Systemy Sztucznej Inteligencji

Dokumentacja Projektu

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

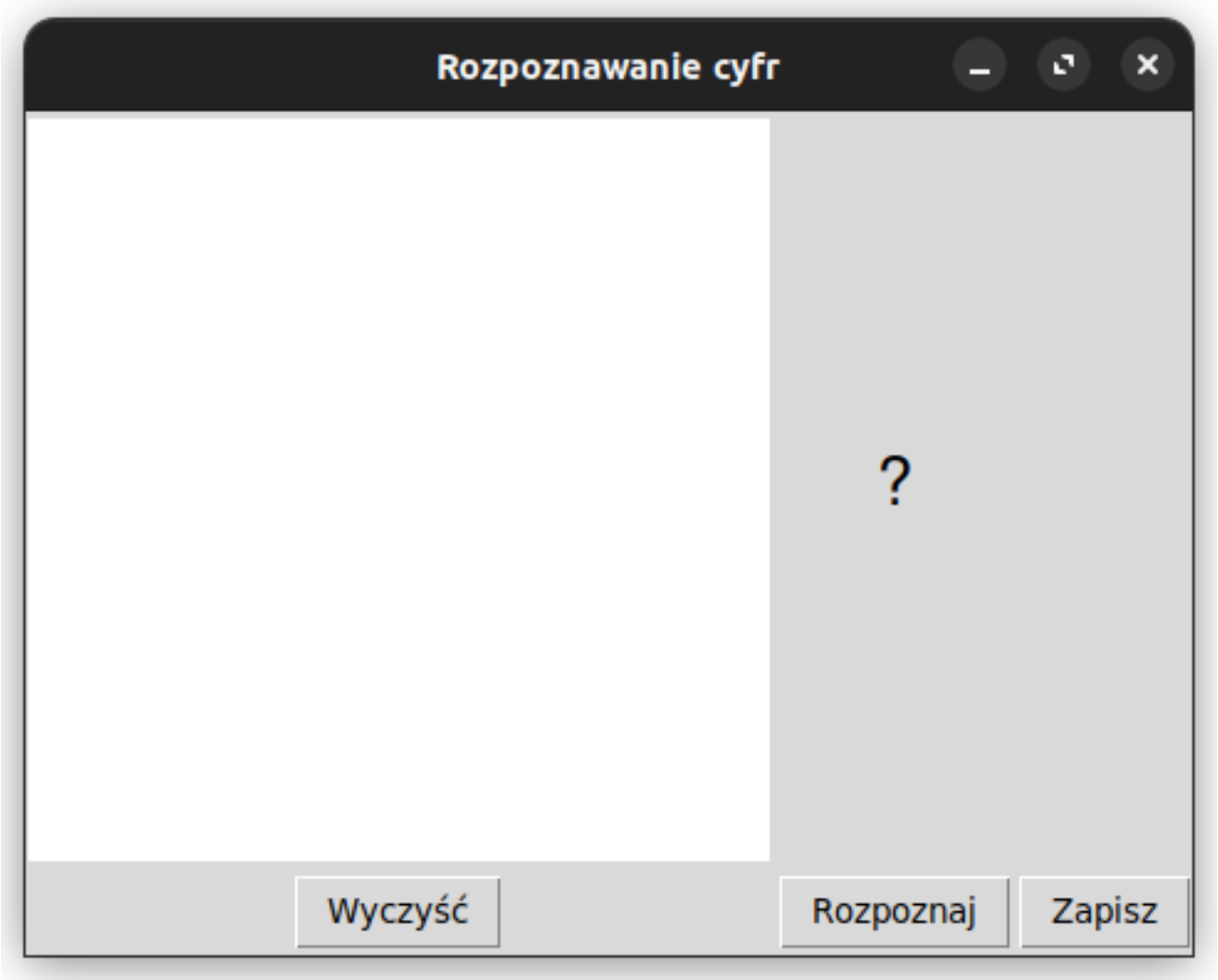
Piotr Skowroński gr. 3/6
Krzysztof Czuba gr. 4/7
Jakub Poreda gr. 3/6

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:

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

Następnie przechodzimy do fazy uczenia, gdzie model otrzymuje nasze dane treningowe. Model wykorzystując klasyfikator KNN, umieszcza w przestrzeni metrycznej cech wartości pikseli cyfr z zestawu treningowego. Wykorzstuje metrykę odległości Manhattan mierzy on sumę bezwzględnych różnic pomiędzy cechami

$$d_n = \Sigma_{i=0}^{n} |wektor1_i - wektor2_i|$$

Przewidywanie odbywa się poprzez wybranie trzech cyfr z zbioru treningowego, których cechy różnią się najmniej od cech sprawdzanego rysunku. Wybór odbywa się poprzez znalezienie dominującej spośród wybranych cyfr

Algorytm

Tutaj opisujemy rozwiązanie zadania. Dla przedmiotu programowanie będzie to wykorzystanie matematyki z poprzedniego zadania itd. Dla SSI będzie to ogólne działanie przetwarzania danych w oparciu o modele matematyczne z poprzedniego zadania.

Pseudokod tworzymy w LaTeX. Przykład:

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

Implementacja

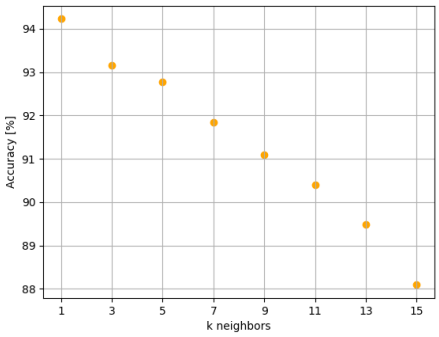
1. Program składa się z statycznej klasy przechowującej ustawienia,
2. Funkcji zamieniającej obrazy na wektory
3. Funkcji przewidującej cyfry
4. Widgeta Tkintera

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.

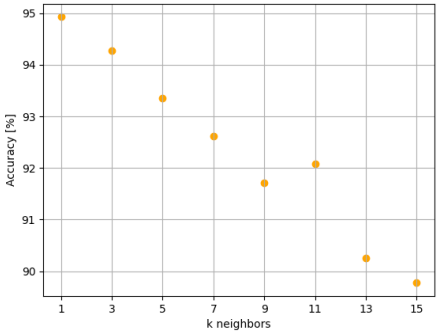
Testy

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

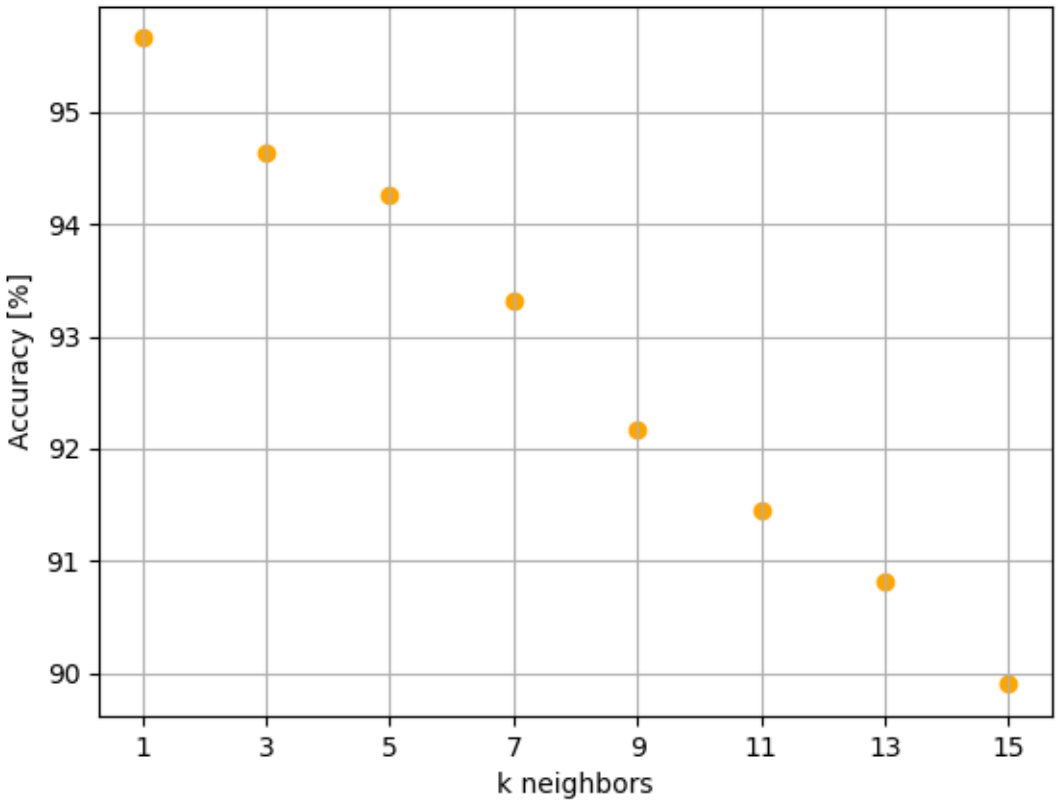
Eksperymenty



Rysunek 3: p=1



Rysunek 4: p=2



Rysunek 5: p=3

Pełen kod aplikacji

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