Vilniaus Universitetas

Perceptrono mokymas

Dirbtinio intelekto pagrindai

2 užduotis

Darbą atliko:

Dovydas Martinkus

Duomenų Mokslas 4 kursas 2 gr.

Vilnius, 2022

**Turinys**

[1 Tikslas ir uždaviniai 3](#_Toc114422146)

[2 Duomenys 4](#_Toc114422147)

[3 Užduoties ataskaita 5](#_Toc114422148)

[3.1 Programinis kodas 5](#_Toc114422149)

[3.2 Tyrimo rezultatai 8](#_Toc114422150)

[4 Išvados 17](#_Toc114422151)

[Priedas 18](#_Toc114422152)

# Tikslas ir uždaviniai

Tikslas: apmokyti vieną dirbtinį neuroną spręsti nesudėtingą dviejų klasių uždavinį, atlikti rezultatų tyrimą su dviem duomenų aibėmis.

Uždaviniai:

Naudojamų duomenų aibių paruošimas.

Duomenų aibių padalijimas į mokymo ir testavimo aibes, dirbtinio neurono mokymo ir testavimo įgyvendinimas naudojant pasirinktą programavimo kalbą.

Klasifikavimo rezultatų priklausomybės nuo epochų skaičiaus, mokymo greičio parametro reikšmės, naudojamos aktyvacijos funkcijos tyrimas.

Visų tyrimų gautų rezultatų pateikimas lentelėse arba grafikuose.

# Duomenys

Užduotyje naudoti 2 duomenų rinkiniai:

Irisų duomenų aibę sudaro 150 stebėjimų, iš kurių kiekvienas turi po 4 skaitinius požymius ir klasę, kuriai priklauso. Šiuose duomenyse yra trys klasės Setosa, Versicolor ir Virginica. Analizei naudotos tik dvi: Versicolor ir Virginica. Prieiga per internetą: <https://archive.ics.uci.edu/ml/datasets/iris>.

Krūties vėžio duomenų aibę sudaro 6873 stebėjimai, turintys po 10 skaitinių požymių ir klasės kintamąjį, kurį sudaro dvi klasės: 2 – nepiktybinis navikas , 4 – piktybinis navikas. Prieiga per internetą: <https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic).>

Abi duomenų aibes pasirinkta dalinti į mokymo ir testavimo aibes naudojant santykį 80-20.

Kai kurie ataskaitoje naudojami terminai:

Epocha – vienas algoritmo pilnas perėjimas pro visus turimus mokymo duomenis.

Iteracija – tai mokymo proceso dalis, kai į perceptroną (arba į dirbtinį neuroninį tinklą) pateikiamas vienas mokymo aibės duomuo.

# Užduoties ataskaita

## Programinis kodas

Užduotis atlikta naudojant programavimo kalbą „Python“. Žemiau pateiktas programinis kodas su tarpiniais paaiškinimais:

**import** pandas **as** pd

**import** numpy **as** np

**import** seaborn **as** sns

**import** math

**from** sklearn.model\_selection **import** train\_test\_split

**from** matplotlib **import** pyplot **as** plt

*# pirmojo duomenų rinkinio nuskaitymas, pertvarkymas ir padalijimas į mokymo ir testavimo aibes*

x1 **=** pd**.**read\_csv('iris.data', sep**=**",", header**=None**)**.**values

x1 **=** x1[x1[:,4] **!=** "Iris-setosa"]

x1 **=** np**.**column\_stack([np**.**ones(x1**.**shape[0]),x1])

x1[:,**-**1] **=** np**.**where(x1[:,**-**1]**==**'Iris-virginica',0,1)

X1\_train, X1\_test, y1\_train, y1\_test **=** train\_test\_split(x1[:,0:**-**1],x1[:,**-**1],

train\_size**=**0.8,random\_state**=**123,stratify**=**x1[:,**-**1])

*# antrojo duomenų rinkinio nuskaitymas, pertvarkymas ir padalijimas į mokymo ir testavimo aibes*

x2 **=** pd**.**read\_csv('breast-cancer-wisconsin.data', sep**=**",", header**=None**)**.**values

x2 **=** x2[x2[:,6] **!=** "?"]

x2[:,**-**1] **=** np**.**where(x2[:,**-**1]**==**2,0,1)

x2 **=** x2[:,1:]

x2 **=** np**.**column\_stack([np**.**ones(x2**.**shape[0]),x2])

x2 **=** x2**.**astype(float)

X2\_train, X2\_test, y2\_train, y2\_test **=** train\_test\_split(x2[:,0:**-**1],x2[:,**-**1],

train\_size**=**0.8,random\_state**=**123,stratify**=**x2[:,**-**1])

Atlikus reikalingus pertvarkymus, pirmojo duomenų rinkinio (irisų) mokymo aibėje turima 80 stebėjimų, testavimo aibėje – 20 stebėjimų. Antrojo duomenų rinkinio (krūties vėžio) mokymo ir testavimo aibėse turimi atitinkamai 546 ir 137 stebėjimai.

Užduotyje naudotos slenkstinė ir sigmoidinė aktyvacijos funkcijos:

**def** sigmoid(row,weights):

*# Sigmoidine aktyvacijos f-ja*

a **=** np**.**dot(row, weights)

**return** 1**/**(1**+**math**.**exp(**-**a))

**def** threshold(row,weights):

*# Slenkstinė aktyvacijos f-ja*

a **=** np**.**dot(row, weights)

**return** int(a **>=** 0)

*# abi aktyvacijos funkcijos skirtos paduoti kaip tolimesnių funkcijų argumentas*

Svoriai atnaujinti naudojantis formule:

**def** update\_weights(weights, row, y\_true, y\_pred, lrate):

*# Svorius atnaujinanti funkcija*

updated\_weights **=** []

**for** i **in** range(len(weights)):

updated\_weights**.**append(weights[i] **+** lrate**\***(y\_true **-** y\_pred)**\***row[i])

**return** updated\_weights

**from** matplotlib.ticker **import** MaxNLocator

**def** lineplot(y, xlabel,ylabel, title,c**=**"b"):

*# funkcija, skirta nubraižyti linijinį grafiką*

fig, ax **=** plt**.**subplots(figsize**=**(8, 4))

ax**.**plot(range(1,len(y)**+**1),y, alpha **=** 0.7,c**=**c)

ax**.**set\_xlabel(xlabel)

ax**.**set\_ylabel(ylabel)

ax**.**set\_title(title)

ax**.**xaxis**.**set\_major\_locator(MaxNLocator(integer**=True**))

Kaip pradines parametrų reikšmes pasirinkta naudoti 20 epochų ir mokymosi greitį (angl. learning rate) lygų 0,5.

Klasifikavimo tikslumas naudojant mokymo ir testavimo aibes skaičiuotas padalijant teisingai priskirtų klasių skaičių iš bendro stebėjimų atitinkamoje aibėje skaičiaus.

**def** train\_perceptron(X, y, activation, lrate, epochs, plot **=** **True**):

weights **=** np**.**full(X**.**shape[1], 1) *# pradiniai svoriai*

errors **=** [] *# paklaidų sąrašas*

accuracies **=** [] *# klasifikavimo tikslumų sąrašas*

**for** e **in** range(epochs):

error **=** 0

accuracy **=** 0

**for** i **in** range(len(X)):

y\_true **=** y[i]

row **=** X[i]

y\_pred **=** activation(row, weights)

weights **=** update\_weights(weights, row, y\_true, y\_pred , lrate)

error **+=** (y\_true **-** y\_pred)**\*\***2

accuracy **+=** round(y\_pred,0) **==** y\_true

errors**.**append(round(error,2)) *# sąrašas papildomas paklaida, gauta po kiekvienos epochos*

accuracies**.**append(round(accuracy**/**len(X),2))

**if** plot:

names **=** {"sigmoid": "sigmoidinę", "threshold": "slenkstinę"}

lineplot(errors, "Epocha", "Paklaida", "Epochų paklaida, naudojant " **+**

names[activation**.**\_\_name\_\_] **+**

" aktyvacijos f-ją","b")

lineplot(accuracies, "Epocha", "Paklaida", "Epochų tikslumas mokymo duomenims, naudojant " **+**

names[activation**.**\_\_name\_\_] **+**

" aktyvacijos f-ją","orange")

**return** errors, accuracies, weights

**def** test\_accuracy(X\_test, y\_test, weights, activation):

*# Apskaičiuoja tikslumą testavimo duomenims, naudojant pasirinktą aktyvacijos funkciją*

correct **=** []

error **=** 0

**for** i **in** range(len(X\_test)):

y\_pred **=** round(activation(X\_test[i],weights),0)

correct**.**append(y\_pred **==** y\_test[i])

error **+=** (y\_test[i] **-** y\_pred)**\*\***2

**return** round(np**.**mean(correct),2), round(error,2)

## Tyrimo rezultatai

Pirmiausia naudotas pirmas (irisų) duomenų rinkinys.

sns**.**set\_context("notebook")

*# kviečiama perceptrono mokymo f-ja su X1 duomenimis, slenkstine aktyvacijos f-ja*

*# 0.5 mokymo greičiu ir naudojant 20 epochų*

errors, \_, weights **=** train\_perceptron(X1\_train, y1\_train, threshold, 0.5, 20)

*# perceptrono mokymas, naudojantis mokymo duomenimis ir slenstine aktyvacijos funkcija*

print("Galutiniai svoriai: ", [round(i,2) **for** i **in** weights],

"\nPaklaida: ", errors[**-**1],

"\nTikslumas testavimo duomenims: " , test\_accuracy(X1\_test, y1\_test, weights, threshold)[0],

"\nPaklaida testavimo duomenims: " , test\_accuracy(X1\_test, y1\_test, weights, threshold)[1])

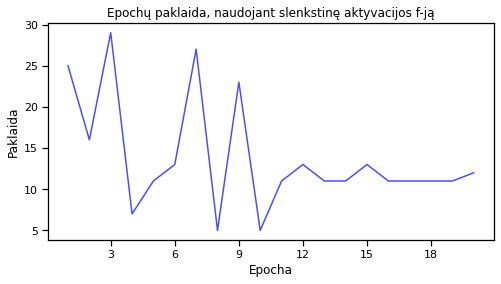
Galutiniai svoriai: [21.0, 21.6, 38.1, -49.2, -49.7]

Paklaida: 12

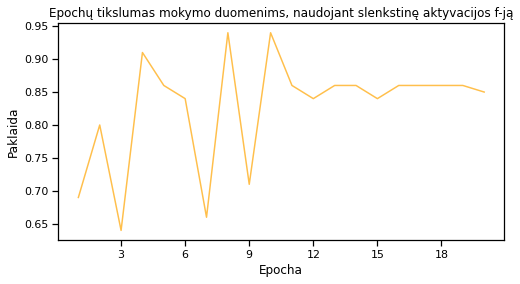
Tikslumas testavimo duomenims: 0.55

Paklaida testavimo duomenims: 9

Paklaidos ir tikslumo priklausomybė nuo epochos pavaizduota grafiškai naudojant linijinę diagramą (atitinkamai 1 ir 2 pav.). Pastebimas paklaidos mažėjimas ir tikslumo didėjimas didėjant atliktų iteracijų skaičiui, tačiau su 0,5 mokymosi greičiu jis yra stipriai nepastovus: neretai sekančios epochos metu gauta paklaida daug didesnė už praėjusios. Tikėtina, kad ši priežastis atsirado dėl per didelio parinkto mokymosi greičio.



1 pav. Paklaida pagal epochą, naudojant slenkstinę aktyvacijos funkciją (pirmas duomenų rinkinys)



2 pav. Tikslumas pagal epochą, naudojant slenkstinę aktyvacijos funkciją (pirmas duomenų rinkinys)

serrors, \_, weights **=** train\_perceptron(X1\_train, y1\_train, sigmoid, 0.5, 20)

*# sigmoidinė aktyvacijos fuhnkcija*

print("Galutiniai svoriai: ", [round(i,2) **for** i **in** weights],

"\nPaklaida: ", errors[**-**1],

"\nTikslumas testavimo duomenims: " , test\_accuracy(X1\_test, y1\_test, weights, sigmoid)[0],

"\nPaklaida testavimo duomenims: " , test\_accuracy(X1\_test, y1\_test, weights, sigmoid)[1])

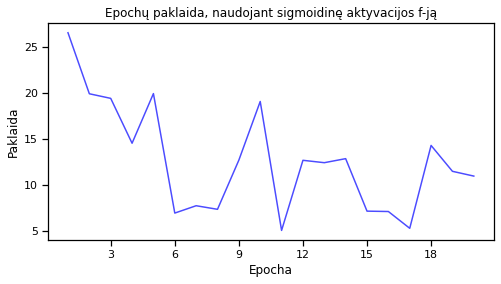
Galutiniai svoriai: [21.72, 20.92, 46.34, -44.19, -43.66]

Paklaida: 10.95

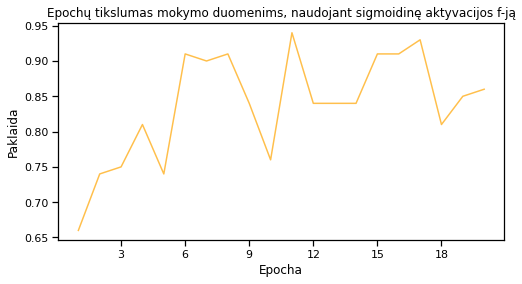
Tikslumas testavimo duomenims: 0.9

Paklaida testavimo duomenims: 2.0

Tokie pat grafikai nubraižyti vietoje slenkstinės aktyvacijos funkcijos naudojant sigmoidinę (atitinkamai 3 ir 4 pav.). Tiek paklaidos, tiek tikslumo tendencijos išlieka labai panašios.



3 pav. Paklaida pagal epochą, naudojant sigmoidinę aktyvacijos funkciją (pirmas duomenų rinkinys)



4 pav. Tikslumas pagal epochą, naudojant sigmoidinę aktyvacijos funkciją (pirmas duomenų rinkinys)

Analogiška procedūra kartota ir naudojant antrąjį duomenų rinkinį:

errors, \_, weights **=** train\_perceptron(X2\_train, y2\_train, threshold, 0.5, 20)

print("Galutiniai svoriai: ", [round(i,2) **for** i **in** weights],

"\nPaklaida: ", errors[**-**1],

"\nTikslumas testavimo duomenims: " , test\_accuracy(X2\_test, y2\_test, weights, threshold)[0],

"\nPaklaida testavimo duomenims: " , test\_accuracy(X2\_test, y2\_test, weights, threshold)[1])

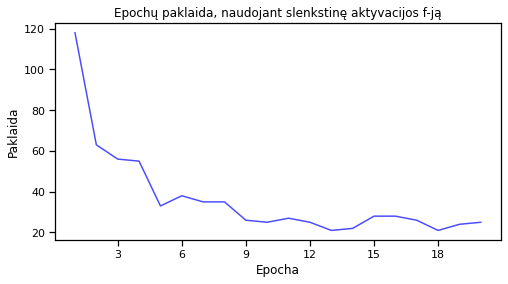
Galutiniai svoriai: [-202.0, 5.0, 12.0, 9.0, 12.0, 8.0, 7.0, 5.0, 5.0, 17.0]

Paklaida: 25.0

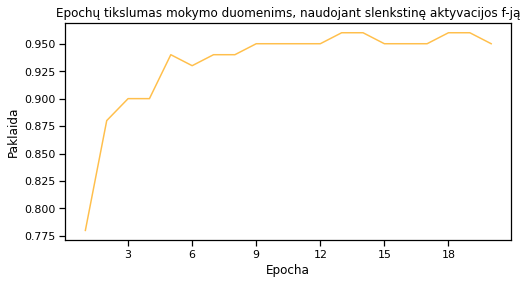
Tikslumas testavimo duomenims: 0.98

Paklaida testavimo duomenims: 3.0

Naudojant slenkstinę aktyvacijos funkciją tikslumo didėjimas daug pastovesnis didėjant epochoms, lyginant su pirmuoju duomenų rinkiniu (6 pav.). Paklaidos tendencijas tarp dviejų turimų duomenų rinkinių taip palyginti sudėtingiau, nes šis dydis nėra normuotas pagal turimą stebėjimų skaičių (antrojo duomenų rinkinio mokymo aibėje yra daugiau stebėjimų negu pirmojo). Taip pat galima matyti, kad paklaida stipriai mažėja iki 5-6 epochos, bet tolimesnėse epochose beveik išvis nustoja keistis.



5 pav. Paklaida pagal epochą, naudojant slenkstinę aktyvacijos funkciją (antras duomenų rinkinys)



6 pav. Tikslumas pagal epochą, naudojant slenkstinę aktyvacijos funkciją (antras duomenų rinkinys)

Lyginant slenkstinę ir sigmoidinę aktyvacijos funkcijas vėl gauname labai panašius rezultatus (8 pav.).

errors, \_, weights **=** train\_perceptron(X2\_train, y2\_train, sigmoid, 0.5, 20)

print("Galutiniai svoriai: ", [round(i,2) **for** i **in** weights],

"\nPaklaida: ", errors[**-**1],

"\nTikslumas testavimo duomenims: " , test\_accuracy(X2\_test, y2\_test, weights, sigmoid)[0],

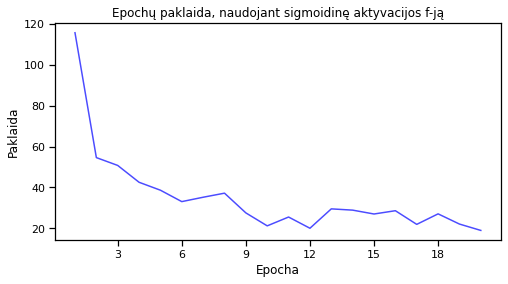
"\nPaklaida testavimo duomenims: " , test\_accuracy(X2\_test, y2\_test, weights, sigmoid)[1])

Galutiniai svoriai: [-200.57, 3.72, 10.81, 15.86, 15.1, 4.6, 10.1, 3.15, -0.76, 18.6]

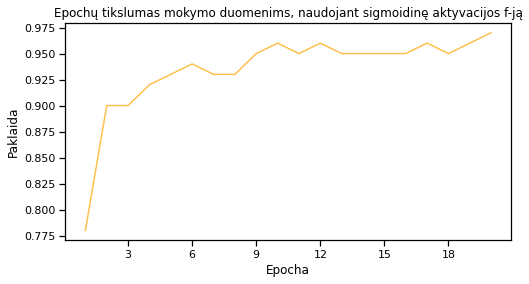
Paklaida: 19.0

Tikslumas testavimo duomenims: 0.96

Paklaida testavimo duomenims: 5.0



7 pav. Paklaida pagal epochą, naudojant sigmoidinę aktyvacijos funkciją (antras duomenų rinkinys)



8 pav. Tikslumas pagal epochą, naudojant sigmoidinę aktyvacijos funkciją (antras duomenų rinkinys)

*# grafikas, parodantys klasifikavimo tikslumo priklausomybę nuo mokymo epochos ir mokymosi greičio parametro reikšmių*

**def** plot\_heatmap(data, ax, title, left**=True**):

cmap **=** sns**.**cm**.**mako\_r

plot**=**sns**.**heatmap(data, vmax**=**1,vmin**=**0.7,center**=**0.85,cbar**=False**,cmap**=**cmap,

linewidth**=**1,annot**=True**, ax**=**ax)

plot**.**set\_title(title)

plot**.**set\_xlabel("Mokymosi greitis")

**if** left:

plot**.**set\_ylabel("Epocha")

plot**.**set\_xticklabels(lrates)

plot**.**set\_yticklabels(range(1,results**.**shape[0]**+**1))

plot**.**tick\_params(axis**=**'x', rotation**=**0)

plot**.**tick\_params(axis**=**'y', rotation**=**360)

lrates **=** np**.**round(np**.**arange(0.1, 1, 0.1),1)

results1 **=** np**.**zeros((20,len(lrates)))

results2 **=** np**.**zeros((20,len(lrates)))

**for** i,j **in** enumerate(lrates):

\_, accuracies, \_\_ **=** train\_perceptron(X1\_train, y1\_train, threshold, j, 20, plot**=False**)

results1[:,i] **=** accuracies

\_, accuracies, \_\_ **=** train\_perceptron(X1\_train, y1\_train, sigmoid, j, 20, plot**=False**)

results2[:,i] **=** accuracies

sns**.**set\_context("talk")

fig, ax **=** plt**.**subplots(1,2,figsize**=**(22, 9))

fig**.**suptitle("Tikslumas pagal mokymosi greitį ir aktyvacijos f-ją " **+** "(iris)")

plot\_heatmap(results1,ax[0], "Slenkstinė")

plot\_heatmap(results2,ax[1], "Sigmoidinė", left**=False**)

Rezultatų priklausomybė nuo epochos, mokymosi greičio ir naudojamos aktyvacijos funkcijos pavaizduota grafiškai (9 pav.). Kaip galime matyti, didžiausias tikslumas gaunamas, pavyzdžiui, naudojant slenkstinę aktyvacijos funkciją, 0,4 mokymosi greitį ir keturias epochas. Nepaisant to, apskritai matoma tendencija gauti geresnius rezultatus naudojant sigmoidinę aktyvacijos funkciją ir didesnį epochų kiekį.

Paveikslėlis, kuriame yra stalas

Automatiškai sugeneruotas aprašymas

9 pav. Mokymo aibės tikslumas pagal mokymosi greitį, epochą ir aktyvacijos f-ją (pirma duomenų aibė)

Nustačius geriausią variantą, išvedami galutiniai svoriai, tikslumas ir paklaida naudojant tiek mokymo duomenys, tiek testavimo duomenis:

errors, accuracy, weights **=** train\_perceptron(X1\_train, y1\_train, threshold, 0.4, 4, plot**=False**)

print("Galutiniai svoriai: ", [round(i,2) **for** i **in** weights],

"\nTikslumas: ", accuracy[**-**1],

"\nPaklaida: ", errors[**-**1],

"\nTikslumas testavimo duomenims: " , test\_accuracy(X1\_test, y1\_test, weights, threshold)[0],

"\nPaklaida testavimo duomenims: " , test\_accuracy(X1\_test, y1\_test, weights, threshold)[1])

Galutiniai svoriai: [3.0, 5.4, 4.44, -7.6, -7.24]

Tikslumas: 0.96

Paklaida: 3

Tikslumas testavimo duomenims: 1.0

Paklaida testavimo duomenims: 0

Taip pat pateikiama lentelė kokias klases kiekvienam testavimo aibės įrašui nustatė neuronas ir kokia klasė turėjo būti (1 priedas).

Tokia pati procedūra kartota ir naudojant antrąjį duomenų rinkinį:

lrates **=** np**.**round(np**.**arange(0.1, 1, 0.1),1)

results1 **=** np**.**zeros((20,len(lrates)))

results2 **=** np**.**zeros((20,len(lrates)))

**for** i,j **in** enumerate(lrates):

\_, accuracies, \_\_ **=** train\_perceptron(X2\_train, y2\_train, threshold, j, 20, plot**=False**)

results1[:,i] **=** accuracies

\_, accuracies, \_\_ **=** train\_perceptron(X2\_train, y2\_train, sigmoid, j, 20, plot**=False**)

results2[:,i] **=** accuracies

sns**.**set\_context("talk")

fig, ax **=** plt**.**subplots(1,2,figsize**=**(22, 9))

plot\_heatmap(results1,ax[0], "Slenkstinė")

plot\_heatmap(results2,ax[1], "Sigmoidinė", left**=False**)

Tokio pat tipo grafikas nubraižytas ir naudojant antrą duomenų rinkinį (10 pav.). Geriausi rezultatai gaunami naudojant 0,7 mokymosi greitį su 11 epochų ir sigmoidine aktyvacijos funkcija. Matoma tendencija mokymo aibės tikslumui didėti didėjant epochų skaičiui, tačiau šiam duomenų rinkinius vos po kelių epochų pasiekiamas labai didelis tikslumas ir rezultatai nustoja gerėti. Taip pat galime pastebėti, kad naudojant antrąjį duomenų rinkinį įprastai gaunamas geresnis klasifikavimo tikslumas, lyginant su rezultatais pirmajam duomenų rinkiniui.

Paveikslėlis, kuriame yra stalas

Automatiškai sugeneruotas aprašymas

10 pav. Mokymo aibės tikslumas pagal mokymosi greitį, epochą ir aktyvacijos f-ją (antra duomenų aibė)

errors, accuracy, weights **=** train\_perceptron(X2\_train, y2\_train, sigmoid, 0.7, 11, plot**=False**)

print("Galutiniai svoriai: ", [round(i,2) **for** i **in** weights],

"\nTikslumas: ", accuracy[**-**1],

"\nPaklaida: ", errors[**-**1],

"\nTikslumas testavimo duomenims: " , test\_accuracy(X2\_test, y2\_test, weights, sigmoid)[0],

"\nPaklaida testavimo duomenims: " , test\_accuracy(X2\_test, y2\_test, weights, sigmoid)[1])

Galutiniai svoriai: [-111.46, -0.98, 10.91, 10.45, 11.23, -0.08, 8.59, 0.08, 1.2, 4.07]

Tikslumas: 0.97

Paklaida: 18.47

Tikslumas testavimo duomenims: 0.96

Paklaida testavimo duomenims: 5.0

Antrajam duomenų rinkiniui taip pat pateiktas tikrų ir perceptrono pateiktų reikšmių palyginimas naudojant testavimo aibę (2 priedas).

# Išvados

Remiantis tyrimo metu gautais rezultatais galima teigti, kad daugeliu atveju didėjant epochoms gautas didesnis klasifikavimo tikslumas (mažesnė paklaida). Nepaisant to, tyrimo metu rasta, kad perceptronas per gana mažą epochų skaičių (didžiajai daliai atvejų nedaugiau nei per 6), pasiekia tikslumą, kuris (beveik) nebegerėja tolimesnių epochų metu.

Taip pat pastebėta problema galinti kilti jeigu parenkamas per didelis mokymosi greitis: svoriai atnaujinami per daug greitai ir tolimesnėje epochoje gauti rezultatai yra prastesni negu epochose prieš tai.

Rasta ir naudojamo duomenų rinkinio įtaka. Su beveik visomis epochų skaičiaus ir mokymosi greičio parametrų reikšmes naudojant antrąjį duomenų rinkinį gautas didesnis klasifikavimo tikslumas mokymo aibėje negu su pirmuoju duomenų rinkiniu. Galima priežastis, kodėl gaunamas šis rezultatas yra tai, kad antrojo duomenų rinkinio mokymo aibėje turima daugiau stebėjimų (vienos epochos metu svoriai atnaujinami daugiau kartų).

# Priedas

|  |  |
| --- | --- |
| Tikra reikšmė | Prognozuota reikšmė |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |

1 priedas Tikros ir prognozuotos klasių reikšmės testavimo duomenims (pirma duomenų aibė)

|  |  |
| --- | --- |
| Tikra reikšmė | Prognozuota reikšmė |
| 0 | 0 |
| 0 | 1 |
| 1 | 1 |
| 0 | 1 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 1 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 1 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 1 | 1 |
| 0 | 0 |
| 0 | 0 |

2 priedas Tikros ir prognozuotos klasių reikšmės testavimo duomenims (antra duomenų aibė)