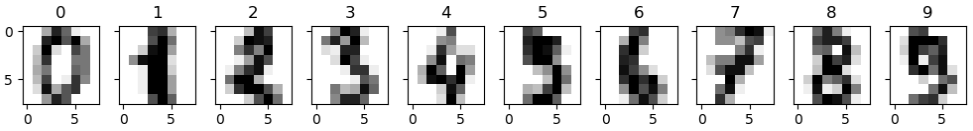
Laboratorium 13

słowa kluczowe: klasyfikacja, rozpoznawanie w otwartym zbiorze.

Zadanie 1:

- Wczytaj zbór danych *digits* (funkcja `load_digits` z biblioteki `scikit-learn`) przedstawiający odręcznie pisane cyfry w niskiej rozdzielczości. Cechy przypisz do zmiennej `X`, etykiety do zmiennej `y`.

  Zauważ, że cechy już są wektorem długości 64 a nie macierzą 8×8. Można sobie podejrzeć jak wyglądają obiekty:



- W tym zadaniu należy zbadać jakość klasyfikacji takich cyfr za pomocą w pełni połączonych sieci neuronowych (`MLPClassifier` z `scikit-learn`) o różnych konfiguracjach warstw ukrytych.
  - Zadeklaruj 5 klasyfikatorów z różnymi parametrami `hidden_layer_sizes`. Wszystkie warstwy ukryte będą miały 10 neuronów, a warstw będzie od 1 do 5.
  - Zbadaj działanie tych klasyfikatorów w 5-krotnie powtórzonej 2-foldowej stratyfikowanej walidacji krzyżowej (`RepeatedStratifiedKFold`). Pamiętaj o użyciu funkcji `clone` przy inicjalizacji klasyfikatorów dla poszczególnych powtórzeń i foldów.
  - Do oceny jakości klasyfikacji wykorzystaj metrykę *balanced accuracy score*.
- Wyniki uśrednij i wypisz w terminalu. Na ich podstawie wybierz najlepszą konfigurację warstw ukrytych sieci dla tego problemu.

Przykładowy efekt zadania 1:

```
Mean results: [0.8922117  0.91409598 0.90218564 0.89549539 0.88913974]
Argmax: 1
```

Zadanie 2:

- Wybierz architekturę sieci która wypadła najlepiej w poprzednim zadaniu. Teraz będziemy używać tylko tej jednej.
- Przekształcimy zbiór *digits* w zbiór **otwarty**, w którym klasami **znanymi** będą wszystkie obiekty przedstawiające cyfry mniejsze od 5 (więc o etykietach `0,1,2,3,4`) a klasami **nieznanymi** obiekty przedstawiające cyfry większe od lub równe 5 (`5,6,7,8,9`).
- W takiej samej konfiguracji walidacji krzyżowej jak w zadaniu 1 zbadaj zdolność klasyfikatora do rozpoznawania:
  - w zamkniętym zbiorze (dla klas znanych),
  - w otwartym zbiorze — będzie to zadanie klasyfikacji **binarnej**, gdzie wszystkie obiekty **znane** będą stanowić klasę pozytywną (etykieta 1), a wszystkie obiekty **nieznane** klasę negatywną (etykieta 0).

  Żeby to zrobić, w pętli walidacji krzyżowej, należy kolejno:
  - Wyuczyć klasyfikator obiektami klas **znanych** — mamy tutaj problem 5-klasowy.

- Zbadać zwykłą zbalansowaną dokładność dla problemu 5-klasowego (to będzie *inner* score).
- Utworzyć zbiór mieszany zawierający obiekty testowe z klas **znanych** (te same którymi testowaliśmy klasyfikator punkt wcześniej) i wszystkie obiekty z klas **nieznanych**.
- Utworzyć binarne etykiety dla tych obiektów — 1 dla klas znanych i 0 dla nieznanych.
- Określić wsparcie dla obiektów z tego zbioru mieszanego za pomocą funkcji `predict_proba` i wyznaczyć maksymalne wsparcie w obrębie klas.

  Macierz będzie miała tyle kolumn, ile było klas w problemie, czyli u nas 5 — trzeba wybrać maksymalną wartość w ich obrębie wykorzystując `np.max`. Nazwijmy tę wartość *wsparciem decyzyjnym*.
- Określ próg `threshold` równy `0.8`. Jeżeli *wsparcie decyzyjne* będzie poniżej progu, obiekty będą uznawane za **nieznane** (0), a powyżej za **znane** (1).
- Zbadaj jakość *balanced accuracy score* dla takiego podejścia do rozpoznawania obiektów nieznanych (to będzie *outer* score).

- Przedstaw uśredniony wynik w terminalu.

Przykładowy efekt zadania 2:

```
Inner score: 0.95328279
Outer score: 0.66574581
```

Zadanie 3:

- Eksperymentalnie znajdź próg `threshold` pozwalający na otrzymanie najlepszego wyniku rozpoznawania obiektów nieznanych. Przetestuj 100 wartości od 0.5 do 1, stabilizując wyniki dzięki wykorzystaniu walidacji krzyżowej.
  Ważne: należy tylko jednokrotnie obliczyć
  wsparcie decyzyjne.
- Pokaż wyniki na wykresie i zaznacz na nim najlepszą znalezioną wartość parametru.

Przykładowy efekt zadania 3: