Wojciech Myszka

Laboratorium 6 wer. 46 z drobnymi modyfikacjami!

2019-11-27 10:24:31 +0100

1. Laboratorium 6: Sztuczna inteligencja

1.1. AI: Odrobina teorii

AI to oczywiście *Artificial Intelligence* czyli Sztuczna Inteligencja.

Natomiast następujących zapisów nie należy w żadnym wypadku traktować jako jakiejkolwiek uporządkowanej informacji na temat sztucznej inteligencji. Umieszczam tu kilka bardzo ogólnych i zupełnie przypadkowych uwag.

Tematy sztucznej inteligencji czy uczenia maszynowego czy *big data* (wszystkie jakoś ze sobą powiązane) odgrywają dziś coraz większą rolę. Stąd nieśmiały pomysł, aby praktycznie (oraz szybko i łatwo) pokazać proste zadania z tym związane. Natomiast tematyka jest bardzo szeroka i warta podjęcia szerszych studiów (na przykład na II stopniu?).

Pod względem matematycznym, na podstawy sztucznej inteligencji składają się:
- algebra liniowa:
  - przestrzenie wektorowe,
  - tensory,
  - przekształcenia liniowe,
  - podprzestrzenie
  - teoria macierzy,
  - wartości i wektory własne,
- analiza matematyczna:
  - granice, ciągłość, nieciągłość,...
  - pochodne,
  - całki,
  - podstawowe twierdzenia rachunku całkowego,
  - ...
- rachunek prawdopodobieństwa:
  - prawdopodobieństwa warunkowe,
  - „prawo wielkich liczb”,
  - statystyka bayesowska,
  - błądzenie losowe (np. łańcuchy Markowa)
  - ...
- optymalizacja:
  - programowanie liniowe,

    — programowanie kwadratowe,
    — zagadnienia kombinatoryczne,
    — ...
— heurystyka:
    — algorytmy genetyczne,
    — ewolucja różniczkowa (*Differential evolution*),
— metody iteracyjne,
    — metoda Newtona,
    — metody gradientowe,
    — metody gradientów sprzężonych
    — interpolacja
— ...

1.1.1. Klasyfikacja

Jednym z najważniejszych zadań sztucznej inteligencji jest klasyfikacja, czyli taki problem decyzyjny, który odpowiada do jakiej kategorii należy badany obiekt. Jednym z przykładów jest rozpoznanie rodzaju schorzenia na podstawie objawów. Dodać trzeba, że bardzo często algorytmy sztucznej inteligencji sprawdzają się tu bardzo dobrze.

Podstawowe metody stosowane przez algorytmy klasyfikacji to:
— „naiwny Bayes" — metoda oparta na twierdzeniu Bayesa o prawdopodobieństwie warunkowym,
— budowa hiperpłaszczyzn (hiperpowierzchni) rozgraniczających w przestrzeni cech,
— k-Najbliższych Sąsiadów — metoda polegająca na zmierzeniu odległości wektora cech klasyfikowanego obiektu od sklasyfikowanych i wyboru tej kategorii, w której znajduje się k najbliższych obiektów.
— drzewa decyzyjne — rozpatruje się sekwencyjnie kolejne cechy klasyfikowanego obiektu i na tej podstawie dokonuje ostatecznej klasyfikacji,
— regresja logistyczna,
— ...

Zazwyczaj klasyfikacja wymaga ciągu uczącego, czyli zestawu obiektów, które są wstępnie sklasyfikowane. Na tej podstawie algorytm „dostraja się taki sposób, żeby jego decyzje były zgodne z zadeklarowanymi klasami obiektów. Bardzo często uczenie się ma kolejny etap, czyli sprawdzian: działanie algorytmu testowane jest na kolejnych preklasyfikowanych obiektach. Jeżeli wyniki nie są zadowalające — dostarcza się kolejny ciąg uczący.

Stąd przyjęła się nazwa „Uczenie maszynowe" lub *Machine Learning*.

Pamiętać trzeba, że nigdy decyzje klasyfikatora nie są w 100% poprawne.

W przypadku klasyfikacji binarnej można stworzyć „tablicę pomyłek” (*confusion matrix*). Niech nasz algorytm służy do decydowania czy pacjent jest chory czy zdrowy. Możemy mieć do czynienia z czterema sytuacjami:

1. Pacjent jest zdrowy i tak jest klasyfikowany przez algorytm; decyzja jest **prawdziwie pozytywna** czyli TP[1].
2. Pacjent jest zdrowy, ale klasyfikowany jako chory — **fałszywie negatywnie** czyli FN.
3. Pacjent jest chory, ale klasyfikowany jako zdrowy —**fałszywie pozytywnie**, FP,
4. Chory pacjent jest klasyfikowany jako chory — **prawdziwie negatywnie**, TN.

Przypadki klasyfikacji fałszywie pozytywnej określane są również mianem błędu pierwszego rodzaju, a przypadki fałszywie negatywne — błędy drugiego rodzaju.

Czasami używa się dodtkowych parametrów: czułość (TPR — *True Positive Rate*), swoistość (TNR — *True Negative Rate*), precyzja (PPV *Positive Predictive Value*) i dokładność (ACC *Accuracy*). Określone są one wzorami:

— TPR:

$$TPR = TP/(TP + FN)$$

— TNR:

$$TNR = TN/(FP + TN)$$

— PPV:

$$PPV = TP/(TP + FP)$$

— ACC:

$$ACC = (TP + TN)/(P + N)$$

gdzie: $P = TP + FN$, a $N = TN + FP$; $P + N$ to wielkość testowanej populacji.

1.1.2. Przykład

Poniższy przykład został wybrany z dokumentacji.

Do klasy $A = \{1, 2\}$ należą dwa elementy, do klasy $B = \{3, 4\}$ kolejne dwa. zbiory są rozłączne i klasyfikacja wydaje się prosta: wszystko co mniejsze od 2,5 to A, a pozostałe — B.

$\mathbf{TrainingSet = \{1 \to A, 2 \to A, 3 \to B, 4 \to B\};}$

$\mathbf{c = Classify[TrainingSet]}$

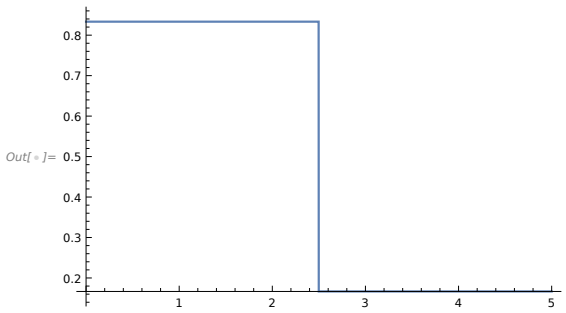
ClassifierFunction [□]

[1] True Positive

c[2.5, "Probabilities"]

$\mathrm{A} \to 0.166667, \mathrm{B} \to 0.833333$

Plot[c[x, "Probability" → A], {x, 0, 5}]



Gdy sytuację skomplikujemy i $A = \{1, 2, 3{,}1\}$, a $B = \{3, 3{,}15, 4, 5\}$ to zbiory znowu, są rozłączne, ale przedziały liczbowe „zachodzą" na siebie. Klasyfikacja nie jest już taka prosta.

TrainingSet = {1 → A, 2 → A, 3.1 → A, 3 → B, 3.15 → B, 4 → B, 5 → B};

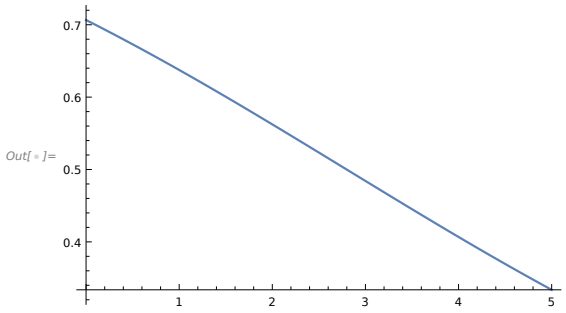
c = Classify[TrainingSet]

ClassifierFunction [□]

c[2.5, "Probabilities"]

$\mathrm{A} \to 0.523397, \mathrm{B} \to 0.476603$

Plot[c[x, "Probability" → A], {x, 0, 5}]



***c*[3]**

B

Co ciekawe, w pierwszym przypadku użyty klasyfikator to **Nearest Neighbors** (najbliżsi sąsiedzi), a w drugim **regresja logistyczna**.

1.1.3. „Odwrotna klasyfikacja”

Pod pojęciem „odwrotnej klasyfikacji” rozumiem zadanie stworzenia obiektu, który będzie miał określone cechy. Takie zadanie już bardziej nadaje się do określenia jako „sztuczna inteligencja” bo algorytm tworzy (na przykład obraz obiektu) mający pożądane cechy. Technologia nazywa się GAN (*Generative Adversarial Network*) i ostatnio nie schodzi z łam internetowych dzienników. Pojawiły się zestawy sztucznie generowanych twarzy, sztucznie generowanych kotków, a nawet obrazki ptaków (i kwiatów) generowane na podstawie słownego opisu („ptak z białym brzuszkiem i zółtym ogonem”) [1].

1.1.4. Lektury

Wydaje się, ża zacząć można od Tadeusiewicza [2] i [3]. Autor jest niewątpliwie autorytetem w tym zakresie. Po polsku dostępnych jest kilka prac: [4], [5], [6]. Tematyka jest tak gorąca, że można napotkać sporo obszernych źródeł (udostępnianych, na przykłąd, przed publikacją) lub jako materiały pomocnicze do kursów: [7], [8], [9]. Ciekawe jest też obszerne zetsawienie zasobów [10].

1.1.5. Uwagi

Sugeruję dokładne zapoznanie się z dokumentacją polecenia **Classify** dostępną bądź na stronach producenta lub — lepiej — w czasie zajęć, po otwarciu notatnika Mathematici. Dokumentacja jest interaktywna i można polecenia w helpach modyfikować i patrzeć na efekty lub kopiować do notatnika. Warto zapoznać się z całym rozdziałem dotyczącym uczenia maszynowego [11].

Polecenie Classify automatycznie tworzy procedurę (funkcję), która podany element klasyfikuje do kategorii.

Dostępne metody klasyfikacji to:
— "DecisionTree"
— “GradientBoostedTrees”
— “LogisticRegression” ’
— “Markov”
— “NaiveBayes”
— “NearestNeighbors”

— “NeuralNetwork”
— “RandomForest”
— “SupportVectorMachine”

Polecenie Classify może wybierać metodę klasyfikacji tak, aby optymalizować jedno z kryteriów:
— użycie pamięci,
— jakość klasyfikacji,
— szybkość klasyfikacji
— szybkość nauki
— może też wybrać kryterum automatycznie, lub użyć wszystkich danych jako danych uczących.

Dodatkowo, oprogramowanie ma już kilka wcześniej przygotowanych funkcji klasyfikujących, pozwalających na rozpoznawanie:
— flag,
— wieku na podstawie twarzy,
— płci na podstawie twarzy,
— języku w jakim napisany jest tekst (lub program)
— ...

1.1.6. Narzędzia

Nie ma **najlepszych** narzędzi do rozwiązywania żadnego problemu. Natomiast obliczenia na potrzeby sztucznej inteligencji prowadzone są/mogą być w różnych językach. Do najpopularniejszych należą:
— Python,
— R,
— Matlab,
— Wolfram Language,
— Julia,
— Prolog
— Lisp
— Java
— ...

2. Zajęcia

2.1. Wstęp

Zacznijmy od prostego przykładu z dokumentacji funkcji **Classify** programu Mathematica.

Wczytujemy dane korzystając z serwisu Gutenberg zawierającego pełne teksty utworów literackich dostępne na wolnych licencjach.

Najpierw Szekspir:

```
Othello =
      Import["http://www.gutenberg.org/cache/epub/2267/pg2267.txt"];
Hamlet =
      Import["http://www.gutenberg.org/cache/epub/2265/pg2265.txt"];
Macbeth =
      Import["http://www.gutenberg.org/cache/epub/2264/pg2264.txt"];
```

Później Hugo:

```
LesMiserables =
      Import["http://www.gutenberg.org/cache/epub/135/pg135.txt"];
NotreDamede =
      Import["http://www.gutenberg.org/cache/epub/2610/pg2610.txt"];
TheManWho =
      Import["http://www.gutenberg.org/cache/epub/12587/pg12587.txt"];
```

I wreszcie Wilde:

```
TheImportance =
      Import["http://www.gutenberg.org/cache/epub/844/pg844.txt"];
ThePicture =
      Import["http://www.gutenberg.org/cache/epub/174/pg174.txt"];
AnIdeal =
      Import["http://www.gutenberg.org/files/885/885-0.txt"];
```

Tworzymy funkcję klasyfikującą wiążąc po dwa teksty z Autorem.

```
author = Classify[<|"William Shakespeare" -> {Othello, Hamlet},
"Oscar Wilde" -> {TheImportance, ThePictureo},
```

```
"Victor Hugo" -> {LesMiserables, NotreDamede}|>]
```

Sprawdzamy pozostałe teksty

```
author[{Macbeth, AnIdeal, TheManWho}]
```

2.2. Zadania

1. Na podstawie dokumentacji opisz jak rozumiesz działanie funkcji `Classify` (niekoniecznie w przypadku rozpoznawania autorów tekstów, ale, być może, patrząc na prostsze przykłady — patrz rozdz. 1.1.2). **Oczekuję sprawozdania!**
2. Powtórz inne przykłady z dokumentacji lub twórczo wykorzystaj gotowe funkcje klasyfikujące.
3. Spróbuj korzystając z serwisu Wolne Lektury powtórzyć zadanie klasyfikacji wybierając jakichś trzech pisarzy. Wolne Lektury również oferują pliki w prostym formacie tekstowym.
4. Rozważmy, że mamy następujący problem (serię uczącą): 500 bananów, 300 pomarańczy i 200 innych owoców. Metodą organoleptyczną sprawdziliśmy które z nich są słodkie. Dodatkowo dokonalismy ich (wzrokowej) klasyfikacji na owoce „długie" i „krótkie". Wyniki przedstawia następująca tabelka:

Owoc	„Długi"	„Krótki"	Słodki	Nie słodki
Banan	400	100	350	150
Pomarańcza	0	300	150	150
Inne	100	100	150	50

Zadanie znakomicie nadaje się do zastosowania naiwnej metody Bayesa[1] do odpowiedzi na następujące pytanie: Jeżeli wylosujemy owoc, który jest długi, ale nie jest słodki, to jakie są szanse, że jest to banan?
Obliczenia można przeprowadzić bezpośrednio na prawdopodobienstwach [1], ale zaproponuj jak powinien wyglądać eksperyment (losowanie) uczenia maszynowego.
Mathematica ma funkcję losowego wyboru **RandomChoice**. Można jej użyć do wylosowania owocu, a później do przyporządkowania mu cech (zgodnie z zadanymi prawdopodoobieństwami). W ten sposób można wygenerować serię uczacą, która posłuży do wygenerowania funkcji klasyfikującej.
Warto sprawdzić jak funkcja klasyfikuje: gennerować elementy serii sprawdzającej (rodzaj owocu oraz jego cechy) i poddawać je klasyfikacji. Niestety, efekty klasyfikacji są zgodne jedynie w sensie probabilistycznym[2], a warto pamiętać

[1] Opisy są tu, tu czy tu.
[2] Co to znaczy?

(wiedzieć?), że tego typu klasyfikatorów używano do wskazywania celów dla dronów działjących na bliskim wschodzie...
Tak na marginesie: metody rozpoznawania spamu bardzo często oparte są na naiwnej metdozie klasyfikacji Bayesa.

2.3. Sprawozdanie

Laboratorium to może być relizowane na dwu zajeciach, ale powinno być zakończone obfitym sprawozdaniem i ciekawymi wynikami.

2.4. Instrukcja w postaci jednego pliku...

...jest również dostępna.

Bibliografia

[1] Ferreira H., *Basics of machine learning and a simple implementation of the naive bayes algorithm*, URL https://medium.com/hugo-ferreiras-blog/basics-of-machine-learning-and-a-simple-implementation-of-the-naive-bayes-algorithm 2018.

[2] Tadeusiewicz R., *Uporządkowane wiadomości na temat sztucznej inteligencji*, URL https://natemat.pl/blogi/ryszardtadeusiewicz/174271, uporzadkowane-wiadomosci-na-temat-sztucznej-inteligencji 2019.

[3] Tadeusiewicz R., *Nowa kolekcja wiadomości na temat sztucznej inteligencji*, URL https://natemat.pl/blogi/ryszardtadeusiewicz/272491, nowa-kolekcja-wiadomosci-na-temat-sztucznej-inteligencji 2019.

[4] Goodfellow I., Bengio Y., Courville A. (red.), *Deep Learning. Systemy uczące się*, Wydawnictwo Naupowe PWN, Warszawa 2018, URL https://libra.ibuk.pl/book/190795.

[5] Koronacki J., Ćwik J., *Statystyczne systemy uczące się*, Akademicka Oficyna Wydawnicza EXIT Andrzej Lang, Warszawa 2015, URL https://libra.ibuk.pl/book/148861.

[6] Raschka S., *Python. Uczenie maszynowe*, Helion 2017, URL https://nasbi.pl/21/pythum/podstawowe-szczegoly.html.

[7] Peyré G., *Mathematical Foundations of Data Sciences* 2019, URL https://mathematical-tours.github.io/book/.

[8] Zhang A., Lipton Z.C., Li M., Smola A.J., *Dive into Deep Learning* 2019, URL http://d2l.ai/.

[9] Gallier J., Quaintance J., *Algebra, Topology, Differential Calculus, and Optimization Theory For Computer Science and Machine Learning* 2019, URL https://drive.google.com/file/d/1sJvLQwxMyu89t2z4Zf9tD7O7efnbIUyB/view.

[10] Allen R., *My curated list of ai and machine learning resources from around the web*, URL https://medium.com/machine-learning-in-practice/my-curated-list-of-ai-and-machine-learning-resources-from-around-the-web-9a97823b85 2017.

[11] *Machine learning*, URL https://reference.wolfram.com/language/guide/MachineLearning.html 2019.