Jakub Tomaszewski 242555 Projekt II Zadanie I

Metody wybrane do redukcji wymiarowosci:

- **Quadrants:** Podział obrazu na cztery równe ćwiartki i zsumowanie wartości pikseli w każdej z nich. Pozwala to uchwycić, w której części obrazu znajduje się największa masa cyfry - jest to cecha silnie związana z jej kształtem i położeniem.

Ręcznie pisane cyfry mają charakterystyczne położenie i kształt, co sprawia, że nawet bardzo prosta reprezentacja – taka jak suma pikseli w czterech ćwiartkach obrazu – pozwala odróżnić wiele z nich. Przykładowo, cyfra „1” skupia większość masy w środkowej części obrazu, „7” dominuje w górnej części, a „0” jest rozłożona niemal równomiernie. Takie podejście do ekstrakcji cech jest bardzo szybkie i proste, a jednocześnie pozwala łatwo wykryć, w której części obrazu znajduje się główny fragment cyfry. Szczególnie dobrze sprawdza się w przypadku cyfr o wyraźnie różnym kształcie i położeniu. Trzeba jednak pamiętać, że metoda ta jest wrażliwa na przesunięcia cyfry względem środka obrazka, co w zbiorze MNIST zdarza się często. Dodatkowo, nie pozwala na rozróżnienie bardziej subtelnych różnic w kształcie, przez co cyfry o podobnym rozkładzie masy, takie jak „5” i „6”, mogą być trudne do odróżnienia.

- **PCA:** Redukcja wymiarów obrazu do dwóch głównych składowych za pomocą analizy głównych składowych (Principal Component Analysis), co umożliwia wizualizację i analizę globalnych wzorców w danych.

PCA automatycznie znajduje takie przekształcenie, które najlepiej rozdziela dane w nowej przestrzeni, niezależnie od położenia czy orientacji cyfry. Dzięki temu cyfry o różnych kształtach często trafiają w różne obszary przestrzeni PCA, co umożliwia wizualizację całego zbioru w dwóch wymiarach i często pozwala dobrze rozróżnić cyfry o odmiennych kształtach. Metoda ta wykorzystuje globalne wzorce obecne w danych, jednak nie zawsze gwarantuje pełne rozdzielenie wszystkich klas, zwłaszcza tych do siebie podobnych. Dodatkowo, interpretacja uzyskanych cech może być mniej intuicyjna niż w przypadku prostych cech geometrycznych, takich jak suma pikseli w ćwiartkach obrazu.

Przykladowe elementy zbioru MNIST:

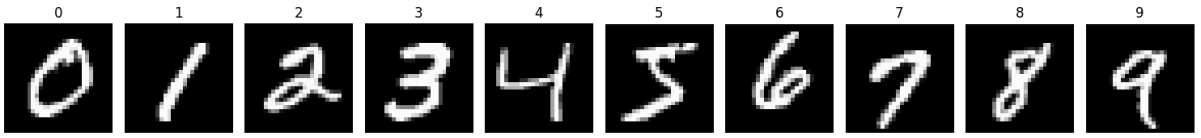


Tabela cech quadrants i PCA dla pierwszego wystąpienia każdej cyfry MNIST:

Digit	Q1	Q2	Q3	Q4	PCA1	PCA2
0	20.309803	43.266670	36.058823	22.305883	3.97	1.16
1	0.533333	29.203920	32.152939	5.305882	-3.13	2.38
2	15.290196	31.411766	37.513725	31.866669	0.72	-0.87
3	15.894118	51.796082	39.564705	33.400002	2.01	1.22
4	12.564706	18.243137	18.109804	27.329411	-0.20	-1.54
5	26.235296	25.337254	26.662746	29.705883	0.49	1.23
6	21.407845	28.062744	34.788235	27.282352	0.94	-0.59
7	24.619608	33.725487	22.796080	18.058823	-0.54	-3.42
8	17.239216	35.478432	40.462746	13.117647	-1.52	0.99
9	18.180391	26.741177	16.423531	29.690197	-1.50	-2.86

W praktyce, rozpoznawanie cyfr na podstawie cech quadrants lub PCA może być utrudnione przez:

Nieregularne kształty cyfr – różne style pisma, grubość linii, odchylenia od typowego wzorca.

Przesunięcia względem środka – centrum cyfry często znajduje się poza środkiem obrazka, co wpływa na wartości cech quadrants.

Nakładanie się rozkładów cech – niektóre cyfry (np. „5” i „6”, „3” i „8”) mogą mieć podobne wartości cech, co utrudnia ich rozróżnienie prostymi metodami.

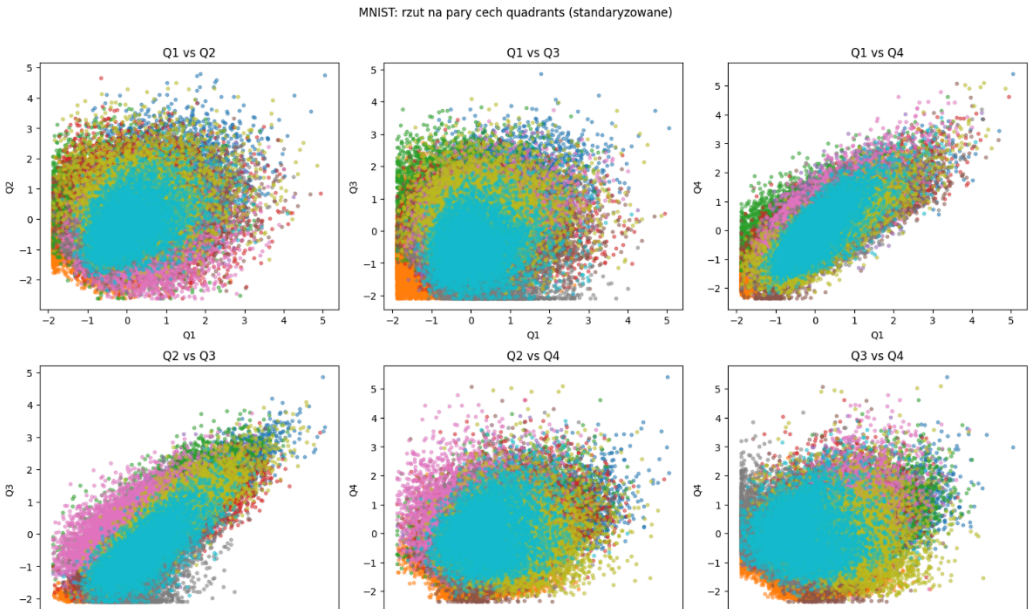
Analiza powyższych wartości pozwala wskazać, które cyfry są łatwe do rozróżnienia na podstawie tych cech:

- **Cyfra 1** wyróżnia się bardzo niskimi wartościami Q1 i Q4 oraz skrajną wartością PCA1, co czyni ją łatwą do odróżnienia od innych.
- **Cyfra 0** ma stosunkowo równomierny rozkład masy (wszystkie quadrants wysokie) oraz wysoką wartość PCA1, co również ją wyróżnia.
- **Cyfry 7 i 9** mają podobne wartości quadrants i PCA, co sugeruje, że mogą być trudniejsze do rozróżnienia prostymi metodami.

Wizualizacja rozkładu cyfr w przestrzeni PCA oraz analiza różnic w cechach quadrants potwierdzają, że cyfry o unikalnym rozkładzie masy lub oddalonych współrzędnych PCA są potencjalnie najłatwiejsze do klasyfikacji.

Na wykresach rzutów na pary cech quadrants widać, że cyfry są slabo rozdzielone a rozkłady silnie się nakładają. Oznacza to, że separowalność klas w tej przestrzeni cech jest ograniczona i wiele cyfr może być mylonych.

Diagramy Woronoja rowniez nie pokazuja wyraźnych stref, co oznacza słabą separowalność.



W celu wyboru najlepszego modelu dla każdego zbioru danych oraz każdej metody ekstrakcji cech, zastosowałem systematyczną procedurę opartą na eksperymentach z różnymi konfiguracjami sieci MLP. Dla każdego zbioru testowałem różne liczby neuronów w warstwie ukrytej, różne wartości learning rate, batch size oraz liczbę epok. Dodatkowo, w przypadku zbiorów o większej liczbie cech, rozważałem także architektury z większą liczbą neuronów lub warstw. Każdy model oceniałem na podstawie accuracy na zbiorze walidacyjnym/testowym oraz analizy macierzy pomyłek.

Takie podejście pozwala znaleźć model, który najlepiej generalizuje – unika zarówno niedouczenia (zbyt prosty model), jak i przeuczenia (zbyt złożony model). Wnioski z projektu pierwszego pokazały, że dobór architektury i hiperparametrów ma kluczowe znaczenie dla skuteczności klasyfikatora, a optymalne ustawienia mogą się różnić w zależności od liczby cech i charakteru danych. Dlatego dla każdego zbioru oraz każdej ekstrakcji cech przeprowadzałem eksperymenty, a do dalszej analizy wybierałem model o najwyższym accuracy na zbiorze testowym.

Zakresy testowanych hiperparametrów:

- liczba neuronów w warstwie ukrytej: 8, 16, 32, 64
- learning rate: 0.01, 0.05, 0.1
- batch size: 32, 64, 128
- liczba epok: 10, 20, 30, 50

Wybrałem powyższe wartości, ponieważ są to **standardowe i najczęściej stosowane zakresy w literaturze oraz praktyce uczenia sieci MLP**. Pozwalają one na skuteczne znalezienie zarówno prostych, jak i bardziej złożonych modeli, a jednocześnie nie prowadzą do nadmiernego wydłużenia czasu obliczeń. Takie ustawienia zapewniają uniwersalność i możliwość zastosowania do każdego zbioru danych, niezależnie od liczby cech czy wielkości zbioru.

W każdym przypadku ostateczny wybór modelu opierał się na wynikach na zbiorze testowym oraz analizie macierzy pomyłek. Dzięki temu mogłem świadomie wybrać architekturę sieci i ustawienia treningu, które zapewniają najlepszą generalizację dla każdego zadania.

Dla zbiorów: Iris, Wine oraz Breast Cancer parametry mozna wybrac poprzez kilka prob z roznym zakresami wartosci manualnie, jednak dla MNIST warto to zapetlic aby automatycznie znalezc najlepszy model.

Wnioski z projektu pierwszego potwierdziły, że **systematyczne eksperymenty z hiperparametrami są kluczowe dla uzyskania najlepszego modelu**. Takie podejście pozwala nie tylko zoptymalizować skuteczność klasyfikatora, ale także porównywać różne sposoby ekstrakcji cech na równych zasadach.

1. Iris

Architektura modelu:
MLP: 4 wejścia, 16 neuronów w warstwie ukrytej, 3 wyjścia (klasy), ReLU, CrossEntropyLoss, SGD.

Accuracy (train): 0.925

```
[[40  0  0]
 [ 0 34  6]
 [ 0  3 37]]
```

Accuracy (test): 0.867

```
[[10  0  0]
 [ 0  7  3]
 [ 0  1  9]]
```

2. Wine

Architektura modelu:
MLP: 13 wejść, 32 neurony w warstwie ukrytej, 3 wyjścia, ReLU, CrossEntropyLoss, SGD.

Accuracy (train): 0.986

```
[[47  0  0]
 [ 0 55  2]
 [ 0  0 38]]
```

Accuracy (test): 1.00

```
[[12  0  0]
 [ 0 14  0]
 [ 0  0 10]]
```

3. Breast Cancer

Architektura modelu:
MLP: 30 wejść, 32 neurony w warstwie ukrytej, 2 wyjścia, ReLU, CrossEntropyLoss, SGD.

Accuracy (train): 0.987

```
[[166   4]
 [  2 283]]
```

Accuracy (test): 0.965

```
[[41  1]
 [ 3 69]]
```

4. MNIST (wszystkie piksele – 784 cechy)

Architektura modelu:
MLP: 784 wejścia, 128 neuronów w warstwie ukrytej, 10 wyjść, ReLU, CrossEntropyLoss, SGD.

Accuracy (train): 0.987

```
[[5867    0    5    3    4    2   22    2    7   11]
 [   1 6695   14    3    7    0    3    9    7    3]
 [   7   11 5884   10    9    0    8   18    9    2]
 [   3    7   28 6000    1   27    4   23   24   14]
 [   1    9    5    0 5766    0   14    7    5   35]
 [   8    6    3   20    3 5342   15    2   11   11]
 [  12    3    0    0    7    5 5886    0    5    0]
 [   3   17   18    5   14    0    1 6181    3   23]
 [   8   29    7   11    5    9   16    2 5751   13]
 [   9    7    1   13   33    8    2   24   10 5842]]
```

Accuracy (test): 0.976

```
[[ 964    0    0    2    0    2    5    1    3    3]
 [   0 1125    2    1    0    1    3    0    3    0]
 [   5    3 1002    0    3    0    5    7    6    1]
 [   1    0    4  983    1    8    0    3    5    5]
 [   1    0    3    0  958    0    4    2    1   13]
 [   2    1    0    6    1  871    6    1    2    2]
 [   5    3    1    0    5    4  938    1    1    0]
 [   0    6    8    4    0    0    0 1001    1    8]
 [   4    0    6    6    3    3    7    3  941    1]
 [   4    6    0    8   10    1    0    5    1  974]]
```

Wnioski:
Wszystkie modele osiągnęły wysoką skuteczność na zbiorze testowym, co świadczy o dobrej generalizacji. Najlepsze wyniki uzyskano dla zbioru Wine (accuracy 100%). W przypadku Iris i Breast Cancer pojawiły się nieliczne błędne klasyfikacje, co widać w macierzach pomyłek. Model dla MNIST (784 cechy) również osiągnął bardzo wysoką skuteczność (accuracy 0.976), a błędy klasyfikacji dotyczą głównie cyfr o podobnych kształtach.

Najlepszy model Quadrants: hidden layers = 64, learning rate= 0.01, batch size= 64, epochs= 50

Quadrants train accuracy: 0.386

```
[[3748  369  120   97  294    0    9   30  966  290]
 [  43 6119    1   18  236    0    2   68   99  156]
 [ 649  279  960   33 1377    0 2356   21  126  157]
 [1605  867  127  494  654    0   56  681 1207  440]
 [ 410  938  235   80 2416    0  987  296  223  257]
 [ 781 1116   89  433  737    0  108  911  872  374]
 [ 411  234  643    9 1544    0 2928    0   53   96]
 [ 128  623   15  378  269    0   36 4325  314  177]
 [1932  800   96  448  346    0   18  185 1609  417]
 [ 794 1383   65  289  715    0   16 1329  783  575]]
```

Quadrants test accuracy: 0.3903

```
[[ 626   51   15   19   67    0    3    5  134   60]
 [  16 1033    0    0   38    0    1    3   21   23]
 [ 118   62  177    4  228    0  375    5   27   36]
 [ 262  152   29   99   87    0   10  110  188   73]
 [  68  172   51    9  419    0  142   42   41   38]
 [ 152  148   12   80  103    0   11  170  146   70]
 [  59   30  135    2  230    0  471    0    9   22]
 [  23  100    1   79   54    0    9  678   53   31]
 [ 328  101    6   78   57    0    1   33  302   68]
 [ 147  216   10   53  136    0   10  207  132   98]]
```

Najlepsze model (PCA): hidden layers = 32, learning rate= 0.1, batch size= 128, epochs= 50

PCA train accuracy: 0.4665

```
[[4604    0  310   59   28  260  580    1   81    0]
 [   0 6380   38   14    4   71    8   55  172    0]
 [ 614  187 1403  929  202  596  833   53 1137    4]
 [ 151  140  545 2938  101  573  317   65 1298    3]
 [  10   85    1    1 3611  173  293 1395   22  251]
 [ 473  178  573  581  427 1040 1062   34 1047    6]
 [ 693  164  510  178  396  478 2205   21 1272    1]
 [   0  222    1    3 2047  192  161 3378   43  218]
 [ 344  208  501  797  276  570  993   21 2140    1]
 [  41  138   16    4 2734  155  229 2300   39  293]]
```

PCA test accuracy: 0.482

```
[[ 762    0   56    5    6   39  102    0   10    0]
 [   0 1074    9    3    0   16    1    4   28    0]
 [ 106   33  287  170   29   95  132    6  173    1]
 [  21   17   88  523   11   96   47   11  196    0]
 [   2   12    0    0  592   26   41  260    2   47]
 [  81   21   91  106   75  174  181    9  153    1]
 [ 138   15   62   19   82   60  435    2  145    0]
 [   0   42    0    2  293   26   32  583   19   31]
 [  81   14   71  119   55  106  181    3  344    0]
 [  11   20    2    0  480   18   36  390    6   46]]
```

Granice decyzyjne są nieregularne i często nachodzą na siebie, co potwierdza, że w przestrzeni dwóch głównych składowych wiele klas nie jest liniowo separowalnych. Widać liczne obszary, gdzie cyfry różnych klas są wymieszane, co tłumaczy umiarkowaną skuteczność klasyfikatora.

Obie metody ekstrakcji cech prowadzą do znacznej utraty informacji i niskiej separowalności klas, co przekłada się na niską skuteczność klasyfikacji. Wizualizacja granic decyzyjnych dla PCA potwierdza, że klasy cyfrowe w tej przestrzeni są mocno nakładające się, co uniemożliwia uzyskanie wysokiego accuracy nawet przy optymalnych parametrach sieci MLP.