Uczenie maszynowe

Paweł Rychlikowski

Instytut Informatyki UWr

28 kwietnia 2023

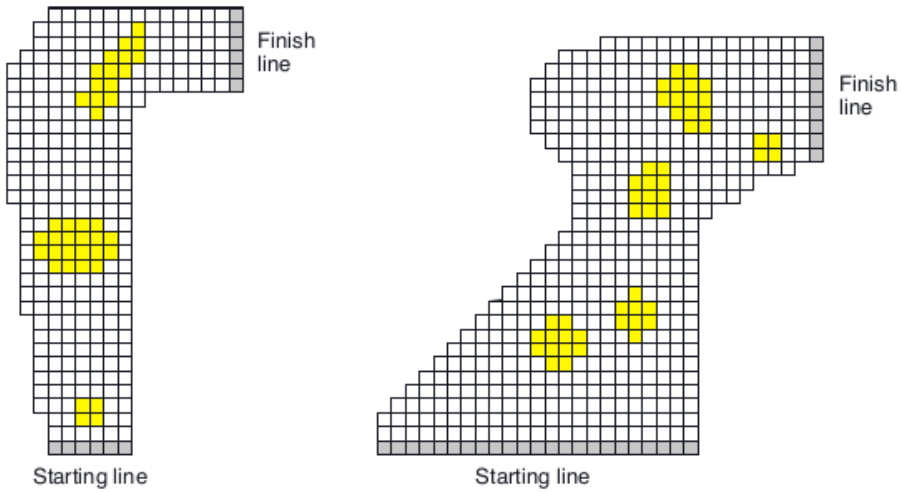
MDP – formalna definicja (przypomnienie)

Definicja

Markowowski proces decyzyjny (MDP) zawiera następujące składowe:

1. S – (skończony) zbiór stanów
2. Stan startowy, $s_{\text{start}} \in S$
3. Actions(s) – zbiór możliwych akcji w stanie s
4. T(s,a,s') – prawdopodobieństwo przejścia z s do s' w wyniku akcji a
5. Reward(s,a,s') – nagroda (wypłata) związana z tym przejściem
6. IsEnd(s) – czy stan jest końcowy?
7. Discount factor, $0 < \gamma \leq 1$ – sprawia, że nagrody w przyszłości cieszą mniej.

Wyścigi samochodzików. Przypomnienie



- Dodajemy plamy po oleju
- Ruch z pola oleju dodaje dodatkową składową losową do prędkości (znamy rozkład).

W tym momencie klasyczne MDP + algorytm Bellmana (czyli iteracji wartości) powinny dać dobry wynik.

Wyścigi samochodzików. Duża plansza

- Problem: dużo większa plansza, dużo większa liczba stanów.
- **Pomysł 1**: położenie „rozmyte", na przykład w kwadracie 10×10 pikseli.
- **Pomysł 2**: dodatkowo informacja, czy jestem 1, 2, czy 3 raz w takim kwadracie ($3 < 100$)

Fundamentalny problem: nie znamy mechaniki takiego świata (i wielu innych)

Wyścigi samochodzików. Float

- Prędkość autka jest wektorem $(v\cos(d), v\sin(d))$,
- Możemy zmieniać d (skręcać), oraz v (przyśpieszać, hamować)
- Celem jest meta.
- W pełni deterministyczny świat, ale bardzo duża liczba stanów, zawierających liczby `float`)

Autka float. Rozwiązanie

- Możemu stworzyć **stan abstrakcyjny** i opisać mechanikę świata dla takich stanów
- Oczywiście będzie ona niedeterministyczna, bo nigdy nie będziemy wiedzieć, czy zmiana w świecie float-ów przenosi się na zmianę w świecie int-ów.

Uwaga

Możemy myśleć o tym, że modelujemy błędy pomiarowe (int zamiast float) za pomocą losowości.

Uczenie ze wzmocnieniem

- Zakładamy, że nie dysponujemy modelem (czyli przejściami, prawdpodobieństwami i nagrodami)
- Możemy wszakże wykonywać pewne eksperymenty w naszym systemie, w wyniku których zdobywamy wiedzę jak nam poszło

Uwaga

Zauważmy, że to pasuje do naszych samochodzików z rozmytymi stanami (eksperyment przeprowadzamy na prawdziwym modelu, ale obserwujemy model „rozmyty")

Ogólny schemat uczenia ze wzmocnieniem

Dla $t \in 1, 2, 3, \ldots$

- Wybieramy akcje $a_t = \pi_{act}(s_{t-1})$ (**jak?**)
- Wykonujemy akcję i obserwujemy nowy stan s_t
- Uaktualniamy parametry (**jak?**)

Metody Monte Carlo

- Estymujemy model podczas eksperymentów
- Rozwiązujemy wyszacowane MDP.

Szacowany MDP

- $\hat{T}(s, a, s') = \frac{\text{cnt}(s,a,s')}{\text{cnt}(s,a)}$
- Nagroda: średnie **r** dla zaobserwowanych **s a r s'**

Eksploracja i eksploatacja

- Jaką polityką mamy badać świat?
  a) Wybierającą akcje losowo (strata czasu?)
  b) Wybierającą akcje prowadzącą do stanu o najlepszej wartości V (ale możemy się zafiksować na nieoptymalnej ścieżce)
- Wybór między a) i b) to wybór między eksploracją i eksploatacją
  - Pamiętamy: strategia ε-zachłanna.

Możemy przeplatać etapy wyznaczania modelu i rozwiązywania MDP (bo w kolejnych iteracjach mamy możliwość wykorzystania lepszej strategii eksploatacyjnej).

Bezmodelowe uczenie ze wzmocnieniem

Mówiliśmy o metodach Monte Carlo, w których przeprowadzamy eksperymenty (losowe przebiegi), żeby estymować (nieznane) parametry MDP.

- Nowy cel: od razu liczyć $Q(s, a)$, nie przejmując się tworzeniem modelu.
- Zaczniemy od obliczenia $Q_\pi(s, a)$

Bezmodelowe Monte Carlo

Definicja (przypomnienie)

$Q_\pi(s, a)$ to oczekiwana sumaryczna nagroda, jaką otrzymamy wykonując w stanie s akcję a, a następnie postępując zgodnie z polityką π

- Użyteczność (dla konkretnego przebiegu): $u_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots$
- $\hat{Q}_\pi(s, a) =$ średnie u_t, gdzie $s_{t-1} = s$, $a_t = a$

Bezmodelowe Monte Carlo – inne sformułowanie

- Zamiast liczyć średnią z całości, można myśleć o uaktualnianiu średniej wraz z pojawieniem się kolejnej informacji.
- Niech: $\eta = \frac{1}{1+\text{cnt}(s,a)}$
- $\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta u$ (gdzie u jest użytecznością zaobserwowaną w konkretnym przebiegu)

Sprawdźmy, czy to się zgadza.

$$\frac{\text{cnt}(s,a)\hat{Q}_\pi(s,a)}{1+\text{cnt}(s,a)} + \frac{u}{1+\text{cnt}(s,a)}$$

Bezmodelowe Monte Carlo – inne sformułowanie (2)

$$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta u$$

- u jest zaobserwowaną użytecznością
- $\hat{Q}_\pi(s,a)$ jest naszą predykcją.

Reguła ta minimalizuje odległość między predykcją a obserwacją.

Bootstraping

Uwaga

W informatyce często, rozwiązując jakieś zadanie, korzystamy z niedoskonałego (tymczasowego) rozwiązania, żeby rozwiązać zadanie lepiej.

Przykład

Szukanie dobrych i złych słów (analizujemy wpisy na jakimś forum), na początku znamy kilka przykładowych dobrych i złych słów.

Będziemy używać Q (poprzedniej wartości) do obliczenia nowego Q

Bootstraping: SARSA

Obserwujemy ciąg akcji i nagród:

$$s_0, a_1, r_1, s_1, a_2, r_2, s_2, \ldots$$

- Uaktualnianie Monte Carlo:

$$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta u$$

- SARSA (obserwujemy s, a, r, s', a'):

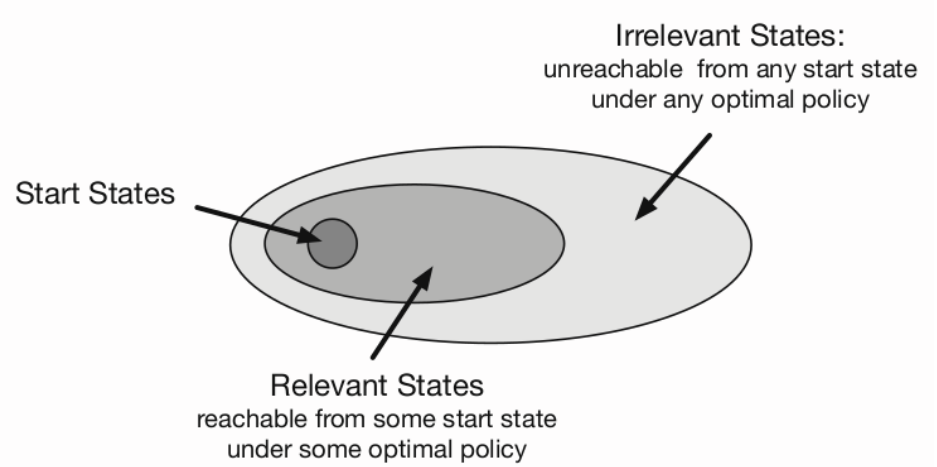
$$\hat{Q}_\pi(s,a) \leftarrow (1-\eta)\hat{Q}_\pi(s,a) + \eta(r + \gamma\hat{Q}_\pi(s',a'))$$

W algorytmie SARSA zamiast konkretnego (zaobserwowanego) u bierzemy zaobserwowaną jego i pierwszą część (r) i estymowaną resztę (zielony jest cel)

Uwaga

Nie musimy czekać do końca epizodu, żeby uaktualnić wartość Q!

Value iteration vs. SARSA



źródło: Sutton, Reinforcement Larning. An introduction

- VI liczy wartości dla stanów „nieoptymalnych”
- VI liczy wartości dla stanów nieosiągalnych (łatwo wymyślić dla autek taką kombinację prędkości i położenia, która jest bezużyteczna)

W momencie, gdy operujemy przebiegami, być może sensownymi, to koncentrujemy się na estymacji rzeczy użytecznych (a na pewno na osiągalnych!)

Q-learning

Uwaga

SARSA estymuje $Q_\pi(s, a)$. Najbardziej naturalnym celem jest znajomość Q_{opt}.

- Algorytm umożliwiający bezpośrednie obliczanie Q_{opt} to właśnie **Q-learning**.
- Również radzimy sobie bez modelu.

Q-learning

Standardowy kształt reguły:

$$Q(s,a)_{\text{opt}} \leftarrow (1-\eta)Q(s,a)_{\text{opt}} + \eta \text{ cel}$$

Celem jest $r + \gamma V_{\text{opt}}(s')$
Natomiast:

$$V_{\text{opt}}(s') = \max_{a' \in \text{Actions(s')}} Q_{\text{opt}}(s', a'))$$

Algorytm **Q-learning**

Dla zaobserwowanych s, a, r, s', a' (dla czytelności bez opt):

$$Q(s,a) \leftarrow (1-\eta)Q(s,a) + \eta(r + \gamma \max_{a' \in \text{Actions(s')}} Q(s', a')))$$

O wadze eksploracji

- Jeżeli chcemy zachowywać się optymalnie powinniśmy wiedzieć coś o każdej parze (s,a)
- Istnieją dwie możliwości:
  1. Rzeczywiście mamy szansę (w granicy) wygenerować przebieg z każdą parą (s,a)
  2. Umiemy jakoś generalizować i wywnioskować coś na temat (s,a) korzystając z podobnego (s', a')

Obok wnioskowania i przeszukiwania jeden z głównych silników sztucznej inteligencji.

Użyteczne między innymi w sytuacjach o których ostatnio mówiliśmy, gdy nie chcemy pamiętać wartości $Q(s, a)$ lecz umieć ją obliczyć

Psy i koty. Konkurs na Kaggle.com

In this competition, you'll write an algorithm to classify whether images contain either a dog or a cat. This is easy for humans, dogs, and cats. Your computer will find it a bit more difficult.

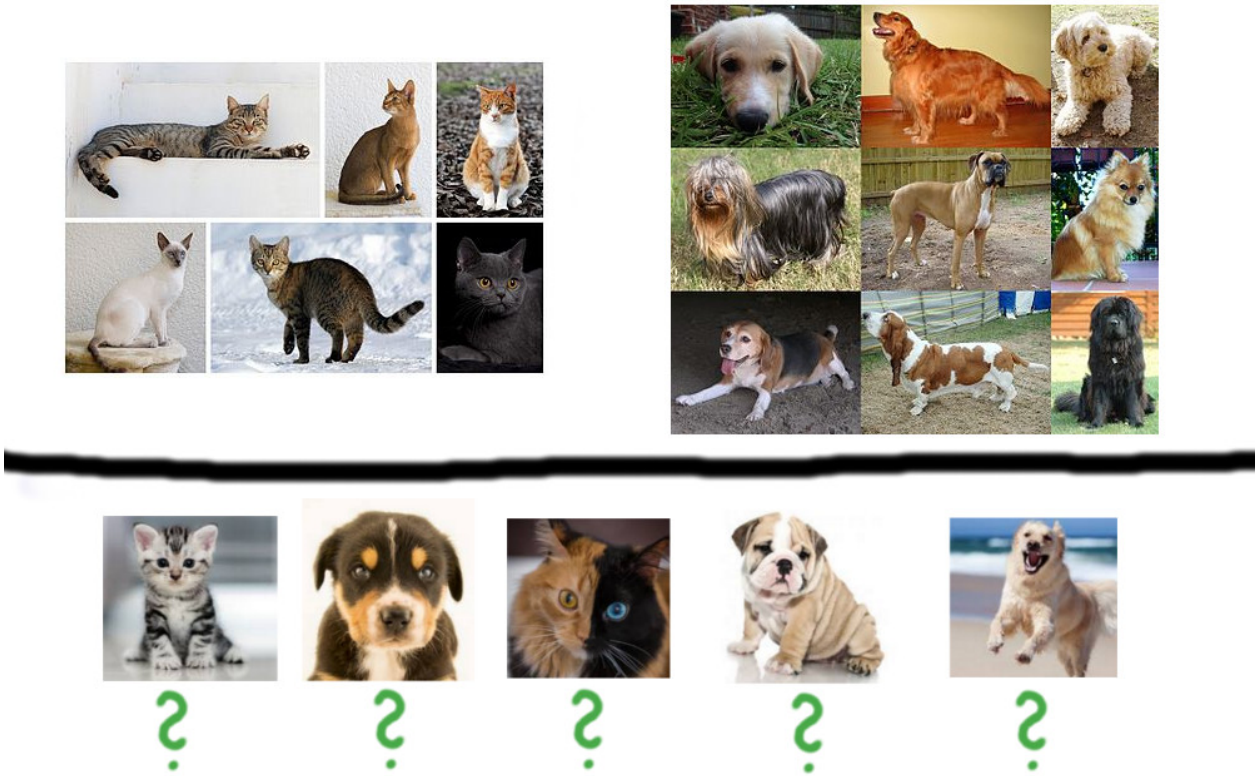
Deep Blue beat Kasparov at chess in 1997.
Watson beat the brightest trivia minds at Jeopardy in 2011.
Can you tell Fido from Mittens in 2013?

Uczenie z nadzorem

Dane uczace

Uczenie z nadzorem

Dane uczace i dane testowe



Uczenie się

- Spróbujemy usystematyzować nasze intuicje związane z uczeniem.
- Co wiemy:
  a. Mamy przykłady, próbujemy je **uogólnić**.
  b. Jedno z podstawowych zadań: **klasyfikacja**, czyli przypisanie przypadkowi jego klasy.
  c. Przykłady:
     - Ocena, czy mail należy do spam czy też nie-spam.
     - Wybór rasy dla zdjęcia psa
     - Czy napis jest adresem e-mail, url-em, nazwą firmy, imieniem i nazwiskiem, czymś innym?

Regresja

- Oczekiwanym wynikiem może być liczba rzeczywista.
- Przykłady:
  - predycja ceny nieruchomości,
  - ocena masy ciała (gdy znamy płeć i wzrost),
  - przewidywanie zużycia wody (dla MPWiK), gdy znamy temperaturę i dzień tygodnia

W klasyfikacji i regresji inaczej oceniamy sukces, w regresji liczba nie musi być dokładna, wystarczy, że jest blisko.

Cechy kota i psa

Uwaga

Zadanie rozróżnienia kota i psa byłoby łatwiejsze, gdybyśmy mieli dane nie obrazki, lecz cechy zwierzęcia

Przykłady?

- masa ciała,
- odległość między oczami,
- odległość nosa i oka,
- długość włosów,
- długość wąsów,
- długość ogona

użyteczne mogłyby być też na przykład proporcje różnych cech i inne wtórne cechy (wyliczone z podstawowych)

Cechy (wektor cech)

- Abstrakcyjny obiekt możemy zamienić na wektor cech.
- Dla (zabawkowego) klasyfikatora **czy-email?**, możemy mieć:
  - Czy długość większa od 10?
  - Jaki procent znaków to znaki alfanumeryczne?
  - Czy zawiera @?
  - Czy kończy się na .com (i tak dalej)

Cechami mogą być też na przykład wartości składowych pikseli, kolejne wartości pliku wave, zbiory pomiarów wszystkich wodomierzy z ostatniej doby, itd.

Dla obiektu x wektor cech oznaczamy często jako $(\phi_1(x), \ldots, \phi_n(x))$.

Dane i uczenie

- Dane uczące – zbiór przykładów, często w postaci:
  (wektor-cech1, wynik1)
  (wektor-cech2, wynik2)
  (wektor-cech3, wynik3)
  ...

Podział dostępnych danych

Definicja 1

Zbiór **uczący** jest podstawowym zbiorem, który będziemy wykorzystywać do zdobywania wiedzy o problemie.

Definicja 2

Zbioru **walidacyjnego** używamy do wyboru rodzaju algorytmu lub do wyboru hiperparametrów algorytmu.

Definicja 3

Zbiór **testowy** używany jest **tylko** do ostatecznego testu, który ma nas przekonać, jak dobrze uogólnia rzeczywistość nasz mechanizm.

Do zbioru testowego nawet nie zaglądamy, nie analizujemy błędów, itd!

Klasyczne zadanie klasyfikacji obrazów

MNIST jest zbiorem około 60K czarnobiałych obrazków 28×28 zawierających ręcznie pisane cyfry.

Jest on powszechnie używany do testowania różnych algorytmów uczenia (głównie klasyfikacji, ale nie tylko)

MNIST – przedstawienie danych

MNIST – wyszukiwanie wzorców

- Naturalnym rozwiązaniem jest stworzenie wzorca (lub wzorców) dla każdej cyfry.
- Jak to zrobić?

Dwa warianty

1. Jeden wzorzec dla wszystkich obrazków danej cyfry.
2. Wiele wzorców (nawet: każda cyfra wzorcem)

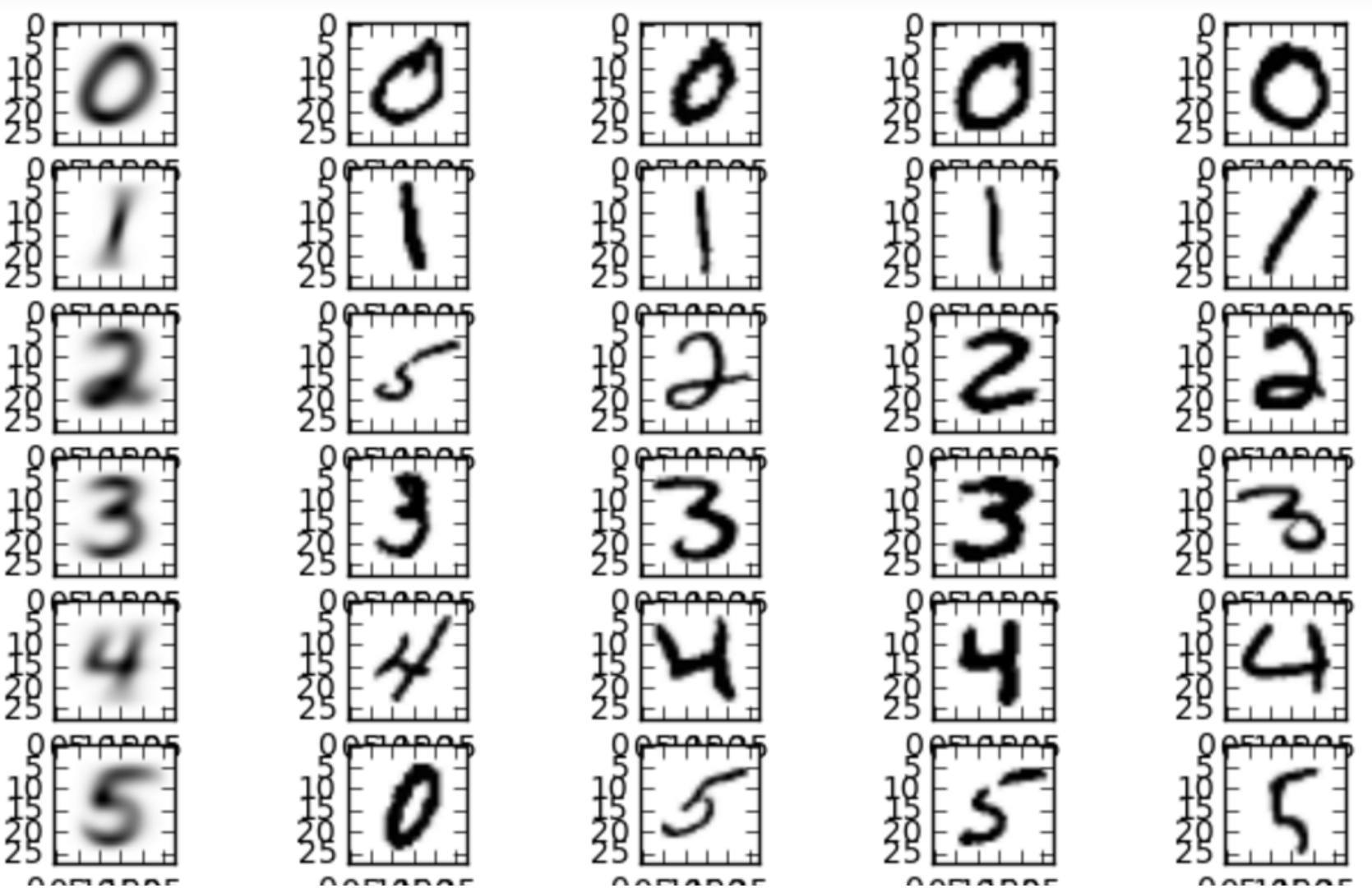
Jeden wzorzec dla cyfry

- Naturalnym wzorcem może być średnia wszystkich egzemplarzy danej cyfry
- Przy klasyfikacji obrazka **o** wybieramy wzorzec **w** najbardziej podobny do **o**
- Co to znaczy podobny?
  - Wysoki iloczyn skalarny?
  - *Wysoki iloczyn skalarny znormalizowanych wektorów?*
  - Niewielka odległość euklidesowa (a może jakaś inna)?

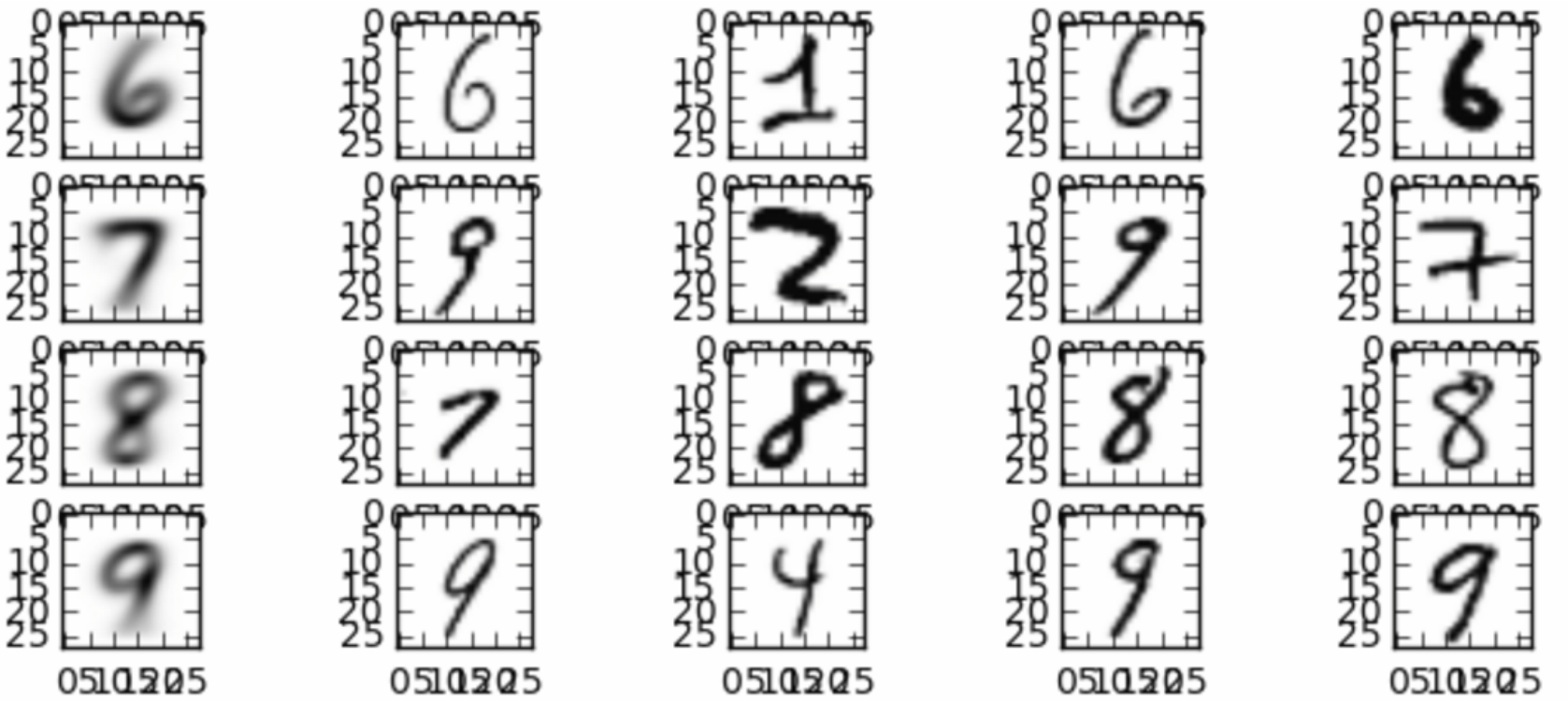
Wyniki eksperymentu

Poprawność klasyfikacji to około **82.1%** (dla iloczynów skalarnych znormalizowanych wektorów)

Wyniki klasyfikacji z „wzorcami średnimi”

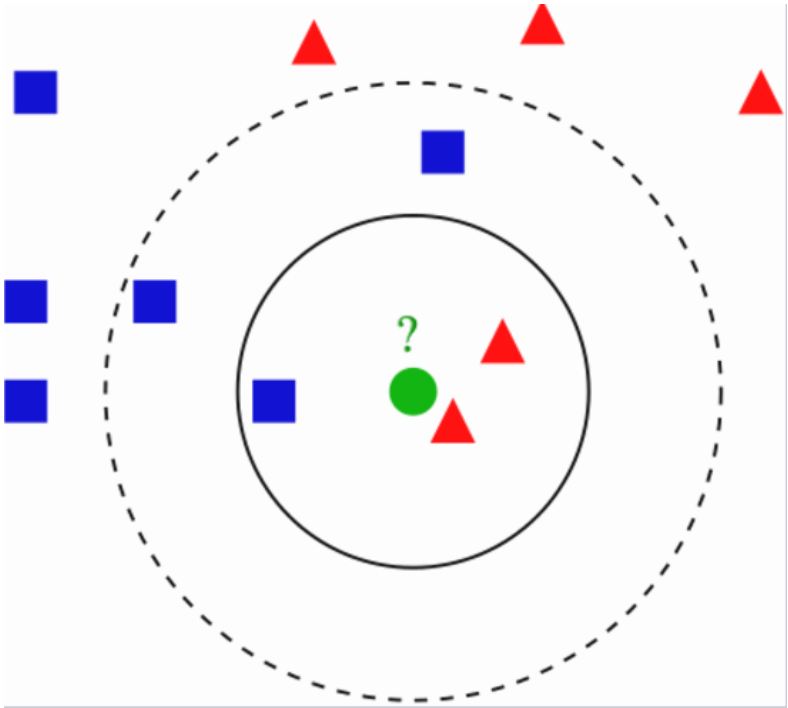


Wyniki klasyfikacji z „wzorcami średnimi”



K najbliższych sąsiadów. KNN

- Pamiętamy wszystkie wektory ze zbioru uczącego.
- Klasyfikując obrazek znajdujemy K najbliższych sąsiadów i pozwalamy im głosować

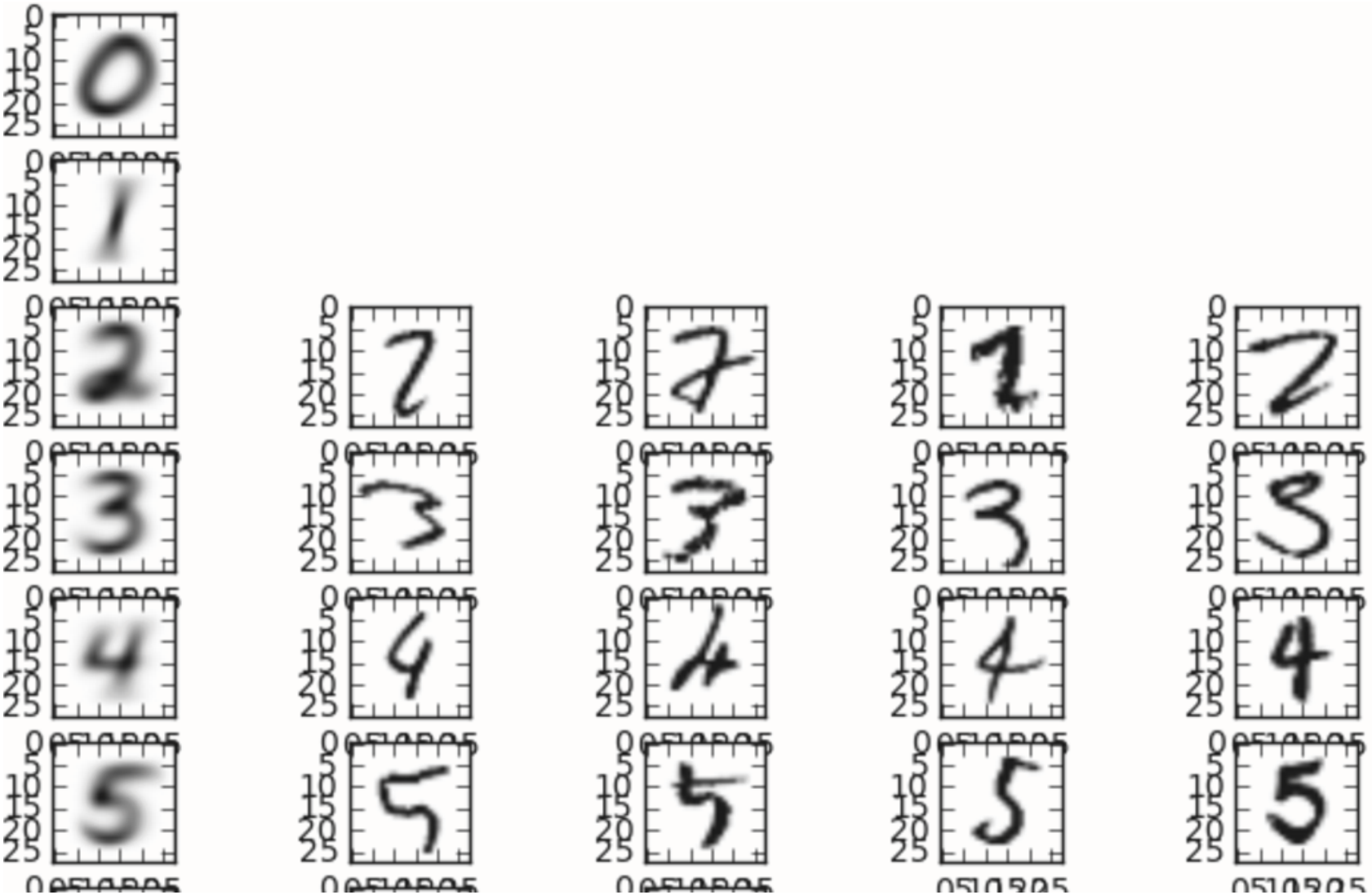


MNIST i KNN

- Podobnieństwo mierzymy iloczynem skalarnym znormalizowanych wektorów (czyli cosinusem)
- Testujemy na próbce (bo inaczej trwa bardzo długo)
- $K = 3$
- Wyniki:
  - Zbiór uczący: **98.55%**
  - Zbiór testowy: około **97%**

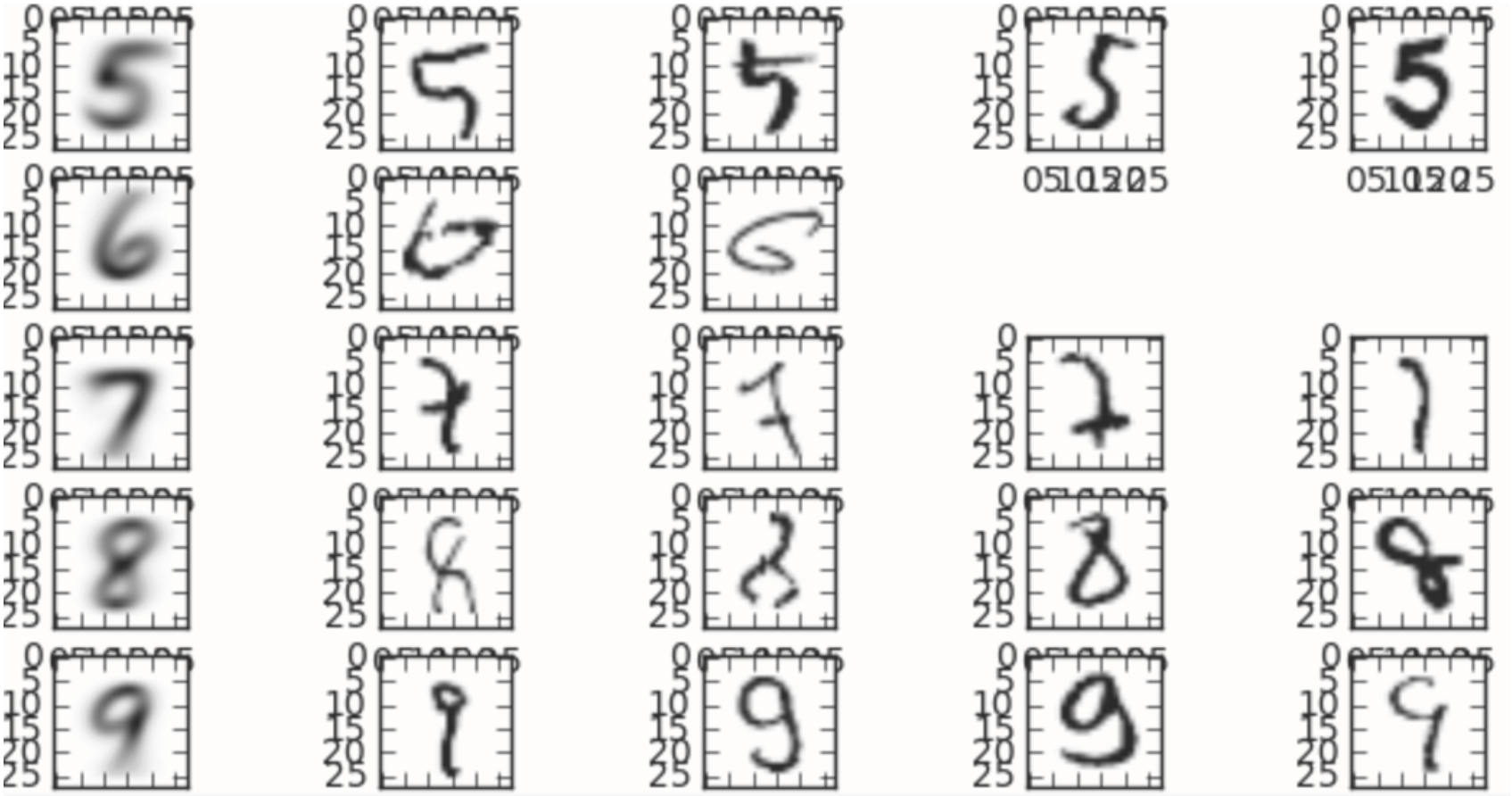
MNIST i KNN. Przykładowe błędy

Z tymi cyframi mieliśmy problemy (zaznaczona prawidłowa klasyfikacja)



MNIST i KNN. Przykładowe błędy

Z tymi cyframi mieliśmy problemy (zaznaczona prawidłowa klasyfikacja)



Zabawkowy problem – dane restauracyjne

Zadanie

Odpowiadamy na pytanie, czy warto czekać na stolik w restauracji

Dostępne dane:

- Alternate – czy w okolicy jest alternatywa
- Bar – wygodna przestrzeń do czekania w restauracji
- Fri/Sat – piątek lub sobota
- Hungry – czy teraz jestem głodny
- Patrons – ile ludzi w restauracji (None, Some, Full)
- Price – \$, \$\$, \$\$\$
- Raining – czy pada?
- Reservation – czy zrobiliśmy rezerwację?
- Type – French, Italian, Thai, burger
- WaitEstimate – 0-10, 10-30, 30-60, > 60

$2^6 \times 3^2 \times 4^2 = 9216$ możliwych kombinacji argumentów

Przykładowe dane uczące

Example	Input Attributes										Output
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*WillWait*
$\mathbf{x}_1$	*Yes*	*No*	*No*	*Yes*	*Some*	*$$$*	*No*	*Yes*	*French*	*0–10*	$y_1 = Yes$
$\mathbf{x}_2$	*Yes*	*No*	*No*	*Yes*	*Full*	*$*	*No*	*No*	*Thai*	*30–60*	$y_2 = No$
$\mathbf{x}_3$	*No*	*Yes*	*No*	*No*	*Some*	*$*	*No*	*No*	*Burger*	*0–10*	$y_3 = Yes$
$\mathbf{x}_4$	*Yes*	*No*	*Yes*	*Yes*	*Full*	*$*	*Yes*	*No*	*Thai*	*10–30*	$y_4 = Yes$
$\mathbf{x}_5$	*Yes*	*No*	*Yes*	*No*	*Full*	*$$$*	*No*	*Yes*	*French*	*>60*	$y_5 = No$
$\mathbf{x}_6$	*No*	*Yes*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Italian*	*0–10*	$y_6 = Yes$
$\mathbf{x}_7$	*No*	*Yes*	*No*	*No*	*None*	*$*	*Yes*	*No*	*Burger*	*0–10*	$y_7 = No$
$\mathbf{x}_8$	*No*	*No*	*No*	*Yes*	*Some*	*$$*	*Yes*	*Yes*	*Thai*	*0–10*	$y_8 = Yes$
$\mathbf{x}_9$	*No*	*Yes*	*Yes*	*No*	*Full*	*$*	*Yes*	*No*	*Burger*	*>60*	$y_9 = No$
$\mathbf{x}_{10}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$$$*	*No*	*Yes*	*Italian*	*10–30*	$y_{10} = No$
$\mathbf{x}_{11}$	*No*	*No*	*No*	*No*	*None*	*$*	*No*	*No*	*Thai*	*0–10*	$y_{11} = No$
$\mathbf{x}_{12}$	*Yes*	*Yes*	*Yes*	*Yes*	*Full*	*$*	*No*	*No*	*Burger*	*30–60*	$y_{12} = Yes$

Figure 19.2 Examples for the restaurant domain.

Drzewa decyzyjne

- W węzłach mają zmienne (atrybuty),
- Krawędzie są etykietowane wartościami atrybutów
- W liściach są odpowiedzi

Uwaga

Wiele stosowanych w codziennym życiu reguł ma postać drzewa (np. co robić z ofiarą wypadku)

Przykładowe drzewo

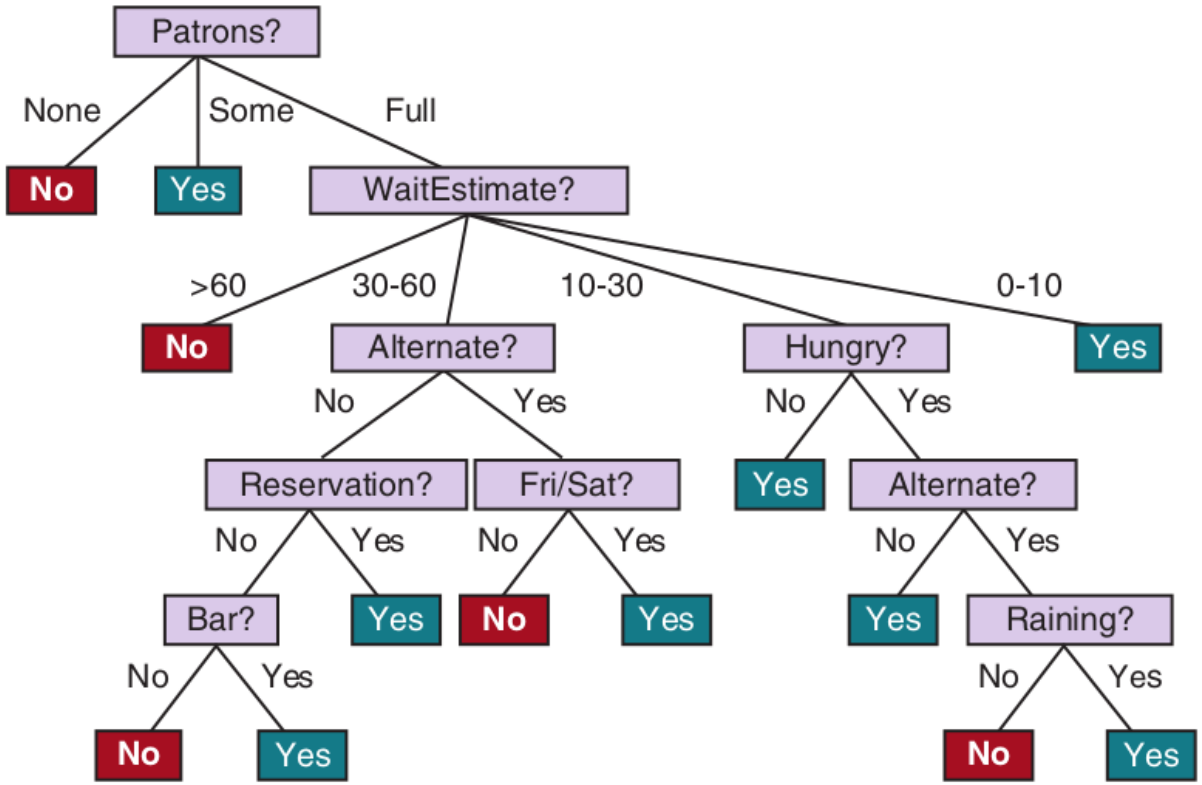


Figure 19.3 A decision tree for deciding whether to wait for a table.