# Systemy Sztucznej Inteligencji

Dokumentacja Projektu

Porównanie algorytmu KNN oraz Naiwnego Klasyfikatora Bayesa przy klasyfikacji odręcznie pisanych cyfr.

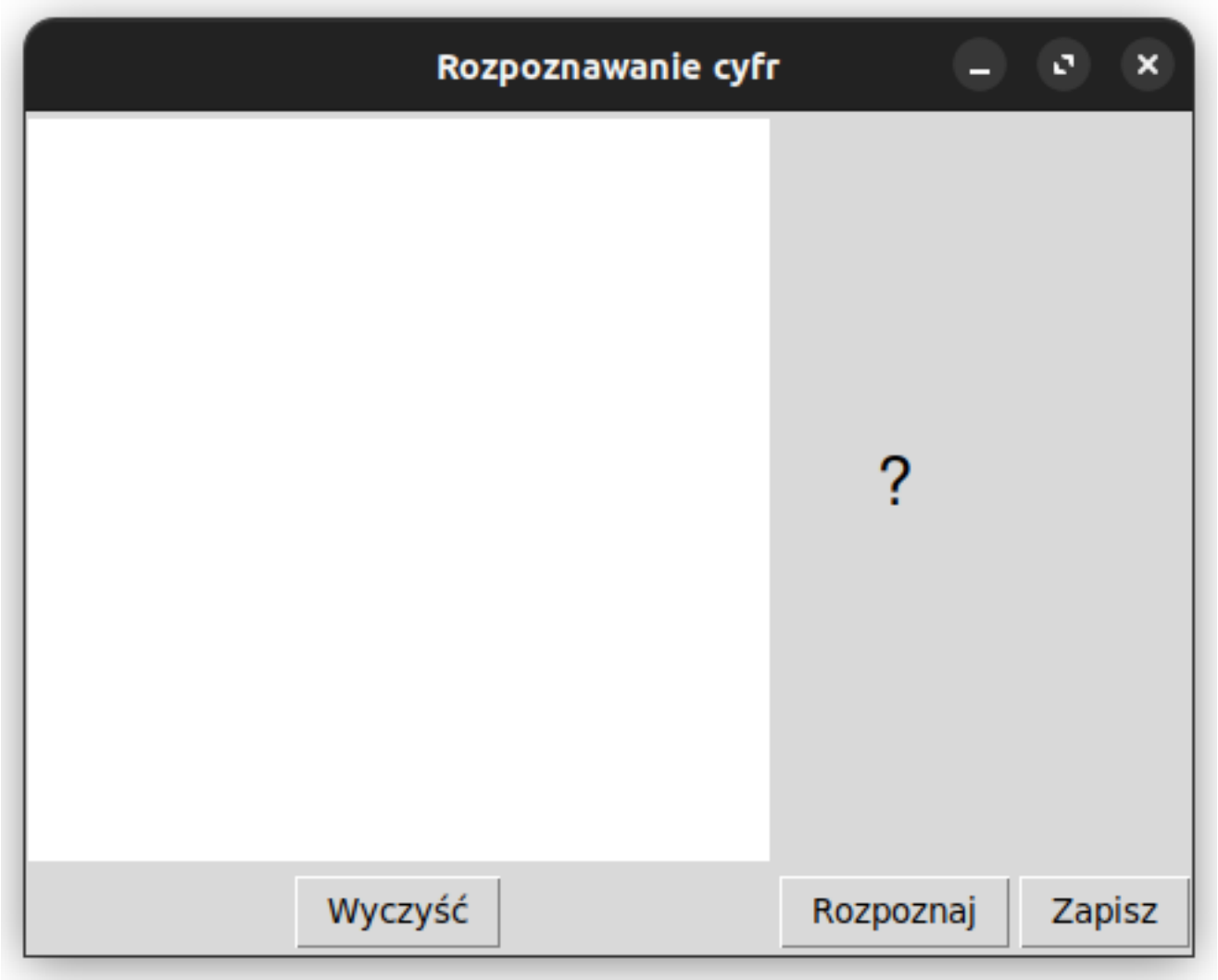
Piotr Skowroński gr. 3/6
Krzysztof Czuba gr. 4/7
Jakub Poreda gr. 3/6

24 czerwca 2023

# Część I

## Opis programu

Celem programu jest klasyfikowanie odręcznie pisanych cyfr przez użytkownika. Aplikacja zawiera proste GUI, które pozwala użytkownikowi narysować cyfrę na płótnie, a następnie po wciśnięciu przycisku 'Rozpoznaj' program klasyfikuje cyfrę za pomocą jednego z klasyfikatorów w celu rozpoznania narysowanej cyfry. Otrzymane wyniki klasyfikacji są wyświetlane w bloku po prawej stronie interfejsu użytkownika.




Rysunek 1: Wygląd aplikacji


Rysunek 2: Rozpoznawanie

## Dodatkowe informacje

Wymagania: Python 3.11
Program korzysta z następujacych zewnętrznych bibliotek:
- Pillow
    - Do transformacji zapisanych cyfr na macierz
    - Do transformacji narysowanej cyfry na macierz
- numpy
- scipy

# Część II

## Opis działania

Piksele wczytanego obrazu są konwertowane na skalę szarości tj. 0 - kolor biały, 255 - kolor czarny oraz są normalizowane do przedziału od 0 do 1 co ułatwia modelowi dopasowanie cyfr. Wzór na normalizację pojedyńczego piksela:

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)}$$

gdzie:

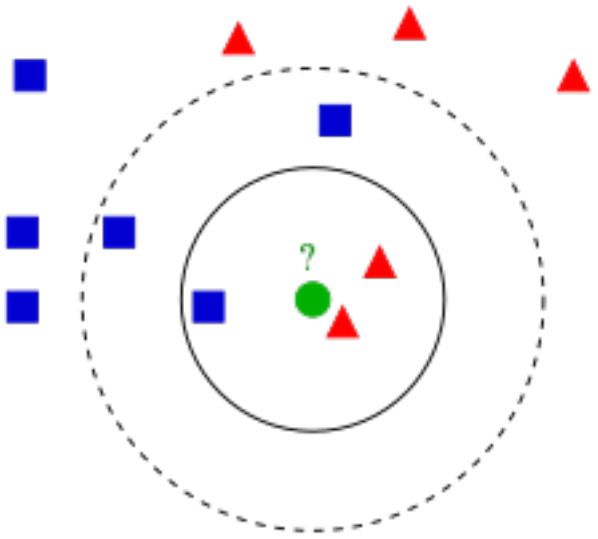
$z_i$ - znormalizowany piksel
$x_i$ - piksel
$x$ - zbiór wszystkich pikseli

Ostatecznie wzór ma postać:

$$z_i = \frac{x_i}{255}$$

### Algorytm k najbliższych sąsiadów

Klasyfikator kNN to jedna z ważniejszych nieparametrycznych metod klasyfikacji. W tej metodzie klasyfikowany obiekt przydzielamy do tej klasy, do której należy większość z k sąsiadów.




Rysunek 3: Przykład klasyfikacji metodą kNN

W przypadku k=3 (mniejszy okrąg), zielona kropka zostanie zakwalifikowana do czerwonych trójkątów. W przypadku k=5 (większy okrąg) - do niebieskich kwadratów.

### Metryka odległości

Użyta została odległość Minkowskiego określona wzorem:

$$L_m(x, y) = \left( \sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}$$

gdzie:
$L_m$ - odległość między punktami x i y
$x$, $y$ - punkty w przestrzeni $n$ wymiarowej
$x_i$, $y_i$ - i'ta współrzędna punktów x i y
$p$ - parametr określający rodzaj metryki

Przetestowaliśmy skuteczność klasyfikatora kNN dla liczby sąsiadów $k \in \{1, 3, 5, 7, 9, 11, 13, 15\}$ oraz dla wartości parametru $p \in \{1, 2, 3\}$.

## Pseudokod algorytmu kNN

**Data:** Dane wejściowe: liczba $k$
**Result:** Brak
$i := 0$;
**while** $i < k$ **do**
    Drukuj na ekran liczbę $i$;
    **if** $i\%2 == 0$ **then**
        Wydrukuj informację, że liczba $i$ jest liczbą parzystą;
    **else**
        Wydrukuj informację, że liczba $i$ nie jest liczbą parzystą;
    **end**
**end**

**Algorithm 1:** Algorytm drukowania informacji o liczbie parzystej/nieprarzystej.

## Bazy danych

Baza danych składa się z 628 obrazów cyfr narysowanych przez nas. Każdy piksel obrazu jest reprezentowany w skali szarości (ma wartość od 0 do 255, gdzie 0 to biały, a 255 to czarny kolor). Obrazy są przechowywane w formacie png.

Rysunek 4: Przykładowy obraz z bazy danych

Rysunek 5: Ten sam obraz po zmianie rozdzielczości na 28x28 pikseli

## Implementacja

1. Program składa się z statycznej klasy przechowującej ustawienia,
2. Funkcji zamieniającej obrazy na wektory
3. Funkcji przewidującej cyfry

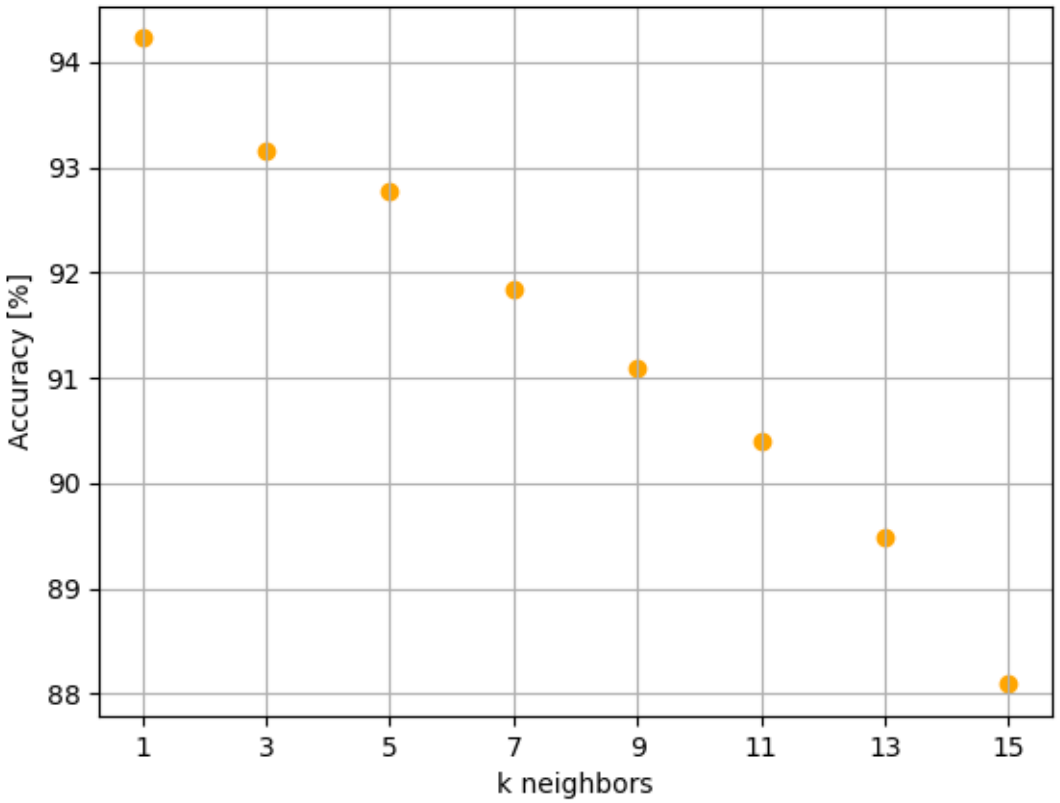
4. Widgeta Tkintera

Widget wyświetla UI, a na nim płótno po którym użytkownik może pisać. Użytkownik bazgrze po nim i klika przycisk z napisem 'Rozpoznaj'. Rysunek jest pobierany i serializowany do postaci odpowiadającej danym testowym. Uruchamiana jest procedura klasyfikująca, a jej rezultat - Dopasowana cyfra - Wyświetlany jest w GUI.

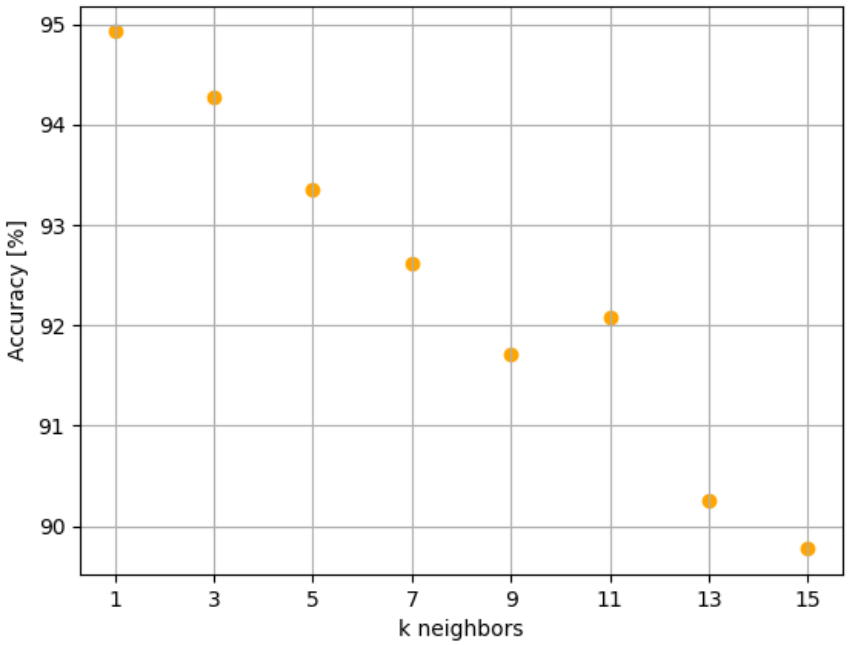
## Testy

Tutaj powinna pojawić się analiza uzyskanych wyników oraz wykresy/pomiary.

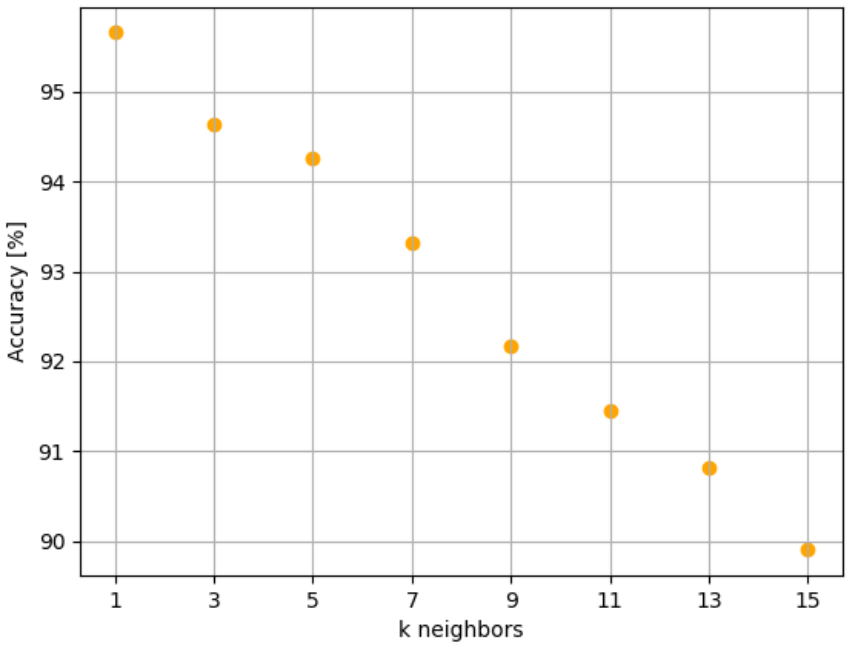
## Eksperymenty




Rysunek 6: p=1




Rysunek 7: p=2




Rysunek 8: p=3

Najlepsze wyniki klasyfikator uzyskuje dla $k = 1$ oraz $p = 3$, gdzie jego dokładność wynosi 95.4%.

# Pełen kod aplikacji

```
1 Tutaj wklejamy pelen kod.
```

---