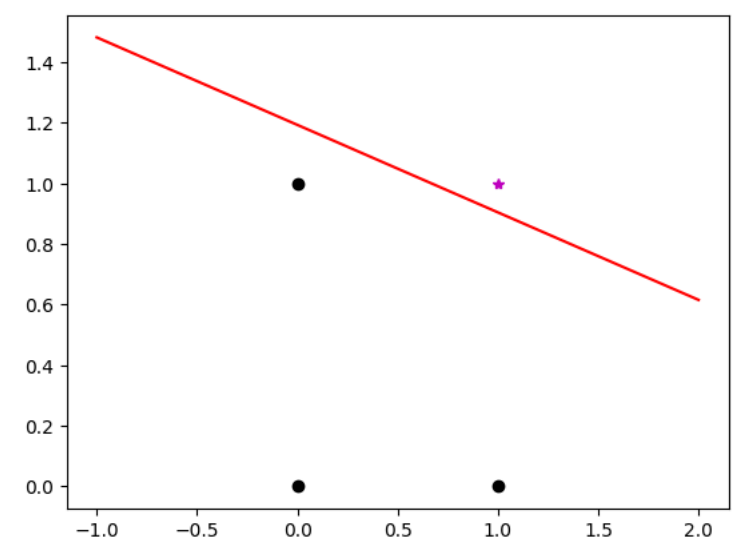
Labolatoria 3

Perceptron przykład

```
import numpy as np

x = np.array(
    [[0,0],
    [0,1],
    [1,0],
    [1,1]])

d = np.array([0,0,0,1])

w = np.random.random(3)


for i in range( len(x) ):
  xx = x[i]
  dd = d[i]

  s = xx[0]*w[1] + xx[1]*w[2] + w[0]*(-1)
  if s >= 0:
    y = 1
  else:
    y = 0

  print("x1: ", xx[0], " x2: ", xx[1], " d: ", dd, " y: ", y)
```

```
x1:  0  x2:  0  d:  0  y:  0
x1:  0  x2:  1  d:  0  y:  1
x1:  1  x2:  0  d:  0  y:  0
x1:  1  x2:  1  d:  1  y:  1
```

```
mi = 0.1
for a in range(1):
  for i in range( len(x) ):
    xx = x[i]
    dd = d[i]

    s = xx[0]*w[1] + xx[1]*w[2] + w[0]*(-1)
    if s >= 0:
      y = 1
    else:
      y = 0

    w[0] = w[0] + mi*(dd-y)*(-1)
    w[1] = w[1] + mi*(dd-y)*xx[0]
    w[2] = w[2] + mi*(dd-y)*xx[1]


import matplotlib.pyplot as plt
xx = np.arange(-1,3)
yy = -(w[1]/w[2]) * xx + (w[0]/w[2])
plt.plot(xx,yy, 'r-')

for i in range( len(x) ):
  if d[i] == 0:
    plt.plot(x[i, 0], x[i, 1], 'ko')
  else:
    plt.plot(x[i, 0], x[i, 1], 'm*')
```



Zadanie

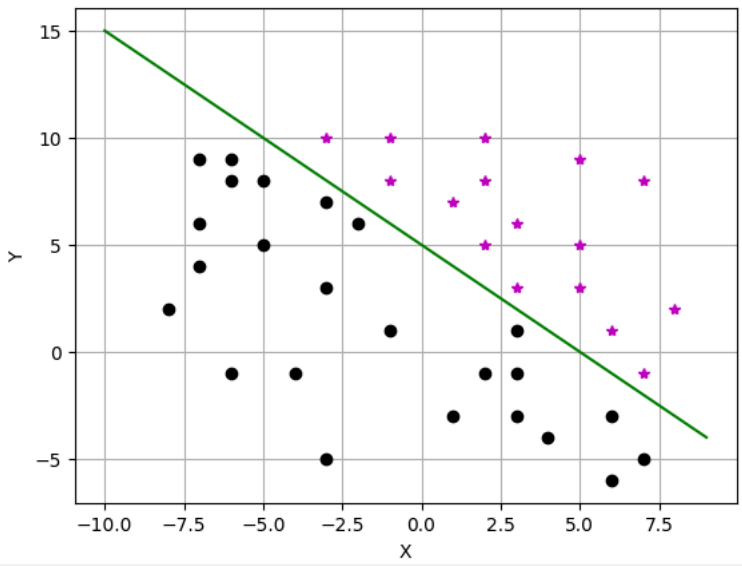
```
x = np.array([[-7.,  9.],
       [ 5.,  5.],
       [ 2., 10.],
       [-6.,  9.],
       [-5.,  8.],
       [-6.,  8.],
       [-3.,  3.],
       [-7.,  4.],
       [-7.,  6.],
       [-5.,  5.],
       [-2.,  6.],
       [-1.,  8.],
       [-3., 10.],
       [-1.,  1.],
       [ 2.,  8.],
       [-1., 10.],
       [ 1.,  7.],
       [-3.,  7.],
       [ 2.,  5.],
       [ 3.,  6.],
       [ 3.,  3.],
       [ 3.,  1.],
       [ 3., -1.],
       [ 2., -1.],
       [ 7., -5.],
       [ 6.,  1.],
       [ 7., -1.],
       [ 8.,  2.],
       [ 3., -3.],
       [ 1., -3.],
       [ 7.,  8.],
       [ 5.,  9.],
       [-6., -1.],
       [-4., -1.],
       [-8.,  2.],
       [-3., -5.],
       [ 4., -4.],
       [ 6., -3.],
       [ 5.,  3.],
       [ 6., -6.]])
```

```
d = np.array([0,
 1,
 1,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 0,
 1,
 1,
```

```
 0,
 1,
 1,
 1,
 0,
 1,
 1,
 1,
 0,
 0,
 0,
 0,
 1,
 1,
 1,
 0,
 0,
 1,
 1,
 0,
 0,
 0,
 0,
 0,
 0,
 1,
 0])

w = np.random.random(3)
```

Wykres poczatkowy

```
xx = np.arange(-10,10)
yy = -xx + 5

plt.plot(xx, yy, 'g-')

for i in range(len(x)):
  if d[i] == 0:
    plt.plot(x[i, 0], x[i, 1], 'ko')
  else:
    plt.plot(x[i, 0], x[i, 1], 'm*')


plt.xlabel("X")
plt.ylabel("Y")
plt.grid(True)
plt.show()
```



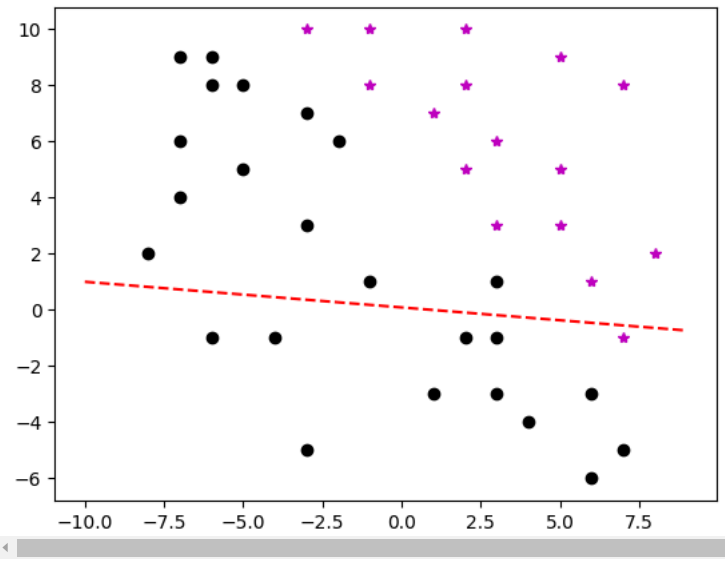

```
for i in range( len(x) ):
  xx = x[i]
  dd = d[i]

  s = xx[0]*w[1] + xx[1]*w[2] + w[0]*(-1)
  if s >= 0:
    y = 1
```

```
  else:
    y = 0

  print(i, ": ", "x1: ", xx[0], " x2: ", xx[1], " d: ", dd, " y: ", y)
```

```
0 :  x1:  -7.0  x2:  9.0  d:  0  y:  1
1 :  x1:  5.0  x2:  5.0  d:  1  y:  1
2 :  x1:  2.0  x2:  10.0  d:  1  y:  1
3 :  x1:  -6.0  x2:  9.0  d:  0  y:  1
4 :  x1:  -5.0  x2:  8.0  d:  0  y:  1
5 :  x1:  -6.0  x2:  8.0  d:  0  y:  1
6 :  x1:  -3.0  x2:  3.0  d:  0  y:  1
7 :  x1:  -7.0  x2:  4.0  d:  0  y:  1
8 :  x1:  -7.0  x2:  6.0  d:  0  y:  1
9 :  x1:  -5.0  x2:  5.0  d:  0  y:  1
10 :  x1:  -2.0  x2:  6.0  d:  0  y:  1
11 :  x1:  -1.0  x2:  8.0  d:  1  y:  1
12 :  x1:  -3.0  x2:  10.0  d:  1  y:  1
13 :  x1:  -1.0  x2:  1.0  d:  0  y:  1
14 :  x1:  2.0  x2:  8.0  d:  1  y:  1
15 :  x1:  -1.0  x2:  10.0  d:  1  y:  1
16 :  x1:  1.0  x2:  7.0  d:  1  y:  1
17 :  x1:  -3.0  x2:  7.0  d:  0  y:  1
18 :  x1:  2.0  x2:  5.0  d:  1  y:  1
19 :  x1:  3.0  x2:  6.0  d:  1  y:  1
20 :  x1:  3.0  x2:  3.0  d:  1  y:  1
21 :  x1:  3.0  x2:  1.0  d:  0  y:  1
22 :  x1:  3.0  x2:  -1.0  d:  0  y:  0
23 :  x1:  2.0  x2:  -1.0  d:  0  y:  0
24 :  x1:  7.0  x2:  -5.0  d:  0  y:  0
25 :  x1:  6.0  x2:  1.0  d:  1  y:  1
26 :  x1:  7.0  x2:  -1.0  d:  1  y:  0
27 :  x1:  8.0  x2:  2.0  d:  1  y:  1
28 :  x1:  3.0  x2:  -3.0  d:  0  y:  0
29 :  x1:  1.0  x2:  -3.0  d:  0  y:  0
30 :  x1:  7.0  x2:  8.0  d:  1  y:  1
31 :  x1:  5.0  x2:  9.0  d:  1  y:  1
32 :  x1:  -6.0  x2:  -1.0  d:  0  y:  0
33 :  x1:  -4.0  x2:  -1.0  d:  0  y:  0
34 :  x1:  -8.0  x2:  2.0  d:  0  y:  1
35 :  x1:  -3.0  x2:  -5.0  d:  0  y:  0
36 :  x1:  4.0  x2:  -4.0  d:  0  y:  0
37 :  x1:  6.0  x2:  -3.0  d:  0  y:  0
38 :  x1:  5.0  x2:  3.0  d:  1  y:  1
39 :  x1:  6.0  x2:  -6.0  d:  0  y:  0
```

Wykres bez uczenia

```
import matplotlib.pyplot as plt
xx = np.arange(-10,10)
yy = -(w[1]/w[2]) * xx + (w[0]/w[2])
yy_before = yy
plt.plot(xx, yy_before, color="red", linestyle="--", label="Przed uczeniem")

for i in range( len(x) ):
  if d[i] == 0:
    plt.plot(x[i, 0], x[i, 1], 'ko')
  else:
    plt.plot(x[i, 0], x[i, 1], 'm*')
```

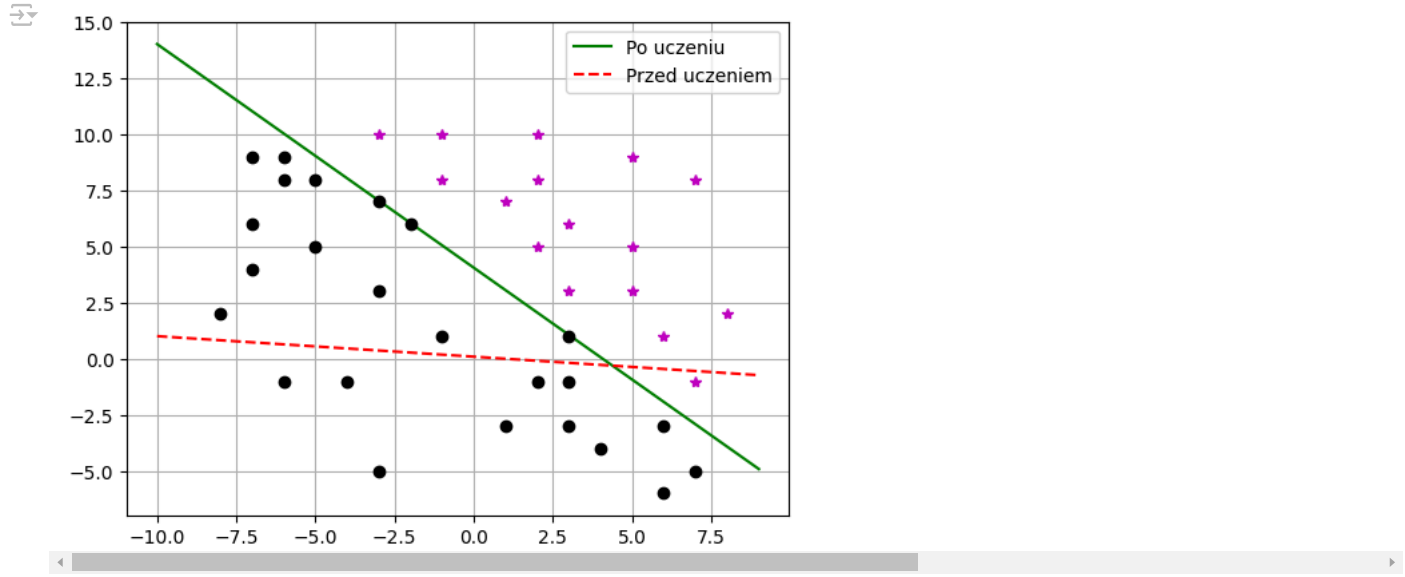


Nauka punktow

```
mi = 0.1
for a in range(100):
  for i in range( len(x) ):
    xx = x[i]
    dd = d[i]

    s = xx[0]*w[1] + xx[1]*w[2] + w[0]*(-1)
    if s >= 0:
      y = 1
    else:
      y = 0

    w[0] = w[0] + mi*(dd-y)*(-1)
    w[1] = w[1] + mi*(dd-y)*xx[0]
    w[2] = w[2] + mi*(dd-y)*xx[1]
```

Wykres po uczeniu punktów

```
import matplotlib.pyplot as plt
xx = np.arange(-10,10)
yy = -(w[1]/w[2]) * xx + (w[0]/w[2])
plt.plot(xx, yy, color="green", linestyle="-", label="Po uczeniu")
plt.plot(xx, yy_before, color="red", linestyle="--", label="Przed uczeniem")


for i in range( len(x) ):
  if d[i] == 0:
    plt.plot(x[i, 0], x[i, 1], 'ko')
  else:
    plt.plot(x[i, 0], x[i, 1], 'm*')

plt.legend()
plt.grid(True)
plt.show()
```



Nauka preceptronu tylko do momentu nauczenia się punktów

```
w = np.random.random(3)
import matplotlib.pyplot as plt

mi = 0.003
max_iterations = 100

errors_per_iteration = []

for a in range(max_iterations):
    errors = 0

    for i in range(len(x)):
        xx = x[i]
        dd = d[i]

        s = xx[0] * w[1] + xx[1] * w[2] + w[0] * (-1)
```

```
        y = 1 if s >= 0 else 0

        if y != dd:
            errors += 1
            w[0] = w[0] + mi * (dd - y) * (-1)
            w[1] = w[1] + mi * (dd - y) * xx[0]
            w[2] = w[2] + mi * (dd - y) * xx[1]

    errors_per_iteration.append(errors)

    if errors == 0:
        print(f"Perceptron nauczony po {a+1} iteracjach.")
        break
else:
    print("Osiągnięto maksymalną liczbę iteracji bez pełnego nauczenia.")


plt.plot(range(1, len(errors_per_iteration) + 1), errors_per_iteration, marker='o')
plt.xlabel("Iteracja")
plt.ylabel("Liczba błędów")
plt.title("Liczba błędów podczas uczenia perceptronu")
plt.grid()
plt.show()
```

```
Perceptron nauczony po 21 iteracjach.
```

Klasyfikacja nowych punktów

```
new_points = []
for i in range(3):
    x1 = float(input(f"Wprowadź x1 dla punktu {i+1}: "))
    x2 = float(input(f"Wprowadź x2 dla punktu {i+1}: "))
    new_points.append([x1, x2])

new_points = np.array(new_points)
classifications = []
for point in new_points:
    s = point[0] * w[1] + point[1] * w[2] + w[0] * (-1)
    y = 1 if s >= 0 else 0
    classifications.append(y)
    print(f"Punkt ({point[0]}, {point[1]}) sklasyfikowany jako: {y}")

for i, point in enumerate(x):
    plt.scatter(point[0], point[1], color='blue' if d[i] == 0 else 'red', label=f'Znane punkty' if i == 0 else "")


for i, point in enumerate(new_points):
  if classifications[i] == 0:
    plt.scatter(point[0], point[1], color='green', marker='x', s=100, label=f'Nowe punkty' if i == 0 else "")
  else:
    plt.scatter(point[0], point[1], color='red' , marker='x', s=100, label=f'Nowe punkty' if i == 0 else "")


xx = np.arange(-10,10)
yy = -(w[1]/w[2]) * xx + (w[0]/w[2])
plt.plot(xx, yy, color="green", linestyle="-", label="Po uczeniu")
```

```
plt.legend()
plt.show()
```

```
Wprowadź x1 dla punktu 1: 4
Wprowadź x2 dla punktu 1: 6
Wprowadź x1 dla punktu 2: 2
Wprowadź x2 dla punktu 2: -3
Wprowadź x1 dla punktu 3: -4
Wprowadź x2 dla punktu 3: 7
Punkt (4.0, 6.0) sklasyfikowany jako: 1
Punkt (2.0, -3.0) sklasyfikowany jako: 0
Punkt (-4.0, 7.0) sklasyfikowany jako: 0
```