Proces ten odbywa się w kierunku z dołu do góry poprzez zdefiniowanie kryteriów oceniających jak duży zysk/jaką stratę spowoduje zamiana danego węzła na liść. Aby poprawić jakość drzew do oceny poziomu przycinania stosuje się wewnętrzny (wbudowany) test krzyżowy który pozwala na bardziej wiarygodną ocenę poziomu przycięcia drzewa. Inną koncepcją optymalizacji drzew jest wykorzystanie meta-heurystyk, które pozwalają optymalizować całość drzewa i nie są ograniczone przez sekwencyjną kolejność konstrukcji drzewa [163].

Drzewa są bardzo popularnymi modelami predykcyjnymi, a swoją popularność zawdzięczają między innymi możliwości interpretacji nauczonego modelu, gdyż można go przedstawić w postaci zbioru reguł, ponadto drzewa to modele o niedużej złożoności obliczeniowej zarówno w trakcie uczenia jak i predykcji. W trakcie uczenia ich złożoność jest liniowo logarytmiczna $O(m \cdot nlog(n))$, natomiast w trakcie predykcji złożoność obliczeniowa jest rzędu $O(logn)$, a w trakcie budowy łatwo dają się zrównoleglić. Co więcej mają niską łożoność przestrzenną, pozwalają na automatyczną selekcję cech, wspierają mieszane typy atrybutów, są nieczułe na skalowanie poszczególnych zmiennych, tolerują wartości brakujące oraz akceptują wstępne informacje np. dotyczących rozkładu klas w problemach klasyfikacji czy też wag opisujących istotność poszczególnych wektorów w zbiorze treningowym. Posiadają jednak szereg wad. Jednym z istotnych problemów związanych z budową drzew decyzji jest pogarszanie się jakości estymacji funkcji kryterialnej w miarę gdy podzbiór obiektów przypadających na dany węzeł staje się mniejszy. Innym problemem występującym w drzewach decyzji jest analiza zbiorów składających się z atrybutów ciągłych, gdyż w ich przypadku liniowa granica decyzji aproksymowana jest za pomocą kolejnych warunków logicznych co powoduje nadmierny rozrost drzewa oraz utratę możliwości jego interpretacji. Powyższy efekt obrazuje Rys.1

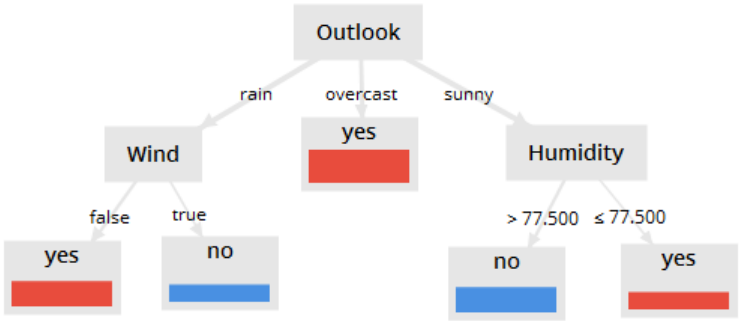


Rysunek 1: Granica drzewa decyzji oraz optymalnego klasyfikatora Bayessowskiego.

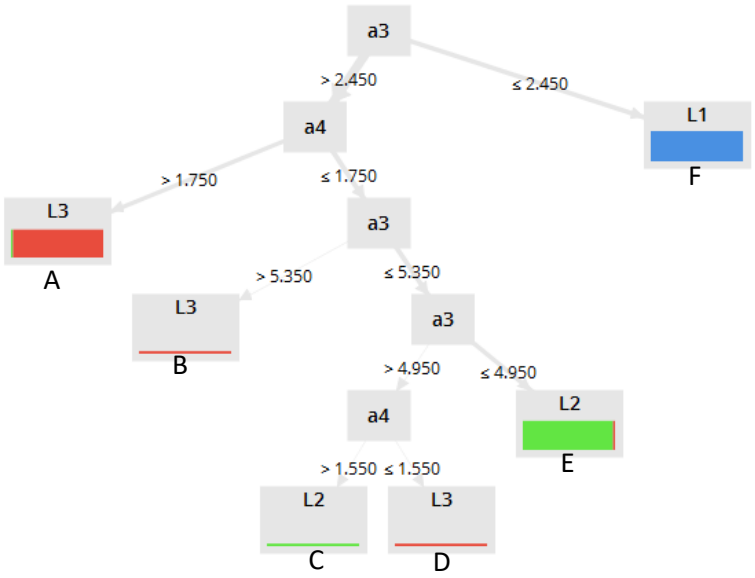
W celu poprawy drzew decyzji stosuje się różne rozwiązania tworząc bardziej skomplikowane funkcje najdujące się w węzłach drzewa, tak iż nie tylko pojedynczy atrybut + warunek stanowią kryterium ale wprowadza się drzewa, które w węzłach posiadają modele predykcyjne np. model linowy [231], lub klasyfikator Bayessowski [140]. Innym rozwiązaniem stosowanym do poprawy szybkości i jakości indukcji drzew decyzji jest korzystanie meta-uczenia, które pozwala na automatyzację doboru parametrów drzewa [106].

Zadania

1) Dla podanego drzewa stworzonego do rozwiązania problemu klasyfikacyjnego (klasy yes, no) wypisz reguły – po jednej dla każdego liścia. Pamiętaj że węzły wskazują na atrybut decyzyjny, a krawędzie odpowiadają poszczególnym wartościom.

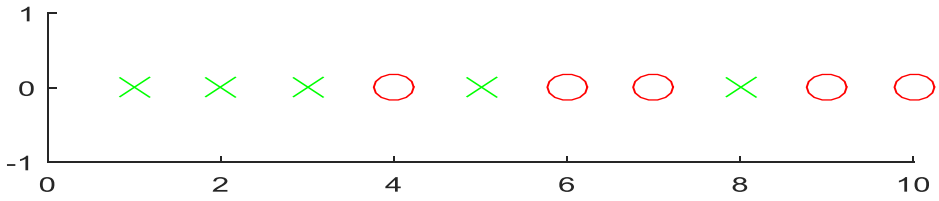


2) Wiedząc, że obiekty mogą należeć do kategorii {L1, L2, L3} oraz wiedząc, że zbiór danych składa się z atrybutów {a1, s2, a3, a4} przeanalizuj podane drzewo i wyznacz, które z liści będą odpowiedzialne za podjęcie decyzji (liście oznaczono kolejnymi literami alfabetu) oraz wpisz etykietę która będzie przypisana podczas klasyfikacji do każdego z poniższych wektorów:

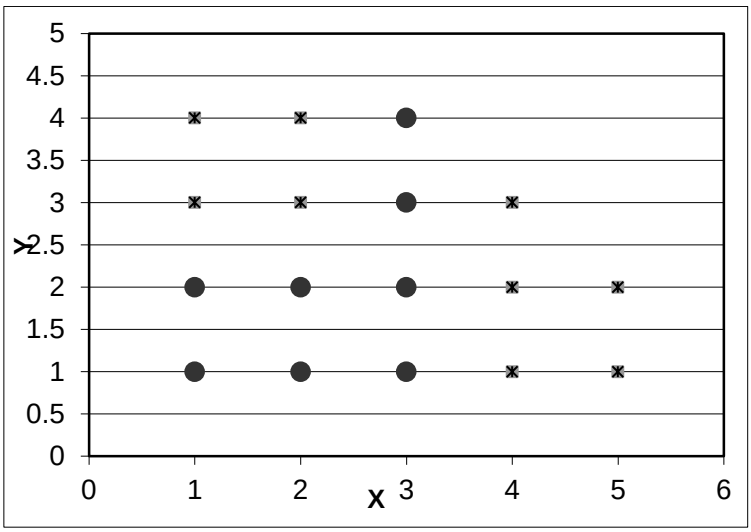


A1	A2	A3	A4	Etykieta	Nr Węzła
1	2	1.5	2.5		
1,5	2,5	2.5	1.5		
3	4	5	1.5		

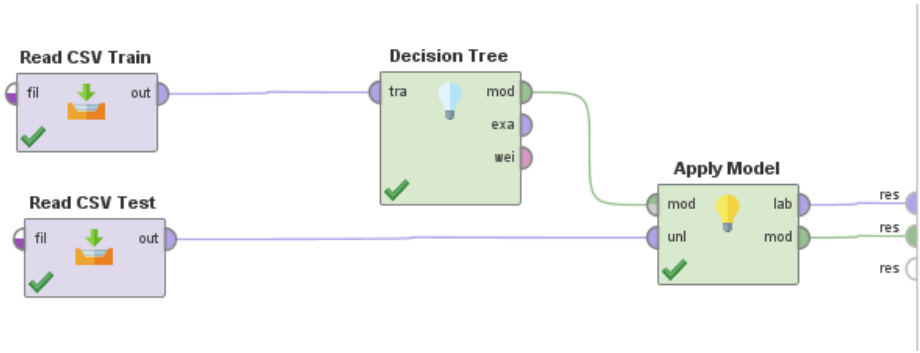
3) Wyznacz dla jakiej wartości x indeks Gini przyjmuje maksymalną wartość



4) Nie licząc indeksów spróbuj „na oko” zaproponować drzewo decyzji, które pokryje poniższe punkty i zapewni ich poprawną klasyfikację wiedząc że poszczególne punkty należą do jednej z dwóch klas {✱, ●}. Narysuj drzewo i określ atrybuty warunkowe oraz wartości warunków



5) W RapidMiner – wczytaj zbiór dataTrain_2G.csv (patrz kNN) i naucz na nim drzewo decyzji (operator Decision Tree). W opcjach drzewa odznacz *apply pruning* oraz *apply preprunning* oraz wczytaj zbiór dataTest_2G.csv i zastosuj na nim nauczone wcześniej drzewo decyzji.



a. Jaka jest optymalna granica decyzji dla przedstawionego problemu predykcyjnego
b. Pokaż jaką granice decyzji uzyskało drzewo decyzji
c. Pokaż drzewo decyzji (gdyby się nie mieściło na ekranie, wówczas skorzystaj z okienka zoom (+),(-)) tak aby było je w całości widać.
d. Zmień opcję *maximal depth* o 3 w górę i w dół (do 4 do 16). Czy zwiększenie głębokości drzewa powoduje zwiększenie generalizacji (uogólnienia), czy może powoduje przeuczenie.
e. Zastanów się jak można opisać złożoność drzewa – podaj propozycję (zwróć uwagę, że w RapidMiner obok widoku drzewa jako grafu można je również przedstawić w formie tekstowej - description)

f. Dowiedz się co to jest *pre-prunning* (przeczytaj w dokumentacji) i opisz to. Ustaw głębokość drzewa na 10 i porównaj jego złożoność (rozmiar) w przypadku gdy *apply preprunning* jest włączony oraz gdy jest wyłączony.

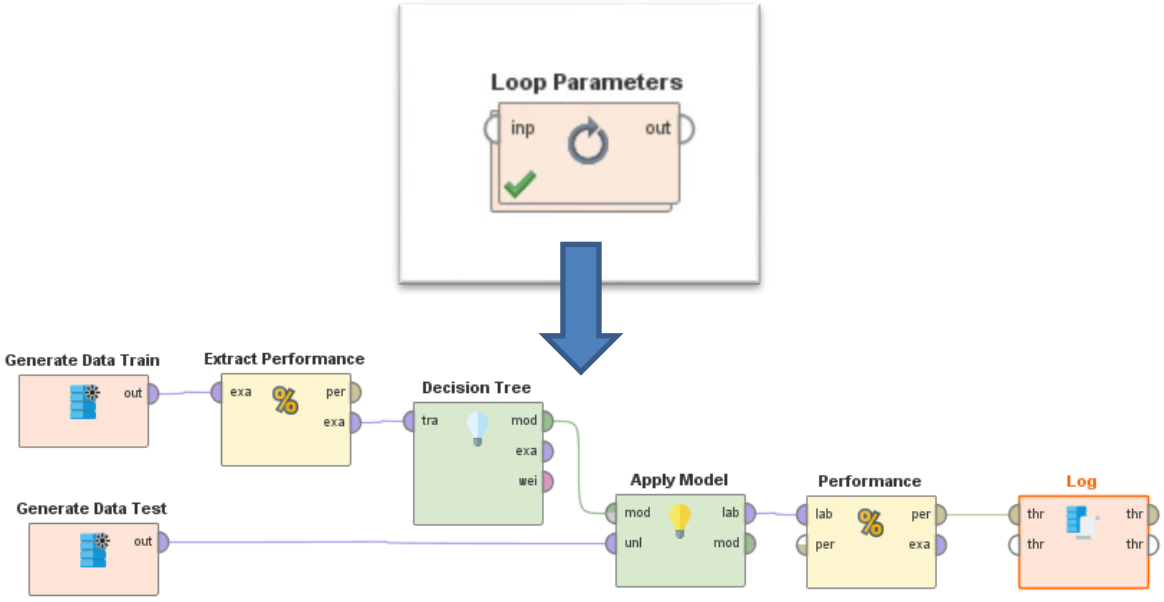
g. Dowiedz się co to jest *apply pruning* (przeczytaj w dokumentacji) i opisz to. Zaznacz opcję *apply pruning* Zmieniaj wartość *Confidence* i sprawdź jak wpływa ona na rozmiar drzewa oraz uzyskaną granicę decyzji (w razie konieczności możesz wyłączyć opcję *apply prepruning*)

h. Jak rozmiar drzewa wpływa na możliwość interpretacji reguł?

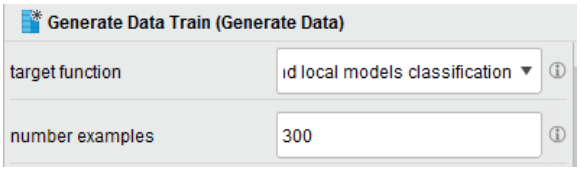
6) Badanie złożoności obliczeniowej drzewa decyzji.

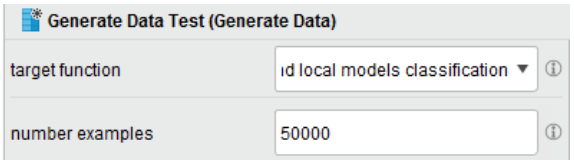
W celu zbadania złożoności obliczeniowej dla zadanego (dużego) zbioru testowego zmieniaj rozmiar zbioru treningowego i zarejestruj czas potrzebny na uczenie i predykcję. W tym celu

a. Stwórz proces jak na obrazku:

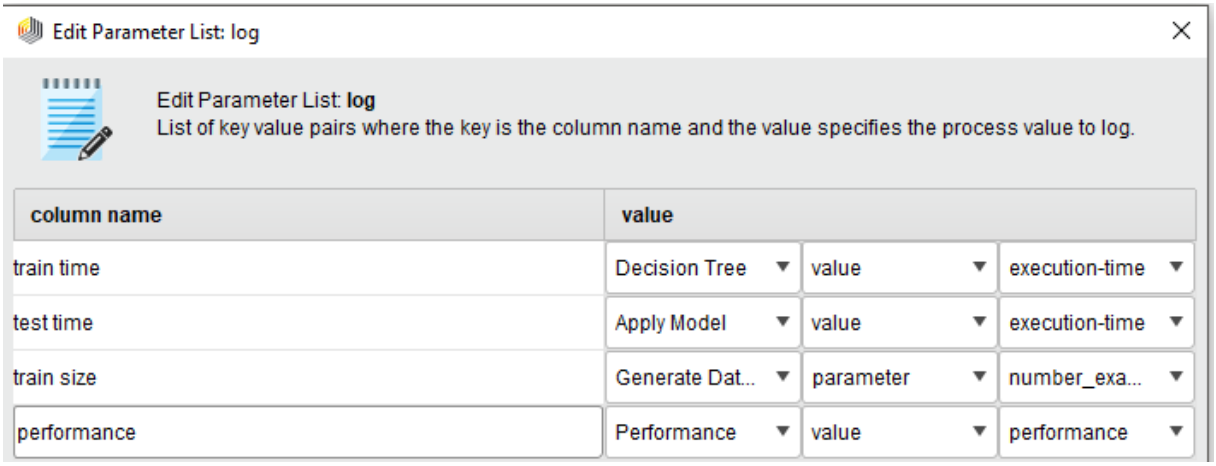


b. Ustaw operatory Generate Data odpowiednio jak na obrazku:





c. Ustaw operator log jak na obrazku:



d. W operatorze loop parameters zmieniaj parametr *number_of_examples* w opratorze *Generate Data Train* w zakresie 100-20000

e. Na podstawie uzyskanych wyników wykreśl zależności pokazujące jak rozmiar zbioru treningowego wpływa na czas predykcji, czas uczenia oraz dokładność drzewa decyzji.