Odpowiedzi na pytania do egzaminu z APU
Wykłady 1–3

1 Wykład

1. Definicja procesów uczenia
   Nie ma jednej definicji procesów uczenia się:

   - "Uczenie się oznacza zmiany w systemie, które mają charakter adaptacyjny w tym sensie, że pozwalają systemowi wykonać za następnym razem takie same zadanie lub zadania podobne bardziej efektywnie" - Herbert Simon (1983)
   - "System uczący się wykorzystuje zewnętrzne dane empiryczne w celu tworzenia i aktualizacji podstaw dla udoskonalania działania na podobnych danych w przyszłości oraz wyrażania tych podstaw w zrozumiałej i symbolicznej postaci" - Donald Miche (1991)
   - "Uczenie sie to konstruowanie i zmiana reprezentacji doświadczanych faktów. W ocenie konstruowanych reprezentacji bierze się pod uwagę:
     (a) wiarygodność - określa stopień w jakim reprezentacja odpowiada rzeczywistości;
     (b) efektywność - charakteryzuje przydatność reprezentacji do osiągania danego celu;
     (c) poziom abstrakcji - odpowiada zakresowi szczegółowości i precyzji pojęć używanych w reprezentacji; określa on tzw. moc opisową reprezentacji. Reprezentacja jest rozumiana jako np. opisy symboliczne, algorytmy, modele symulacyjne, plany obrazy." - Ryszard Michalski (1986)
   - Elementem wspólnym tych definicji są: Wejście (dane empiryczne), miara oceny (Zmiany i poprawa działania) oraz postulat zdobywania wiedzy, reprezentowania jej wewnątrz systemu i stosowania jej do wykonania zadania (nacisk na zrozumiałość reprezentacji)

2. Przykłady problemów rozwiązywanych przez systemy uczące się

   - Uczenie się rozpoznawania mowy
   - Uczenie się kierowania pojazdem (np. ALVINN)
   - Uczenie się klasyfikacji obiektów astronomicznych (NASA Sky Survey)
   - Uczenie się rozgrywania pewnych gier
   - Uczenie się rozpoznawania chorób na podstawie symptomów
   - Uczenie się rozpoznawanie pisma na podstawie przykładów
   - Uczenie się klasyfikowania tekstów do grup tematycznych
   - Uczenie się aproksymacji nieznanej funkcji na podstawie próbek
   - Uczenie się odnajdowania drogi w nieznanym środowisku
   - Automatyczne odkrywanie zależności funkcyjnych w danych
   - Przewidywanie trendów w danych finansowych

3. Motywacje dla budowy systemów uczących się

   - Zadania eksploracji i analizy danych, gdzie duże rozmiary zbiorów danych uniemożliwiają ich analizę w sposób nieautomatyczny (np. ekonomiczne lub medyczne bazy danych)
   - Środowiska gdzie system musi się dynamicznie dostosowywać do zmieniających się warunków (np. systemy sterowania)
   - Problemy które są złożone, trudne do opisu i często nie posiadają wystarczających modeli teoretycznych albo ich uzyskanie jest bardzo kosztowne lub mało wiarygodne.

4. Klasyfikacja metod maszynowego uczenia się

   - Uczenie indukcyjne - na podstawie znanych faktów i obserwacji tworzona jest uogólnienie, które próbuje dopasować wszystkie znane fakty do hipotezy
   - "Nabywanie umiejętności" - optymalizacja zastosowania już posiadanej wiedzy w sposób zwiększający jej efektywność

5. Tworzenie modelu uczenia maszynowego

   (a) Przygotowanie danych (zebranie danych, uzupełnienie braków, normalizacja)
   (b) Zdefiniowanie zadania (regresja, klasyfikacja, grupowanie, inne)
   (c) Wybór metody (regresja liniowa, logistyczna, drzewa decyzyjne, sieci neuronowe)
   (d) Strojenie parametrów
   (e) Ocena modelu

6. Język R. Wykonywanie instrukcji
   Sekwencyjne, możliwe wpisywanie kolejnych poleceń z konsoli lub z pliku skryptu o rozszerzeniu *.R. W konsoli występuje znak zachęty `>`. Komentarz tworzony jest przez `#`. Zapisywanie ma postać `zmienna <- wartość`.

7. Korzystanie z pomocy R

   - `help(max)` - klasyczna pomoc
   - `?max` - skrócona wersja
   - `example(max)` - przykłady użycia
   - `RSiteSearch("max function")` - przeszukiwanie forum
   - `apropos("max", mode = "function")` - wyszukiwanie z funkcji z nazwą "max"
   - `data()` - nazwy obiektów w pakiecie
   - `vignette()` - pdf z dokumentacją

8. Zarządzanie obszarem roboczym R

   - `getwd()` - ścieżka do aktualnego katalogu roboczego
   - `setwd("..")` - ustawia nową ścieżkę do katalogu roboczego
   - `ls()` - wyświetla zmienne obszaru roboczego
   - `remove(a, b, c)` - usuwa zmienne a, b, c z obszaru roboczego
   - `history(3)` - 3 ostatnio użyte instrukcje
   - `savehistory("plik.txt")` - zapisuje historię do pliku
   - `loadhistory("plik.txt")` - odczyt historii z pliku
   - `save.image("workspace.RData")` - zapis obszaru roboczego do pliku
   - `save(a, b, c, file = "workspace.RData")` - zapis tylko kilku zmiennych do pliku
   - `load("workspace.RData")` - odczyt obszaru roboczego z pliku

9. Pakiety rozszerzające R

   - `installed.packages()` - lista zainstalowanych paczek
   - `search()` - wyszukiwanie w załadowanych paczkach
   - `install.packages("RWeka")` - instalowanie paczki RWeka
   - `library(RWeka)` - ładowanie paczki RWeka

10. Skalary i wektory R
    Skalary obejmują pojedyncze liczby, łańcuchy znaków i wartości logiczne. Typ każdej zmiennej i wyrażenia można sprawdzić funkcją `class(obj)`. Np. `class(2)` zwróci `[1] "numeric"`. Każda liczba (bez względu czy double czy int) to numeric.
    Operacje na liczbach:

    - +, -, *, / (dodawanie, odejmowanie, mnożenie, dzielenie)
    - %% (modulo)
    - ^(potęga)
    - sqrt(n) (pierwiastek kwadratowy)

    Dla znaków łączenie nie działa z + tylko z funkcją `paste("a", "b", "c", sep=" ")`. Dla typów logicznych operacje porównania są takie same jak w C++ czy Csharp.
    Wektor obejmuje wiele skalarów tego samego typu. Są jednowymiarowe. Do utworzenia wektora używa się funkcji `c(obj, ...)`. Próba utworzenia wektora o różnych typach spowoduje konwersję typów.
    Można utworzyć sekwencję, która będzie również zachowywała się jak wektor. Tworzy się ją przy użyciu operatora `<liczba>:<liczba>` lub funkcji `seq(<liczba>, <liczba>, <krok>)`. Wektory indeksujemy od 1, a nie od 0. Ujemne wartości w indeksie to pobranie wszystkiego **oprócz** ostatnich N. W indekserze można użyć wektora indeksów. Zwróci on wtedy wektor z wartościami o indeksach jakie zostały wskazane. Można też użyć sekwencji do pobierania tych wartości.
    Do sprawdzenia długości wektora używa się `length(vect)`.

11. Ramka danych R
    Ramka danych to powiązanie dwóch wektorów o tej samej długości. Np. imię i wiek. Do jej utworzenia używa się `data.frame(label = vector, label = vector)`. `names(ramka)` pozwala na wyświetlenie nazw kolumn. Konstrukcja `names(ramka) <- c ("abc", "cde")` umożliwia nadpisanie nazw w ramce. Indeksując ramkę indeksujemy wiersze, ale możemy użyć konstrukcji `frame[1, 1]` co zwróci nam dane tylko z pierwszej kolumny. Korzystając z operatora `ramka$abc` można pobrać kolumnę po nazwie. Można indeksować podając warunek: `ramka[ramka$abc < 10]`.
    Dodawanie nowego wiersza możliwe jest przy użyciu funkcji `rbind(old, new)`. Odpowiednio dodawanie kolumny wymaga funkcji `cbind(old, new)`.
    Dodatkowe funkcje które pozwalają uzyskać informacje o ramce:

    - `lenght(ramka)` - Liczba kolumn w ramce
    - `dim(ramka)` - Wymiary ramki
    - `str(ramka)` - Tekstowy opis zawartości ramki
    - `head(ramka, n = 2)` - Wyświetlenie pierwszych 2 wierszy w ramce
    - `tail(ramka, n = 2)` - Wyświetlenie ostatnich 2 wierszy w ramce

12. Przegląd wykresów

- Histogram

```
hist(vector,
        breaks = unique(vector),
        right = FALSE,
        main = "Main title",
        xlab = "Label x",
        ylab = "Label y")
```

- Wykres pudełkowy - wizualizuje minimum, pierwszy kwartyl (0.25), medianę, drugi kwartyl (0.25) i wartość maksymalną

```
boxplot(iris$Sepal.Width,
        main = "Podstawowe staystyki dla szerokosci kwiatu sepal")
```

- Wykres kołowy

```
pie(table(new_iris$Species) / length(new_iris$Species))
```

- Wykres wachlarzowy

```
install.packages("plotrix")
library(plotrix)
fan.plot(percentage, labels = names(percentage))
```

- Wykres słupkowy

```
count <- table(new_iris$Species)
colors <- rainbow(3)
barplot(count,
        main = "Liczebnosc gatunkow",
        ylim = c(0, 50),
        xlab = "Gatunek",
        ylab = "Ilosc",
        col = colors)
```

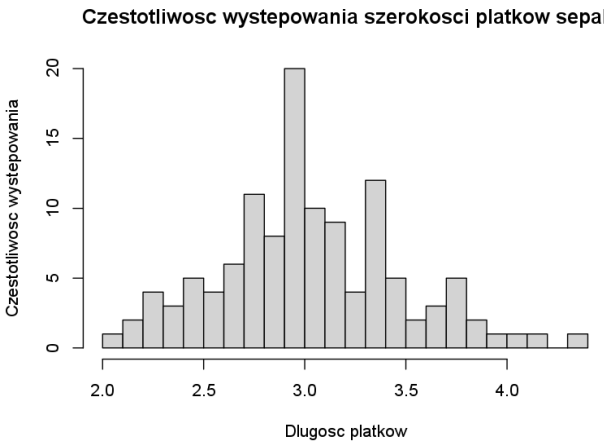

Figure 1: Histogram

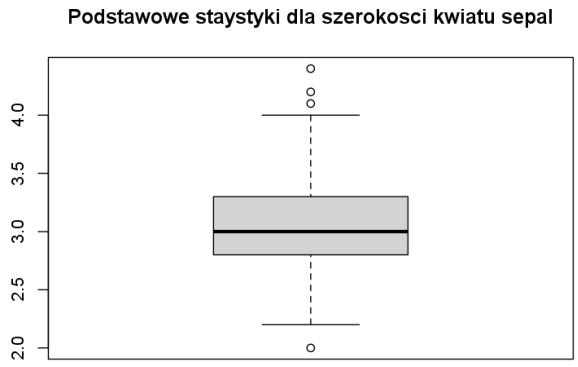

Figure 2: Wykres pudełkowy

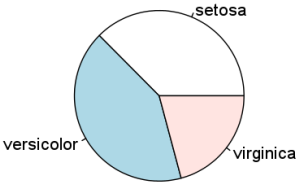

Figure 3: Wykres kołowy

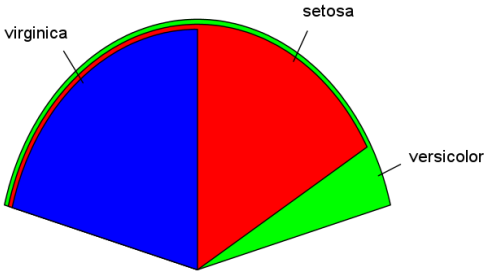

Figure 4: Wykres wachlarzowy

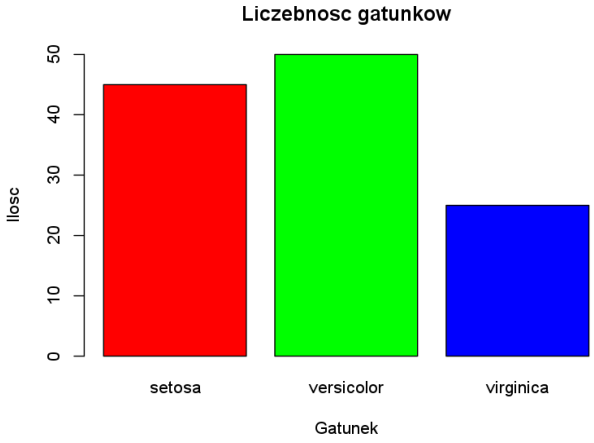

Figure 5: Wykres słupkowy

13. Język R i uczenie maszynowe

- Składniki uczenia maszynowego - szereg konceptów uczenia maszynowego
- Zbiór uczący - doświadczenie w postaci danych. Zbiór może być etykietowany lub nieetykietowany.
- Atrybuty (cechy) - opis obiektów zbioru uczącego
- Instancja - zestaw atrybutów opisujących jeden obiekt
- Model - utworzone na podstawie danych rozwiązanie zadania
  - Geometryczny - oparty o przestrzeń euklidesową. Tworzony przez regresję liniową, logistyczną lub maszynę wektorów wspierających (SVM, SMO)
  - Logiczny - oparty o logiki matematyczne. Zawiera proste reguły w postaci warunków prowadzących do decyzji. Tworzony przez drzewa decyzyjne, JRipper, PART
  - Probabilistyczny - oparty o prawdopodobieństwo warunkowe i regułę Bayesa
  - Hybrydowe
- Ewaluacja - sposoby oceny skuteczności modelu
  - Podział na zbiory uczący i testowy
  - 10-fold cross-validation - 10-krotne powtórzenie podziału zbioru uczącego. Efekt końcowy jest uśrednieniem poprzednich pomiarów.

2 Wykład

1. Analiza eksploracyjna i analiza potwierdzająca
   Analiza potwierdzająca polega na odrzuceniu fałszywych wzorców. Do tego używa się najczęściej metod:

   - testowania formalnego - sprawdzenie zachowania modelu przy użyciu danych nie użytych w analizie eksploracyjnej;
   - teorii prawdopodobieństwa - sprawdzenie czy wzorce w pierwotnej próbce mogły powstać przypadkowo.

   Analiza eksploracyjna to stosowanie różnych technik prowadzących do wykrycia zależności między danymi i powstania modeli. Ludzie mają tendencję do znajdowania fałszywych wzorców, które nie istnieją. Analiza eksploracyjna musi być uzupełniona analizą potwierdzającą.

2. Czym są dane w uczeniu maszynowym? Zbiór danych to tabela w której wiersze odpowiadają za obserwacje zjawisk, a kolumny opisują cechy (atrybuty) tej obserwacji. Dane układające się w formie tabeli nazywa się modelem danych prostokątnych.
   Operacje na takich danych można wizualizować rysunkami. Innym sposobem na uproszczenie zbioru jest podsumowanie liczbowe.

3. Wnioskowanie o typach danych w kolumnach

   - `is.numeric(obj)`
   - `is.character(obj)`
   - `is.factor(obj)` - sprawdza czy wartości są skategoryzowane (zmienne typu factor). Zmienna taka wewnętrznie jest liczbą, ale gdy jest wyświetlana to tłumaczona zostaje na ciąg znaków.

4. Podsumowania liczbowe w R
   `summary(ramka)` - tworzy zestawienie składające się z minimum, kwantyla pierwszego, mediany, średniej, kwantyla trzeciego i wartości maksymalnej.

5. Średnie, mediany i dominanty w R

   - `mean(obj)` - średnia
   - `median(obj)` - mediana
   - dominanty nie da się wyznaczyć wbudowaną funkcją ze względu na brak możliwości porównywania liczb zmiennoprzecinkowych, co jest wymagane definicją.

6. Kwantyle w R

   - `quantile(vec)` - zwraca wartości dla 0, 25%, 50%, 75% i 100%.
   - `quantile(vec, probs=seq(0, 1, by=0.20))` - zwraca wartości dla co 20% od 0 do 100%.

7. Odchylenia standardowe i wariancje w R

   - `var(vec)` - wariancja
   - `sd(vec)` - odchylenie standardowe

8. Eksploracyjne wizualizacje danych Do użycia funkcji `ggplot` potrzeba załadować bibliotekę

- Histogram

```
ggplot(iris, aes(x = Sepal.Length)) +
geom_histogram(binwidth = 0.1) +
labs(x = "Sepal.Length", y = "liczba")
```

- Wykres gęstości (ang. *kernel density estimates* - KDE)

```
ggplot(iris, aes(x = Sepal.Length)) +
geom_density() +
labs(x = "Sepal.Length", y = "liczba")
```

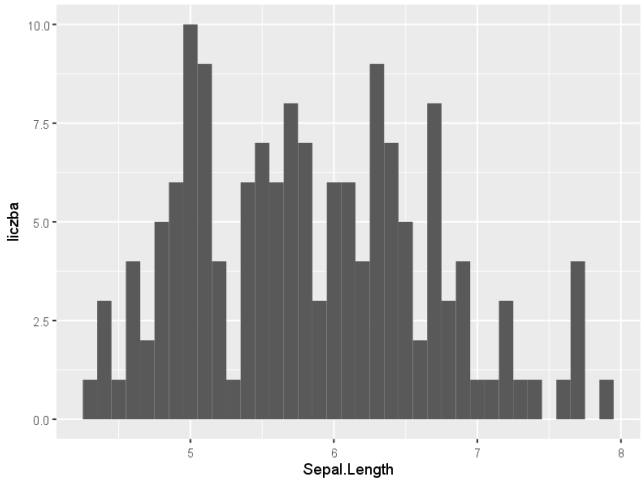

Figure 6: Histogram z ggplot

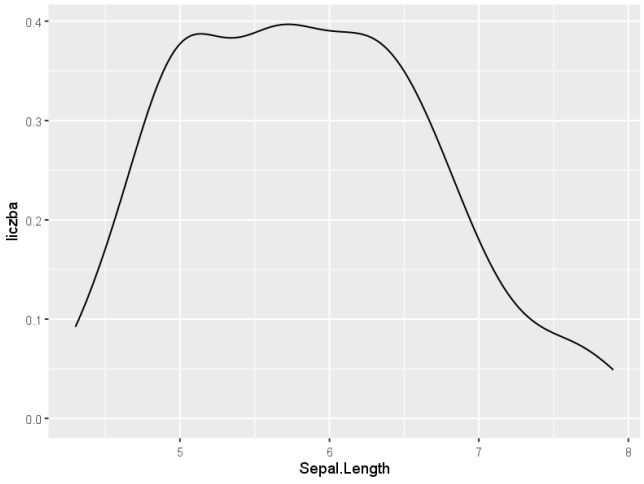

Figure 7: Wykres gęstości

Generowanie rozkładów:

- Normalny - `rnorm(próbki, ymin, ymax)`
- Cauchy - `rcauchy(próbki, ymin, ymax)`
- Gamma - `rgamma(próbki, ymin, ymax)`
- Wykładniczy - `rexp(próbki, ymin, ymax)`

9. Wizualizowanie powiązań pomiędzy kolumnami Wykresy przedstawiające powiązania między atrybutami:

   - Wykres punktowy

     ```
     ggplot(iris, aes(x = Sepal.Length, y = Petal.Length, color = Species)) +
     geom_point() +
     labs(x = "Sepal.Length", y = "Petal.Length", color = "Species")
     ```

   - Wygładzony wykres punktowy

     ```
     ggplot(iris, aes(x = Sepal.Length, y = Petal.Length, color = Species)) +
     geom_point() +
     labs(x = "Sepal.Length", y = "Petal.Length", color = "Species") +
     geom_smooth()
     ```

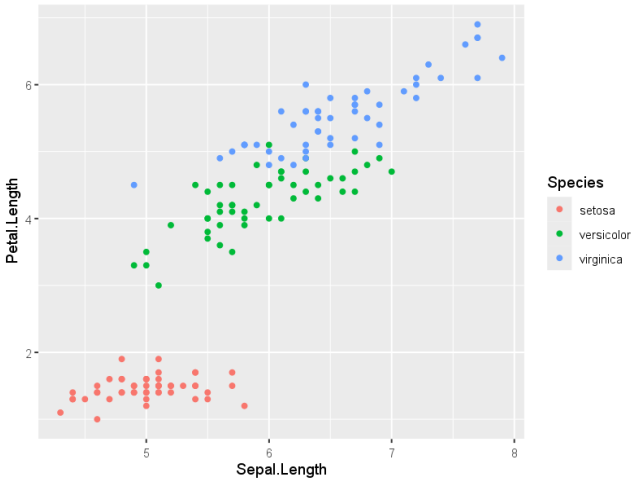

Figure 8: Wykres punktowy

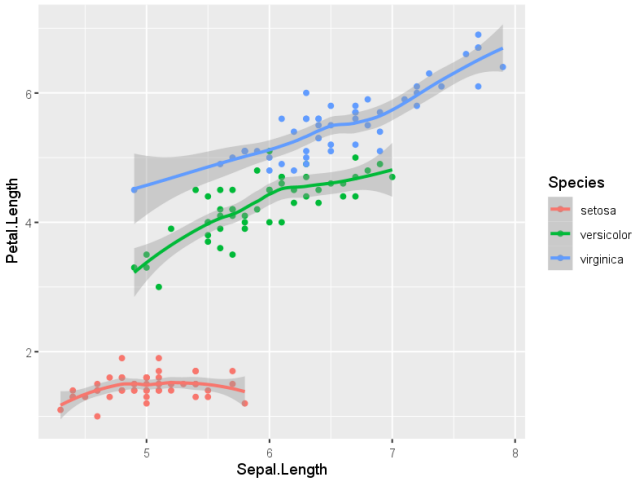

Figure 9: Wykres punktowy wygładzony

10. Klasyfikacja; Zdefiniowanie zadania

    - Klasyfikacja - znajdowanie odwzorowywania zbioru danych w zbiór predefiniowanych klas
    - Klasyfikacja to metoda uczenia z nadzorem. Głównym celem klasyfikacji jest zbudowanie formalnego modelu zwanego **klasyfikatorem** dla każdej klasy.
    - Danymi wejściowymi są krotki będące listą opisywanych atrybutów. Wartości atrybutu decyzyjnego decydują o przydziale do konkretnej kategorii.

11. Trening i testowanie klasyfikacji

    - Trening - opiera się o podział danych na zbiór uczący i testowy. Efektem procesu trenowania jest powstanie modelu zawierającego reguły klasyfikacyjne.
    - Testowanie klasyfikacji - opiera się o sprawdzenie poprawności modelu w oparciu o dane testowe.

12. Kryteria porównawcze metod klasyfikacji

    - Dokładność modelu - procent przykładów poprawnie sklasyfikowanych przez model
    - Efektywność (szybkość) - koszt obliczeniowy treningu i wykorzystania klasyfikatora
    - Odporność modelu - zdolność rozpoznawania zaszumionych lub niepełnych danych
    - Skalowalność - możliwość przetwarzania dowolnie dużych zbiorów
    - Interpretowalność - odnosi się do stopnia w jakim konstrukcja klasyfikatora pozwala na zrozumienie mechanizmu klasyfikacji

13. Metody klasyfikacji

    - Klasyfikacja poprzez indukcję drzew decyzyjnych
    - Klasyfikatory Bayes'owskie
    - Sieci Neuronowe
    - Analiza statystyczna
    - Metaheurystyki (np. algorytmy genetyczne)
    - Zbiory przybliżone
    - k-NN – k-najbliższych sąsiadów

14. Drzewa decyzyjne
    To struktury drzewiaste, gdzie od korzenia odchodzą tylko dwie gałęzie. Każdy z węzłów zawiera test od którego zależy czy wybrać lewą gałąź czy prawą. Liść zawiera odpowiedź.

15. Funkcje testu w celu konstruowania drzew decyzyjnych

    - Univariate tree - testy operują na pojedynczym parametrze
    - Multivariate tree - testy biorą pod uwagę wiele atrybutów

16. Konstrukcja drzew decyzyjnych
    Budowę drzewa decyzyjnego można opisać korzystając z 3. funkcji pomocniczych:

    - Warunek stopu - zatrzymaj tworzenie drzewa gdy zbiór jest pusty lub zawiera obiekty wyłącznie jednej klasy decyzyjnej lub żaden test nie może podzielić zbioru na dwa mniejsze;
    - Wyznaczenie etykiet - wybór najliczniej reprezentowanej klasy w zbiorze;
    - Wybór testu - heurystyka oceniająca testy. Funkcja wyboru musi być monotoniczna oraz przyjmować niskie wartości jeżeli różnorodność zbioru jest niska.

    Lepsze drzewo to takie, które jest mniejsze. Stosuje się do tego przycinanie, czyli zastępowanie liściem lub mniejszym poddrzewem większego poddrzewa.

17. Problem brakujących wartości przy konstruowaniu drzew decyzyjnych
    Możliwe rozwiązania

    - Wypełnienie nieznanych wartości atrybutu najczęściej występującą wartością w zbiorze obiektów związanych z aktualnym węzłem
    - Wypełnienie nieznanych danych średnią ważoną wyznaczoną na zbiorze wartości
    - Zatrzymanie procesu klasyfikacji i zwrócenie większościowej etykiety dla liścia
    - Wypełnienie nieznanej wartości według jednej z heurystyk przy konstruowaniu drzewa
    - Połączenie wszystkich pozostałych możliwości i zwrócenie rozkładu prawdopodobieństwa

18. Analiza ROC jakości klasyfikacji. Krzywe ROC
    Analiza ROC to porównanie wyników pozytywnie pozytywnych (PP lub TP), fałszywie pozytywnych (FP, FP), pozytywnie negatywnych (PN, TN) i fałszywie negatywnych (FN, FN). FP to zjawiska niepoprawnie sklasyfikowane jako negatywne, a FN to zjawiska niepoprawnie sklasyfikowane jako pozytywne.

19. Czułość, a specyficzność klasyfikacji binarnej

    - Czułość -
    $$sensitivity = \frac{TP}{TP + FP}$$
    - Specyficzność -
    $$specificity = \frac{TN}{FP + TN} = 1 - \frac{FP}{TN + FP}$$

- Dokładność -
$$precision = \frac{TP}{TP + FP}$$

20. Konstruowanie krzywych ROC
Każdy punkt na wykresie to wartość czułości i specyficzności. Wartość ROC to pole pod wykresem. Najgorszym rezultatem będzie ROC = 0.5 (czułość i specyficzność są sobie równe), a najlepszym ROC = 1.0 (czułość zawsze o wartości 100%, a specyficzność dowolna).

21. Pakiet ROCR

```
install.package("ROCR")
library(ROCR)
perf <- performance(frame, optX, optY)
plot(perf)
```

opt może przyjąć wartość:

- acc - dokładność
- err - error rate
- fpr - false positive
- tpr, sens, rec - true positive
- fnr - false negative
- tnr, spec - true negative
- ppv, prec - positive prediction value

3 Wykład

1. Wieloatrybutowe problemy decyzyjne
problemy wyznaczenia takiej opcji decyzyjnej spośród skończonego (liczbowo niedużego) zbioru dopuszczalnych opcji, która zapewni jak najlepsze osiągnięcie wszystkich rozpatrywanych przez decydenta kryteriów – atrybutów.
Inne sformułowania:

- Problem sortowania wieloatrybutowego - przyporządkowanie opcji do z góry określonej kategorii
- Problem przyporządkowywania wieloatrybutowego - podział opcji na klasy jednakowo dobrych

Szkoły rozwiązywania problemów:

- Amerykańska - metody teorii użyteczności wieloatrybutowej - AHP (Proces analitycznej hierarchizacji)
- Europejska - metody relacji przewyższania - ELECTRE

2. Proces analitycznej hierarchizacji problemu decyzyjnego
To systematyczna procedura oparta na hierarchicznym przedstawieniu elementów problemu decyzyjnego, a dokładnie takich, które określają jego istotę. Metoda polega na dekompozycji problemu do możliwe jak najprostszych składowych, a następnie na przetwarzaniu sekwencji ocen opartych o porównania parami.

3. Kroki rozwiązywania problemu AHP

(a) dekompozycja i przedstawienie problemu w formie hierarchicznej
(b) określenie/zdefiniowanie ocenianych atrybutów
(c) specyfikacja opcji decyzyjnych i ostateczne graficzne przedstawienie hierarchii
(d) tworzenie macierzy porównań parami

(e) obliczanie priorytetów

4. Podstawy wieloatrybutowej teorii użyteczności
5. Agregacja ocen z wykorzystaniem macierzy porównań parami
6. Skala preferencji względnej
7. Ocena spójności macierzy porównań parami
8. Krok V – obliczenie priorytetów AHP
9. Obliczanie przybliżonego wektora własnego macierzy porównań parami
10. Inne metody rozwiązywania problemu AHP
11. Przykłady zastosowań metody AHP