**Reguły logiczne: jak wydobyć   
i zrozumieć wiedzę ukrytą w danych**

**Streszczenie**

Odkrywanie wiedzy w danych służy po pierwsze wzbogacaniu naszej wiedzy o procesach opisanych tymi danymi, a po drugie upewnieniu użytkownika, że inteligentny system podjął odpowiednią decyzję poprzez przedstawienie użytkownikowi uzasadnienia tej decyzji. Pierwszy cel jest istotny w pracy naukowej i inżynierskiej, a drugi w codziennym funkcjonowaniu systemów wspomagania decyzji.

Pprzedstawię poglądowo podejście do odkrywania wiedzy w danych i prezentowania jej w formie reguł logicznych w systemach wykorzystujących sztuczną inteligencję, ze szczególnym uwzględnieniem najnowszych rozwiązań. Wiele rzeczy zostanie przedstawionych w sposób uproszczony, lub w ogółe pominiętych. Jest to jednak konieczne celem spójnego przedstawienia tej tematyki w ramach jednego wykładu.

1. **Wstęp**

W ostatnich czasach systemy inteligentne są coraz powszechniej stosowane w medycynie. Systemy takie są pomocne np. w wykrywaniu wczesnych symptomów choroby i zbliżającego się zagrożenia (np. udar, zawał, itd.) w oparciu o wyniki różnych badań pacjenta i ich zestawienie   
z dużą populacją pacjentów. Często system potrafi wykryć symptomy, których lekarze jeszcze nie dostrzegają. Natomiast zwłaszcza w medycynie istotnym jest to, że aby lekarz mógł zaufać takiemu systemowi, to musi rozumieć jego przesłanki. Musi wiedzieć dlaczego system wygenerował dane ostrzeżenia lub postawił daną diagnozę. Znajomość tego po pierwsze pozwoli lekarzowi zastosować się do wskazań systemu, lub stwierdzić, że zgodnie z jego wiedzą system się myli. Po drugie zaś   
w odpowiednim uzasadnieniu jest pośrednio zawarta również podpowiedź, co należy zrobić, aby poprawić zdrowie pacjenta. Kolejną sprawą, zaobserwowaną w różnych branżach, nie tylko   
w medycynie jest to, że fachowcy również się uczą od takiego systemu, gdyż uzmysławia im on istnienie pewnych zależności, których przedtem nie byli świadomi.

Innym przykładem mogą być systemy używane w branży finansowej do oceny ryzyka kredytowego, gdzie w sytuacji, gdy bank odmówi klientowi udzielenia kredytu, to w niektórych państwach jest on zobowiązany podać klientowi konkretny powód odmowy. Przy czym powód ten nie może brzmieć np. „nie uzyskał Pan wystarczająco dużej ilości punktów (credit score)”. Jeśli natomiast system oceny ryzyka oprócz końcowego wyniku generuje również reguły opisujące dlaczego podjął taką decyzję, to te reguły są już gotowym uzasadnieniem odmowy dla klienta. Dodatkowo klient na podstawie takich reguł może się dowiedzieć, co konkretnie ma poprawić.

Kolejnym zastosowaniem są praktyczne badania eksperymentalne przy tworzeniu nowego produktu, np. w przypadku doboru odpowiedniego składu chemicznego, gdzie ilość możliwych kombinacje jest praktycznie nie ograniczona, a własności materiału wcale nie zmieniają się liniowo wraz ze zwiększaniem udziału procentowego różnych składników i z czasem, w jakim zachodzi proces. Niektóre zależności są na tyle skomplikowane, że bez odpowiedniego inteligentnego systemu nie sposób ich dostrzec. Dopiero zastosowanie takich rozwiązań w kilku przypadkach posunęło prace istotnie do przodu. Odkrycie tych zależności powiększyło wiedzę inżynierów   
i umożliwiło otrzymanie materiałów o wymaganych właściwościach.

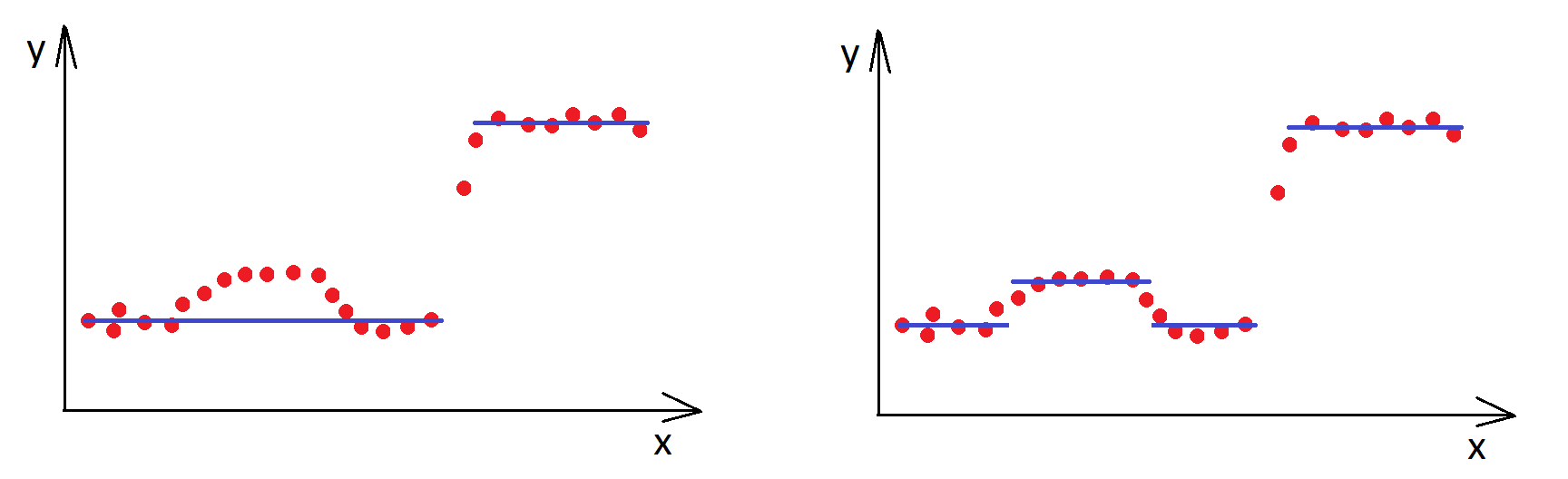
Najczęściej do przedstawienia wiedzy wydobytej z danych używa się reguł logicznych, gdyż jest to forma opisu dobrze zrozumiała dla człowieka. Same reguły mogą mieć rozmaitą postać, nie koniecznie „jeżeli… to…”. Również istnieje wiele sposobów ich pozyskiwania. Zarówno postać reguł jak i sposób ich otrzymywania powinny być dostosowane do konkretnego zagadnienia. Zgłębienie tego tematu będzie celowe zarówno dla osób, którzy jedynie będą korzystać   
z takich reguł, jak i dla specjalistów tworzących takie systemy. W pierwszym przypadku użytkownicy zrozumieją zasadę działania takich systemów, co pozwoli im przekonać się do ich stosowania oraz będą wiedzieć, jak rozmawiać z twórcami tych systemów, by spełnili ich oczekiwania. W drugim przypadku wykład dostarczy cennych informacji osobom, które będą chciały się zająć tworzeniem takich systemów. Oczywiście, żeby takie systemy tworzyć trzeba wiedzieć więcej, niż to można przekazać w ciągu godzinnego wykładu, nie mniej jednak zawsze najważniejszymi i zarazem najtrudniejszymi sprawami jest po pierwsze uświadomienie sobie, że coś w ogóle można zrobić, a po drugie zdobycie ogólnego rozeznania i zrozumienie idei rozwiązań. Mając takie rozeznanie i wiedząc, czego szukać, samodzielne opracowanie szczegółów implementacyjnych jest już znacznie prostsze.

Wykorzystanie wiedzy zawartej w danych jest nam potrzebne w dwóch podstawowych celach: poznanie zjawiska i uzasadnienie decyzji. Poznanie zjawiska, czyli np. co wpływa na to, że materiał podczas produkcji robi się kruchy, albo jakie czynniki wpływają na to, że pacjent będzie dobrze reagować na dane lekarstwo. Często wynik danego procesu jest uwarunkowany wieloma czynnikami. Najczęściej występujące uwarunkowania są znane inżynierowi, czy lekarzowi, jednakże rzadziej występujące bywają nieznane. Używając odpowiednich metod można je odkryć na podstawie dostępnych danych. W pracach o charakterze naukowym, innowacyjnym, doświadczalnym najczęściej dopiero staramy się proces odpowiednio zaprojektować i tu już niezbędnym bywa odpowiednia analiza otrzymanych danych doświadczalnych, bo brak nam jeszcze odpowiedniego doświadczenia, na podstawie którego moglibyśmy znać zestawy czynników decydujących o powodzeniu procesu.

Uzasadnienie decyzji jest potrzebne gdy inteligentny system proponuje rozwiązanie, które ma zostać zastosowane. Wówczas system powinien uzasadnić swoją decyzję, aby być wiarygodnym dla ludzie, którzy mają mu zaufać. Jest to szczególnie istotne w przypadkach, gdy decyzja systemu jest inna, niż wiedza, czy intuicja człowieka, który ma ją wdrożyć. W takiej sytuacji człowiek na ogół zignoruje system, choćby to system miał rację i wdroży swoją własną decyzję. Jednakże, jeśli system tą decyzję dobrze uzasadni, to jest w stanie przekonać człowieka do swojej decyzji. Co więcej, jest też w stanie nauczyć człowieka nowej informacji.

Systemy inteligentne mogą spełniać różne zadania, a w szczególności wyznaczać wartości dyskretne, wyznaczać wartości ciągłe, obliczać prawdopodobieństwo różnych zdarzeń, grupować razem podobne przypadki. Na wykładzie skoncentrujemy się na dwóch pierwszych przypadkach: gdy zadaniem systemu jest regresja (wyznaczenie konkretnej wartości wielkości liczbowej)   
i klasyfikacja (wyznaczenie konkretnej wartości wielkości symbolicznej/dyskretnej/logicznej). Sposób, w jaki system prezentuje wiedzę człowiekowi musi być dostosowany do możliwości percepcyjnych człowieka, czyli możliwie prosty – czym prostszy, tym lepiej. W zagadnieniu poznania zjawiska można sobie jeszcze pozwolić na trochę bardziej złożony opis, ale przy uzasadnieniu decyzji opis musi być na tyle prosty, by człowiek to szybko zrozumiał. Podstawową formą opisu wiedzy są reguły logiczne. Reguły te mogą być w różnej postaci, np. w postaci jeżeli… to, w postaci równania, w postaci reguł typu N of M (co najmniej N przesłanek z M musi zachodzić), w postaci reguł rozmytych (operujących pojęciami typu „mało”, „dużo”), w postaci reguł prototypowych. Czasami bywa, że rzeczywiste zjawisko jest nieco bardziej złożone i nie da się go całkowicie odwzorować w postaci prostych reguł. Wówczas należy znaleźć pewien kompromis między prostotą, a dokładnością opisu.

Wiedza wydobyta z danych powinna opisywać zjawisko, którego dane dotyczą w sposób zrozumiały dla człowieka. Najczęściej przedstawia się tą wiedzę w formie reguł, a proces jej pozyskiwania nazywa również ekstrakcją reguł logicznych. Właśnie taką formą opisu wiedzy zajmiemy się w tej prezentacji. Zadaniem opisu zjawiska przy pomocy reguł logicznych jest to, aby opisać zjawisko w procy i zrozumiały sposób, jednocześnie z możliwie największa dokładnością.

****

Rys. 1. Kompromis między zrozumiałością a dokładnością reguł logicznych.

To, czy ważniejsza jest prostota opisu, czy jego dokładność zależy od specyficznych wymagań   
w każdym przypadku. Większość metod ekstrakcji reguł pozwala na określanie za pomocą pewnych parametrów tego na ile ważna jest dokładność, a na ile prostota otrzymanego opisu. Pozyskiwanie reguł logicznych z dobrego modelu opisującego dane może pozwolić na lepszą reprezentację danych, niż same dane, gdyż model pozwala odfiltrować szum i zakłócenia.

1. **Formy reguł logicznych**

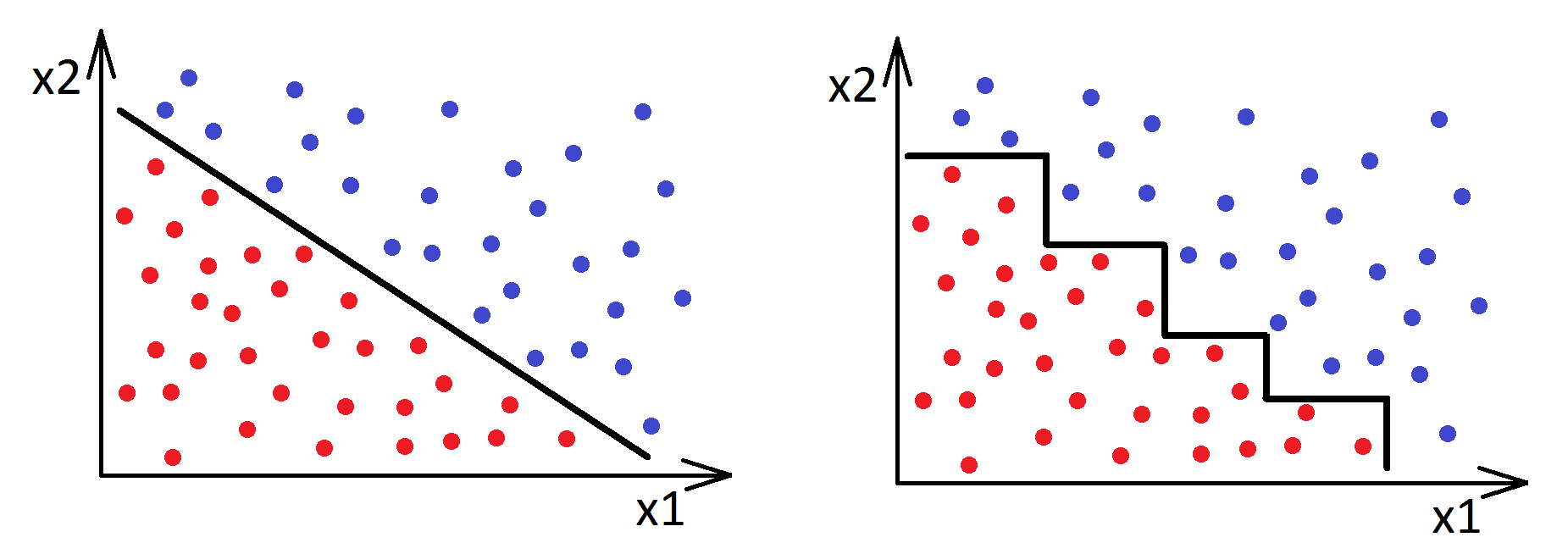
**Reguły ostre (crisp rules)**

Najczęściej stosowanymi regułami są proste reguły ostre typu *jeżeli…to (if…then)* przedstawiane   
w formie: Jeżeli [warunki] to [decyzja]

Większość algorytmów ekstrakcji reguł generuje reguły, których warunki zwane też przesłankami opisują oddzielne, nie nachodzące na siebie przestrzenie. Przesłanki tych reguł są z reguły prostopadłe do osi atrybutów (Rys. 2 po prawej).

**Reguły skośne (oblique rules)**

Granice decyzyjne przesłanek reguł skośnych są skośne do osi atrybutów, co w niektórych przypadkach lepiej opisuje rozkład danych np.*if a\*x1 + b\*x2 < c then Class=1*(Rys. 2 po lewej).

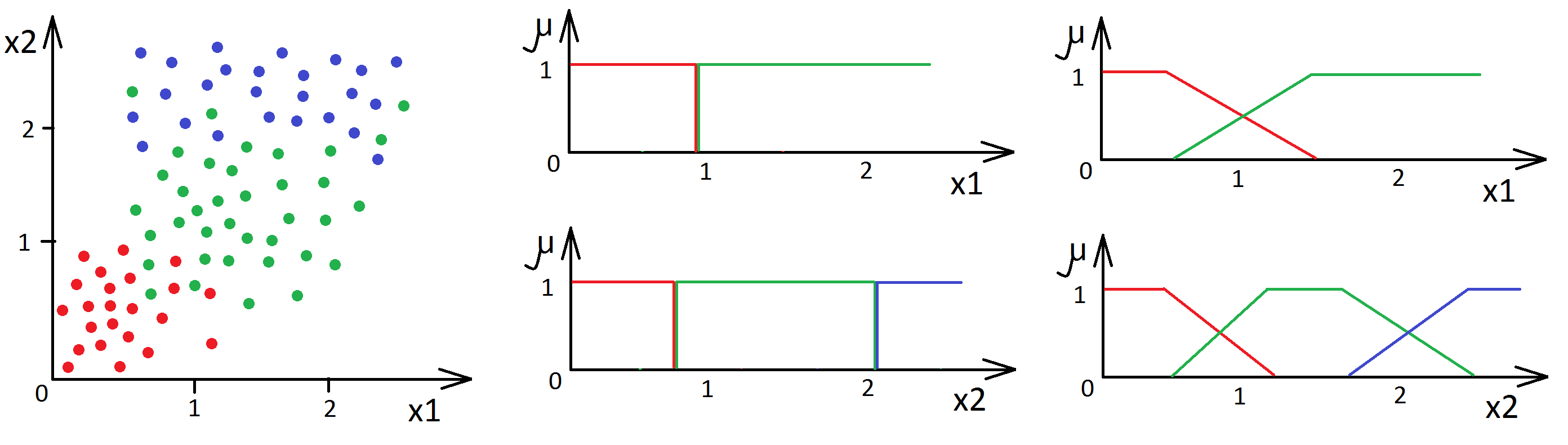


Rys. 2. Rozkład danych, dla opisu którego warto zastosować reguły skośne. Kolor oznacza klasę.

**Reguły rozmyte (fuzzy rules)**

W regułach rozmytych (Rys. 3 po prawej) istnieje pewna funkcja przynależności, która mówi o tym na ile dany przypadek należy do danego przedziału, np. na ile jest duży. Zaprezentowane na rys. 3 funkcje przynależności są trapezowe, ale mogą być też inne, np. gaussowskie. Zwykłe ostre reguły można by traktować jako reguły o prostokątnej funkcji przynależności z dwoma tylko możliwymi stopniami przynależności: 0 i 1. Natomiast dla reguł rozmytych stopień przynależności μ może być określony dowolną liczbą rzeczywistą z przedziału od 0 do 1. Reguły rozmyte są w wielu przypadkach naturalnie używane przez człowieka w jego naturalnym sposobie rozumowania, np. podczas kierowania samochodem: „jeżeli prędkość jest duża, a odległość do przeszkody mała to hamuj mocno”. Logika rozmyta znajduje więc zastosowanie nie tylko w generowaniu reguł, ale także w budowie sterowników różnych urządzeń i pojazdów, naśladując działanie człowieka sterującego nimi. Reguły można też wykorzystać do przedstawienia prawdopodobieństwa poszczególnych przypadków, tj. prawdopodobieństwa przynależności wektorów do poszczególnych klas. Reguły rozmyte operują na atrybutach ciągłych, w przypadku atrybutów symbolicznych wymagają dodatkowych operacji.

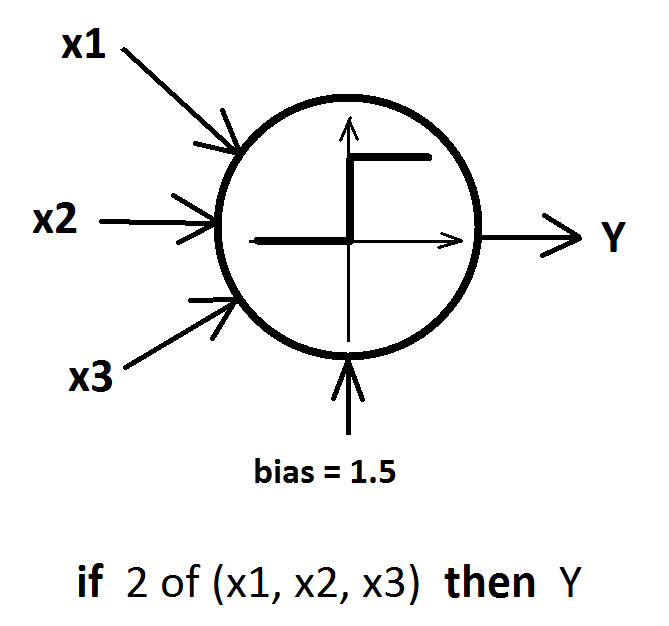
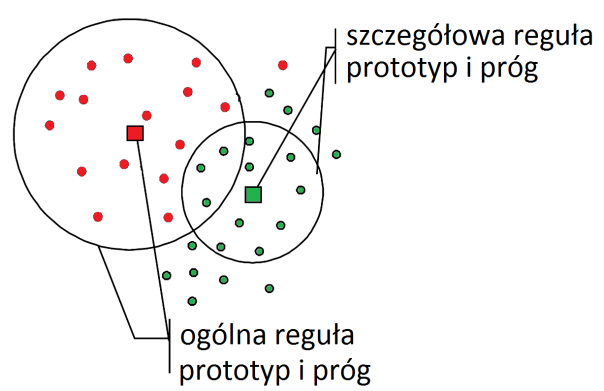
Załóżmy, że mamy dwa atrybuty x1 i x2. Stopień przynależności μ danego wektora do danego klastra reprezentującego klasę może być określony na przykład jako odwrotność jego odległości od środka tego klastra, albo np. jako funkcja trapezowa, jak na rys. 3. Zależy to, jak zdefiniujemy funkcję przynależności. Różne operatory zwane T-normami i S-normami są używane do wyznaczania iloczynu i części wspólnej zbiorów rozmytych (odpowiedniki AND i OR w systemach reguł ostrych). Systemy rozmyte i neuronowo-rozmyte to obszerna dziedzina wiedzy wychodząca poza zakres niniejszego wykładu i godna oddzielnej prezentacji.



Rys. 3. Porównanie reguł ostrych (środkowa kolumna) i rozmytych (po prawej). Kolor oznacza klasę.

**Reguły N-of-M**

Reguła N-of-M (N z M) jest spełniona, gdy zachodzi co najmniej N z możliwych M przesłanek. Reguły N-of-M realizują neurony w sieciach neuronowych typu MLP jeśli sygnały wejściowe są binarne, a funkcja transferu jest skokowa. W przypadku sigmoidalnej funkcji transferu i sygnałów ciągłych realizują one pewną „rozmytą” wersję tej reguły.

 ****

Rys. 4. Po lewej: istota reguł prototypowych, po prawej: neuron generujący regułę N-of-M.

**Reguły prototypowe**

Opis przypadków może być też oparty nie tylko na wartościach poszczególnych atrybutów, jak   
w poprzednich regułach, lecz także na ich podobieństwie do przypadków wzorcowych (prototypów). Ludzie podejmują decyzje bardzo często na podstawie podobieństwa do innych sytuacji zakładają, że skoro dane czynności są odpowiednio podobne to i wynik musi być podobny do znanych już przypadków. Można wyróżnić dwa rodzaje reguł prototypowych: reguły oparte na najbliższych sąsiadach, czyli na algorytmie k-NN, które klasyfikują dany przypadek do tej samej grupy, jak większość jego sąsiadów oraz reguły oparte na odległości progowej, gdzie każdy prototyp ma swoją strefę wpływu, jeśli dany przypadek znajduje się bliżej prototypu niż odległość progowa, to zostaje zaklasyfikowany do grupy (klasy) tego prototypu. Sposób wyznaczania odległości, który wpływa zarówno na dokładność, jak i złożoność reguł prototypowych. Stosuje się różne miary odległości począwszy od najprostszej euklidesowej, aż do miar heterogenicznych.

**Reguły opisane równaniami**

Reguły opisane równaniami (jednym lub kilkoma) wykorzystywane są zwłaszcza w zagadnieniach regresji. Przykładowo opis postać reguł może być następującą postać: y = x1 + 0.25 \* x22.

**First Order Rules – Inductive Logic Programming**

Tym się tu nie będziemy zajmować, ale wspomnę dla komplementarności prezentacji, że istnieje coś takiego jak reguły pierwszego rzędu, które charakteryzują się, tym, że mogą opisywać relacje pomiędzy zmiennymi:

np. if Father(x,y) and Female(y) then Dauthter (y,x)

i tu szczególnie język PROLOG dobrze się nadaje do dokonywania operacji na takich regułach.

1. **Dokładność modeli klasyfikacyjnych**

Na ogół czym dany model osiąga większą dokładność, tym jest bardziej złożony i trudniejszy   
w interpretacji. Znalazło to potwierdzenie w licznych eksperymentach, np. opisanych przez Richa Caruana i Alexandra Niculescu-Mizil, którzy porównali dokładność predykcji wielu modeli dla różnych zbiorach danych i otrzymali wyniki przedstawione w skondensowanej formie w poniższej tabeli.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| model | komitety  (średnia) | SVM | sieci neuronowe | k-NN | drzewa decyzyjne |
| średnia dokładność | 86% | 82% | 81% | 76% | 66% |

Jednakże łatwość interpretacji wyników działania tych modeli i łatwość pozyskiwania z nich opisu wiedzy za pomocą reguł logicznych jest uszeregowana w odwrotnej kolejności. Dlatego pojawia się wyzwanie jak otrzymać proste reguły ze skomplikowanych modeli, a zwłaszcza z komitetów.

(Komitet to zbiór wielu modeli klasyfikacyjnych, który podejmuje ostateczną decyzję uwzględniającą wyniki poszczególnych modeli korzystając z określonego algorytmu;   
w najprostszym przypadku algorytmem tym może być zwykłe głosowanie.)

Naukowcy na różne sposoby próbowali ten problem rozwiązać. Ogólnie można wyróżnić trzy grupy metod:

1. metody dekompozycyjne, czyli wykorzystujące wewnętrzne parametry standardowego modelu, np. budujące reguły w oparciu o wartości poszczególnych wag sieci neuronowej.
2. metody również dekompozycyjne, ale operujące na specjalnie zmodyfikowanym modelu, tak, by ułatwić pozyskiwanie reguł, np. na sieci neuronowej, której wagi mogą przyjmować jedynie wartości -1, 0 lub 1.
3. metody pedagogiczne, które traktują model jak czarną skrzynkę, wykorzystując jedynie wynik jego predykcji zamiast oryginalnych danych i w oparciu o takie dane tworząc reguły z wykorzystaniem łatwo interpretowalnych metod, np. pokrywania sekwencyjnego lub drzew decyzyjnych.

Metody pedagogiczne mają jednak pewną wadę: generują jedynie reguły, nie pozwalając natomiast zrozumieć, co się dzieje wewnątrz modelu. Tym samym nie dają pełnego zaufania do decyzji modelu, nie ma bowiem całkowitej pewności, że dla nowego przypadku działanie modelu będzie takie samo jak dla przypadków na podstawie których zostały stworzone reguły pedagogiczne.

1. **systemy bezpośrednio generujące reguły**

Sekwencyjne Pokrywanie (Sequential Covering)

Historycznie było to pierwsze metody ekstakcji reguł logicznych, nad którymi pracowani już w latach 1970-tych.

Zaczynając od pustego zbioru reguł, pokrywanie sekwencyjne szuka reguły, która klasyfikuje dużą ilość przypadków jednej klasy. Jeżeli jej dokładność jest powyżej określonego progu, wówczas ta reguła jest dodawana do zbioru już istniejących reguł i algorytm powtarza swoje działanie na pozostałej części zbioru (niepokrytej przez żadną z dotychczasowych reguł). Jeżeli dokładność znajdowanych reguł jest już poniżej tego progu to algorytm kończy swoje działanie.

Jest to powtarzane osobno dla każdej klasy. Za każdym zarem reguła pokrywa tylko przypadki z jednej klasy. Są one nazwane pozytiwnymi, a pozostałe negatywnymi.

Ponieważ znalezione reguły mogą nachodzić na siebie, reguły te są najpierw sortowane wg ich dokładności, a następie prezentowane użytkownikowi. Nowe przypadki są klasyfikowane przez pierwszą regułę, która je opisuje. Opracowane kilka algorytmów pokrywania sekwencyjnego, jak AQ, CN2, RIPPER, różniących się szczegółowymi rozwiązaniami implementacyjnymi.

Z podejściem takim wiąże się jednak kilka problemów: po pierwsze otrzymany zezstaw reguł przeważnie nie pokrywa calej przestrzenii atrybutów. Po drugie poszczególne raguły mogą częsciowo nachodzić na siebie.

Powstaje zatem problem, jak klasyfikować nowe przypadki przez zestaw reguł, zwłaszcze, że etap generowania reguł i etap klasyfikacji, to dwa różne etapy. Przeważnie są stosowane różne schematy głosowania. Wagi przypisane poszczególnym regułom biorącym udzial w głosowaniu są wyznaczanie w oparciu o statystyki wyznaczane na podstawie liczby przypadków pokrywanych przez te reguły, np. liczba przypadków pokrywana przez daną regułę. Biorąc jednak pod uwagę jak że te reguły były wyznaczane tylko na przypadkach jeszcze nie pokrytych przez poprzednie reguły, nie zawsze too odzwierciedla rzeczywistą siłę poszczególnych reguł.

Aby zaradzić tym brakom algorytmów pokrywania sekwencyjnego zaproponowano w późniejszych latach różne udoskonalenia, np.

A New Sequential Covering Strategy for Inducing Classification Rules with Ant Colony Algorithms Fernando E. B. Otero, Alex A. Freitas and Colin G. Johnson (2013)

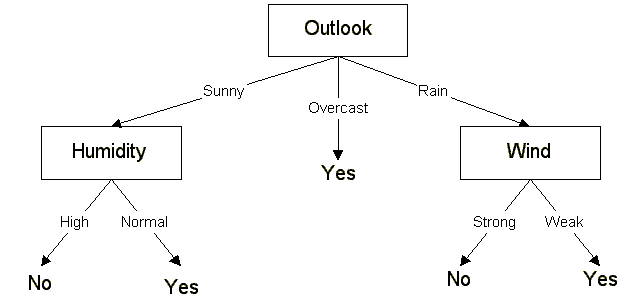
This paper proposes a new sequential covering strategy for ACO classification algorithms to mitigate the problem of rule interaction, where the order of the rules is implicitly encoded as pheromone values and the search is guided by the quality of a candidate list of rules.

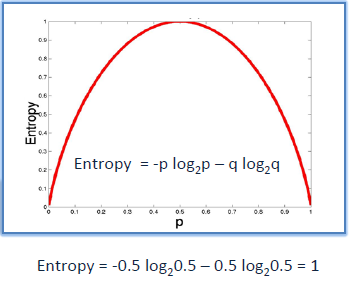
Rule Quality Measures Settings in a Sequential Covering Rule Induction Algorithm - an Empirical Approach

Marcin Michalak∗, Marek Sikora∗†, Łukasz Wróbel†∗

**Drzewa Decyzyjne**

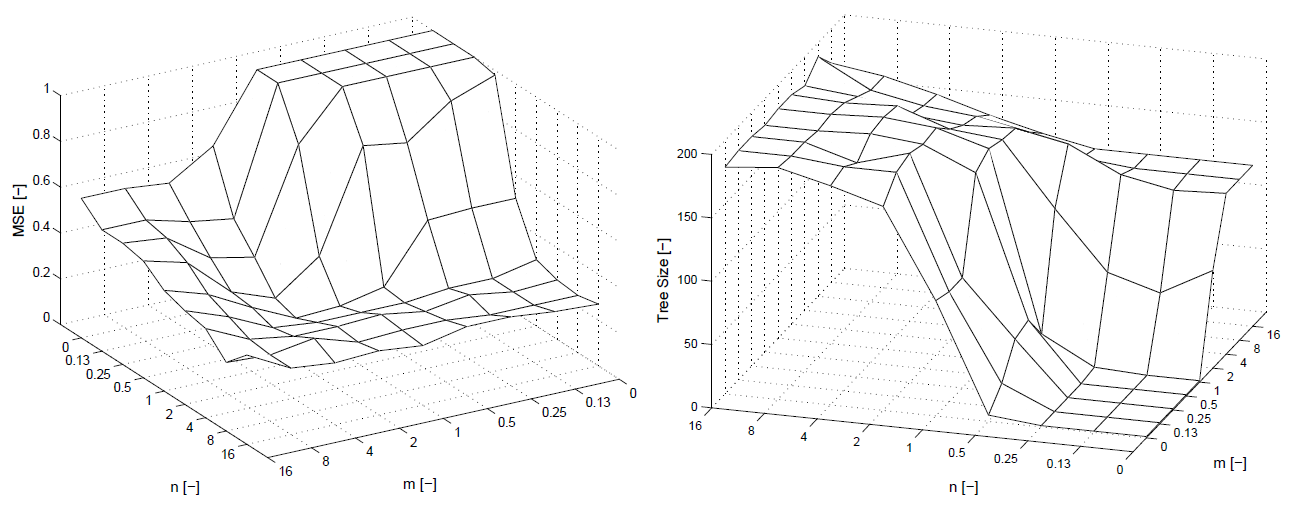
Opracowano wiele różnych modeli drzew decyzyjnych, jednak działają one na wspólnej zasadzie najczęściej poszukując takiego atrybutu i takiego punktu podziału na tym atrybucie, który podzieli istniejący zbiór na dwa podzbiory możliwie najbardziej różniące się między sobą i możliwie najbardziej jednorodne wewnątrz. Poszczególne rozwiązania różnią się głównie stosowaną miarą różnorodności i jednorodności i sposobem jej wyznaczania. Pokażemy sposób generowania reguł otrzymanych dla klasyfikacji z drzewa C4.5 oraz dla regresji z innego drzewa, które daje również możliwość uzyskania reguł skośnych.

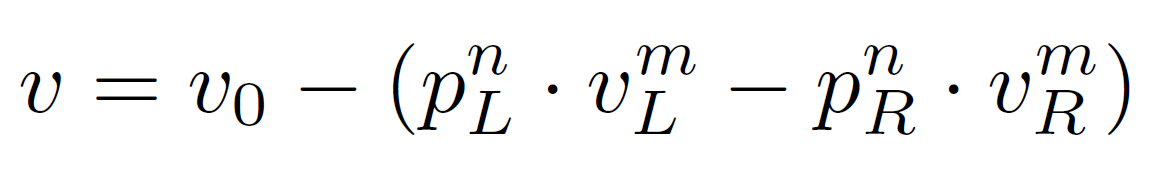


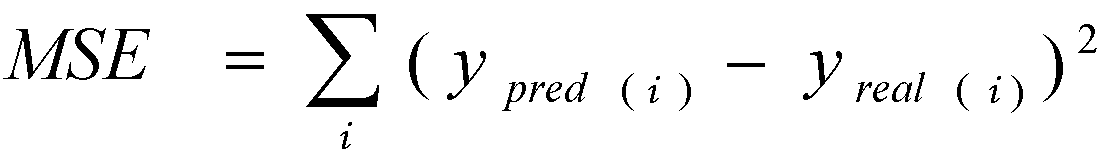


C5.0 – dalsze ulepszenia, ale ta sama idea

CART stands for Classification and Regression Trees (Breiman *et al*., 1984).









Jaki jest problem z drzewami decyzyjnymi?

**Reguły Prototypowe**

Do generowania reguł prototypowych można wykorzystać zarówno zmodyfikowane drzewa decyzyjne (HDT - Heterogeneous Decision Trees), jak i zmodyfikowany algorytm pokrywania sekwencyjnego (PTDL - Prototype Threshold Decision List). Drzewa heterogeniczne (HDT) wykorzystują oba rodzaje testów: tradycyjne oparte na wartości atrybutów, które dzielą przestrzeń płaszczyznami, jak i oparte na podobieństwie do prototypów, które wyodrębniają w przestrzeni hipersfery wokół prototypów. Oprócz możliwości wpływania na postać reguł dają one też często dokładniejsze wyniki klasyfikacji od drzew tradycyjnych. PTDL wykorzystuje pokrywanie sekwencyjne, generuje więc listę reguł nachodzących na siebie, które potem sortuje wg ilości pokrywanych przypadków. Oba algorytmy muszą wyznaczyć najpierw wektory prototypowe, lub wektory te mogą zostać określone przez eksperta z danej dziedziny.

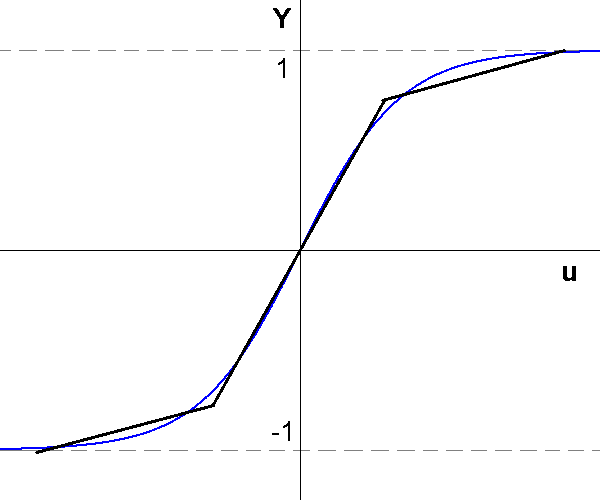
1. **Systemy nie generujące bezpośrednio reguł**

**na przykładzie sieci neuronowych**

**Reguły dekompozycyjne dla klasyfikacji – standardowa sieć MLP**

Opracowano kilka dekompozycyjnych algorytmów ekstrakcji reguł z sieci neuronowych dla zagadnienia klasyfikacji. Omówimy tu algorytm N-of-M ze względu na jego prostotę.   
W algorytmie tym po nauczeniu sieci, dla każdego neuronu dokonujemy klasteryzacji jego wag   
i wagi, które znalazły się w tym samym klastrze zastępujemy średnią wartością dla tego klastra. Przy czym usuwamy klastry, których wartość średnia jest zbyt mała (co do wartości bezwzględnej), ponieważ nie mają one istotnego wpływu na wartość wyjścia neuronu. Następnie zamrażamy wartości wszystkich wag i douczamy sieć, aby zoptymalizować wartości biasów. Tak zmodyfikowane wartości umożliwiają nam ekstrakcję reguł typu N-of-M dla każdego neuronu,   
a więc i dla całej sieci. Przykładowo, jeśli 3 wagi (reprezentujące atrybuty x1, x2, x3) danego neuronu przyjmą wartość równą 2, a bias wartość równą 3, to mamy regułę: „jeżeli co najmniej   
2 z (x1, x2, x3) to aktywuj neuron”, a w przypadku neuronu warstwy wyjściowej: to „jeżeli co najmniej 2 z (x1, x2, x3) to wektor należy do klasy reprezentowanej przez ten neuron” (Rys. 4 po prawej stronie).

**Reguły dekompozycyjne dla klasyfikacji – zmodyfikowana sieć MLP**



Rys. 5. Po lewej: dostosowanie architektury sieci neuronowej do ekstrakcji reguł,

po prawej aproksymacja trój-odcinkowa funkcji transferu tanh.

Jako przykład przedstawiona zostanie sieć regułowo-dyskretyzacyjna (Rys. 5), która operuje na danych dyskretnych (FD1) i symbolicznych (FC1). Dane ciągłe należy zdyskretyzować albo przed uczeniem sieci, albo włączyć wyznaczanie przedziałów dyskretyzacji do uczenia sieci   
w dodatkowym bloku za cechą FC1, jak na rys. 5. Ekstrakcja reguł dokonywana jest na podstawie wartości wag nauczonej sieci. Aby można było otrzymać proste i jednoznaczne reguły, nauczona sieć ma skokowe funkcje transferu neuronów oraz wagi będące liczbami –1, 0 lub +1, dzięki czemu każdy jej neuron realizuje operację M-of-N, a przeważnie jest możliwe uproszczenie reguł zamieniając operatory N-of-M na OR i AND.

Biasy neuronów warstwy ukrytej mają wartości będące liczbą całkowitą ±0.5, zaś biasy warstwy wyjściowej zawsze +0.5. Każdy neuron warstwy ukrytej dedykowany jest do określonej klasy (nie ma pełnych połączeń między warstwą ukrytą a wyjściową), co znacznie upraszcza otrzymane reguły. Na początku uczenia sieć ma losowe rzeczywiste wartości wagi sigmoidalne funkcje transferu, a do funkcji błędu dodany jest człon regularyzacyjny wymuszający ustawianie wag   
i biasów na zadanych wartościach oraz stopniowo zwiększany jest skos sigmoidy, aż stanie się ona praktycznie funkcją skokową.

Pierwszy neuron ukryty klasyfikuje zawsze najwięcej przypadków, jeżeli nie sklasyfikował wszystkich, to dodawany jest kolejny neuron ukryty połączony dodatnią wagą z neuronem wyjściowym, by sklasyfikował pozostałe, Jeśli zaś pierwszy neuron sklasyfikował (przepuścił) także przypadki nie należące do jego klasy, to dodawany jest kolejny neuron połączony ujemną wagą z neuronem wyjściowym, którego zadaniem jest klasyfikacja wyjątków.

**Reguły pedagogiczne dla klasyfikacji**

Przykładową metodą jest TREPAN, który buduje drzewo decyzyjne, które bezpośrednio dostarcza nam reguły, używając sieci neuronowej (ale może to być też dowolny inny algorytm klasyfikacji) do generowania wartości wyjściowych dla tego drzewa. W dwóch aspektach różni się on od podstawowych metod tworzenia drzew decyzyjnych: jeśli w danym węźle jest zbyt mało przypadków, to generuje on dodatkowe przypadki, do podziału węzła nie używa pojedynczego atrybutu, tylko wyrażenia N-of-M. Na podobnej zasadzie działa kilka innych algorytmów pedagogicznych, różniąc się tylko sposobem budowania i ostateczną postacią drzewa decyzyjnego.

**Reguły dekompozycyjne dla regresji – standardowa sieć MLP**

Zaprezentowane zostanie rozwiązanie przedstawione o nazwie REFANN (Rule Extraction from Function Approximation Neural Networks) polegające na zastąpieniu sigmoidalnej funkcji transferu jej aproksymacją trzema prostymi odcinkami (Rys. 5 po prawej). Ponieważ jest to zagadnienie regresji, więc funkcja wyjścia neuronu warstwy wyjściowej jest liniowa. W wyniku tego otrzymujemy dość spory na ogół układ równań liniowych opisujący działanie całej sieci. Jednakże w przypadku ekstrakcji reguł dla pojedynczego wektora układ ten się znacznie upraszcza.

**Reguły dekompozycyjne dla regresji – zmodyfikowana sieć MLP**

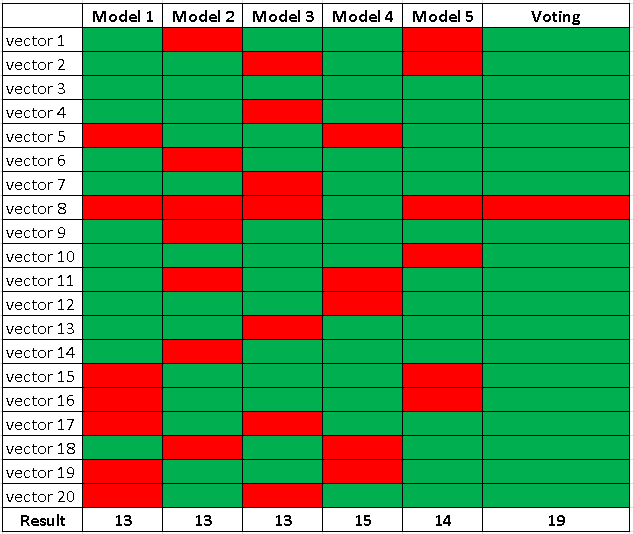
Podobna modyfikacja sieci neuronowej jak dla klasyfikacji może być zastosowana do regresji. Przy czym w tym wypadku również wielkość wyjściowa zostajezdyskretyzowana i problem regresji zostaje zamieniony na problem klasyfikacji wieloklasowej. Celem zwiększenia dokładności można zastosować podobne rozwiązanie, jak w liściach omawianego drzewa regresyjnego, czyli zastosowanie regresji liniowej na danych wchodzących osobno do każdego neuronu wyjściowego i ograniczenie jej do jednego lub dwóch wymiarów pochodzących od atrybutów o największej lokalnej korelacji z wyjściem. W porównaniu z poprzednim rozwiązaniem otrzymujemy mniej skomplikowany układ równań (duża część równań się w praktyce łączy tworząc znacznie mniejszą ilość) i prostsze reguły, łatwiejsze do zrozumienia o porównywalnej dokładności.

**Reguły pedagogiczne dla regresji**

W przypadku regresji można używać te same metody pedagogiczne, co dla klasyfikacji po dokonaniu dyskretyzacji danych, lub ich inne podobnie działające metody różniące się tym, że budują drzewo dostosowane do regresji, np. skośne z ograniczoną liczbą atrybutów na których jest dokonany podział w węźle.

1. **Generowanie reguł z komitetów**

Opłaca się generować erguly, bo komitety mają największą dokładność.



Nie ma problemu, żeby do ekstrakcji reguł z komitetów klasyfikatorów użyć metod pedagogicznych i tak się rzeczywiście postępuje. Natomiast chcąc użyć reguł wygenerowanych przez metody dekompozycyjne, sprawa się nieco komplikuje. Niemniej jednak powzięto pewne próby stworzenia metod dekompozycyjnych dedykowanych komitetom, a zwłaszcza komitetom sieci neuronowych. Szły one w dwóch kierunkach: ograniczenia liczby sieci neuronowych   
w komitecie np. do trzech oraz zastosowania zmodyfikowanych sieci jak DIMMLP (Discretized Interpretable MLP).

Przedstawiona zostaną także rozwiązania zmierzające do możliwości otrzymywania reguł dekompozycyjnych z komitetów cechujących się prostotą porównywalną z regułami otrzymanymi   
z pojedynczych modeli i dokładnością oddającą dokładność komitetów. W tym celu zostaje wybrany początkowo jeden najlepszy model z całego komitetu i zostają mu przydzielone wszystkie wektory, które prawidłowo klasyfikuje. Następnie zostają szukane kolejne modele, które   
w największej części poprawnie klasyfikują wektory błędnie sklasyfikowane przez pierwszy model, potem przez dwa pierwsze. Podejście jest analogiczne do selekcji cech metodą przeszukiwania   
w przód. Natomiast należy wypracować prosty lecz skuteczny mechanizm decydujący o tym, które wektory będą opisywane wg reguł którego klasyfikatora w przypadku nowych wektorów i prace   
w tym zakresie zostaną również przedstawione w prezentacji.

Guido Bologna Is it worth generating rules from neural network ensembles?, Journal of Applied Logic, 2004

*Multi-Layer Perceptron* (DIMLP) is a special neural network model for which

symbolic rules are generated to explain the knowledge embedded within the connections

and the activations of neurons.