Klasyfikatory w uczeniu maszynowym

Joanna Jaworek-Korjakowska

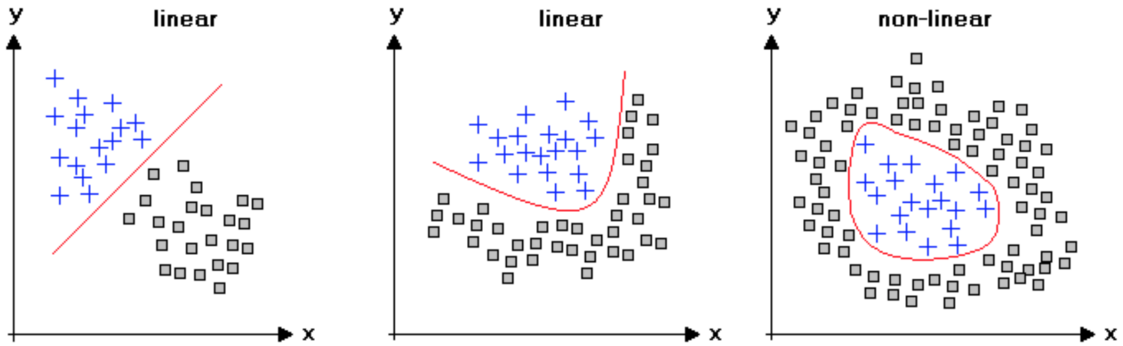
WEAIIB, Katedra Automatyki i Robotyki, ISS

2018/2019

Algorytm k-najbliższych sąsiadów (model leniwego uczenia)

Algorytm k-najbliższych sąsiadów

Algotym k-najbliższych sąsiadów (ang. *k-nearest neighbor classifier* - kNN) jest bardzo intuicyjną metodą, która klasyfikuje nieoznaczone przykłady na podstawie ich podobieństwa z przykładami w zestawie treningowym



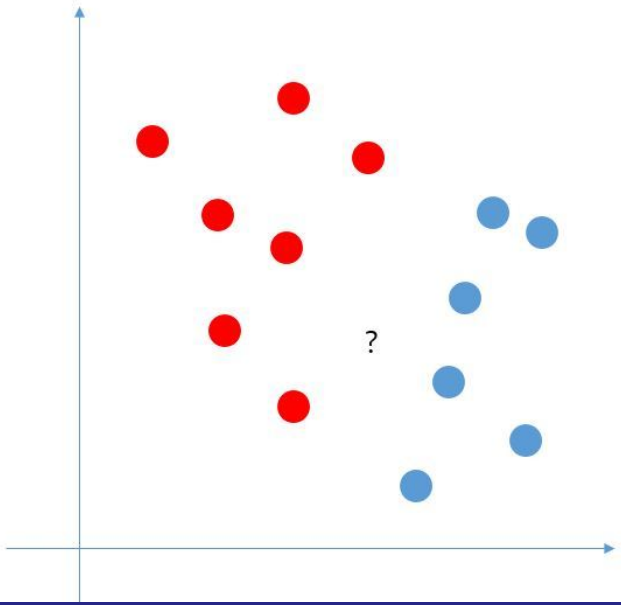
- Jeden z najprostszych algorytmów uczenia maszynowego
- kNN typowym przykladem **leniwego klasyfikatora** (brak f. dyskryminacyjnej)
- Podgrupa algorytmów nieparametrycznych - uczenie z przykładów (ang. *instance-based learning*)

Definicja algorytmu

Dla punktu x_0, znajdujemy k punktów $x_{(r)}, r = 1, ..., k$ najbliżej położonych (metryka odległości) od punktu x_0, a następnie sklasyfikuj używając głosowania większościowego dla k sąsiadów.

Algorytm kNN wymaga podania:

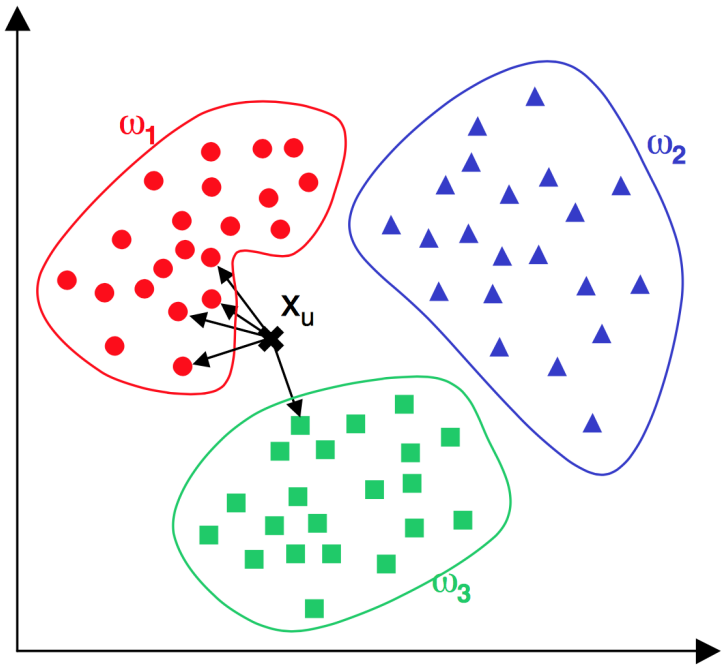
- Liczby całkowitej k
- Zestaw oznaczonych przykładów (zestaw szkoleniowy)
- Wybranej metryki odległości



Przykład

Algorytm można podsumować następująco:

- Znajdź klasę dla przykładu x_u
- Wybierz wartość parametru k oraz metrykę.
- Wybieramy odległość euklidesową oraz $k = 5$
- Z 5 najbliższych sąsiadów, 4 należą do $\omega 1$ oraz 1 do $\omega 3$, stąd x_u zostaje przypisany do $\omega 1$ - klasy dominującej.



Miary odległości

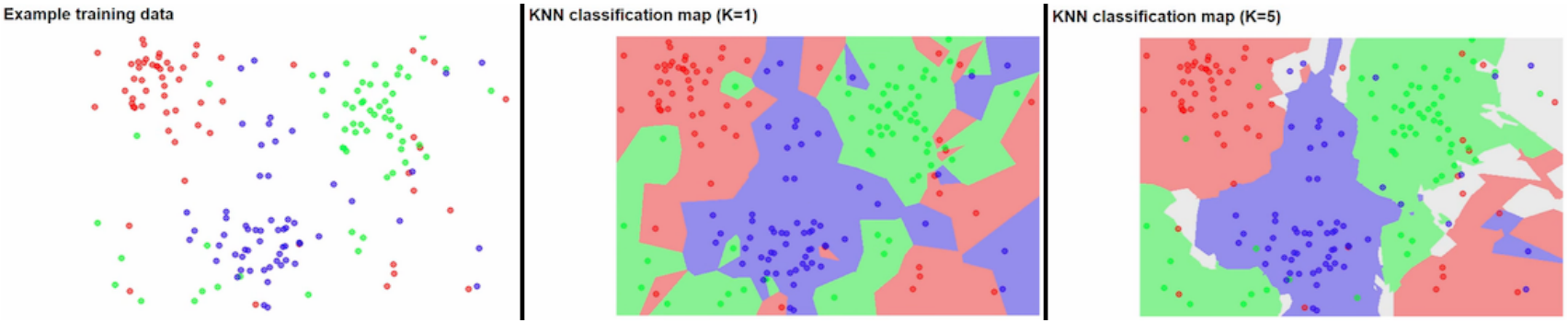
Miary podobieństwa powinny być wybierane konkretnie dla typu danych analizowanych: inne są bowiem miary typowo dla danych binarnych, inne dla danych nominalnych a inne dla danych numerycznych.
Wartości ciągłe:

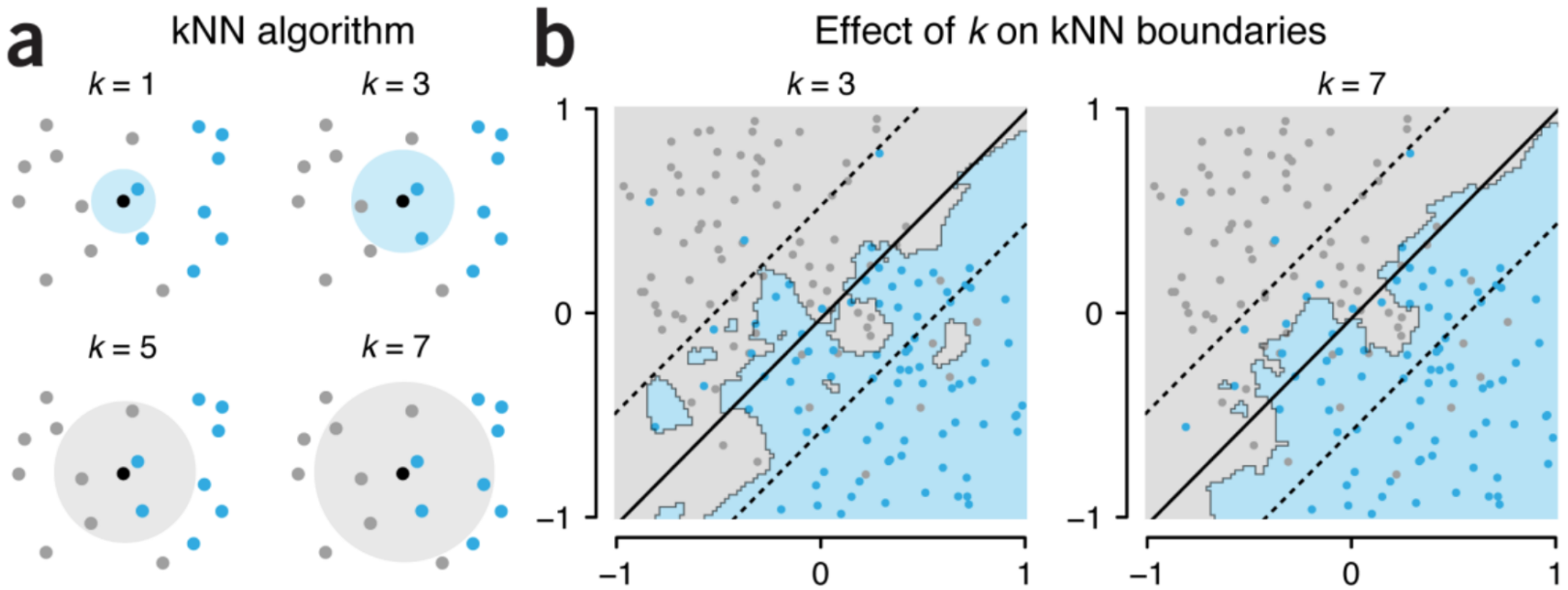
- Euclidean $$d_E(x,y) = [\sum_{i=1}^{n}(x_i - y_i)^2]^{1/2}$$
- Minkowsky $$d_E(x,y) = [\sum_{i=1}^{n}(x_i - y_i)^q]^{1/q}$$
- Distance absolute value $$d_{ABS}(x,y) = \sum_{i=1}^{n}| x_i - y_i |$$

Dane nominalne:

$$d_a(ex_1^a, ex_2^a) = \begin{cases} 1 & \text{if } ex_1^a \neq ex_2^a \\ 0 & \text{otherwise} \end{cases}$$

Algorithm


Example training data
KNN classification map (K=1)
KNN classification map (K=5)


a
kNN algorithm
k = 1
k = 3
k = 5
k = 7
b
Effect of k on kNN boundaries
k = 3
k = 7
1
0
−1
−1
0
1

1 NN

W przypadku przestrzeni dwuwymiarowej, dla danego zbioru n punktów, dzieli się płaszczyznę na n obszarów, w taki sposób, że każdy punkt w dowolnym obszarze znajduje się bliżej określonego punktu ze zbioru n punktów, niż od $n - 1$ pozostałych.

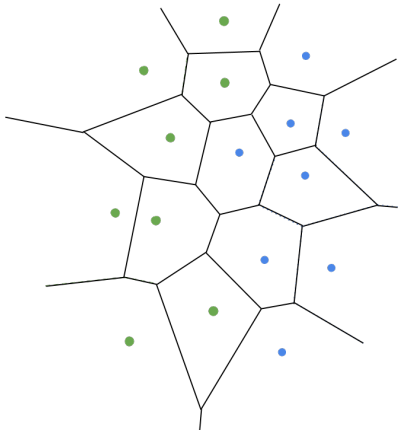
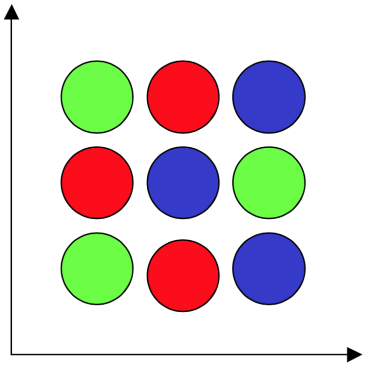


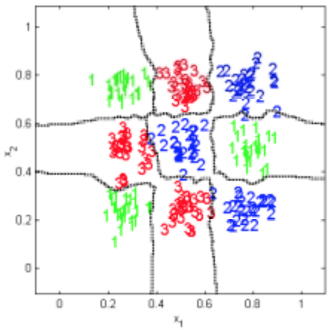
Diagram Woronoja:

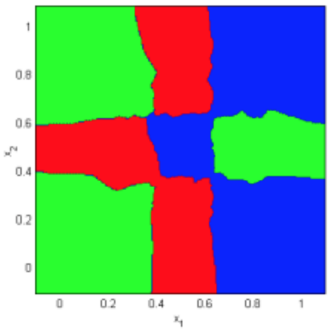
- Komórki Woronoja będąc intersekcją półpłaszczyzn są wielobokami wypukłymi.
- Krawędź - równa odległość od dwóch punktów.

k-NN example 1

We have generated data for a 2-dimensional 3- class problem, where the class-conditional densities are multi-modal, and non-linearly separable, as illustrated in the figure ($k = 5$, Euclidean distance).

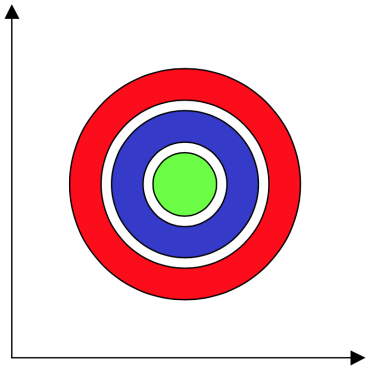


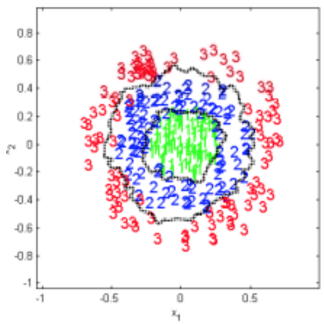


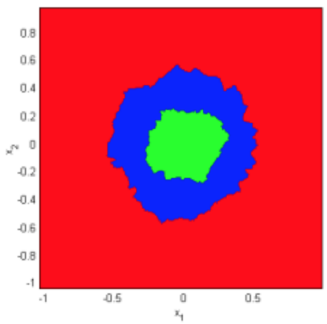


k-NN example 2

We have generated data for a 2-dimensional 3-class problem, where the class-conditional densities are unimodal, and are distributed in rings around a common mean. These classes are also non-linearly separable, as illustrated in the figure ($k = 5$, Euclidean distance).

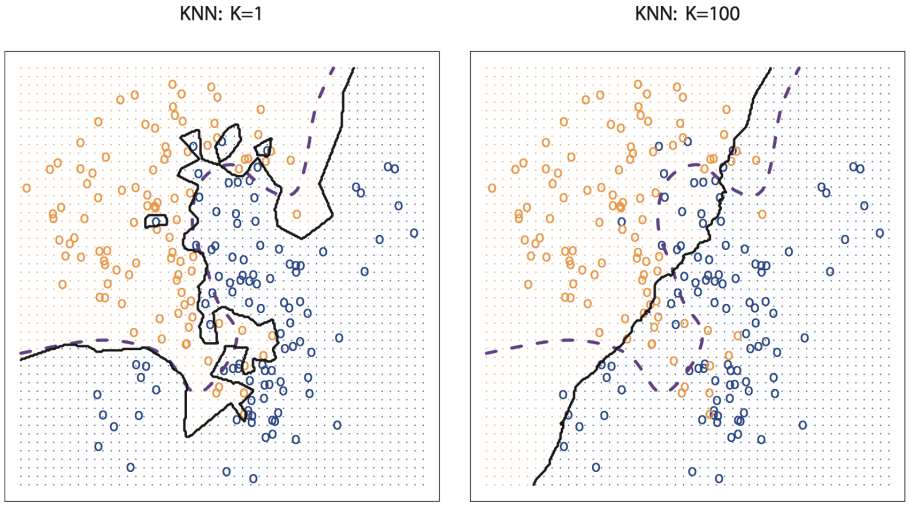


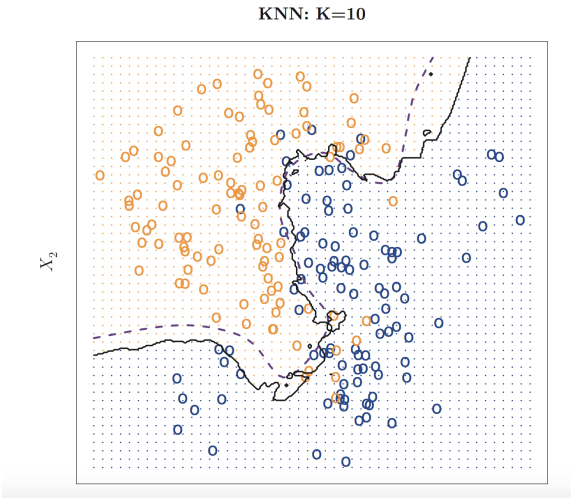




Effect of K - choosing K

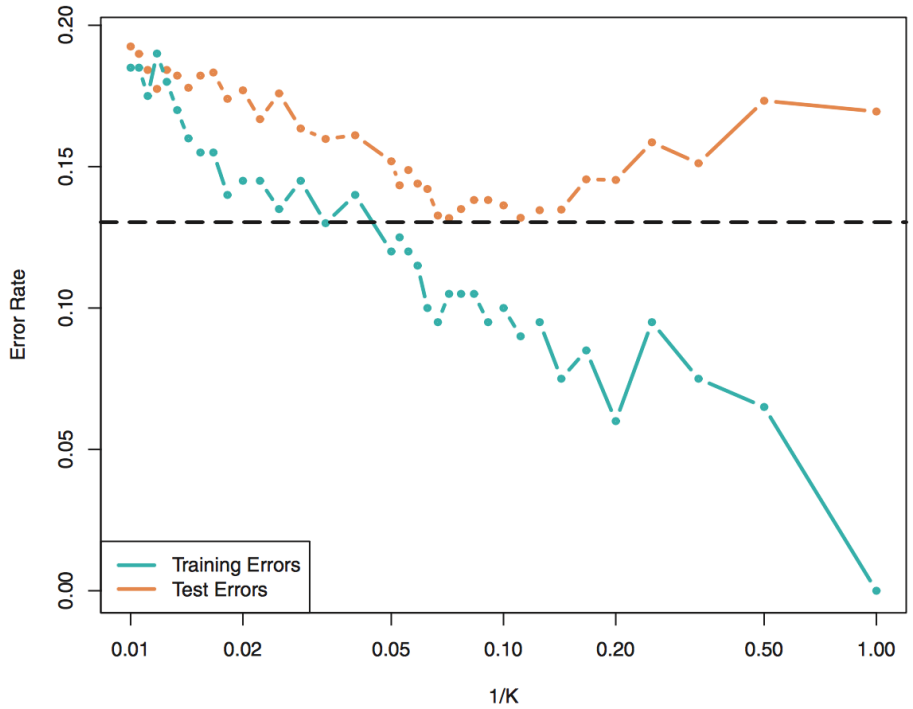
- Larger number of neighbors
- Larger regions
- Smoother class boundaries
- Large k reduces the impact of class label noise





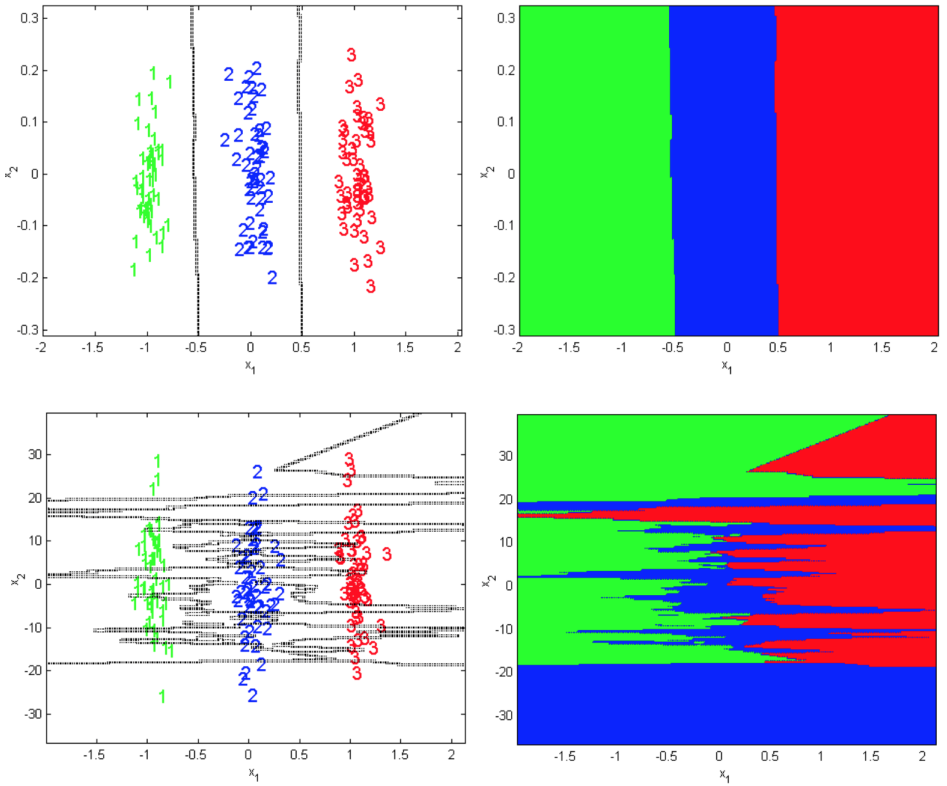
Error rate

The KNN training error rate (blue, 200 observations) and test error rate (orange, 5,000 observations) on the data from Figure 2.13, as the level of flexibility (assessed using 1/K) increases, or equivalently as the number of neighbors K decreases. The black dashed line indicates the Bayes error rate. The jumpiness of the curves is due to the small size of the training data set



Achilles' heel of the k-NNR classifier

- In the first example, both axes are scaled properly
- In the second example, the magnitude of the second axis has been increased two order of magnitudes (see axes tick marks)



Feature weighting

The solution to this problem is to modify the Euclidean metric by a set of weights that represent the information content or “goodness” of each feature.

IDEA: Give higher weights to k-instances:

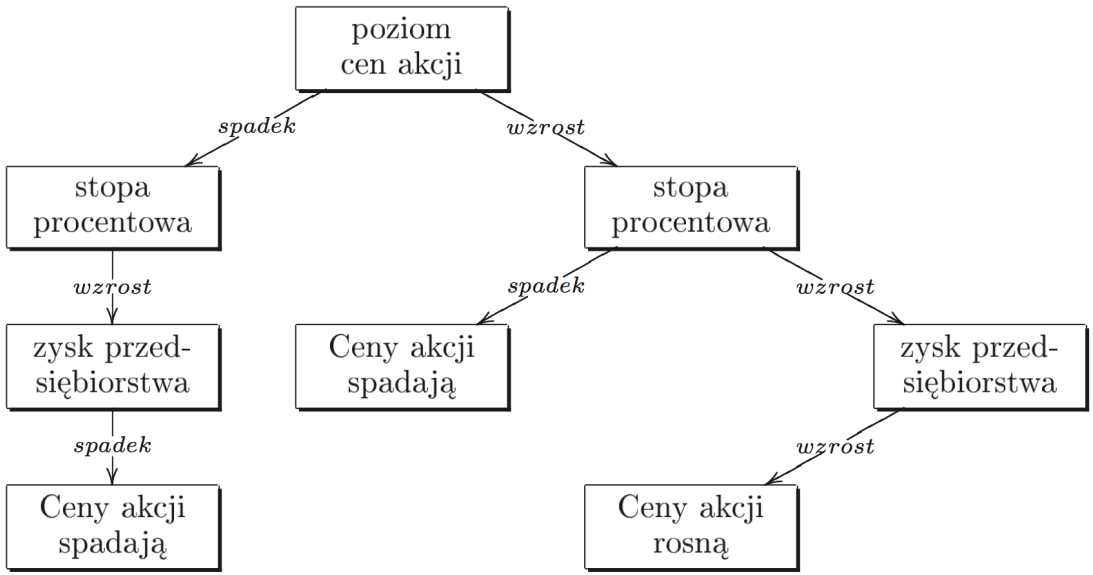
- expert knowledge -provides weights
- closer instances
- parameters
- special algoithms

Drzewa decyzyjne

Wprowadzenie

Drzewem decyzyjnym (klasyfikacyjnym) określimy drzewo reprezentujące proces podziału zbioru obiektów na jednorodne klasy.
W takim drzewie:

- wewnętrzne węzły opisują sposób dokonania podziału na jednorodne klasy (dokonywany w oparciu o wartości cech obiektów)
- liście klasom, do których obiekt należy
- krawędzie drzewa reprezentują wartości cech, na podstawie ktoórych dokonano podziału.



Algorytm

Drzewo decyzyjne buduje się w sposób rekurencyjny od korzenia do liścia, metodą "dziel i zwyciężaj". Pierwszy algorytm **ID3** i jego późniejsze rozszerzenie **C4.5** opracował Rossa Quinlan (profesor University of New South Wales, Australia).

Ogólna idea algorytmu indukcji drzewa jest następująca:

- Drzewo zaczyna od pojedynczego węzła reprezentującego cały zbiór treningowy.
- Jeżeli wszystkie przykłady należą do jednej klasy decyzyjnej, to zbadany węzeł staje się liściem i jest on etykietowany tą decyzją.
- W przeciwnym przypadku algorytm wykorzystuje miarę entropii (funkcja przyrostu informacji) jako heurystyki do wyboru atrybutu, który najlepiej dzieli zbiór przykładów treningowych.
- Dla każdego wyniku testu tworzy się jedno odgałęzienie i przykłady treningowe są odpowiednio rozdzielone do nowych węzłów (poddrzew).
- Algorytm działa dalej w rekurencyjny sposób dla zbiorów przykładów przydzielonych do poddrzew.

Algorytm kończy się, gdy kryterium stopu jest spełnione.

Algorytm

Algorytm indukcji drzewa decyzyjnego zatrzymuje się, gdy jeden z podanych niżej warunków jest spełniony:

- Wszystkie przykłady przydzielone do danego węzła należą do jednej klasy decyzyjnej, lub
- Nie istnieje atrybut, który może dalej podzielić zbiór przykładów. W tym przypadku, liść jest etykietowany nie jedną wartością decyzji, lecz wektorem wartości zwanym rozkładem decyzji, lub
- Wszystkie liście mają założoną z góry przewagę jednej klasy decyzyjnej (np. w żadnym nie ma mniej, niż 1

Miara podziału

Jak mierzyć jakość podziałów zbioru danych?

Wykorzystywane jest pojęcie **zysku informacyjnego** dla podjęcia decyzji między alternatywnymi podziałami. Pojęcie pochodzi z teorii informacji i wykorzystuje pojęcie **entropii**.

Entropia stowarzyszona ze zbiorem zdażeń -miara nieuporządkowania zbioru :

Wzór na entropię:

$$H(x) = \sum_{i=1}^{n} p(i) \log \frac{1}{p(i)} = -\sum_{i=1}^{n} p(i) \log p(i)$$

gdzie $p(i)$ - prawdopodobieństwo zajścia zdarzenia i.

Entropia w ramach teorii informacji jest definiowana jako średnia ilość informacji (liczba bitów), przypadająca na znak symbolizujący zajście zdarzenia z pewnego zbioru. Zdarzenia w tym zbiorze mają przypisane prawdopodobieństwa wystąpienia.

Z punktu widzenia teorii informacji:

- **miara 0 (czyli entropia 0)**, to wskazanie, że nie potrzebujemy żadnych dodatkowych informacji w celu sklasyfikowania obserwacji w obrębie zestawu danych - wszystkie obserwacje należą do tej samej klasy.
- **miara 1** wskazuje, że potrzebujemy maksymalną, możliwą liczbę dodatkowych informacji w celu sklasyfikowania naszych obserwacji.
- Jeżeli podział danej obserwacji jest różny od 50% / 50%, ale wynosi np. 75% / 25%, to potrzebujemy mniej informacji w celu sklasyfikowania naszych obserwacji - zestaw danych już zawiera informacje na temat w jaki sposób klasyfikować dane.

Przykład

OUTLOOK	TEMPERATURE	HUMIDITY	WINDY	PLAY
sunny	85	85	FALSE	Don't Play
sunny	80	90	TRUE	Don't Play
overcast	83	78	FALSE	Play
rain	70	96	FALSE	Play
rain	68	80	FALSE	Play
rain	65	70	TRUE	Don't Play
overcast	64	65	TRUE	Play
sunny	72	95	FALSE	Don't Play
sunny	69	70	FALSE	Play
rain	75	80	FALSE	Play
sunny	75	70	TRUE	Play
overcast	72	90	TRUE	Play
overcast	81	75	FALSE	Play
rain	71	80	TRUE	Don't Play

Przykład

W celu utworzenia drzewa decyzyjnego liczymy entropię dla różnych przypadków. Gdy entropia jest równa 0 to jest to koniec "konaru" drzewa decyzyjnego. I tak np. dla przykładu drugiego mamy, że średnia informacja I w danym węźle wynosi:
Outlook = sunny

$$I([2,3]) = H\left(\frac{2}{5},\frac{3}{5}\right) = -2/5\log\left(\frac{2}{5}\right) - \frac{3}{5}\log\left(\frac{3}{5}\right) = 0.971\,bits$$

Outlook = overcast

$$I([4,0]) = H(1,0) = -1\log(1) - 0\log(0) = 0\,bits$$

Outlook = rainy

$$I([3,2]) = H\left(\frac{3}{5},\frac{2}{5}\right) = -\frac{3}{5}\log\left(\frac{3}{5}\right) - \frac{2}{5}\log\left(\frac{2}{5}\right) = 0.971\,bits$$

Wartość dla atrybutu:

$$I([3,2],[4,0],[3,2]) = \left(\frac{5}{14}\right) * 0.971 + \left(\frac{4}{14}\right) * 0 + \left(\frac{5}{14}\right) * 0.971 = 0.693\,bits$$

Przyrost informacji (ang. information gain)

- **zysk informacji** (ang. *information gain*) służy do wyboru atrybutu decyzyjnego
- przyrost informacji IG jest to różnica ilosści informacji przed rozdzieleniem oraz po rozdzieleniu.
- w każdym węźle drzewa wybierany jest atrybut o największym zysku informacji (o największej redukcji entropii), gdyż jest to atrybut najlepiej rozróżniający (dyskryminujący) obiekty ze zbioru.

$$Gain(T, X) = Entropy(T) - Entropy(T, X)$$

Przykład

Przyrost informacji IG jest to różnica ilości informacji przed rozdzieleniem oraz po rozdzieleniu. Dla danych z przykładu 2 wygląda to następująco

$$IG(Outlook) = I([9,5]) - I([2,3],[4,0],[3,2]) = 0.940 - 0.693 = 0.247\,bits$$

Możemy budować drzewa decyzyjne również w oparciu o zysk informacyjny. Proces budowy drzewa zatrzymujemy gdy dalszy podział nie jest możliwy.

Algoryt C4.5

C4.5 jest algorytmem do tworzenia drzew decyzyjnych, które mogą być wykorzystane do klasyfikacji.
Kroki algorytmu:

1. wybierz atrybut, który najbardziej różnicuje wyjściowe wartości atrybutów
2. stwórz oddzielną gałąź dla każdej wartości wybranego atrybutu
3. podziel przypadki na podgrupy, tak aby odzwierciedlały wartości w wybranym węźle
4. dla każdej podgrupy zakończ proces selekcji jeżeli:
   - wszystkie podgrupy mają taką sama wartość dla wyjściowego atrybutu
   - podgrupa zawiera pojedynczy węzeł
5. dla każdej podgrupy stworzonej w 3 punkcie, która nie została określona jako liść powtórz powyższy proces

- **zysk informacji** (ang. *information gain*) służy do wyboru atrybutu decyzyjnego
- przyrost informacji IG jest to różnica ilosści informacji przed rozdzieleniem oraz po rozdzieleniu.

Ocena działania klasyfikatora

Kryteria (ang. Classification / Predictive accuracy):

- Szybkość i skalowalność:
  - czas uczenia się,
  - szybkość samego klasyfikowania
- Odporność (Robustness)
  - szum (noise),
  - missing values,
- Zdolności wyjaśniania: np. drzewa decyzyjne vs. sieci neuronowe
- Złożoność struktury:
  - rozmiar drzew decyzyjnego,
  - miary oceny reguły

Macierz pomyłek/błędów (ang. confusion matrix)

- Analiza pomyłek w przydziale do różnych klas przy pomocy tzw. macierz pomyłek (ang. confusion matrix)
- Macierz *r* x *r*, gdzie wiersze odpowiadają poprawnym klasom decyzyjnym, a kolumny decyzjom przewidywanym przez klasyfikator, na przecięciu wiersza *i* oraz kolumny *j* - liczba przykładów *n* – *ij* należących oryginalnie do klasy *i* – *tej*, a zaliczonej do klasy *j* – *tej*

Przykład:

	Przewidywane klasy decyzyjne		
Oryginalne klasy	K_1	K_2	K_3
K_1	50	0	0
K_2	0	48	2
K_3	0	4	46

Klasyfikacja binarna

Oryginalne klasy	Przewidywane klasy decyzyjne	
	Pozytywna	Negatywna
Pozytywna	*TP*	*FN*
Negatywna	*FP*	*TN*

- TP (ang. true positive) – liczba poprawnie sklasyfikowanych przykładów
- FN (ang. false negative) – liczba błędnie sklasyfikowanych przykładów z tej klasy, tj. decyzja negatywna podczas gdy przykład w rzeczywistości jest pozytywny (błąd pominięcia - z ang. miss),
- TN (ang. true negative) – liczba przykładów poprawnie nie przydzielonych do wybranej klasy (poprawnie odrzuconych z ang. correct rejection),
- FP (ang. false positive) – liczba przykładów błędnie przydzielonych do wybranej klasy, podczas gdy w rzeczywistości do niej nie należą (ang. false alarm).

Miary stosowane w ocenie klasyfikatora

		True condition				
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**, Power	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$	
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$	
		True positive rate (TPR), Recall, Sensitivity, probability of detection = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$	F_1 score = $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$
		False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	True negative rate (TNR), Specificity (SPC) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$		

Czułość (ang. ”sensitivity, recall, TPR): $TPR = TP/P = TP/(TP + FN)$
Swoistość (ang. ”specificity”, TNR): $TNR = TN/N = TN/(FP + TN)$
Precyzja (ang. ”precision”, PPV – wartość predykcyjna dodatnia):
$precyzja = TP/(TP + FP)$
Dokładność (ang. ”accuracy”, ACC): $ACC = (TP + TN)/(P + N)$