Projekt - perceptron

Andżelika Daczkowska

Wczytanie danych z pliku

```
import numpy as np
import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import io
%matplotlib inline
from google.colab import files

data = pd.read_csv('/letters.data', header=None)
#data.tail()

X = data.iloc[[0, 1, 3, 10, 16, 17, 18, 21, 22, 23], 0:35].values

print(f"wartosci X: \n{X}")
print("Wymiary tablicy X:", X.shape)
print("Rozmiar tablicy X:", X.size)
```

Powyższy fragment kodu przedstawia wczytanie danych z pliku letters.data do zmiennej, oraz przypisanie do zbioru X pierwszych 35 wartości z 10 wybranych wierszy.

Zawartość X:

```
Zawartość X:
[[-1  1  1  1 -1  1 -1 -1 -1  1  1 -1 -1 -1  1  1  1  1  1  1  1 -1 -1 -1
   1  1 -1 -1 -1  1  1 -1 -1 -1  1]
 [ 1  1  1  1 -1  1 -1 -1 -1  1  1 -1 -1 -1  1  1  1  1  1 -1  1 -1 -1 -1
   1  1 -1 -1 -1  1  1  1  1  1 -1]
 [ 1  1  1  1 -1  1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1 -1 -1
   1  1 -1 -1 -1  1  1  1  1  1 -1]
 [ 1 -1 -1 -1  1  1 -1 -1  1 -1  1 -1  1 -1 -1  1  1 -1 -1 -1  1 -1  1 -1
  -1  1 -1 -1  1 -1  1 -1 -1 -1  1]
 [-1  1  1  1 -1  1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1  1 -1
   1  1 -1 -1  1  1 -1  1  1  1  1]
 [ 1  1  1  1 -1  1 -1 -1 -1  1  1 -1 -1 -1  1  1  1  1  1 -1  1 -1  1 -1
  -1  1 -1 -1  1 -1  1 -1 -1 -1  1]
 [-1  1  1  1 -1  1 -1 -1 -1  1  1 -1 -1 -1 -1 -1  1  1  1 -1 -1 -1 -1 -1
   1  1 -1 -1 -1  1 -1  1  1  1 -1]
 [ 1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1 -1 -1
   1 -1  1 -1  1 -1 -1 -1  1 -1 -1]
 [ 1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1 -1 -1  1  1 -1  1 -1
   1  1 -1  1 -1  1 -1  1 -1  1 -1]
 [ 1 -1 -1 -1  1  1 -1 -1 -1  1 -1  1 -1  1 -1 -1 -1  1 -1 -1 -1  1 -1  1
  -1  1 -1 -1 -1  1  1 -1 -1 -1  1]]
Wymiary tablicy X: (10, 35)
Rozmiar tablicy X: 350
```

Wypełnianie zbioru y

```python
y = np.zeros((10, 10))
for i in range(10):
    for j in range(10):
        if i==j:
            y[i, j] = 1
        else:
            y[i, j] = -1

y = y.astype(np.int64)
print(f"wartosci y: \n{y}")
print("Wymiary tablicy y:", y.shape)
print("Rozmiar tablicy y:", y.size)
```

Zbiór wartości oczekiwanych o wymiarach 10x10 został wypełniony wartościami -1 wszędzie poza główną przekątną.

Zawartość y:

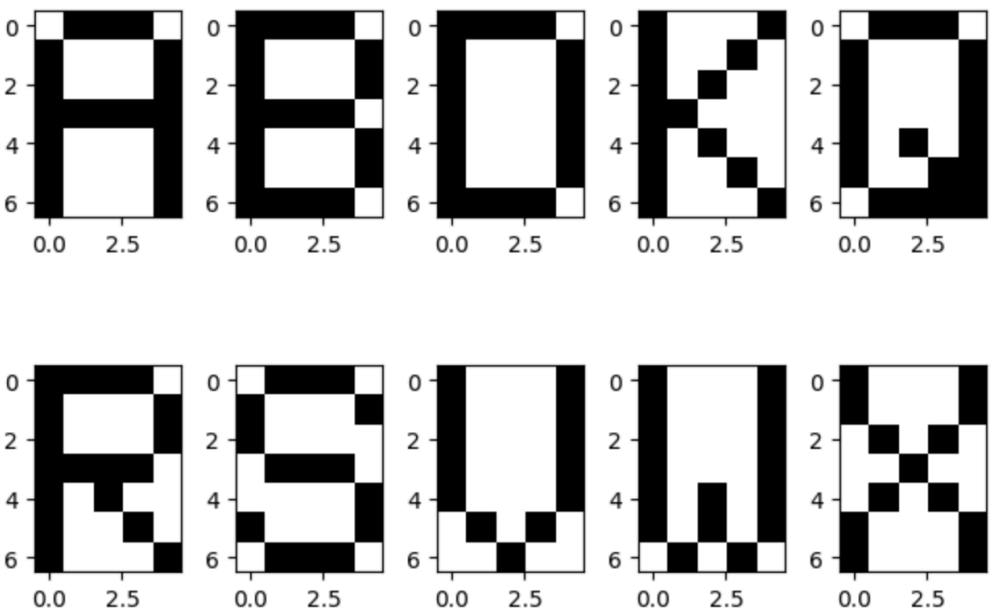
```
wartosci y:
[[ 1 -1 -1 -1 -1 -1 -1 -1 -1 -1]
 [-1  1 -1 -1 -1 -1 -1 -1 -1 -1]
 [-1 -1  1 -1 -1 -1 -1 -1 -1 -1]
 [-1 -1 -1  1 -1 -1 -1 -1 -1 -1]
 [-1 -1 -1 -1  1 -1 -1 -1 -1 -1]
 [-1 -1 -1 -1 -1  1 -1 -1 -1 -1]
 [-1 -1 -1 -1 -1 -1  1 -1 -1 -1]
 [-1 -1 -1 -1 -1 -1 -1  1 -1 -1]
 [-1 -1 -1 -1 -1 -1 -1 -1  1 -1]
 [-1 -1 -1 -1 -1 -1 -1 -1 -1  1]]
Wymiary tablicy y: (10, 10)
Rozmiar tablicy y: 100
```

Utworzenie obiektu klasy SLP o nazwie net

```
net = SLP()
```

Wyświetlenie danych ze zbioru X

```
net.show(X)
```



Uczenie modelu za pomocą metody fit(X, y)

```
net.fit(X, y)
```

Wynik predict() dla zbioru uczącego X

```
net.predict(X)
```

```
[array([ 1, -1, -1, -1, -1, -1, -1, -1, -1, -1]),
 array([-1,  1, -1, -1, -1, -1, -1, -1, -1, -1]),
 array([-1, -1,  1, -1, -1, -1, -1, -1, -1, -1]),
 array([-1, -1, -1,  1, -1, -1, -1, -1, -1, -1]),
 array([-1, -1, -1, -1,  1, -1, -1, -1, -1, -1]),
 array([-1, -1, -1, -1, -1,  1, -1, -1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1,  1, -1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1, -1,  1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1, -1, -1,  1, -1]),
 array([-1, -1, -1, -1, -1, -1, -1, -1, -1,  1])]
```

Wynik działania metody misclassified(X,y)

```
net.misclassified(X,y)
```

```
0
```

Funkcja damage() uszkadzająca dany % litery ze zbioru X

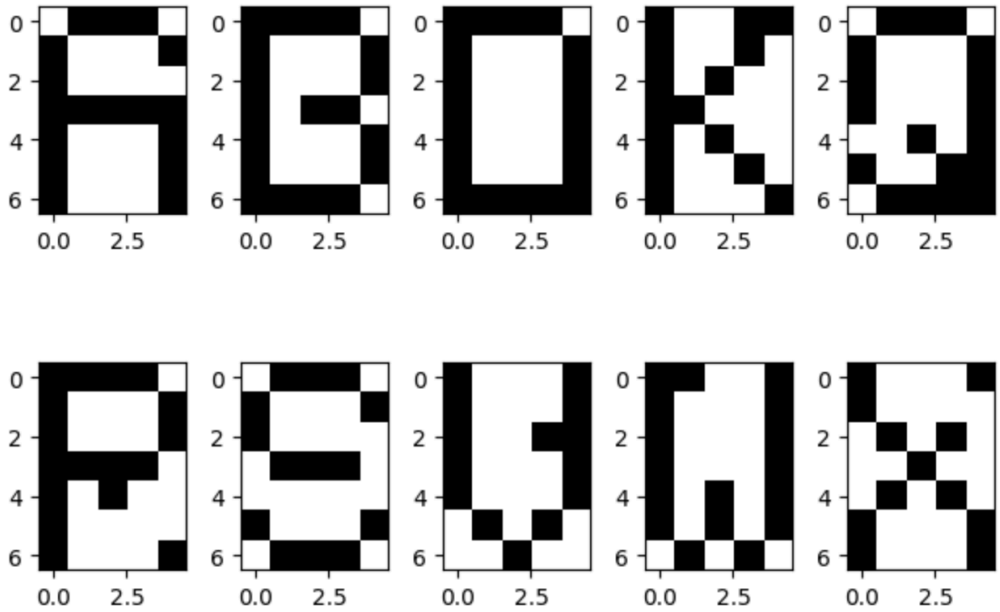
```
def damage(X, percent, seed=1):
        rgen = np.random.RandomState(seed)
        result = np.array( X )
        count = int(X.shape[1]*percent/100)

        for indeks_example in range( len(X) ):
            order = np.sort(rgen.choice(X.shape[1], count, replace=False))
            for indeks_pixel in order:
                result[indeks_example][indeks_pixel] *= -1
        return result

damaged5 = damage(X,5)
damaged15 = damage(X,15)
damaged40 = damage(X,40)
```

Zawartość zbioru X po uszkodzeniu 5%:

```
net.show(damaged5)
```



Wynik predict() i liczba błędnie sklasyfikowanych liter dla zbioru z 5% uszkodzeniem:

```
[320] net.predict(damaged5)
```

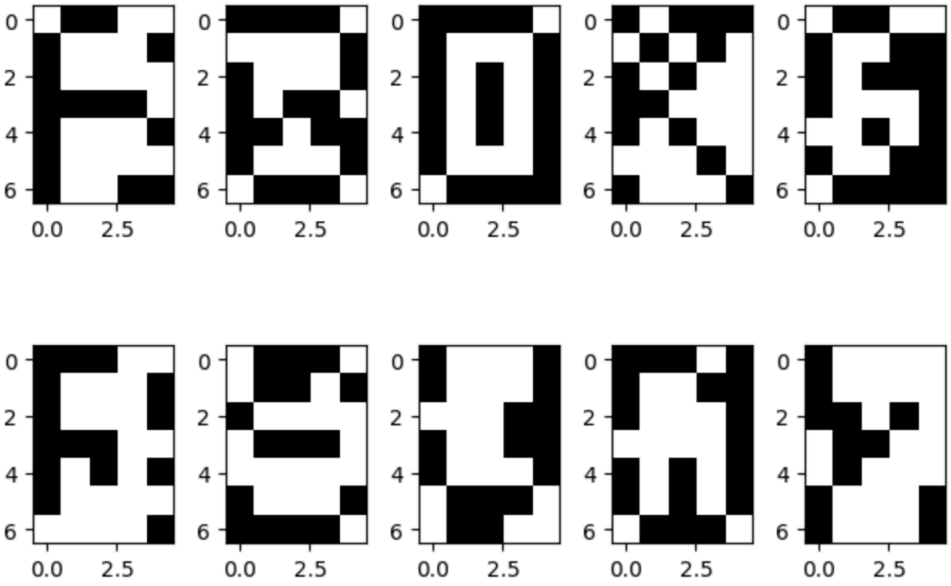
```
[array([ 1, -1, -1, -1, -1, -1, -1, -1, -1, -1]),
 array([-1,  1, -1, -1, -1, -1, -1, -1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1, -1, -1, -1, -1]),
 array([-1, -1, -1,  1, -1, -1, -1, -1, -1, -1]),
 array([-1, -1, -1, -1,  1, -1,  1, -1, -1, -1]),
 array([-1, -1, -1, -1, -1,  1, -1, -1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1,  1, -1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1, -1,  1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1, -1, -1,  1, -1]),
 array([-1, -1, -1, -1, -1, -1, -1, -1, -1,  1])]
```

```
net.misclassified(damaged5, y)
```

```
2
```

Zawartość zbioru X po uszkodzeniu 15%:

```
net.show(damaged15)
```



Wynik predict() i liczba błędnie sklasyfikowanych liter dla zbioru z 15% uszkodzeniem:

```
[19] net.predict(damaged15)
```

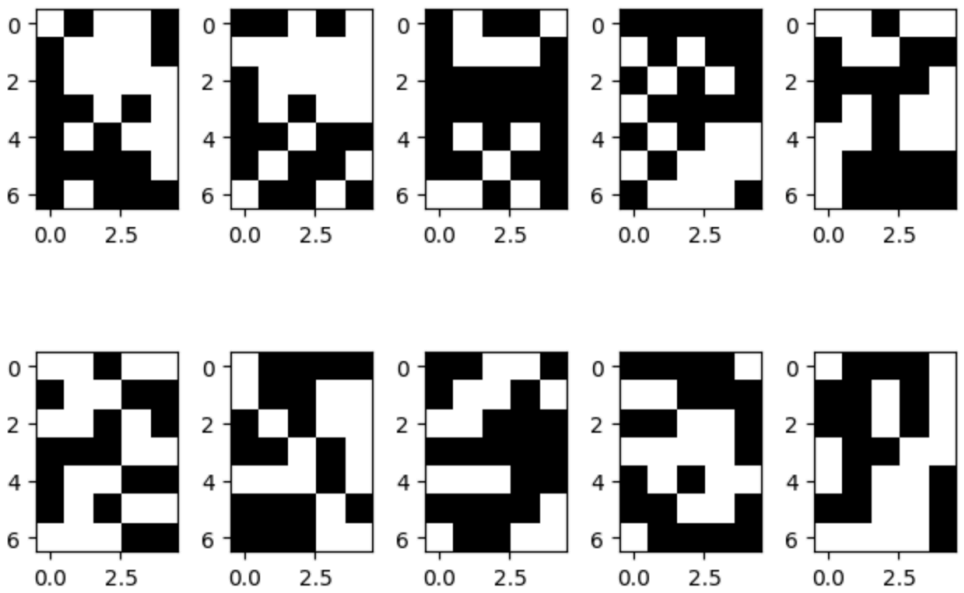
```
[array([-1, -1, -1, -1, -1, -1, -1, -1, -1, -1]),
 array([-1,  1, -1, -1, -1, -1,  1,  1, -1, -1]),
 array([-1, -1, -1,  1, -1, -1, -1, -1, -1, -1]),
 array([-1, -1, -1,  1, -1, -1, -1, -1, -1, -1]),
 array([-1, -1,  1,  1,  1, -1, -1, -1, -1, -1]),
 array([-1, -1, -1,  1, -1, -1,  1, -1, -1, -1]),
 array([-1,  1, -1, -1, -1, -1,  1, -1, -1,  1]),
 array([-1, -1, -1, -1, -1, -1, -1,  1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1, -1, -1,  1, -1]),
 array([-1, -1, -1, -1, -1, -1,  1, -1, -1,  1])]
```

```
net.misclassified(damaged15, y)
```

```
13
```

Zawartość zbioru X po uszkodzeniu 40%:

```
net.show(damaged40)
```



Wynik predict() i liczba błędnie sklasyfikowanych liter dla zbioru z 40% uszkodzeniem:

```
[324] net.predict(damaged40)
```

```
[array([-1, -1, -1,  1, -1, -1, -1, -1, -1,  1]),
 array([-1, -1, -1, -1,  1, -1,  1,  1, -1, -1]),
 array([-1, -1, -1, -1, -1, -1,  1,  1, -1, -1]),
 array([ 1, -1, -1, -1,  1, -1,  1, -1, -1, -1]),
 array([-1,  1, -1, -1,  1, -1,  1, -1,  1, -1]),
 array([ 1, -1,  1,  1, -1, -1,  1,  1, -1, -1]),
 array([-1, -1, -1, -1,  1, -1,  1,  1, -1,  1]),
 array([-1, -1, -1, -1,  1, -1, -1,  1, -1, -1]),
 array([-1, -1, -1, -1,  1, -1, -1,  1, -1, -1]),
 array([-1, -1, -1, -1,  1, -1, -1, -1, -1,  1])]
```

```
net.misclassified(damaged40, y)
```

```
31
```