

VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

INFORMACINIŲ SISTEMŲ INŽINERIJOS STUDIJŲ PROGRAMA

**Pirma užduotis (Dirbtinis neuronas)**

Laboratorinio darbo ataskaita

Atliko: Klaidas Kubilius

VU el. p.: klaidas.kubilius@mif.stud.vu.lt

Vilnius

2025

1. Darbo eiga
   1. Užduoties tikslas

**Tikslas** - Išanalizuoti dirbtinio neurono modelio veikimo principus.

* 1. Sugeneruoti duomenys

1.1 lentelė. Sugeneruotų duomenų koordinatės

|  |  |  |
| --- | --- | --- |
| X | Y | Class |
| 1.62 | 2.14 | 0 |
| 3.96 | 3.42 | 0 |
| 2.05 | 4.78 | 0 |
| 3.13 | 4.41 | 0 |
| 1.06 | 1.01 | 0 |
| 4.67 | 3.08 | 0 |
| 4.60 | 3.21 | 0 |
| 1.13 | 2.94 | 0 |
| 4.83 | 4.07 | 0 |
| 1.55 | 1.64 | 0 |
| 9.06 | 8.81 | 1 |
| 6.08 | 7.31 | 1 |
| 6.54 | 7.34 | 1 |
| 6.47 | 9.91 | 1 |
| 7.24 | 8.50 | 1 |
| 8.69 | 9.80 | 1 |
| 7.88 | 9.07 | 1 |
| 9.26 | 9.30 | 1 |
| 7.16 | 7.63 | 1 |
| 8.93 | 7.81 | 1 |

Duomenys išskaidyti į dvi klases: pirma klasė (0) turi x ir y koordinates nuo 1 iki 5, o antroji klasė (1) – nuo 6 iki 10.

A screenshot of a computer

AI-generated content may be incorrect.

1.1 pav. Sugeneruoti duomenys koordinačių sistemoje

* 1. Programos kodas

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

import sys

sys.stdout.reconfigure(encoding='utf-8')

# sėklos reikšmė atkuriamumui

np.random.seed(12)

# Generuojama pirma klasė (žalia) (nuo 1 iki 5)

class1\_x = np.random.uniform(1, 5, 10)

class1\_y = np.random.uniform(1, 5, 10)

class1 = np.column\_stack((class1\_x, class1\_y))

# Generuojama antra klasė (violetinė) (nuo 6 iki 10)

class2\_x = np.random.uniform(6, 10, 10)

class2\_y = np.random.uniform(6, 10, 10)

class2 = np.column\_stack((class2\_x, class2\_y))

# Sujungiami duomenys

labels = np.array([0] \* 10 + [1] \* 10) # 0 - pirma klasė, 1 - antra klasė

X = np.vstack((class1, class2))

# Vizualizuojame duomenis

plt.figure(figsize=(6, 6))

plt.scatter(class1[:, 0], class1[:, 1], color='green', label='Klasė 0')

plt.scatter(class2[:, 0], class2[:, 1], color='purple', label='Klasė 1')

plt.xlabel("X ašis")

plt.ylabel("Y ašis")

plt.title("Sugeneruoti duomenys")

plt.legend()

plt.grid(True)

plt.show()

# Išspausdinti taškų koordinates ir klases

for (x, y), label in zip(X, labels):

print(f"Koordinatės: ({x:.2f}, {y:.2f}), Klasė: {label}")

# Aktyvacijos funkcija

activation = 'sigmoid' # 'threshold' (slenkstinė) arba 'sigmoid' (sigmoidinė)

weights\_bias\_sets = []

for i in range(3): # Trys tinkami rinkiniai

for i in range(10000): # Bandymų skaičius

w = np.random.uniform(-1, 1, 2)

b = np.random.uniform(-1, 1)

a = np.dot(X, w) + b

if activation == 'threshold':

output = np.where(a >= 0, 1, 0)

elif activation == 'sigmoid':

output = np.round(1 / (1 + np.exp(-a)))

else:

raise ValueError("Nežinoma aktyvacijos funkcija")

if np.array\_equal(output, labels):

weights\_bias\_sets.append((w, b))

print(f"Rasti {activation} tinkami svoriai {len(weights\_bias\_sets)}: {w}, Bias: {b}")

break

# Vizualizuojame duomenis, tieses ir svorių vektorius

plt.figure(figsize=(5, 5))

plt.scatter(class1[:, 0], class1[:, 1], color='green', label='Klasė 0')

plt.scatter(class2[:, 0], class2[:, 1], color='purple', label='Klasė 1')

colors = ['red', 'blue', 'orange'] # Tiesių ir vektorių spalvos

for i, (w, b) in enumerate(weights\_bias\_sets):

if w is not None: # Tikriname ar svoriai rasti

x\_vals = np.linspace(0, 10, 100)

y\_vals = -(w[0] \* x\_vals + b) / w[1]

plt.plot(x\_vals, y\_vals, color=colors[i], linestyle='dashed', label=f'Tiesė {i+1}')

# Pasirenkame tašką ant tiesės (x=5)

x0 = 5

y0 = -(w[0] \* x0 + b) / w[1]

# Vektoriai atvaizduojami kaip rodyklės

plt.quiver(x0, y0, w[0], w[1], color=colors[i], angles='xy', scale\_units='xy', scale=0.08, width=0.005)

plt.xlabel("X ašis")

plt.ylabel("Y ašis")

plt.title("Duomenys su klasifikavimo tiesėmis ir vektoriais")

plt.legend()

plt.grid(True)

plt.axis('equal') # Vektoriai statmeni

# Nustatome x ir y ašių ribas nuo 0 iki 10

plt.xlim(0, 10)

plt.ylim(0, 10)

plt.show()

* 1. Svorių ir poslinkių paieška

Svoriai ir poslinkiai buvo ieškomi atsitiktinai generuojant skaičius tarp -1 ir 1 iki 10000 bandymų, po 3 rinkinius: 2 svorius ir poslinkį. Aktyvacijai naudojama slenkstinė arba sigmoidinė funkcija. Kai naudojama slenkstinė funkcija, jei gauta reikšmė a≥0a, tai priklauso klasei 1, o jei a<0, tada klasei 0. Kai naudojama sigmoidinė funkcija, apskaičiuojama pagal formulę ir rezultatas apvalinamas iki 0 arba 1. Tada gauta reikšmė palyginama su tikrąja taško klase.

* 1. Svorių ir poslinkio reikšmių rinkiniai

Svoriai ir poslinkio rinkiniai slenkstinei ir sigmoidinei funkcijai gaunasi tokie patys.

1.2 lentelė. Svorių ir poslinkio rinkiniai

|  |  |  |  |
| --- | --- | --- | --- |
| Rinkiniai | Svoris W1 | Svoris W2 | Poslinkis b |
| 1 | 0.15342852 | -0.00286854 | -0.7511472642633321 |
| 2 | 0.04687523 | 0.13270317 | -0.9936566860936882 |
| 3 | 0.05669221 | 0.08075526 | -0.8292815333363901 |

* 1. Duomenų taškų, tiesių ir vektorių grafikas

Grafike pavaizduoti klasių duomenų taškai, jas skiriančios tiesės ir jų neurono svorius atitinkantys vektoriai. Žali taškai – pirmos klasės (0), violetiniai taškai – antros klasės (1). Raudona – pirma tiesė (1), mėlyna – antra tiesė (2), geltona – trečioji tiesė (3). Taip pat vektorių spalvos atitinka tas pačias tieses.

A screen shot of a computer screen

AI-generated content may be incorrect.

1.2 pav. Duomenų taškų ir klases skiriancių tiesių ir ju svorių vektorių grafikas

1. Išvados

Naudojant slenkstinę ar sigmoidinę funkcija rezultatai gaunasi tokie patys, nes sigmoidinė funkcija apvalinama iki 0 ar 1.

Gautos tieses ir vektoriai vizualiai parodo kaip atskiriamos klases pagal dirbtinį neoroną, tai leidžia matyti kaip modelis priima sprendimus ir kokia įtaką turi skirtingi svoriai ir poslinkiai.

Nors modelis veikia šiuo atveju, jis neveiktų jeigu duomenys persidengtų.

1. Papildoma

Generuota:

Vektorių vaizdavimo kodas

Gramatinėm klaidom taisyti ataskaitoje