VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

Praktinė užduotis Nr. 2

Vieno neurono mokymas sprendžiant klasifikavimo uždavinį

Atliko: Programų sistemų 4 k. 1 gr. stud. Rokas Petrauskas

VILNIUS, 2023

Turinys

Tikslas

Šios užduoties tikslas yra apmokyti vieną neuroną spręsti dviejų klasių uždavinį ir atlikti tyrimą su dviem duomenų aibėm bei nustatyti optimalius hiperparametrus (mokymosi greitis, epochų skaičius).

Naudojami duomenys

Dirbtinio neurono mokymui ir testavimui buvo naudotos dvejos duomenų aibės:

1. Irisų duomenų aibė https://archive.ics.uci.edu/dataset/53/iris.
2. Krūties vėžio duomenų aibė https://archive.ics.uci.edu/dataset/15/breast+cancer+wisconsin+original.

Irisų duomenų aibę sudaro 150 eilučių duomenų, 50 iš jų priklauso Iris – setosa klasei, 50 – Iris – versicolor, 50 – Iris – virginica. Neuronas apmokomas dvejoms klasėms, todėl Iris – setosa kategorijos eilutės pašalinamos, tad naudota 50 Iris – versicolor klasės duomenų eilučių ir 50 Iris – virginica klasės duomenų eilučių, sumoje 100. Vieną duomenų eilutę sudaro 5 atributai: taurėlapio ilgis (cm), taurėlapio plotis (cm), žiedlapio ilgis (cm), žiedlapio plotis (cm), klasė. Duomenų aibėje tuščių atributų nėra.

Krūties vėžio duomenų aibę sudaro 699 eilutės duomenų, 458 iš jų priklauso nepiktybinio naviko klasei, 241 – piktybinio naviko klasei. 16 Duomenų aibės eilučių turi atributus neapibrėžtomis reikšmėmis, tos eilutės pašalintos, tad naudotos 683 duomenų eilutės. Vieną duomenų eilutę sudaro 11 atributų: pavyzdinis kodo numeris (id), gumbo storis (1 – 10), ląstelių dydžio vienodumas (1 – 10), ląstelių formos vienodumas (1 – 10), ribinis sukibimas (1 – 10), Vienos epitelinės ląstelės dydis (1 – 10), neapsaugotas branduolys (1 – 10), chromatinas (1 – 10), įprasti branduoliai (1 – 10), mitozė (1 – 10), klasė (2 – nepiktybinis navikas, 4 – piktybinis navikas).

Duomenų paruošimas

Duomenų aibėse naudojami klasių pavadinimai pakeičiami į vertes 0 ir 1. Irisų duomenų aibės klasių reikšmės Iris – versicolor keičiamos į 0, o Iris – virginica keičiamos į 1. Krūties vėžio duomenų aibės klasių reikšmės 2 keičiamos į 0 (nepiktybinis navikas), o reikšmės 4 keičiamos į 1 (piktybinis navikas). Po to iš krūties vėžio duomenų aibės pašalinami pavyzdinio kodo numerio atributai, kadangi jie skirti duomenų identifikacijai. Galiausiai abejos duomenų aibės padalinamos į treniravimo ir testavimo aibes santykiu 80:20. Imamos atsitiktinės eilutės, nes irisų duomenų aibėje eilutės išrikiuotos pagal klasės atributą.

Užduoties variantas

Studento numeris 1911087. Paskutinio skaitmens dalybos iš 3 liekana 1. Užduoties variantas 1. Neurono mokymui naudoti stochastinį gradientinį nusileidimą ir sigmoidinį neuroną.

Pradinių svorių ir poslinkio pasirinkimas

Pradinių svorių aibė užpildoma nuliais pagal atributų skaičių, pradinei poslinkio reikšmei priskiriamas nulis. Neuronui mokantis šios reikšmės kinta.

Programos kodas

```
import os
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
# duomenų failo apdorojimas. 1 dalis

# Iris-versicolor -> 0
# Iris-virginica -> 1

with open("data/iris.data", "r") as file:
    lines = file.readlines()

with open("data/iris_chg.data", "w") as output_file:
    for line in lines:
        if "Iris-versicolor" in line:
            line = line.replace("Iris-versicolor", "0")
        elif "Iris-virginica" in line:
            line = line.replace("Iris-virginica", "1")
        output_file.write(line)

# 2 -> 0 (nepiktybinis navikas)
# 4 -> 1 (piktybinis navikas)

with open("data/breast-cancer-wisconsin.data", "r") as file:
    lines = file.readlines()

with open("data/breast-cancer-wisconsin_chg.data", "w") as output_file:
    for line in lines:
        parts = line.strip().split(',')
        if len(parts) > 1:
            if parts[-1] == "2":
                parts[-1] = "0"
            elif parts[-1] == "4":
                parts[-1] = "1"
            new_line = ','.join(parts) + '\n'
            output_file.write(new_line)

# duomenų failo apdorojimas. 2 dalis

with open("data/breast-cancer-wisconsin_chg.data", 'r') as f:
    lines = f.readlines()

with open("data/breast-cancer-wisconsin_chg2.data", 'w') as f:
    for line in lines:
        data = line.strip().split(',')
        f.write(','.join(data[1:]) + '\n')
```

```
# duomenų failo apdorojimas. 3 dalis
def split_data(original_file, train_file, test_file):
    data = pd.read_csv(original_file, header=None)
    train, test = train_test_split(data, test_size=0.2)
    train.to_csv(train_file, index=False, header=False)
    test.to_csv(test_file, index=False, header=False)

split_data("data/iris_chg.data", "data/iris/iris_train.data",
"data/iris/iris_test.data")
split_data("data/breast-cancer-wisconsin_chg2.data", "data/bc/bc_train.data",
"data/bc/bc_test.data")

# pašalinama paskutinioji (tuščia) failo eilutė
for filename in ["data/iris/iris_train.data", "data/iris/iris_test.data",
"data/bc/bc_train.data","data/bc/bc_test.data"]:
    with open(filename, "rb+") as filehandle:
        filehandle.seek(-2, os.SEEK_END)
        filehandle.truncate()

class ArtificialNeuron:
    """
    ArtificialNeuron Klasė realizuojanti dirbtinį neuroną.
    """

    def __init__(self):
        """
        __init__ metodas, šiuo metu nepriskiriame pradinių svorių ir poslinkio.
        """
        self.weights = None
        self.bias = None

    def activate(self, inputs):
        """
        activate metodas sumuoja svorių ir įėjimų sandaugas ir prideda poslinkio
vertę, grąžinama numatoma klasė.
        """
        weighted_sum = np.dot(inputs, self.weights) + self.bias
        output = self.sigmoid(weighted_sum)
        return output

    def sigmoid(self, x):
        """
        sigmoid metodas naudojamas activate funkcijoje, tam, kad apibrėžtų išeities
reikšmę pagal logistinę funkciją.
        """
        return 1 / (1 + np.exp(-x))

    def train(self, training_file, learning_rate, epochs):
```

```
        """
        train metodas apmoko dirbtinį neuroną naudojant stochastinį gradientinį
nusileidimą.
        """
        # nustatomas svorių skaičius
        weight_number = 0
        with open(training_file, 'r') as f:
            line = f.readline()
            weight_number = line.count(",")

        # inicializuojami svoriai ir poslinkis
        self.weights = [0] * weight_number
        self.bias = 0

        # apmokomas neuronas
        errors = []
        accuracies = []
        for epoch in range(epochs):
            total_error = 0
            correct_guesses = 0
            total_guesses = 0
            with open(training_file, 'r') as file:
                for line in file:
                    parts = line.strip().split(',')
                    inputs = [float(part) for part in parts[:-1]]  # inputs pvz.:
[6.8, 2.8, 4.8, 1.4]
                    inputs = np.array(inputs)
                    target = float(parts[-1])  # target pvz.: 1.0

                    prediction = self.activate(inputs)
                    error = target - prediction

                    self.weights += learning_rate * error * inputs
                    self.bias += learning_rate * error

                    total_error += error

                    total_guesses += 1
                    if round(prediction) == target:
                        correct_guesses += 1
            errors.append(total_error)
            accuracy = correct_guesses / total_guesses
            accuracies.append(accuracy)
            print(f"Epocha Nr. {epoch+1}. Gauta paklaida: {total_error}")
            print(f"Tikslumas: {accuracy}")

        # atvaizduojame klaidos grafiką
        plt.plot(range(epochs), errors)
        plt.xlabel('Epochos')
```

```
        plt.ylabel('Klaida')
        plt.show()

    def test(self, test_file):
        """
        test metodas leidžia įvertinti modelio tikslumą su testavimo duomenimis.
        """
        total_squared_error = 0
        total_samples = 0

        correct_guesses = 0
        total_guesses = 0

        with open(test_file, 'r') as file:
            for line in file:
                parts = line.strip().split(',')
                inputs = [float(part) for part in parts[:-1]]
                inputs = np.array(inputs)
                target = float(parts[-1])

                prediction = self.activate(inputs)

                squared_error = (target - prediction) ** 2
                total_squared_error += squared_error
                total_samples += 1

                total_guesses += 1
                if round(prediction) == target:
                    correct_guesses += 1

                print("Spėjimas: " + str(round(prediction)) + ", Tikroji klasė: " +
str(target))


        mse = total_squared_error / (2 * total_samples)
        print(f"Gauta paklaida: {mse}")
        print(f"Tikslumas: {correct_guesses/total_guesses}")

        print(self.weights)
        print(self.bias)


neuron1 = ArtificialNeuron()
neuron1.train("data/bc/bc_train.data", 0.001, 500) # čia keičiamas mokymosi
greitis, epochų skaičius
neuron1.test("data/bc/bc_test.data")

neuron2 = ArtificialNeuron()
```

```
neuron1.train("data/iris/iris_train.data", 0.001, 100) # čia keičiamas mokymosi greitis, epochų skaičius
neuron1.test("data/iris/iris_test.data")
```

Programinio kodo pakeitimas

Programoje naudojami svorių ir poslinkio nustatymai neatitinka sąlygos. Žemiau aprašyti rezultatai apskaičiuoti šia programa. Rezultatai, atlikus programinio kodo pakeitimus, kad jie atitiktų sąlygą aprašyti skyriuje „Atnaujinti tyrimo rezultatai“. Programinio kodo pakeitimai:

```
                    self.weights -= learning_rate * (prediction - target) * prediction * (1 - prediction) * inputs
                    self.bias -= learning_rate * (prediction - target) * prediction * (1 - prediction)
```

Pakeistos šios 2 eilutės „train“ metode.

Sąvokos

Stochastinis gradientinis nusileidimas – gradientinis optimizavimo metodas, kai imamos pradinių svorių reikšmės kai gradientinio nusileidimo algoritmu judama antigradiento kryptimi, svorių reikšmes keičiant pagal iteracinę formulę. Gradientas aproksimuojamas gradientu, gautu pagal vieną mokymo duomenų įrašą (pav. 1).

$$w_k := w_k - \eta \nabla E_i(W)$$

Pav 1. Stochastinio gradiento nusileidimo formulė

η – mokymo greitis, kuriuo reguliuojamas gradientinio optimizavimo žingsnio ilgis.

Epocha – viena mokymo ciklo iteracija, kai visi mokymo duomenys yra panaudojami modeliui mokyti. Per vieną epochą visi mokymo duomenys yra naudojami bent vieną kartą modeliui atnaujinti ir optimizuoti.

Sigmoidinis neuronas – dirbtinis neuronas, naudojantis sigmoidinę aktyvacijos funkciją. Sigmoidinės aktyvacijos funkcija gali būti išreikšta formule (pav. 2).

$$f(a) = \frac{1}{1 + e^{-a}}$$

Pav. 2. Sigmoidinės funkcijos išraiška formule

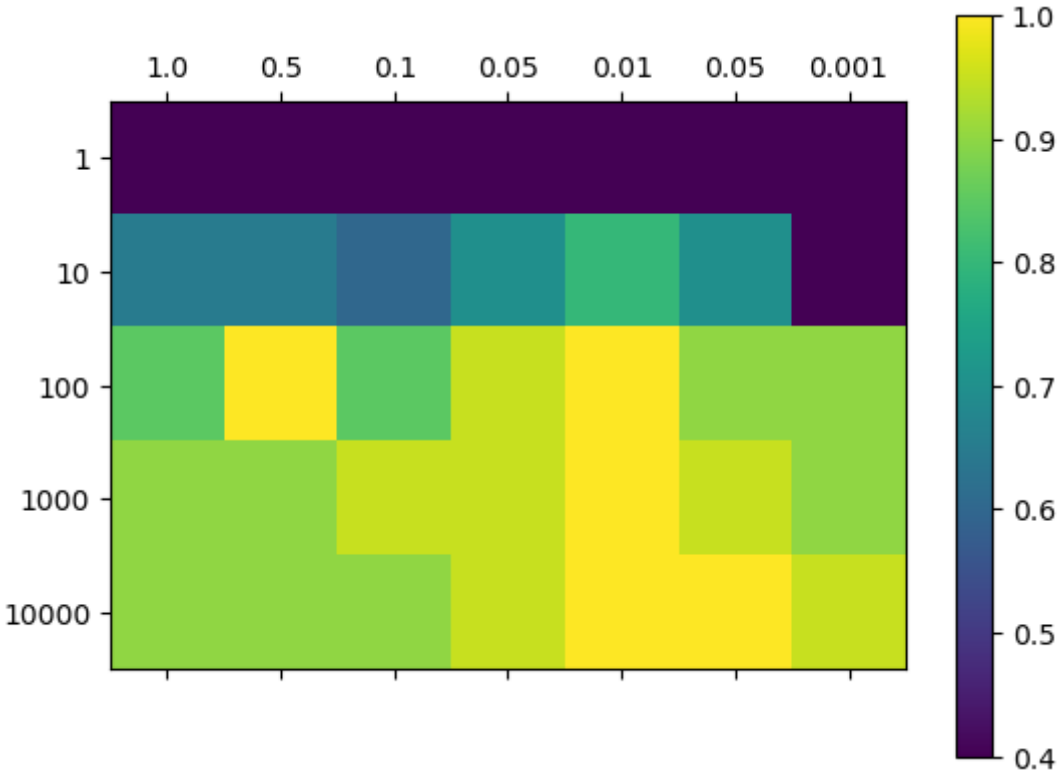
a – svorių ir įėjimų sandaugų bei poslinkio suma.

Tyrimo rezultatai

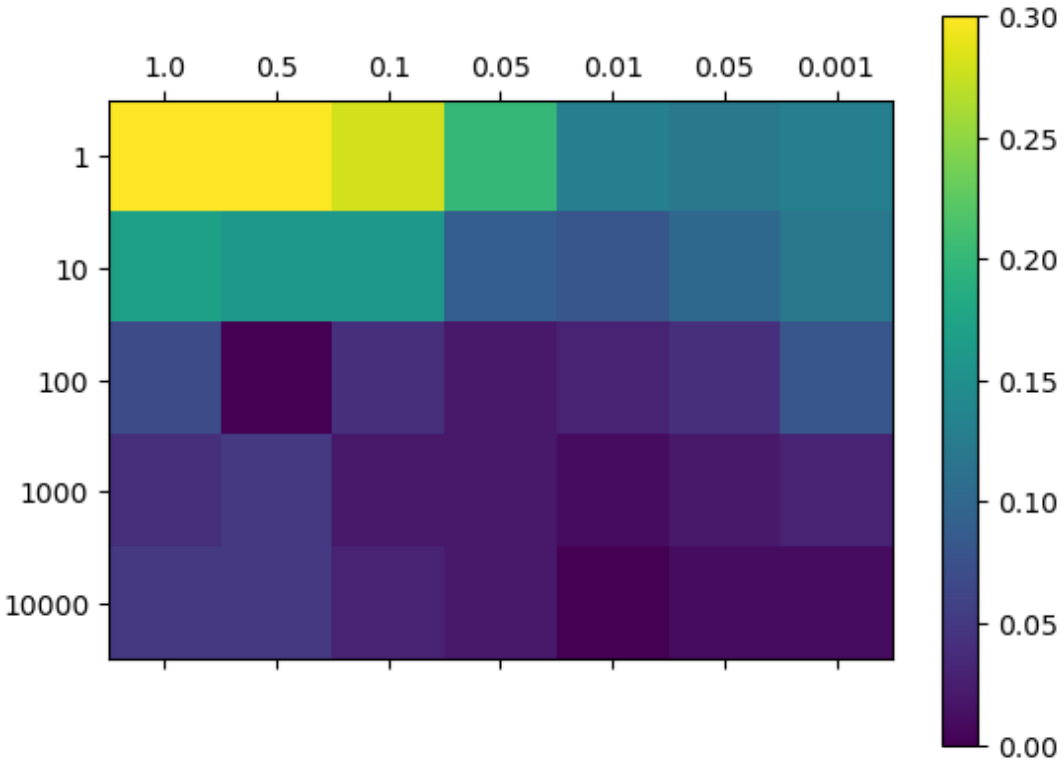
Tyrimo metu bandomi skirtinga hiperparametrų kombinacija, siekiant atrasti optimalų variantą. Pirmoje lentelės eilutėje – mokymosi greičiai (pažymėta žalia spalva), pirmame lentelės stulpelyje – epochų skaičius (pažymėta mėlyna spalva). Irisų duomenų aibę klasifikuojančio neurono testavimo paklaidos ir tikslumai užrašyti lentelėje 1, krūties vėžio – lentelėje 2. Geriausi rezultatai paryškinti oranžine spalva. Paveikslėliuose 3,4 ir 5,6 paklaidos ir tikslumo vertės išskaidomos pagal tiriamą duomenų aibę.

Lentelė 1. Irisų testavimo paklaidos ir tikslumo nuo mokymosi greičio ir epochų skaičiaus mokymo metu priklausomybė.

	1	0,5	0,1	0,05	0,01	0,005	0,001
1	Paklaida 0,3 Tikslumas 0,4	Paklaida 0,3 Tikslumas 0,4	Paklaida 0,28 Tikslumas 0,4	Paklaida 0,2 Tikslumas 0,4	Paklaida 0,13 Tikslumas 0,4	Paklaida 0,12 Tikslumas 0,4	Paklaida 0,13 Tikslumas 0,4
10	Paklaida 0,17 Tikslumas 0,65	Paklaida 0,16 Tikslumas 0,65	Paklaida 0,16 Tikslumas 0,6	Paklaida 0,09 Tikslumas 0,7	Paklaida 0,08 Tikslumas 0,8	Paklaida 0,1 Tikslumas 0,7	Paklaida 0,12 Tikslumas 0,4
100	Paklaida 0,07 Tikslumas 0,85	Paklaida 0,00 Tikslumas 1	Paklaida 0,04 Tikslumas 0,85	Paklaida 0,02 Tikslumas 0,95	Paklaida 0,03 Tikslumas 1	Paklaida 0,04 Tikslumas 0,9	Paklaida 0,08 Tikslumas 0,9
1000	Paklaida 0,04 Tikslumas 0,9	Paklaida 0,05 Tikslumas 0,9	Paklaida 0,02 Tikslumas 0,95	Paklaida 0,02 Tikslumas 0,95	Paklaida 0,01 Tikslumas 1	Paklaida 0,02 Tikslumas 0,95	Paklaida 0,03 Tikslumas 0,9
10000	Paklaida 0,05 Tikslumas 0,9	Paklaida 0,05 Tikslumas 0,9	Paklaida 0,03 Tikslumas 0,9	Paklaida 0,02 Tikslumas 0,95	Paklaida 0,00 Tikslumas 1	Paklaida 0,01 Tikslumas 1	Paklaida 0,01 Tikslumas 0,95



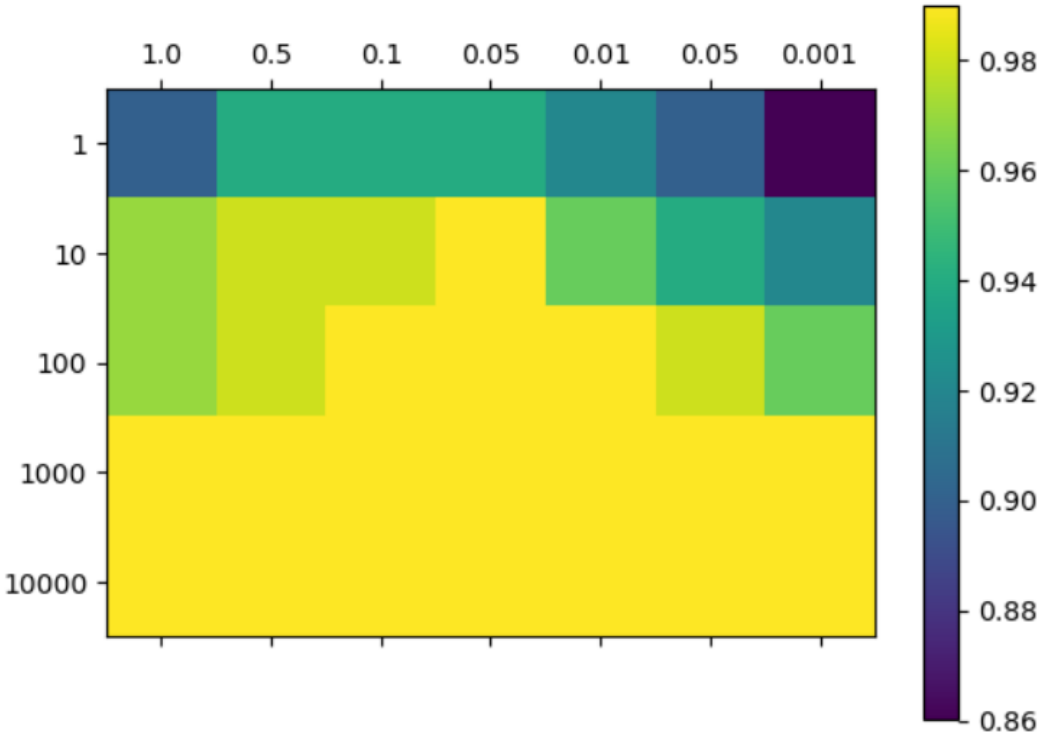
Pav. 3. Irisų testavimo tikslumas



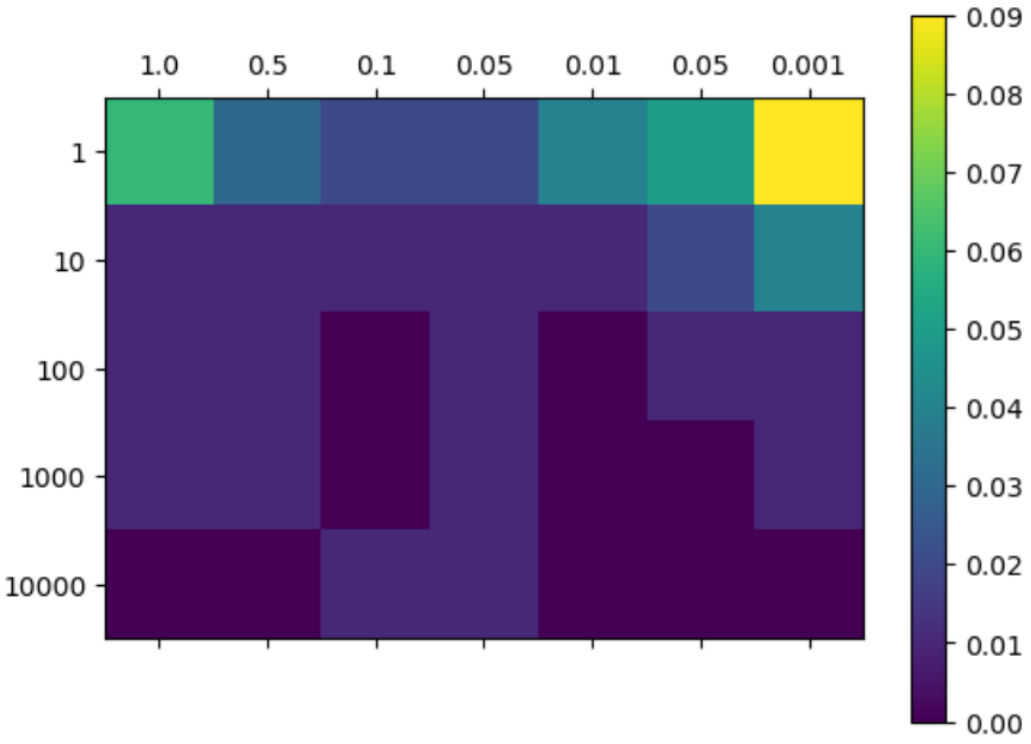
Pav. 4. Irisų testavimo paklaidos reikšmės

Lentelė 2. Krūties vėžio testavimo paklaidos ir tikslumo nuo mokymosi greičio ir epochų skaičiaus mokymo metu priklausomybė.

	1	0,5	0,1	0,05	0,01	0,005	0,001
1	Paklaida 0,06 Tikslumas 0,9	Paklaida 0,03 Tikslumas 0,94	Paklaida 0,02 Tikslumas 0,94	Paklaida 0,02 Tikslumas 0,94	Paklaida 0,04 Tikslumas 0,92	Paklaida 0,05 Tikslumas 0,9	Paklaida 0,09 Tikslumas 0,86
10	Paklaida 0,01 Tikslumas 0,97	Paklaida 0,01 Tikslumas 0,98	Paklaida 0,01 Tikslumas 0,98	Paklaida 0,01 Tikslumas 0,99	Paklaida 0,01 Tikslumas 0,96	Paklaida 0,02 Tikslumas 0,94	Paklaida 0,04 Tikslumas 0,92
100	Paklaida 0,01 Tikslumas 0,97	Paklaida 0,01 Tikslumas 0,98	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,01 Tikslumas 0,99	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,01 Tikslumas 0,98	Paklaida 0,01 Tikslumas 0,96
1000	Paklaida 0,01 Tikslumas 0,99	Paklaida 0,01 Tikslumas 0,99	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,01 Tikslumas 0,99	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,01 Tikslumas 0,99
10000	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,01 Tikslumas 0,99	Paklaida 0,01 Tikslumas 0,99	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,00 Tikslumas 0,99	Paklaida 0,00 Tikslumas 0,99



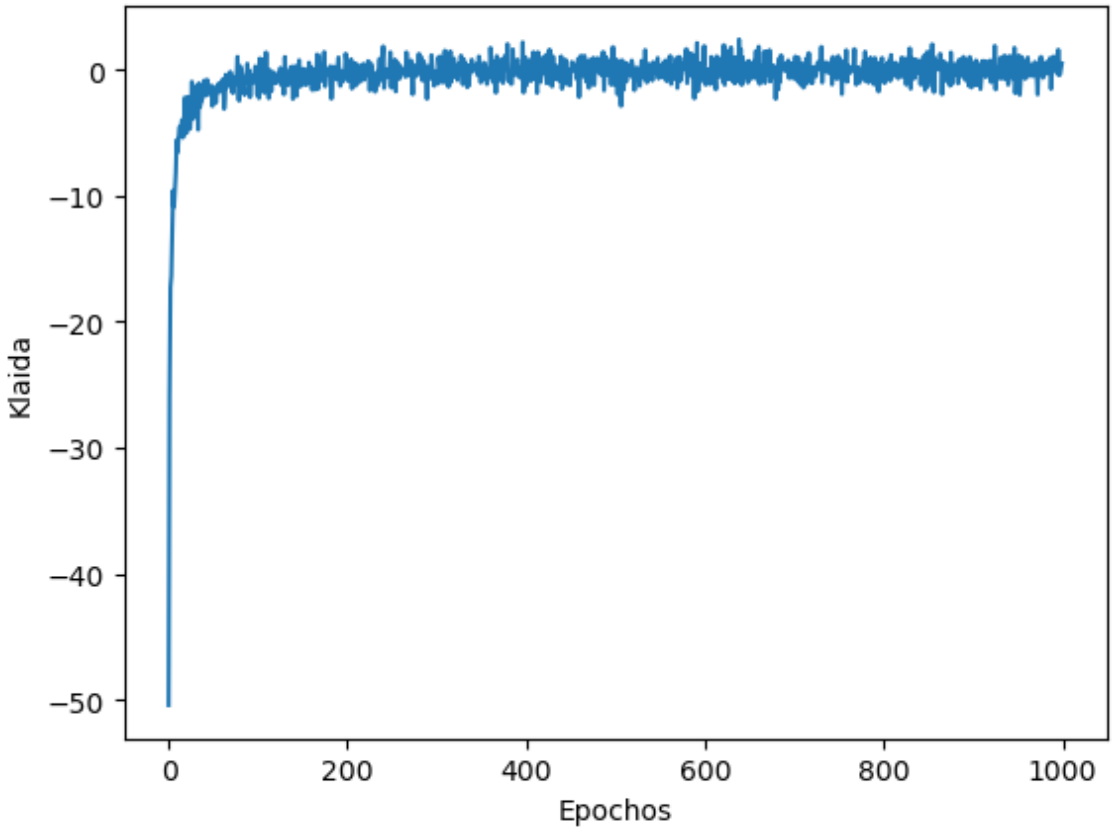
Pav. 5. Krūties vėžio testavimo tikslumas



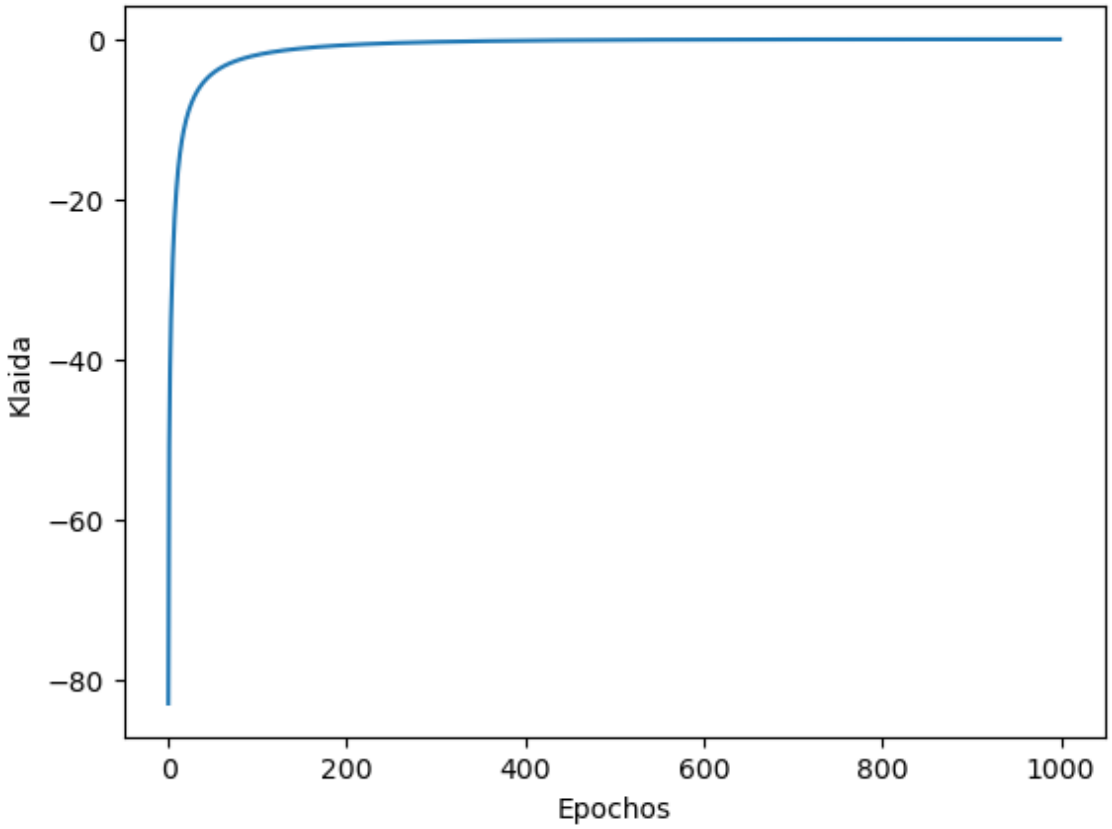
Pav. 6. Krūties vėžio testavimo paklaidos reikšmės

Paklaidos reikšmės priklausomybė nuo epochų skaičiaus

Analizuojant 4 ir 6 paveikslėlius galime daryti išvadą, kad paklaidos reikšmė mažėja didinant epochų skaičių iki tam tikro lygio, kurį apriboja mokymosi greičio vertė, tą galime pamatyti 7 paveikslėlyje, kuriame pavaizduota klaidos vertė po kiekvienos epochos, čia krūties vėžio aibė, mokymosi greitis – 0,5. Apie 50–ą epochą klaidos reikšmė pradeda stagnuoti. Sumažinus mokymosi greitį reikšmės ima tolydžiau artėti 0, žr. 8 paveikslėlį. Čia aibė ir hiperparametrai tie patys, pakeista tik mokymosi greičio vertė, dabar ji 0,01. Tačiau analizuojant tik klaidos reikšmę pagal epochų skaičių matyti, kad pirmos kelios epochos turi labai didelę svarbą klaidos mažėjimui, vėlesnės epochos didelio skirtumo nesudaro.



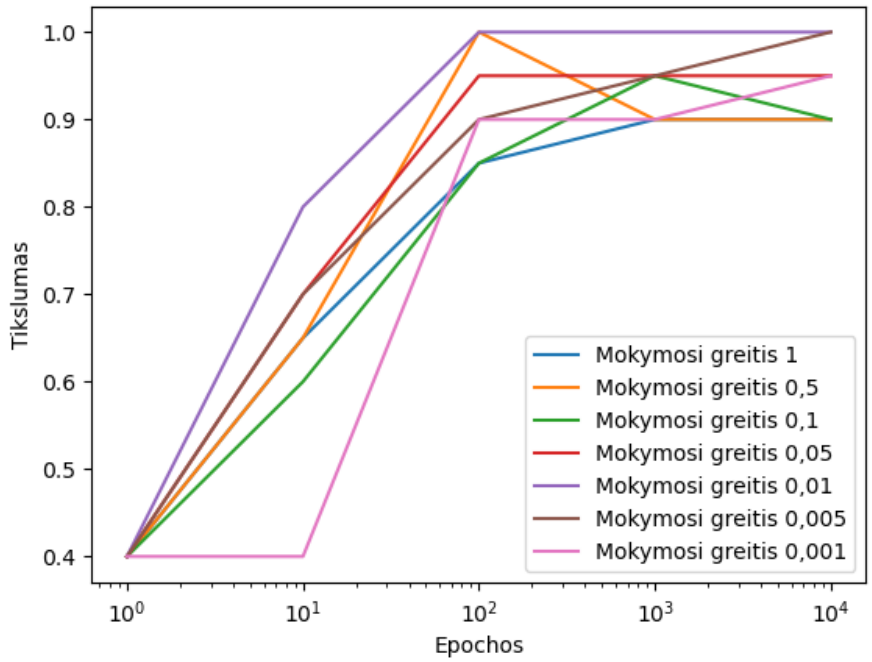
Pav. 7. Krūties vėžio aibės klaidos reikšmė po kiekvienos epochos, kai mokymosi greitis 0,5



Pav. 8. Krūties vėžio aibės klaidos reikšmė po kiekvienos epochos, kai mokymosi greitis 0,01

Klasifikavimo tikslumo priklausomybė nuo epochų skaičiaus

Analizuojant abiejų duomenų aibių duomenis matoma klasifikavimo tikslumo kitimo tendencija, tačiau ji ryškesnė irisų duomenų aibėje, nes ten mokymo aibė yra gerokai mažesnė, todėl epochų skaičius daro didesnę įtaką klasifikavimo tikslumui, šį pavyzdį ir analizuoju. 9 paveikslėlyje pavaizduota tikslumo priklausomybė nuo epochų skaičiaus, nesunku pastebėti, kad epochų skaičiui augant, auga ir tikslumas, kuo mažesnis mokymosi greitis, tuo didesnis epochų skaičius reikalingas, norint gauti didesnį tikslumą. Pasiekus maksimalų modelio tikslumą (tai priklauso nuo architektūros bei tiriamų duomenų aibės kokybės), didesnis epochų skaičius didelės įtakos nesudaro.



Pav. 9. Irisų aibės tikslumo priklausomybė nuo epochų skaičiaus

Rezultatų nuo mokymosi greičio priklausomybė

Rezultatų priklausomybę nuo mokymosi greičio galima pamatyti jau apžvelgtuose paveikslėliuose 7, 8, 9. Turint mažesnį mokymosi greitį reikalingas didesnis epochų skaičius norint geriems rezultatams gauti. Kuo mažesnis mokymosi greitis, tuo mažesni klaidos artėjimo prie 0 svyravimai žr. pav. 7, 8, tačiau norint gauti aukštą tikslumą kai mokymosi greitis mažas, reikia didesnio epochų skaičiaus.

Neurono mokymo rezultatai

Abiems duomenų aibėms geriausi rezultatai gauti, kai mokymosi greitis lygus 0,01, o epochų skaičius lygus 10000, žr. lentelė 1, 2. Radau ir kitas hiperparametrų reikšmes, kai gaunami toki patys rezultatai, tačiau jie nesutampa duomenų aibėms.

Irisų duomenų aibės rezultatai

Gauti svoriai: -3,3310031; -4,50019187; 6,56997263; 11,83746343;

Gautas poslinkis: -17,940260908425152

Epochų skaičius: 10 000

Paklaida paskutinėje epochoje mokymo duomenims: -0,0768197243374164

Klasifikavimo tikslumas paskutinėje epochoje mokymo duomenimis: 0,975

Paklaida testavimo duomenims: 0,0039917594305293675

Klasifikavimo tikslumas testavimo duomenims: 1

Neurono spėjimai ir trokštama klasė:

Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0

Krūties vėžio duomenų aibės rezultatai

Gauti svoriai: 2,17499445; -0,28104685; 2,03283526; 0,96965931; 0,33617457; 0,63658297;

1,68297917; 0,51702183; 2,05167079

Gautas poslinkis: -31,789483681937682

Epochų skaičius: 10 000

Paklaida paskutinėje epochoje mokymo duomenims: -0,5782778344286715

Klasifikavimo tikslumas paskutinėje epochoje mokymo duomenimis: 0,9487179487179487

Paklaida testavimo duomenims: 0,003190415419079878

Klasifikavimo tikslumas testavimo duomenims: 0,9927007299270073

Neurono spėjimai ir trokštama klasė:

Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0

Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0

Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0

Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 0, Tikroji klasė: 0.0
Spėjimas: 1, Tikroji klasė: 1.0
Spėjimas: 0, Tikroji klasė: 0.0

Atnaujinti neurono mokymo rezultatai

Šiame skyriuje aprašomi rezultatai gauti naudojant sigmoidinio neurono mokymo taisyklę, ja remiantis svoriai ir poslinkis skaičiuojamas pagal 10 paveikslėlyje pateiktą formulę.

$$w_k := w_k - \eta(y_i - t_i)y_i(1 - y_i)(x_{ik})$$

Pav. 10. Sigmoidinio neurono mokymo formulė

Neuronui mokyti mokymo greitį parinkau kaip 0,01, o epochų skaičių 10000, kadangi su šiais hiperparametrais buvo gauti geriausi rezultatai abejoms aibėms.

Irisų duomenų aibės rezultatai

Gauti svoriai: -4,31823526; -4,1915452; 6,21356776; 7,75207854;

Gautas poslinkis: -4.500023017443272

Epochų skaičius: 10 000

Paklaida paskutinėje epochoje mokymo duomenims: -0,10258469764803829

Klasifikavimo tikslumas paskutinėje epochoje mokymo duomenimis: 0,975

Paklaida testavimo duomenims: 0,029626127714474988

Klasifikavimo tikslumas testavimo duomenims: 0,95

Krūties vėžio duomenų aibės rezultatai

Gauti svoriai: 0,73068929; 0,53908808; -0,04663171; 0,832495; 0,05335846; 0,8882289; 0,21759953; 0,50637487; 0,75293031;

Gautas poslinkis: -13,440112403613202

Epochų skaičius: 10 000

Paklaida paskutinėje epochoje mokymo duomenims: -6,542965062739941

Klasifikavimo tikslumas paskutinėje epochoje mokymo duomenimis: 0,9835164835164835

Paklaida testavimo duomenims: 0,018435209810903585

Klasifikavimo tikslumas testavimo duomenims: 0,948905109489051

Išvados

Neurono mokymo metu pastebėtos tendencijos, kuriomis remdamasis formuluoju išvadas. Pirmasis pastebėjimas yra duomenų aibių kokybės svarbumas ir teisingas duomenų paruošimas, kadangi visos įvestys turi būti užpildytos arba su duomenų nebuvimu reikia susitvarkyti metodais, kaip duomenų trynimas ar kokios nors reikšmės užpildymas (vidurkiu ar kita). Taip pat svarbu suvienodinti duomenų skales bei patikrinti, kad visi tos įvesties duomenys būtų vienoduose rėžiuose. Modeliui apmokant labai svarbu išbandyti ir pasirinkti kuo daugiau skirtingų hiperparametrų kombinacijų, kadangi galutiniai rezultatai stipriai priklauso tiek nuo mokymosi žingsnio, tiek nuo epochų skaičiaus. Pastebėjau, kad didesnis epochų skaičius dažnai reiškia geresnį modelio tikslumą, tačiau tai kainuoja daugiau resursų, o nuo tam tikro taško epochų kiekis pradeda rezultatų negerinti arba gerinti juos labai nežymiai. Mažesnis mokymosi greitis padeda sumažinti klaidos variaciją, kas dažnai lemia geresnius galutinius rezultatus, tačiau mažinant mokymosi greitį per daug, epochų skaičius gali būti per mažas norint gauti optimalisu rezultatus. Klaidos vertė mokymosi metu padeda įvertinti neurono mokymąsi, tačiau tai nevisada atspindi neurono kalsifikavimo tikslumą. Neurono, su blogesnėmis klaidos vertėmis, rezultatai gali būti geresni nei neurono su geresnėmis klaidos vertėmis.