Zastosowanie Informatyki w Medycynie

Komputerowe rozpoznawanie diagnozowania choroby niedokrwiennej u dzieci z wykorzystaniem algorytmu k-NN *

Szynal Paweł[226026] and Zdeb Kamil[235871]

Politechnika Wrocławska (Wrocław University of Science and Technology)

Streszczenie Dokument zawiera realizację pierwszego etapu tematu numer 6 projektu z kursu Zastosowania Informatyki w Medycynie. Został w nim przedstawiony cel projektu, opis problemu medycznego, charakterystyka klas danych, a także cech, na podstawie których odbędzie się klasyfikacja. Zawiera również opis algorytmu k-najbliższych sąsiadów oraz ranking cech sporządzony metodą chi-kwadrat. Opisano w nim również implementację środowiska eksperymentalnego oraz zebrane wyniki eksperymentu oraz analizy statystycznej wraz z zebranymi przez nas wnioskami.

Keywords: k-najbliższych sąsiadów · chi-kwadrat · anemia · test t-studenta

1 Cel projektu

Celem projektu jest nabycie umiejętności zastosowania algorytmu k-NN w zadaniu diagnostyki medycznej wraz z selekcją cech i eksperymentalną oceną skuteczności algorytmu na danych rzeczywistych. Projekt obejmuje sprawdzenie jak jakość klasyfikacji jest skorelowana z liczbą stosowanych cech. W naszym przypadku wybrany algorytm posłuży w próbie usprawnienia procesu diagnozowania choroby niedokrwiennej u dzieci.

* Realizowany w ramach zajęć projektowych Zastosowanie Informatyki w Medycynie PWR

2 Opis problemu medycznego

2.1 Podstawowe informacje na temat anemii

Anemia (Niedokrwistość) jest to stan chorobowy, w którym dochodzi do spadku stężenia hemoglobiny, liczby krwinek czerwonych RBC (erytrocytów), wskaźnika hematokrytowego poniżej wartości prawidłowych.

Niedokrwistość może być spowodowana utratą krwi, zmniejszoną produkcją czerwonych krwinek i zwiększonym rozpadem czerwonych krwinek. Przyczyny utraty krwi obejmują urazy i krwawienie z przewodu pokarmowego. Przyczyną anemi może być niedobór żelaza, witaminy B12 i szereg nowotworów szpiku kostnego. W przypadku osób wymagających operacji niedokrwistość może zwiększać ryzyko konieczności przetoczenia krwi po operacji.

Najczęstszą postacią niedokrwistości jestniedokrwistość niedoborowa związana z deficytem żelaza. Rzadziej występuje anemia hemolityczna (z rozpadu czerwonych krwinek), megaloblastyczna (charakteryzująca się zwiększoną objętością erytrocytów, np. w niedoborze witaminy B12) oraz aplastyczna (spadek poziomu wszystkich rodzajów krwinek na skutek zaniku szpiku kostnego). Wszystkie typy niedokrwistości, które zostały poruszone w naszym projekcie, zostały opisane w rozdziale 2.2. Charakterystyka danych.

Wyróżniamy następujące rodzaje niedokrwistości (klasyfikacja według przebiegu):

- **łagodną** - stężenie hemoglobiny wynosi od 10 g/dl do 12 g/dl,
- **umiarkowaną** – stężenie hemoglobiny wynosi od 8 g/dl do 9,9 g/dl,
- **ciężką** – stężenie hemoglobiny wynosi od 6,5 g/dl do 7,9 g/dl,
- **zagrażającą życiu** – stężenie hemoglobiny wynosi poniżej 6,5 g/dl.

Wczesne rozpoznanie anemii i wdrożenie odpowiedniego leczenia można wyleczyć i zapobiec spusteszeniu urganizmu.Dotyczy to anemii: z niedoboru żelaza, kwasu foliowego lub witaminy B12. W pozostałych przypadkach nie zawsze udaje się do końca wyleczyć pacjenta. Niedokrwistość jest najczęstszą chorobą krwi, dotykającą około jednej trzeciej światowej populacji.[8] Niedokrwistość z niedoboru żelaza dotyka prawie 1 miliard ludzi.

[12] W 2013 r. Niedokrwistość jest jednym z sześciu globalnych celów żywieniowych WHO [3] na 2025 r.

2.2 Charakterystyka danych

Wszystkie klasy oraz cechy zostały dokładnie opisane poniżej.

Klasy jednostki chorobowej

1. **Niedokrwistość normocytowa** - Anemia wywołana przez monochromatyczne występowanie mikrocytów ($Hb > 9\frac{g}{dl}$).

2. **Niedokrwistość megaloblastyczna** - Anemia wywołana brakiem witaminy B12 oraz kwasu foliowego.
3. **Niedokrwistość z niedoboru żelaza** - Anemia wywołana brakiem żelaza w organizmie.
4. **Pierwotna niedokrwistość aplastyczna** - Wrodzona aplazja szpiku spowodowana zaburzeniem prawidłowego działania komórek macierzystych w szpiku kostnym. Niedokrwistości aplastyczne są wynikiem zaniku utkania szpikowego, przez zastąpienie go tkanką łączną lub tłuszczową, co powoduje zanik wszystkich elementów krwi.
5. **Wtórna niedokrwistość aplastyczna** - Nabyta aplazja szpiku spowodowana zaburzeniem prawidłowego działania komórek macierzystych w szpiku kostnym.
6. **Wrodzona sferocytoza** - Najczęściej diagnozowana wrodzona anemia hemolityczna. W przebiegu choroby krwinki czerwone przyjmują kulisty kształt, zamiast prawidłowego dwuwklęsłego, co sprzyja ich niszczeniu.
7. **Wrodzona stomatocytoza** - Niedokrwistości hemolityczna, wywołana defektami błon erytrocytów.
8. **Wrodzona eliptocytoza** - Rzadkie zaburzenie genetyczne dotyczące błony komórkowej erytrocytów,charakteryzującym się niedokrwistością hemolityczną
9. **Akantocytoza** - Obecność akantocytów we krwi obwodowej.
10. **Niedokrwistość wywołana niedoborem G-6-PD** - Choroba spowodowana mutacją w genie G6PD. Prowadzi do utleniania grup sulfhydrylowych hemoglobiny i białek błony erytrocytu oraz wewnątrznaczyniowej hemolizy.
11. **Kinaza pirogronianowa** - Defekt enzymatyczny krwinek czerwonych polegający na braku lub niedoborze ważnego enzymu szlaku glikolizy.
12. **Niedokrwistość śródziemnomorska** - Ilościowe zaburzenia syntezy hemoglobiny, spowodowane wrodzonym defektem biosyntezy łańcuchów globiny.
13. **Niedokrwistość sierpowatokrwinkowa** - Zagrażająca życiu choroba krwi, powoduje zlepianie się krwinek, co prowadzi do zatorów w naczyniach krwionośnych.
14. **Niedokrwistość autoimmunohemolityczna** - Najczęściej występująca anemia hemolityczna. Wywoływana jest przez przeciwciała skierowane przeciwko własnym krwinkom czerwonym.
15. **Połowiczna niedokrwistość immunohemolityczna** - Choroba z grupy niedokrwistość hemolityczna charakteryzująca się skróceniem czasu połowicznego rozpadu erytrocytów.
16. **Niedokrwistość jatrogenna** - Niedokrwistość spowodowana następstwem nieodpowiedniego leczenia pacjenta.
17. **Krwotoczna utrata krwi** - Silne krwawienie, gwałtowna utrata krwi w jej pełnym składzie na skutek choroby. Przy anemii może to dotyczyć hemofilii - choroby krzepnięcia krwi.
18. **Krwotok wywołany ankilostomozą** - Najczęstszy objawy wywołane przez ankilostomozę - chorobę zakaźną określaną inaczej anemią górników lub chorobą tęgoryjcową. Do rozwoju choroby dochodzi na skutek zarażenia tęgoryjcem dwunastnicy.

19. **Krwotok wywołany wrzodem jelita** - Krwawienie, w którym krew przedostaje się do światła przewodu pokarmowego. Zwykle jest bezobjawowe i objawia się niedokrwistością z niedoboru żelaza.
20. **Krwotok wywołany nadżerką** - Krwawienie z dróg rozrodczych kobiety, wywołane zmianą patologiczną.

Cechy i ich charakterystyka Podane razem z danymi cechy na podstawie których odbywać się będzie klasyfikacja wyglądają następująco:

NR	Nazwa cechy	Wartości
Obraz krwi		
1.	Koncentracja hemoglobiny (g/100 ml)	1 - powyżej 12 2- od 9 do 12 3- od 6 do 9 4- od 3 do 6 5- poniżej 3
2.	Liczba erytrocytów (10ˆ4)	1- powyżej 350 2- od 300 do 350 3- od 250 do 300 4- od 100 do 250 5- poniżej 100
3.	Średnia objętość krwinki MCV	1- poniżej 80 2- od 80 do 96 3- powyżej 96
4.	Średnie stężenie HB w krwince MCHC (%)	1- poniżej 30 2- od 30 do 36
5.	Wielkość erytrocytów	1- powiększone 2- zmniejszone 3- normalne
6.	Rodzaj erytrocytów	1- sferyczne 2- eliptyczne 3- stomatyczne 4- kolczyste 5- sierpowate 6- owalne 7- zygzakowate
7.	Tkanka siateczkowata	1- normalna 2- powiększona 3- zmniejszona
Obraz Szpiku		
8.	Szpik kostny	1- krańcowo aktywny 2- średnio aktywny 3- aktywny 4- mało aktywny 5- nieaktywny
Stan komórki		
9.	Wielkość	1- duża 2 - mała
10.	Stosunek jądrowo-cytoplazmatyczny	1- duży 2- mały
11.	Rodzaj jądra	1- sferyczne 2- listkowe 3- zdeformowane
12.	Struktura chromatyny jądrowej	1- drobna 2- rozrzucona 3- chropowata 4- spoista
13.	Jąderko	1- obecne 2- nieobecne

NR	Nazwa cechy	Wartości
Stan komórki		
14.	Pasożyty	1- obecne 2- nieobecne
15.	Ziarenka żelaza	1- typu I 2- typu II 3- typu III 4 - typu IV
Osocze		
16.	Poziom żelaza	1- obniżony 2- normalny
17.	Poziom trwałych związków żelaza	1- podwyższony 2- obniżony
18.	Poziom witaminy B12	1- podwyższony 2- obniżony
19.	Poziom kwasu foliowego	1- podwyższony 2- obniżony
Test odpornościowy		
21.	Reakcja	1- negatywna 2- pozytywna
Test urobilinowy, urobilinogenowy, urobilirubinowy		
22.	Reakcja	1- negatywna 2- pozytywna
Test ruchliwości komórki		
23.	Reakcja	1- negatywna 2- pozytywna
Wrażenia kliniczne		
24.	Płeć	1- mężczyzna 2- kobieta
25.	Wiek	1- poniżej miesiąca 2- od miesiąca do roku 3- od roku do 3 lat 4- od 3 do 6 lat 5- od 6 do 9 lat 6- powyżej 9 lat
26.	Gorączka	1- obecna 2- nieobecna
27.	Krwawienie	1- obecne 2- nieobecne
28.	Skóra	1- blada 2- żółta 3- sina 4- obrzęknięta
29.	Węzły chłonne	1- powiększone 2- niepowiększone
30.	Szmery secowe	1- obecne 2- nieobecne
31.	Wątroba, śledziona	1- powiększone 2- niepowiększone

3 Ranking cech pod względem ich przydatności do klasyfikacji

3.1 Test zgodności chi-kwadrat

Ranking cech został sporządzony z wykorzystaniem biblioteki sci-kit learn do języka Python. Na podstawie zaimportowanego wektora z cechami oraz klasami, do których przyporządkowano obiekty przeprowadza test zgodności chi-kwadrat[6]. Metoda zwraca wartości chi-kwadrat oraz wartość p. Wartość p to prawdopodobieństwo tego, że dana zależność mogła wystąpić przypadkowo. Im niższa ta wartość, tym większe znaczenie ma dana cecha i tym wyżej znajduje się z w rankingu cech.

Test zgodności chi-kwadrat jest używany w statystyce do przetestowania niezależności dwóch zdarzeń. Porównuje on zaobserwowane dane z rozkładem prawdopodobieństwa chi-kwadrat. Mierzy jak wartość zmierzona oraz wartość oczekiwania się od siebie różnią. Im wyższa jest ta wartość, tym bardziej prawdopodobne jest to, że dana cecha wpływa na klasyfikację w sposób znaczący. Test zgodności korzysta z następującego wzoru.

$$x_c^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

gdzie:

O_i - wartość zaobserwowana

E_i - wartość oczekiwana

c - liczba stopni swobody, czyli (liczba klas-1)*(liczba możliwych wartości -1)

3.2 Kod rankingu cech

Kod, który odpowiada za przeprowadzenie rankingu cech prezentuje się następująco. Wykorzystuje on edytowany plik Excela z danymi w taki sposób, aby przy każdej próbce znajdował się numer klasy, do której ona należy. Taka edycja pliku pozwoliła na proste przeprowadzenie testu zgodności eliminując konieczność operacji przeprowadzanych na wczytanej z pliku macierzy przed podaniem jej na wejście funkcji przeprowadzającej test chi-kwadrat. Po przeprowadzeniu testu macierze z wartościami chi-kwadrat i p zostają wypisane na ekran.

```
import pandas as pd
from sklearn import feature_selection

input_file = "anemia.xlsx"
df = pd.read_excel(input_file, header=0, engine='openpyxl',)
x = df.iloc[:, 2:]
y = df.iloc[:, 0]
chi2, pval = feature_selection.chi2(x, y)
print(chi2)
print(pval)
```

3.3 Ranking

Wartości chi-kwadrat oraz p zostały zebrane w tabeli, która została posortowana zgodnie z wartością p. Zgodnie z przyjętym poziomem istotności (0.05) za istotne można uznać dane mające p na poziomie mniejszym od 0.05, a więc pierwsze 10 cech z tabeli uznanych zostało za istotne.

Nr Cechy	**Chi-kwadrat**	**Wartość p**
6	268.3222	5.902E-46
8	131.5664	7.363E-19
2	81.7284	9.348E-10
12	71.3369	5.506E-08
15	70.9707	6.339E-08
11	61.0469	2.640E-06
3	52.1683	6.243E-05
5	49.5012	1.551E-04
1	42.6910	1.428E-03
25	34.9065	1.433E-02
26	19.5977	4.191E-01
7	18.4541	4.923E-01
13	15.2396	7.073E-01
17	14.8839	7.299E-01
9	12.9984	8.387E-01
19	11.9207	8.890E-01
4	11.8910	8.902E-01
16	11.2797	9.141E-01
14	11.1366	9.192E-01
23	11.0629	9.217E-01
10	10.6678	9.345E-01
24	8.3011	9.834E-01
18	8.0786	9.859E-01
20	8.0037	9.866E-01
27	7.7946	9.886E-01
28	7.6336	9.900E-01
22	7.0778	9.938E-01
21	5.8679	9.982E-01
29	3.7351	9.999E-01
30	3.0138	1.000E+00
31	1.7623	1.000E+00

Tabela 1: Ranking cech

4 Algorytm k-najbliższych sąsiadów

Algorytm k-najbliższych (ang. k-nearest neighbors, knn)[9] sąsiadów jest jednym z algorytmów uczenia nadzorowanego wykorzystywanym w uczeniu maszy-

nowym oraz data science. Algorytm ten klasyfikuje dane na podstawie tego jak podobne są do innych danych. Metoda K Najbliższych Sąsiadów (k-Nearest Neighbors) należy do grupy algorytmów leniwych (lazy algorithms), czyli takich, które nie tworzą wewnętrznej reprezentacji wiedzy o problemie na podstawie danych uczących, lecz szukają rozwiązania dopiero w momencie pojawienia się wzorca testowego do klasyfikacji. Metoda przechowuje wszystkie wzorce uczące, względem których wyznacza odległość wobec wzorca testowego.

4.1 Zbiór wzorców uczących

W problemach klasyfikacji, głównym zadaniem jest stworzenie algorytmu (programu), który na podstawie znanych sobie, opisanych wzorców, będzie w stanie efektywnie rozpoznawać wzorce nieopisane i dotąd sobie nieznane. Zbiór wzorców uczących (learning patterns, training examples) składa się ze zbioru par $< x^i, y^i >$, gdzie x^i jest zbiorem parametrów $x^i = (x^i_1, ...,^i_n)$ definiujących obiekty (zwykle w postaci wektora lub macierzy danych), zaś y^i jest wartością przewidywaną/powiązaną/skojarzoną, np. indeksem lub nazwą klasy, do której obiekt x^i należy i którą razem z innymi obiektami tej klasy definiuje.

4.2 Algorytm

Algorytm knn przedstawia się następująco:

1. Podziel dane na zbiór uczący i testujący.
2. Wybierz parametr k, który wskazuje na to ilu najbliższych sąsiadów będzie branych pod uwagę.
3. Dla każdej danej ze zbioru testującego oblicz dystans od danych ze zbioru uczącego za pomocą wybranej miary odległości.
4. Weź k obiektów z najmniejszym dystansem.
5. Sprawdź, do której klasy należy najwięcej wybranych obiektów.
6. Przydziel dane, do klasy, która wystąpiła najczęściej.

4.3 Klasyfikacja

W fazie klasyfikacji k jest stałą zdefiniowaną przez użytkownika, a nieoznaczony wektor (zapytanie lub punkt testowy) jest klasyfikowany przez przypisanie etykiety, która występuje najczęściej wśród k próbek uczących znajdujących się najbliżej tego punktu zapytania.

4.4 Miary odległości

Szczególnie ważne jest przyjęcie właściwej odległości [1], a w zasadzie miary niepodobieństwa obiektów. Funkcję $\rho : \chi \times \chi \rightarrow R$ nazywamy miarą niepodobieństwa jeśli:

1. $\rho(x, y) \geqslant 0$
2. $\rho(x, y) = 0$ wtedy i tyko wtedy, gdy $x = y$
3. $\rho(y, x) = \rho(x, y)$

Metryka euklidesowa

- Niech $x, y \in R$
- oraz $x = (x_1, x_2, ..., x_n)$ $y = (y_1, y_2, ..., y_n)$
- Metryka euklidesowa zdefiniowana jest wzorem:

$$\rho_e(x, y) = \sqrt{\sum_{k=1}^{n}(x_k - y_k)^2} = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + ... + (x_n - y_n)}$$

Metryka Czebyszewa, szachowa

- Niech $x, y \in R^n$
- oraz $x = (x_1, x_2, ..., x_n)$ $y = (y_1, y_2, ..., y_n)$
- Metryka Czebyszewa zdefiniowana jest wzorem:

$$\rho_{ch}(x, y) = imax) \left|x_i - y_i\right| = \lim_{m \to \propto} \left(\sum_{i=1}^{n} \left|x_i - y_i\right|^m\right)^{\frac{1}{m}}$$

Metryka kolejowa, centrum, węzła kolejowego

- Niech $x, y \in R^2$
- oraz $x = (x_1, x_2)$ $y = (y_1, y_2)$
- Metryka kolejowa zdefiniowana jest wzorem:

$$\rho_k(x, y) = \begin{cases} \rho_e(x, y) & \text{gdy } x, y \text{ oraz } \theta \text{ leżą na jednej prostej} \\ \rho_e(x, \theta) + d_3(\theta, y) & \text{w przeciwnym wypadku} \end{cases}$$

Odległość dwóch punktów w tej metryce jest sumą euklidesowych ich odległości od punktu = (0,0) lub – w przypadku, kiedy prosta łącząca te punkty przechodzi przez punkt – zwykła euklidesowa odległość.

Metryka Frécheta

- Niech $x, y \in R$
- oraz $x = (x_1, x_2, ..., x_n)$ $y = (y_1, y_2, ..., y_n)$
- Metryka Frécheta zdefiniowana jest wzorem:

$$\rho_{ch}(x, y) = \sum_{i=1} \frac{\left|x_i - y_i\right|}{1 + 2^i \left|x_i + y_i\right|}$$

4.5 Metryka Manhattan

- Niech $x, y \in R$
- odległość dwóch punktów w tej metryce to suma wartości bezwzględnych różnic ich współrzędnych
- W przestrzeni R^n metryka Manhattan zdefiniowana jest wzorem:

$$\rho_k(x, y) = \sum_{k=1}^{n} \left|x_k - y_k\right|$$

5 Plan eksperymentu

Za istotne statystycznie zostało uznane pierwsze 10 cech z rankingu cech, ponieważ dały p-wartość poniżej przyjętego poziomu istotności. Zostanie wykorzystanych 6 klasyfikatorów. Każdy z klasyfikatorów wykorzysta inną kombinację 3 wartości liczby sąsiadów (5,9,16) oraz 2 metryk (metryka euklidesowa i metryka Manhattan). Po wczytaniu z pliku danych z pliku dla każdego z klasyfikatorów i pierwszej z istotnych cech zostanie przeprowadzona klasyfikacja danych testowych. Po jej dokonaniu zostanie dodana kolejna z cech i proces powtórzony. Aby zapewnić wiarygodność osiągniętych wyników klasyfikacji została wykorzystana 5-krotnie powtórzona 2-krotna walidacja krzyżowa [5]. Po ukończeniu klasyfikacji dane dla każdego klasyfikatora i liczby cech zostaną uśrednione. Dla każdego klasyfikatora zostanie wybrana liczba cech, dla której klasyfikator osiągnął najlepsze wyniki. Następnie serie wyników, które miały najlepszą średnią dla danego klasyfikatora, zostaną porównane z analogicznymi seriami wyników dla innych klasyfikatorów za pomocą testu parowego t-studenta [4].

6 Środowisko programistyczne

Środowisko eksperymentalne zostało zaprojektowane z wykorzystaniem biblioteki scikit-learn [11] do języka Python[2]. Zostały także użyte takie biblioteki jak Numpy[7] czy Pandas[13]. Test parowy t-studenta został przeprowadzony przy wykorzystaniu biblioteki scipy[14], która jest jedną z zależności biblioteki scikit-learn.

Program działa w ten sposób, że na podstawie listy z 10 najlepszymi cechami z rankingu cech, wybiera pierwszą cechę, a następnie na jej podstawie klasyfikuje dane. Program używa 5-krotnie powtórzonej 2-krotnej walidacji krzyżowej[5] dla każdego zestawu metryk oraz liczby sąsiadów, aby zwiększyć wiarygodność otrzymaych wyników.. Wyniki zapisuje do macierzy. Następnie dodawana jest kolejna cecha z listy cech i proces jest powtórzony. Po 10 krotnym wykonaniu takiej pętli wyniki są uśredniane, a następnie wybierane są najlepsze wyniki dla danego klasyfikatora, które są wykorzystywane do przeprowadzenia parowego testu t-studenta. Poza wypisaniem na ekran wyniku testu wypisane są również macierz przewagi oraz macierz istotności.

6.1 Kod

```
import numpy as np
import pandas as pd
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split,\
    RepeatedStratifiedKFold
from sklearn.base import clone
from scipy.stats import ttest_ind
```

```
import xlrd
from tabulate import tabulate

np.set_printoptions(suppress=True)
input_file = "ANEMIA.XLS"
features = [7, 9, 3, 13, 16, 12, 4, 6, 2, 26]

# Import data

with xlrd.open_workbook(input_file) as wb:
    df = pd.ExcelFile(wb).parse()
    df["Unnamed: 0"].fillna(method='pad', inplace=True)
    data = df.to_numpy()
clfs = {
        "man5": KNeighborsClassifier(n_neighbors=5, p=1),
        "euk5": KNeighborsClassifier(n_neighbors=5, p=2),
        "man9": KNeighborsClassifier(n_neighbors=9, p=1),
        "euk9": KNeighborsClassifier(n_neighbors=9, p=2),
        "man16": KNeighborsClassifier(n_neighbors=16, p=1),
        "euk16": KNeighborsClassifier(n_neighbors=16, p=2),
    }
n_splits = 2
n_repeats = 5
rskf = RepeatedStratifiedKFold(n_repeats=n_repeats,
n_splits=n_splits, random_state=312)
scores = np.zeros((len(clfs), len(features),
n_splits * n_repeats))
for i in range(len(features)):
    features_selected = features[0: i + 1]
    # print(features_selected)
    X = data[:, features_selected]
    y = data[:, 0].astype(int)
    # Replace Nan with previous number
    X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=.3,
    random_state=42,
    )

    for fold_id, (train, test) in enumerate(rskf.split(X, y)):
        for clf_id, clf_name in enumerate(clfs):
            clf = clone(clfs[clf_name])
            clf.fit(X[train], y[train])
            y_pred = clf.predict(X[test])
            scores[clf_id, i, fold_id] = accuracy_score(
```

```
            y[test], y_pred)
np.save('results', scores)

scores = np.load('results.npy')
print("\nScores:\n", scores.shape)
mean_scores = np.mean(scores, axis=2).T
print("\nMean scores:\n", mean_scores)
best_indices = mean_scores.argmax(axis=0)
print(best_indices)
best_scores = np.zeros((len(clfs), n_splits * n_repeats))
for i, clf_id in enumerate(scores):
    best_scores[i] = clf_id[best_indices[i]]

alfa = .05
t_statistic = np.zeros((len(clfs), len(clfs)))
p_value = np.zeros((len(clfs), len(clfs)))

for i in range(len(clfs)):
    for j in range(len(clfs)):
        t_statistic[i, j], p_value[i, j] = ttest_ind(
        best_scores[i], best_scores[j])
print("t-statistic:\n", t_statistic, "\n\np-value:\n", p_value)

headers = ["Man5", "Euk5", "Man9", "Euk9", "Man16", "Euk16" ]
names_column = np.array([["Man5"], ["Euk5"], ["Man9"], ["Euk9"],
["Man16"], ["Euk16"]])
t_statistic_table = np.concatenate((names_column, t_statistic), axis=1)
t_statistic_table = tabulate(t_statistic_table, headers, floatfmt=".2f")
p_value_table = np.concatenate((names_column, p_value), axis=1)
p_value_table = tabulate(p_value_table, headers, floatfmt=".2f")
print("t-statistic:\n", t_statistic_table, "\n\np-value:\n",
p_value_table)
advantage = np.zeros((len(clfs), len(clfs)))
advantage[t_statistic > 0] = 1
advantage_table = tabulate(np.concatenate(
    (names_column, advantage), axis=1), headers)
print("Advantage:\n", advantage_table)
significance = np.zeros((len(clfs), len(clfs)))
significance[p_value <= alfa] = 1
significance_table = tabulate(np.concatenate(
    (names_column, significance), axis=1), headers)
print("Statistical significance (alpha = 0.05):\n", significance_table)
```

7 Wyniki eksperymentu

7.1 Wyniki ewaluacji eksperymentalnej

Wyniki zostały zebrane za pomocą programu za pomocą algorytmu knn z wykorzystaniem protokołu badawczego 5-krotnie powtórzonej 2-krotnej walidacji krzyżowej.

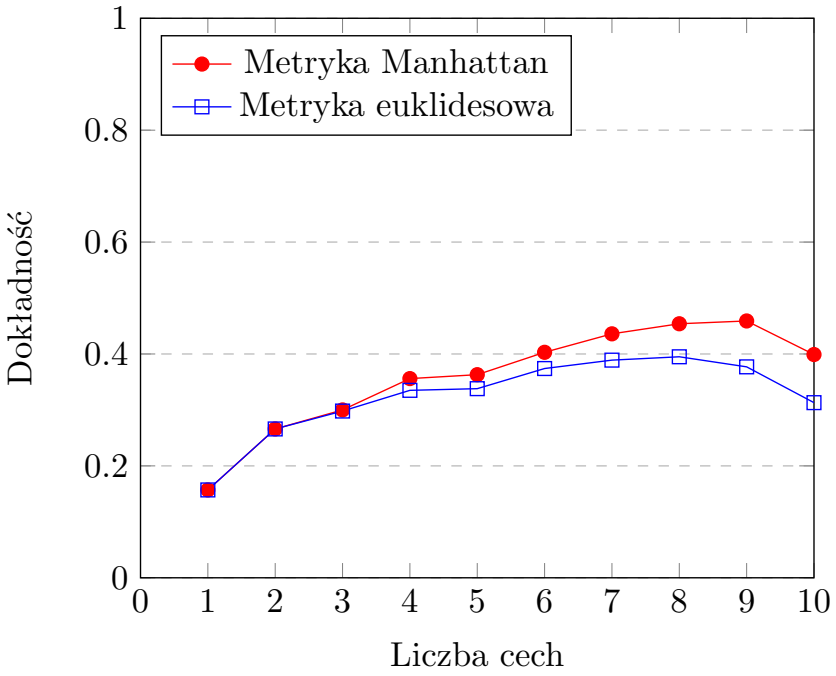
Sąsiedzi	**Metryka**	**Cechy**	**Wartość**
5	Manhattan	1	0.157
5	Euklidesowa	1	0.157
9	Manhattan	1	0.159
9	Euklidesowa	1	0.159
16	Manhattan	1	0.177
16	Euklidesowa	1	0.177
5	Manhattan	2	0.266
5	Euklidesowa	2	0.266
9	Manhattan	2	0.279
9	Euklidesowa	2	0.277
16	Manhattan	2	0.270
16	Euklidesowa	2	0.269
5	Manhattan	3	0.300
5	Euklidesowa	3	0.298
9	Manhattan	3	0.322
9	Euklidesowa	3	0.307
16	Manhattan	3	0.341
16	Euklidesowa	3	0.307
5	Manhattan	4	0.356
5	Euklidesowa	4	0.335
9	Manhattan	4	0.363
9	Euklidesowa	4	0.337
16	Manhattan	4	0.347
16	Euklidesowa	4	0.303
5	Manhattan	5	0.363
5	Euklidesowa	5	0.338
9	Manhattan	5	0.378
9	Euklidesowa	5	0.336
16	Manhattan	5	0.371
16	Euklidesowa	5	0.300

Sąsiedzi	**Metryka**	**Cechy**	**Wartość**
5	Manhattan	6	0.403
5	Euklidesowa	6	0.374
9	Manhattan	6	0.419
9	Euklidesowa	6	0.357
16	Manhattan	6	0.423
16	Euklidesowa	6	0.330
5	Manhattan	7	0.436
5	Euklidesowa	7	0.389
9	Manhattan	7	0.436
9	Euklidesowa	7	0.377
16	Manhattan	7	0.434
16	Euklidesowa	7	0.337
5	Manhattan	8	0.454
5	Euklidesowa	8	0.395
9	Manhattan	8	0.471
9	Euklidesowa	8	0.361
16	Manhattan	8	0.458
16	Euklidesowa	8	0.343
5	Manhattan	9	0.459
5	Euklidesowa	9	0.377
9	Manhattan	9	0.462
9	Euklidesowa	9	0.369
16	Manhattan	9	0.442
16	Euklidesowa	9	0.344
5	Manhattan	10	0.399
5	Euklidesowa	10	0.313
9	Manhattan	10	0.412
9	Euklidesowa	10	0.304
16	Manhattan	10	0.421
16	Euklidesowa	10	0.290

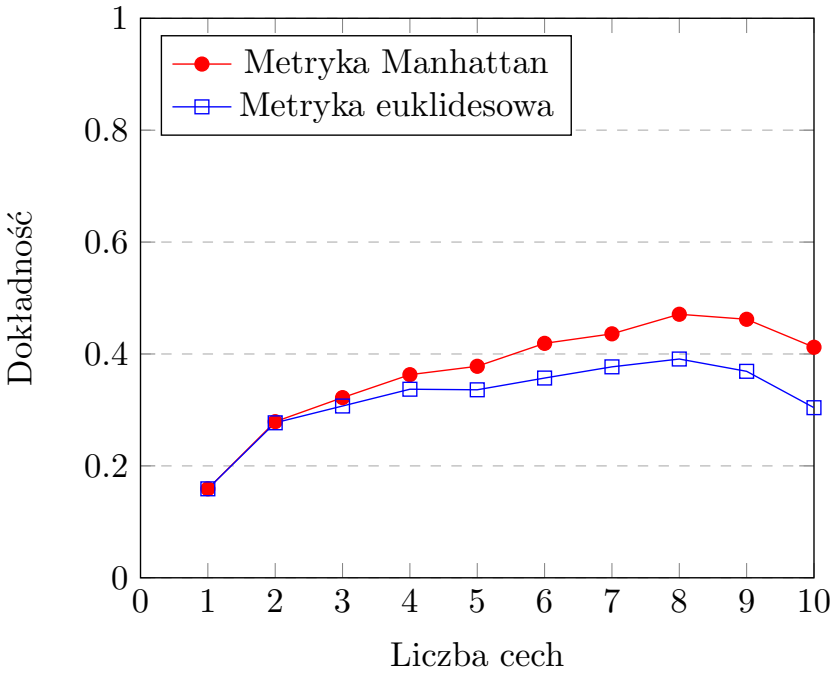
Tabela 2: Uśrednione wyniki dla każdego eksperymentu

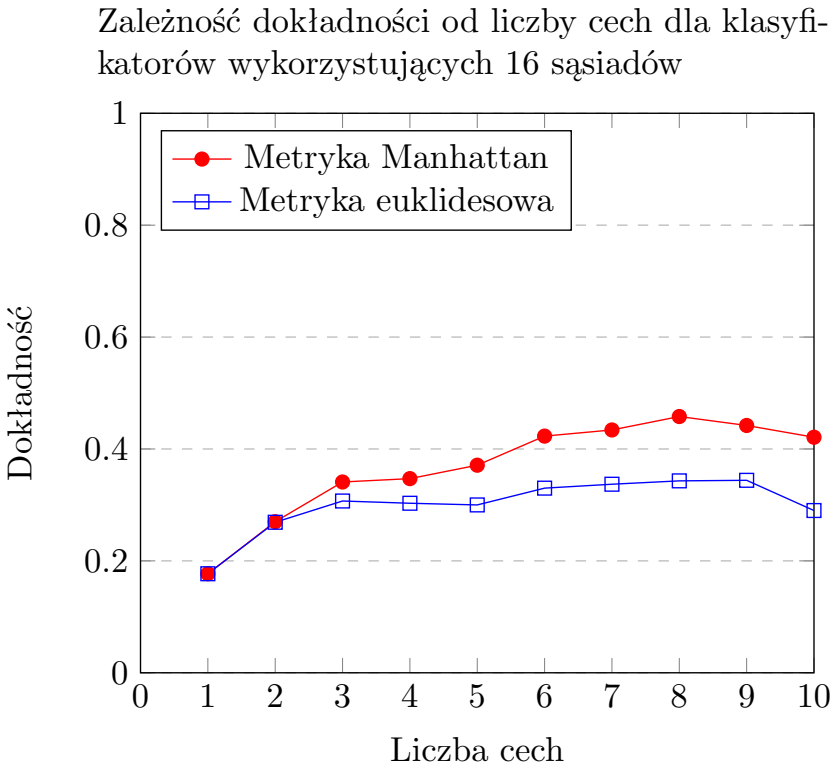
Wyniki dla klasyfikatorów wykorzystujących taką samą liczbę sąsiadów i różne metryki zostały ze sobą przedstawione na poniższych wykresach.

Zależność dokładności od liczby cech dla klasyfikatorów wykorzystujących 5 sąsiadów



Zależność dokładności od liczby cech dla klasyfikatorów wykorzystujących 9 sąsiadów





Na podstawie powyższych wyników dla każdego z klasyfikatorów została wybrana liczba cech, dla których spisał się on najlepiej. Przedstawia się to następująco:

- 5 sąsiadów, metryka Manhattan - 9 cech (średnia dokładność: 0.459)
- 5 sąsiadów, metryka Euklidesa - 8 cech (średnia dokładność: 0.395)
- 9 sąsiadów, metryka Manhattan - 8 cech (średnia dokładność: 0.471)
- 9 sąsiadów, metryka Euklidesa - 8 cech (średnia dokładność: 0.361)
- 16 sąsiadów, metryka Manhattan - 8 cech (średnia dokładność: 0.458)
- 16 sąsiadów, metryka Euklidesa - 9 cech (średnia dokładność: 0.344)

7.2 Test t-studenta

Test t-studenta [4] porównuje ze sobą dwie średnie i określa czy są one od siebie różne oraz jak duża jest ta różnica. Pozwala określić czy takie różnice mogły wystąpić przez przypadek. Często wykorzystuje się go, aby porównać rozwiązanie z określoną grupą kontrolną. Im większa jest wartość t, tym większa jest różnica między grupami. Im mniejsza wartość t, tym bardziej grupy są do siebie podobne.

Każdej wartości t jest przyporządkowana wartość p. Wartość p, to prawdopodobieństwo tego, że różnica między grupami wystąpiła przez przypadek. Im mniejsza wartość p, tym bardziej prawdopodobne, że zależność między grupami nie jest przypadkowa. Zazwyczaj przyjmuje się poziom istotności na poziomie

0.05 (5%). Wyniki otrzymanych z parowego testu t-studenta wartości p zostały przedstawione w poniższej tabeli:

	5/Man	**5/Euk**	**9/Man**	**9/Euk**	**16/Man**	**16/Euk**
5/Man	1.00	0.00	0.36	0.00	0.95	0.00
5/Euk	0.00	1.00	0.00	0.78	0.00	0.00
9/Man	0.36	0.00	1.00	0.00	0.45	0.00
9/Euk	0.00	0.78	0.00	1.00	0.00	0.01
16/Man	0.95	0.00	0.45	0.00	1.00	0.00
16/Euk	0.00	0.00	0.00	0.01	0.00	1.00

Tabela 3: Wyniki testu parowego t-studenta

Wyniki testu poniżej przyjętego poziomu istotności uzyskały pary klasyfikatorów mające wynik '1' w poniższej tabeli. Uzyskanie takiego wyniku oznacza to, że różnice w próbkach można uznać za nieprzypadkowe:

	5/Man	**5/Euk**	**9/Man**	**9/Euk**	**16/Man**	**16/Euk**
5/Man	0	1	0	1	0	1
5/Euk	1	0	1	0	1	1
9/Man	0	1	0	1	0	1
9/Euk	1	0	1	0	1	1
16/Man	0	1	0	1	0	1
16/Euk	1	1	1	1	1	0

Tabela 4: Wyniki testu w odniesieniu do poziomu istotności

8 Dyskusja otrzymanych wyników

Z uzyskanych wyników można wywnioskować, że w przypadku stworzonego przez nas klasyfikatora większe znaczenie od liczby branych pod uwagę sąsiadów ma przyjęta metryka. W przypadku każdego z klasyfikatorów różnice w rozkładzie wartości były uznawane za istotne w każdym przypadku porównania z klasyfikatorem korzystającym z innej metryki. Poza tym klasyfikator wykorzystujący 16 sąsiadów oraz metrykę euklidesową wykazał istotne różnice z pozostałymi klasyfikatorami wykorzystującymi metrykę euklidesową. Klasyfikatory korzystujące z metryki Manhattan nie wykazały różnic poniżej przyjętego poziomu istotności. Może to świadczyć o tym, że pozostałe przyjęte wartości liczby sąsiadów w zbyt małym stopniu się od siebie różniły.

Dla każdej wartości liczby sąsiadów lepszą dokładność dał klasyfikator wykorzystujący metrykę Manhattan. Najlepsze średnie wyniki dał klasyfikator o parametrach 9 sąsiadów i metryce Manhattan. Osiągnął on średnią wartość dokładności na poziomie 0.471, a najwyższy nieuśredniony wynik dokładności na

poziomie 0.541. Najwyższa średnia wartość dokładności uzyskana przez klasyfikator wykorzystujący metrykę Euklidesową wyniosła 0.395 i uzyskał ją klasyfikator z 5 sąsiadami. Widać zatem, że dla różnych metryk najlepsze wyniki uzyskały klasyfikatory o różnej liczbie sąsiadów.

Uzyskane wyniki dokładności na poziomie poniżej 0.5 pozwalają nam sądzić, że stworzone przez nas klasyfikatory nie mogłyby zostać zastosowane do prawdziwych zastosowań. Przeprowadzone przez nas poza eksperymentem próby manipulowania liczbą sąsiadów nie przyniosły istotnej poprawy dokładności. Możemy zatem domniemywać, że algorytm knn nie radzi sobie z klasyfikacją przy takiej liczbie klas i atrybutów. Udało nam się znaleźć informację w artykule naukowym [10], że skuteczność algorytmu knn jest relatywnie niska w porównaniu do innych algorytmów klasyfikujących.

Literatura

1. 9 Distance Measures in Data Science. https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa, (data dostępu: 16.05.2021)
2. Dokumentacja języka Python w wersji 3.9. https://docs.python.org/3.9/, (data dostępu: 25.04.2021)
3. World Health Organization. https://www.who.int, (data dostępu: 16.03.2021)
4. Aankul, A.: T-test using Python and Numpy. https://towardsdatascience.com/inferential-statistics-series-t-test-using-numpy-2718f8f9bf2f, (data dostępu: 20.04.2021)
5. Brownlee, J.: Repeated k-Fold Cross-Validation for Model Evaluation in Python. https://machinelearningmastery.com/repeated-k-fold-cross-validation-with-python/, (data dostępu: 27.04.2021)
6. Gajawada, S.K.: Chi-Square Test for Feature Selection in Machine learning. https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223, (data dostępu: 20.04.2021)
7. Harris, C.R., Millman, K.J., van der Walt, S.J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N.J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M.H., Brett, M., Haldane, A., del Río, J.F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., Oliphant, T.E.: Array programming with NumPy. Nature **585**(7825), 357–362 (Sep 2020). https://doi.org/10.1038/s41586-020-2649-2, https://doi.org/10.1038/s41586-020-2649-2
8. Janz TG, Johnson RL, R.S.: Anemia in the emergency department: evaluation and treatment. Emergency Medicine Practice. 15 (11): 1–15, quiz 15–16. PMID 24716235
9. Jose, I.: KNN (K-Nearest Neighbors). https://towardsdatascience.com/knn-k-nearest-neighbors-1-a4707b24bd1d, (data dostępu: 20.04.2021)
10. K.U. Syaliman, E.N., Sitompul, O.: Improving the accuracy of k-nearest neighbor using local mean based and distance weight
11. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
12. T, V.: Years lived with disability (YLDs) for 1160 sequelae of 289 diseases and injuries 1990-2010: a systematic analysis for the Global Burden of Disease Study 2010. https://www.documentation.ird.fr/hor/fdi:010059240, (data dostępu: 16.03.2021)
13. pandas development team, T.: pandas-dev/pandas: Pandas (Feb 2020). https://doi.org/10.5281/zenodo.3509134, https://doi.org/10.5281/zenodo.3509134
14. Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S.J., Brett, M., Wilson, J., Millman, K.J., Mayorov, N., Nelson, A.R.J., Jones, E., Kern, R., Larson, E., Carey, C.J., Polat, İ., Feng, Y., Moore, E.W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E.A., Harris, C.R.,

Archibald, A.M., Ribeiro, A.H., Pedregosa, F., van Mulbregt, P., SciPy 1.0 Contributors: SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods **17**, 261–272 (2020). https://doi.org/10.1038/s41592-019-0686-2