Uczenie maszynowe: Projekt - dokumentacja wstępna.

Jan Kwiatkowski, Jarosław Nachyła

February 8, 2023

1 Temat projektu

Las losowy z naiwnym klasyfikatorem bayesowskim (NBC) w zadaniu klasyfikacji. Postępujemy tak jak przy tworzeniu lasu losowego, tylko co drugi klasyfikator w lesie to NBC. Jeden z klasyfikatorów (NBC lub drzewo ID3) może pochodzić z istniejącej implementacji.

2 Algorytm budowania drzewa Id3

Do budowy drzewa będzie wykorzystywany algorytm ID3 o nazwie Iteratywny Dychotomizer 3 - ang. Iterative Dichotomiser 3. Jest to algorytm rekurencyjny, który tworzy drzewo w kolejności top down od korzenia do liścia metodą „dziel i zwyciężaj". Jego celem jest wybór atrybutów które będą tworzyć podział zbioru danych na zbiory o największej wartości zysku informacji IG – Information Gain (opisane w podrozdziale 2.3). Algorytm ten w każdym kroku wybiera atrybut, który najlepiej dzieli zbiór danych, mierząc ten podział przyrostem informacji po podziale zbioru na 2 części w danym węźle(drzewo binarne). Precyzyjny opis algorytmu znajduje się w punkcie 2.4.

2.1 Entropia

Entropia danego zbioru danych jest miarą nieuporządkowania docelowej cechy zbioru danych. Im jest większa tym nieuporządkowanie jest większe, z kolei im mniejsza tym w zbiorze panuje większy porzadek (istnieje wiecej jednoznacznych/podobnych wartości w zbiorze). Wartość entropii zdefiniowana jest nastepująco:

$$E(S) = -\sum_{x \in X} p(x) * log_2(x) \tag{1}$$

gdzie:

S - zbiór dla którego liczona jest entropia

X - zbiór klas w zbiorze S

$p(x)$ - stosunek liczby elementów z klasy x do liczby wszystkich elementów w zbiorze S

2.2 Entropia warunkowa

Entropia warunkowa bierze pod uwagę dwie zmienne losowe i daje odpowiedź jak jedna zmienna losowa pomaga w porządkowaniu wartości drugiej zmiennej losowej. Zbiór zdarzeń dzielimy na dwie grupy ze wzgledu na wybrane kryterium (atrybut „A").

$$H(X \mid A) = \sum_{a \in A} \sum_{x \in X} p(a, x) * \log \frac{p(a)}{p(a, x)} \tag{2}$$

gdzie:

$X \mid A$ - prawdopobieństwo zajścia X pod warunkiem A

A - zbiór atrybutów zbioru

X - zbiór klas w zbiorze S

$p(a, x)$ - stosunek liczby wierszy, które posiadają jednocześnie dla atrybutu wartość a i klasy x do liczby wszystkich wierszy

2.3 Information Gain (Zysk informacji)

Zysk informacji jest różnicą entropii całego zbioru przed podziałem i entropii po podziale. Jest miarą na podstawie której wybierany jest atrybut dla drzewa w ID3. Zysk informacji dla danego zbioru S i atrybutu A możemy zdefiniować następująco:

$$IG(S, A) = H(S) - \sum_{t \in T} p(t) * H(t) = H(S) - H(S \mid A) \tag{3}$$

gdzie:

$H(S)$ - funkcja entropii dla rozpatrywanego zbioru/ podzbioru

T - definiujemy zbiory uzyskane poprzez podział zbioru S względem wartości atrybutu A

t - t oznacza każdy podzbiór należący do T

$p(t)$ - jest to proporcja definiowana jako iloraz liczby elementów w t przez liczbę elementów w S

$H(t)$ - funkcja entropii dla podzbioru t

2.4 Formalny zapis algorytmu ID3 dla lasu losowego.

Algorytm Id3 dla lasu losowego posiada modyfikację w postaci losowania podzbioru atrybutów biorących udział w podziale drzewa. Losowanie atrybutów do podziału będzie sparametryzowane liczbą losowanych atrybutów - *split_features* z możliwymi watościami: None - wszystkie atrybuty, sqrt - pierwiatek l. atrybutów i log2 - logarytm l. atrybutów. Poniżej znajduje się pseudokod algorytmu ID3, który zostanie omówiony na przykładzie:

Algorithm 1 Algorytm ID3

Input: A - zbiór atrybutów, S - zbiór danych, ytarget - atrybut przewidywany, split_features- parametr określający losowanie podzbioru atrybutów dla drzewa (las losowy)

Output: $Node$ (główny węzęł zbudowanego drzewa)

1: **function** ID3(S, A)
2: **if** S jest pusty **then**
3: **return** stwórz węzeł
4: **end if**
5: **if** S zawiera te same wartości ytarget= v **then**
6: **return** stwórz liść(label=v)
7: **end if**
8: **if** A jest pusty **then**
9: **return** stwórz liść(label = najczęstsza wartość ytarget w S)
10: **end if**
11: $A\_to\_split \leftarrow$ losuj_podzbiór(A, split_features)
12: $Amax \leftarrow argmax(IG(S, A\_to\_split))$
13: $Node \leftarrow$ stwórz węzeł($Amax$)
14: **for** v_i in Amax.values **do**
15: Dodaj poddrzewo (Node, v_i)
16: $Sv_i \leftarrow$ S | $value = v_i$
17: **if** Sv_i jest pusty **then**
18: stwórz liść(label = najczęstsza wartość ytarget w S)
19: **else**
20: **return** ID3 (Sv_i, $A - \{Amax\}$)
21: **end if**
22: **end for**
23: **return** $Node$
24: **end function**

Postaramy się zaprezentować działanie algorytmu dla przykładowych danych. Poniżej zaprezentowany jest zbiór, na którym dokonana zostanie klasyfikacja - określenie czy pacjent jest zainfekowany czy nie (kolumna Infected). W opisywanym przykładzie dla uproszczenia przyjmiemy wartość split_features = None - bierzemy wszystkie atrybuty do podziału.

ID	Fever	Cough	Breathing issues	Infected
1	NO	NO	NO	NO
2	YES	YES	YES	YES
3	YES	YES	NO	NO
4	YES	NO	YES	YES
5	YES	YES	YES	YES
6	NO	YES	NO	NO
7	YES	NO	YES	YES
8	YES	NO	YES	YES
9	NO	YES	YES	YES
10	YES	YES	NO	YES
11	NO	YES	NO	NO
12	NO	YES	YES	YES
13	NO	YES	YES	NO
14	YES	YES	NO	NO

Stosując powyższy algorytm określamy paramatry wejściowe:

S - zbiór danych to przedstawiona tabela.

A - zbiór atrybutów to: Fever(Gorączka), Cough(Kaszel) i Breathing issues (Problemy z oddychaniem).

ytarget - atrybut przewidywany to kolumna Infected.

Zaczynając mamy niespełnione warunki linii pseudokodu od 2 do 10: zbióry S i A nie są puste i S nie zawiera tych samych wartości ytarget. Przechodząc do kolejnej lini kodu(12), wyznaczamy wartości Information Gain dla wszystkich atrybutów i bierzemy z nich atrybut o wartości maksymalnej.

Pokażemy obliczanie IG dla atrybutu Fever(gorączka).

Wyznaczmy najpierw Entropię dla całego zbioru S:

$$\text{E(S)} = \text{- (8/14) * } log_2(8/14) \text{ - (6/14) * } log_2(6/14) = 0.99$$

Poniżej w tabeli zliczono ile osób z gorączką i bez było zainfekowanych, przyda się to do dalszych wyliczeń.

Fever	Infected	Not Infected
YES	6	2
NO	2	4

Odnosząc się do wyrażenia na IG - Zysk informacji (podrozdział 2.3), obliczyliśmy H(S), a potrzebujemy jeszcze wartości H(t) i p(t), gdzie wartości t są poszczególnymi wartościami atrybutu Fever(YES,NO).Oznaczmy zbiory Syesfever i Snofever.

Liczymy entropię E dla wartości YES atrybutu Fever na zbiorze Syesfever , na podstawie powyższej tabeli.

$$E(Syesfever) = -(6/8) * log_2(6/8) - (2/8) * log_2(2/8) = 0.81$$

Analogicznie obliczamy entropię dla wartości NO atrybutu Fever na zbiorze Snofever.

$$E(Snofever) = -(2/6) * log_2(2/6) - (4/6) * log_2(4/6) = 0.91$$

Potrzebujemy jeszcze obliczyć częstości p(t). Dla wartości zmiennej przewidywanej Infected = YES jest to:

$$p(Fever = YES) = 8/14$$

Dla wartości zmiennej przewidywanej Infected = NO jest to:

$$p(Fever = NO) = 6/14$$

Możemy teraz policzyć wartość IG dla atrybutu Fever:

$$IG(S, A = Fever) = 0.99 - (8/14) * 0.81 - (6/14) * 0.91 = 0.13$$

Analogicznie liczymy wartość IG dla atrybutu Cough - Kaszel:

$$IG(S, A = Cough) = 0.04$$

IG dla Breathing Issues - problemy z oddychaniem jest nastepujące:

$$IG(S, A = BreathingIssues) = 0.4$$

Ostatni atrybut: Problemy z oddychaniem ma najwyższą wartość Information Gain = 0.4, a więc ten atrybut zostanie przypisany jako węzeł, a w tym przypadku także korzeń drzewa - (zmienna Nroot - linia 9).

Następnie iterujemy po możliwych wartościach wybranego atrybutu: {YES,NO} (pętla w liniach 14-22). W pierwszej iteracji, dla wartości YES w lini 15 dodajemy podrzewo które będzie zawierać przodków węzła Nroot. W lini 16 tworzony jest podzbiór naszego zbioru głównego dla S(A = Breathnig Issues,Breathnig Issues=YES). Poniższa tabela pokazuje ten podzbiór:

Fever	Cough	Breathing issues	Infected
YES	YES	YES	YES
YES	NO	YES	YES
YES	YES	YES	YES
YES	NO	YES	YES
YES	NO	YES	YES
NO	YES	YES	YES
NO	YES	YES	YES
NO	YES	YES	NO

Sprawdzamy następnie warunek w lini 17, ponieważ zbiór nie jest pusty wywołujemy rekurencyjnie z powrotem funkcję budowy drzewa - ID3 (linia 20). W przeciwnym razie, gdy zbiór Sv_i będzie pusty wykonana została by linia - 18. Wtedy musimy węzeł Nroot oznaczyć jako liść w drzewie i przypisać mu wartość, której jest najwięcej dla atrybutu Infected.

Wracamy do wywołania rekurencyjnego - linia 16. Przekazujemy do funkcji zbiór danych zdefiniowany w tabeli powyżej (S = S(A = Breathnig Issues,Breathnig Issues=YES)) i zbiór atrybutów z usuniętym atrybutem Breathing Issues {Fever, Caugh}. Warto zauważyć, że warunek z lini 5-7 jest warunkiem decydującym czy dokonywać kolejnych podziałów czy uznać węzęł jako liść. Postępując analogicznie możemy zbudować całe drzewo decyzyjne.

3 Klasyfikator Bayesowski NBC

3.1 Twierdzenie Bayesa

Algorytm ten oparty jest na rachunku prawdopodobieństwa. Na podstawie bazy etykietowanych przykładów D dla nowego przykładu d_j klasyfikator ma stwierdzić jaka jest najbardziej prawdopodobna wartość decyzji dla tego przykładu lub jakie jest prawdopodobieństwo przypisania nowego przykładu do jakiejś klasy na podstawie posiadanego zbioru treningowego.

Do wyznaczania wartości prawdopodobieństwa algorytm wykorzystuje zmodyfikowaną wersję twierdzenia o prawdopodobieństwie warunkowym przedstawionym poniżej:

$$P(h_i \mid d_j) = \frac{P(h_i) * P(d_j \mid h_i)}{P(d_j)} \tag{4}$$

gdzie:

$h_i \in H$ - klasa decyzyjna

d_j - nowy atrybut składający się z wartości $< a_1, ..., a_n >$

Taki zapis równania posiada jednak pewne słabości takie jak:

- liczebność danej klasy decyzyjnej h_i bezpośrednio głosuje za wynikiem (im większa liczba przykładów z danej klasy tym prawdopodobieństwo, że wynik będzie tej samej klasy również rośnie)
- istnieje możliwość iż dla danej decyzji h_i nie występuje krotka d_j o konkretnych wartościach atrybutu zawartych w analizowanym przykładzie.

3.2 Naiwny klasyfikator bayesowski

W związku przedstawionymi powyżej słabościami zamiast prawdopodobieństwa całej krotki $P(h_i) * P(d_j \mid h_i)$ możemy je przybliżyć korzystając z dodatkowego, „naiwnego" założeni, że poszczególne prawdopodobieństwa warunkowe są niezależne. Dla takich zdarzeń prawdopodobieństwo koniunkcji jest równe iloczynowi prawdopodobieństw wystąpienia każdej wartości atrybutu we wskazanej klasie.

$$P(h_i \mid d_j) = P(a_1 \mid h_i) * P(a_2 \mid h_i) * ... * P(a_n \mid h_i) \tag{5}$$

$$P(h_i \mid d_j) = \frac{\Pi P(a_j \mid h_i) * P(h_i)}{P(d_j)} \tag{6}$$

Powyższy wzór a dokładnie założenie o zamianie współwystępowania wartości atrybutów dla danej klasy decyzyjnej na występowanie iloczynu pojedyńczych wartości: $P(a_1 \mid h_i) * P(a_2 \mid h_i) * ... * P(a_n \mid h_i)$ świadczy o naiwności klasyfikatora Bayesowskiego.

W związku z tym, że zależy nam jedynie na stwierdzeniu, które z prawdopodobieństw $P(h_i \mid d_j)$ dla różnych h_i jest największe, to mianownik możemy potraktować jako stałą (nie zależy on od d_j) i pominąć w dalszych rozważaniach.

Ostatecznie łącząc powyższe przekształcenia i założenia otrzymujemy wzór na **Naiwny klasyfikator bayesowski**, który nie wykorzystuje żadnych innych hipotez do wyznaczenia wyniku:

$$argmax_{h_i \in H} P(h_i \mid a_1, ..., a_n) = C * P(a_1 \mid h_i) * P(a_2 \mid h_i) * ... * P(a_n \mid h_i) * P(h_i) \tag{7}$$

3.3 Implementacja algorytmu

Algorytm ten będzie pochodził z gotowej implementacji z biblioteki scikit-learn. Do implementacji zostanie użyty wariant - Categorical Naive Bayess znajdujący się w: sklearn.naive_bayes.CategoricalNB. Poniżej zamieszczamy link do dokumentacji klasyfikatora:
https://scikit-learn.org/stable/modules/generated /sklearn.naive_bayes.CategoricalNB.html

4 Las losowy

Algorytm lasu losowego, jako model bazowy wykorzystuje model drzewa. Podobnie jak pojedyncze drzewo decyzyjne jest zdolny do rozwiązywania zarówno problemów regresyjnych jak i klasyfikacyjnych. Ponieważ pojedyńcze drzewo jest bardzo wrażliwe na trenujący zbiór danych (istnieje prawdopodobieństwo, że będzie ono bardzo szczegółowe/przeuczone), tworzy się z nich las to znaczy zbiór drzew decyzyjnych. Zastosowanie lasu losowego ma na celu zwiększenie odporności algorytmu na przeuczenie (zredukowanie wariancji) oraz zwiększanie uogólnienia modelu (nie szukamy jednej szczegółowej hipotezy a tworzymy ich wiele).

Do nauki drzew decyzyjnych można wykorzystać jedną z dwóch metod:

- Bagging
- Boosting

W tworzonym przez nas systemie będziemy korzystać z metody bagging.

4.1 Tworzeniu lasu losowego metodą bagging

Podstawą algorytmu jest drzewo decyzyjne, które z założenia powinno być jak najprostsze. Ilość takich prostych klasyfikatorów w systemie może być bardzo duża. Każdy klasyfikator do nauki wykorzystuje pewną próbkę danych (część) z orginalnego zbioru.

W metodzie bagging, każdy element (atrybut) ma takie samo prawdopodobieństwo pojawienia się w nowej próbce danych treningowych. Takie podejście zmniejsza wariancję poprzez zwiększenie prawdopodobieństwa wylosowania próby, która została niepoprawnie zaklasyfikowana przez poprzedni klasyfikator (uśrednia model i pomaga uniknąć nadmiernego dopasowania).

Najpierw losowana jest ze zwracaniem wskazana liczba podzbiorów (bootstraping). Podzbiory te służą następnie uczeniu modeli algorytmem NBC lub

zmodyfikowanym algorytmem id3. Modyfikacja wykorzystywanego do budowy drzew algorytmu polega na ograniczeniu ilości atrybutów przekazanych w każdym podzbiorze algorytmowi id3 (opisane w 2.4 - split_features).

4.2 Predykcja lasu losowego

Predykcja lasu losowego polega na wskazaniu przez każde z drzew klasy do której należy obserwowany przykład (Aggregating). Następnie na podstawie wyników wskazywanych przez każdy z klasyfikatorów podawany jest ostateczny wynik, będący wynikiem przewidzianym przez większość drzew.

4.3 Algorytm trenowania lasu losowego

Zainicjalizuj listę modeli M.

dane wejściowe N - liczba modeli, S - zbiór danych

Dla każdego n : od 1 do N:

1. Wylosuj ze zwracaniem k-elementową próbę bootstrapową Sn ze zbioru S.
2. if n modulo 2 == 0:

   Mn = stwórz model przy użyciu drzewa używając algorytmu Id3 dla próby Sn.

   else:

   Mn = Stwórz model naiwnego klasyfikatora bayesowskiego dla próby Sn.
3. Dodaj model Mn do listy modeli M.

4.4 Algorytm predykcji lasu losowego

Zainicjalizuj listę P predykcji modeli M .

dane wejściowe Sp - przykład testowy

Dla każdego modelu Mn : lista modeli M:

1. pn = Wykonaj predykcję Mn przykładu Sp
2. Dodaj pn Do listy P

Zwróć wartość p równą większości wartości w P.

4.5 Błąd predykcji - OOB

Metoda OOB (out-of-bag) służy do estymacji błędów klasyfikacji oraz istotności poszczególnych zmiennych i jest alternatywą dla liczenia błędu z podziałem zbioru danych na treningowy i walidacyjny.

Błąd predykcji OOB liczony jest dla każdego modelu z lasu osobno na próbkach na których dany model nie był trenowany. W ten sposób jesteśmy w stanie wykorzystać efektywniej zbiór danych. Ma to zastosowanie w przypadku małych zbiorów danych gdzie mała ilość przykładów może wpływać negatywnie na naukę modelu i zwiększać błąd estymacji.

Implementacja Out of Bag zostanie rozważona jako alternatywna dla zbioru z małą ilością przykładów - zbiór z samochodami (5.2).

5 Zbiory danych

5.1 Zbiór klasyfikacji grzybów

Pierwszym zbiorem danych do testowania którego chcemy użyć jest zbiór Mushroom Dataset. Dostępny jest pod linkiem:
https://archive.ics.uci.edu/ml/datasets/Mushroom

Wybraliśmy go na początek jako średni zbiór - około 8 tysięcy przykładów do klasyfikacji. Służy on do określania czy dany grzyb jest jadalny(klasa - edible) czy trujący(klasa poisonous). Zbiór nie jest oryginalnie podzielony, a więc do oceny modeli zastosujemy walidację krzyżową.

Liczba przykładów	Atryb. dyskret.	Atryb. rzecz.	Brak. atryb.	Liczba klas
8124	22	0	2480	2

Podsumowanie zbioru: Mushrooms classification

Zbiór posiada aż 2480 atrybytów, które nie posiadają wartości i zostanie to uwzględnione w przetwarzaniu i implementacji.
Klasy w tym zbiorze są względnie równo dystrybuowane:

```
 class      N          N[%]
------------------------------

 edible     4208     (51.8%)
 poisonous  3916     (48.2%)
```

5.2 Zbiór do przewidywania jakości samochodów

Drugim zbiorem danych do testowania którego chcemy użyć jest zbiór Car Evaluation. Jest to mały zbiór zawierający 1728 przykładów. Wstępnie nie jest podzielony na zbiory testowy i treningowy, zastosujemy więc dla niego walidację krzyżową.
Dostępny jest pod linkiem:

https://archive.ics.uci.edu/ml/datasets/car+evaluation

Liczba przykładów	Atryb. dyskret.	Atryb. rzecz.	Brak. atryb.	Liczba klas
1728	6	0	0	4

Podsumowanie zbioru: Car Evaluation.

Zbiór posiada nierówno reprezentowane klasy. Poniżej zamieszczono dystrybucję klas w zbiorze:

```
 class      N          N[%]
------------------------------
unacc     1210     (70.023 %)
acc        384     (22.222 %)
good        69     ( 3.993 %)
v-good      65     ( 3.762 %)
```

Klasy: unacc i acc (nieakceptowalny i akceptowalny) to aż ponad 92% przykładów, a klasy good i v-good (dobry i bardzo dobry) to prawie 8%. Zostanie to uwzględnione przy analizie wyników (obliczaniu metryk per klasa), a także przy podziałach zbiorów na treningowy i testowy(zachowanie proporcji po podziale).

6 Plan eksperymentów i metryki klasyfikacji

Poniższe eksperymenty bedą przeprowadzone dla 2 opisanych zbiorów i oceniane metrykami opisanymi w 6.3 (accuracy, precision, recall i F1).

6.1 Plan eksperymentów

6.1.1 Eksperyment porównujący modele:

Do porównania użyjemy następujących modeli na 2 opisanych wcześniej zbiorach danych.

- zaimlementowany model: las losowy(drzewa ID3 i naiwny bayess)
- lasu losowego z biblioteki scikit-learn: sklearn.ensemble.RandomForestClassifier
- naiwny bayess z biblioteki scikit-learn: sklearn.naive_bayes.CategoricalNB

6.1.2 Eksperyment porównujący modyfikację zaimplementowanego modelu:

W eksperymencie porównamy zaimplementowany model dla różnych wartości parametrów:

- N - oznaczającego liczbę modeli w lesie: *10, 20, 50, 100, 200.*
- split_features: funkcja losowania atrybutów w drzewie Id3 dla lasu dla wartości: *None, sqrt i log2.*

6.2 Podział zbiorów

Do eksperymentów wykorzystamy dwa zbiory i postaramy się porównać uzyskane wyniki. Zbiory różnią się miedzy sobą ilością atrybutów, klas i przykładów więc spodziewamy się różnic. Zbiory danych wstępnie przygotujemy do klasyfikacji np. uwzględniając odpowiednie postępowanie z wartościami brakującymi. Nastepnie przetrenujemy zbudowany model na każdym z 2 zbiorów. Zbiory podzielimy stosując walidację krzyżową z podziałem równym 10 w celu osiągnięcia lepszej estymacji dokładności modeli. Dodatkowo w każdym splicie przetrenujemy model 10-krotnie, ponieważ algorytm lasu ma w sobie losowania. Będziemy więc uczyli 10*10 razy każdy model na każdym zbiorze kolejno zamieniając 9/10 frakcji zbioru treningowego i 1/10 frakcji zbioru testowego. Na uzyskanych wynikach policzymy metryki, które są opisane w kolejnych podrozdziałach: Accuracy, Precision, Recall i F. Na końcu policzymy także macierz pomyłek z uśrednieniem rezultatów.

6.3 Metryki

6.3.1 Accuracy

Accuracy czyli dokładność jest podstawową metryką używaną do oceny jakości klasyfikacji, zdefiniowana jest następująco:

$$accuracy = \frac{\text{liczba poprawnych predykcji}}{\text{liczba wszystkich predykcji}} \tag{8}$$

6.3.2 Precision, Recall, F1

Kolejną metryką do oceny jest Precyzja - ang. Precision, zdefiniowana jako stosunek liczby poprawnie sklasyfikowanych przykładów (True Positive) do liczby wszystkich przewidzianych przykładów Positive:

$$Precision = \frac{TruePositives}{TruePositives + FalsePositives} \tag{9}$$

Następnie możemy zdefiniować Czułość ang. Recall jako stosunek poprawnie sklasyfikowanych przykładów (TP) do wszystkich przykładów, które powinny zostać sklasyfikowane jako Positive(TP + FN):

$$Recall = \frac{TruePositives}{TruePositives + FalseNegatives} \tag{10}$$

Metryka F1 jest średnią harmoniczną Precyzji i Czułości:

$$F = \frac{2 * Precision * Recall}{Precision + Recall} \tag{11}$$

6.3.3 Macierz pomyłek

Dla zbioru o dużej liczbie klas przydatna może być też macierz pomyłek - pokazująca wartości błędnie sklasyfikowane na rzecz innych klas w wygodnej do analizy formie dwuwymiarowej.