VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ BAKALAURO STUDIJŲ PROGRAMA

Vieno neurono mokymas sprendžiant klasifikavimo uždavinį

Skaitmeninis intelektas ir sprendimų priėmimas

Užduoties ataskaita

Atliko: 4 kurso 4 grupės studentas
Vilius Puškunalis

Vilnius – 2023

TURINYS

1. Įvadas

Užduoties tikslas

Apmokyti vieną neuroną spręsti dviejų klasių uždavinį ir atlikti tyrimą su dviem duomenų aibėm.

Užduoties variantas

Studento numeris – 2016025.

Paskutinio skaitmens 5 dalybos iš 3 liekana – 2.

2-as variantas – neurono mokymui naudoti stochastinį gradientinį nusileidimą ir ADALINE mokymo taisyklę.

Uždaviniai

- Parsisiųsti ir paruošti irisų bei krūties vėžio navikų duomenų aibes.
- Sukurti programą, kuri įgyvendintų vieno neurono mokymo ir testavimo procesą, sprendžiant klasifikavimo užduotį.
- Irisų ir krūties vėžio navikų duomenų aibėms ištirti:
  - Paklaidos reikšmės priklausomybę nuo epochų skaičiaus.
  - Klasifikavimo tikslumo priklausomybė nuo epochų skaičiaus.
  - Rezultatų priklausymą nuo skirtingų mokymosi greičio reikšmių.

2. Duomenys

Dirbtinio neurono mokymui ir testavimui buvo naudojamos irisų [Fis36] bei krūties vėžio [Wol92] duomenų aibės.

Irisų duomenų aibėje yra 3 rūšys (Setosa, Versicolor, Virginica) po 50 duomenų įrašų, tačiau darbe naudojamos Versicolor ir Virginica rūšys. Kiekvienas įrašas turi po 4 požymius. Taigi, apdorotoje duomenų aibėje yra 100 įrašų.

Krūties vėžio duomenų aibėje yra 699 duomenų įrašai. Kiekvienas įrašas turi po 9 požymius. Atmetus eilutes, kur kai kurie duomenys trūkstami, lieka 683 įrašai. Duomenys klasifikuojami į nepiktybinius ir piktybinius navikus.

Duomenys buvo padalinti į mokymo ir testavimo aibes santykiu 70:30.

3. Programos kodas su komentarais

Dirbtinis neuronas įgyvendintas Python programavimo kalba. Gali būti modifikuojami šie hiperparametrai: mokymo greitis, epochų skaičius, santykis tarp mokymo ir testavimo aibių.

```
import pandas as pd
import numpy as np

# Nuskaito ir paruosia irisu duomenu aibe
def fetch_and_prepare_iris_data():
    # Skaitome CSV formato faila
    df = pd.read_csv('iris/iris.data', header=None)

    # Filtruojame duomenis
    filtered_indices = df[df.iloc[:, -1].isin(['Iris-versicolor', 'Iris
        -virginica'])].index
    df = df.loc[filtered_indices]

    # Pakeiciame klasiu pavadinimus i numerius
    df.iloc[:, -1] = df.iloc[:, -1].map({'Iris-versicolor': 0, 'Iris-
        virginica': 1})

    return df

# Nuskaito ir paruosia kruties vezio duomenu aibe
def fetch_and_prepare_breast_cancer_data():
    # Skaitome CSV formato faila
    df = pd.read_csv('breast-cancer/breast-cancer-wisconsin.data',
        header=None)

    # Pasaliname pirma stulpeli
    df = df.iloc[:, 1:]

    # Ismetame eilutes su klaustukais
    indices_to_keep = ~df.isin(['?']).any(axis=1)
    df = df[indices_to_keep]

    # Konvertuojame visus stulpelius i skaiciu formata (problema del
        buvusiu klaustuku)
    for col in df.columns:
        df[col] = pd.to_numeric(df[col], errors='coerce')

    # Pakeiciame klasiu numerius
    df = df.loc[indices_to_keep].copy()
```

```
    df.iloc[:, -1] = df.iloc[:, -1].map({2: 0, 4: 1})

    return df

# Skaido duomenis i mokymo ir testavimo rinkinius
def split_data(train_test_split, df):
    # Maisome duomenis
    df = df.sample(frac=1, random_state=0)

    # Apskaiciuojame indeksa, kuriuo padalinsime duomenis
    split_index = int(train_test_split * len(df))

    # Daliname duomenis
    train_data = df[:split_index]
    test_data = df[split_index:]

    return train_data, test_data

# Neurono mokymas naudojant ADALINE taisykle ir stochastini gradientini
    nusileidima
def adaline_SGD(train_data, epochs, learning_rate):
    features = train_data.iloc[:, :-1].values
    classes = train_data.iloc[:, -1].values

    # Inicializuojame svorius ir bias
    weights = np.zeros(features.shape[1])
    bias = 0

    # Mokymo ciklas
    error_per_epoch = []
    accuracy_per_epoch = []
    for epoch in range(epochs):
        total_error = 0
        for xi, t in zip(features, classes):
            # Aktyvacijos funkcija yra tiesine
            y = np.dot(xi, weights) + bias
            weights += learning_rate * (t - y) * xi
            bias += learning_rate * (t - y)

            error = (t - y) ** 2
            total_error += error

        error_per_epoch.append(total_error)
```

```
78          accuracy_per_epoch.append(accuracy(train_data, weights, bias,
                False))
79
80      return weights, bias, error_per_epoch, accuracy_per_epoch
81
82  # Skaiciuoja klasifikavimo tiksluma
83  def accuracy(data, weights, bias, should_print):
84      features = data.iloc[:, :-1].values
85      classes = data.iloc[:, -1].values
86
87      y = np.dot(features, weights) + bias
88
89      # Slenkstine funkcija
90      y = np.where(y > 0.5, 1, 0)
91
92      if should_print:
93          for real_class, predicted_class in zip(classes, y):
94              print(f"Real class: {real_class}, predicted class: {
                    predicted_class}, prediction correct: {real_class ==
                    predicted_class}")
95
96      return np.mean(y == classes)
97
98  # Skaiciuoja vidutine kvadratine paklaida
99  def mean_squared_error(data, weights, bias):
100     features = data.iloc[:, :-1].values
101     classes = data.iloc[:, -1].values
102
103     # Aktyvacijos funkcija yra tiesine
104     y = np.dot(features, weights) + bias
105
106     # MSE skaiciavimas
107     mse = np.mean((classes - y)**2)
108
109     return mse
110
111 if __name__ == "__main__":
112     train_test_split = 0.7 # Dalinimas i mokymo/testavimo aibes 70:30
113     epochs = 100  # Epochu skaicius
114     learning_rate = 0.0001  # Mokymosi greitis
115
116     # Pasirinkti irisu arba krutu vezio duomenu aibes
117     df = fetch_and_prepare_iris_data()
```

```
    #df = fetch_and_prepare_breast_cancer_data()

    # Duomenu dalinimas i mokymo ir testavimo aibes
    train_data, test_data = split_data(train_test_split, df)

    # Neurono mokymas
    weights, bias, error_per_epoch, accuracy_per_epoch = adaline_SGD(
        train_data, epochs, learning_rate)

    # Tikslumo skaiciavimas
    test_accuracy = accuracy(test_data, weights, bias, True)
    test_mse = mean_squared_error(test_data, weights, bias)

    print(f"Gauti svoriai: {np.round([bias, *weights], decimals=4)}")
    print(f"Gautos paklaidos po kiekvienos epochos mokymo duomenims: {
        np.round(error_per_epoch, decimals=4)}")
    print(f"Gauta paklaida testavimo duomenims: {test_mse:.4f}")
    print(f"Gautas klasifikavimo tikslumas po kiekvienos epochos mokymo
         duomenims: {np.round(accuracy_per_epoch, decimals=4)}")
    print(f"Gautas klasifikavimo tikslumas testavimo duomenims: {
        test_accuracy:.4f}")
```

4. Neurono mokymo taisyklės

4.1. Pradinių svorių reikšmių parinkimas

Pradiniai svoriai bei poslinkis visi nustatyti su reikšme lygia 0.

4.2. Stochastinis gradientinis nusileidimas

Stochastinis gradientinis nusileidimas (angl. Stochastic Gradient Descent; SGD) yra optimizavimo algoritmas, skirtas rasti funkcijos lokalų minimumą arba maksimumą. Stochastinis gradientinis nusileidimas yra vienas iš gradientinio nusileidimo algoritmo variantų.

Stochastinio gradientinio nusileidimo atveju viena epocha atitinka m iteracijų, čia m yra mokymo duomenų kiekis [Kur21].

4.3. ADALINE taisyklė

ADALINE (angl. Adaptive Linear Neuron) yra dirbtinio neurono mokymo taisyklė, sukurta 1959-aisiais. ADALINE esminis dalykas – naudojama tiesinė aktyvacijos funkcija. ADALINE mokymosi taisyklių formulė yra ta pati kaip ir perceptreono. Svoriai keičiami (atnaujinami) pagal šią mokymo taisyklę (formulę) [Kur21]:

$$w_k \coloneqq w_k - \eta(t_i - y_i)(-x_{ik})$$

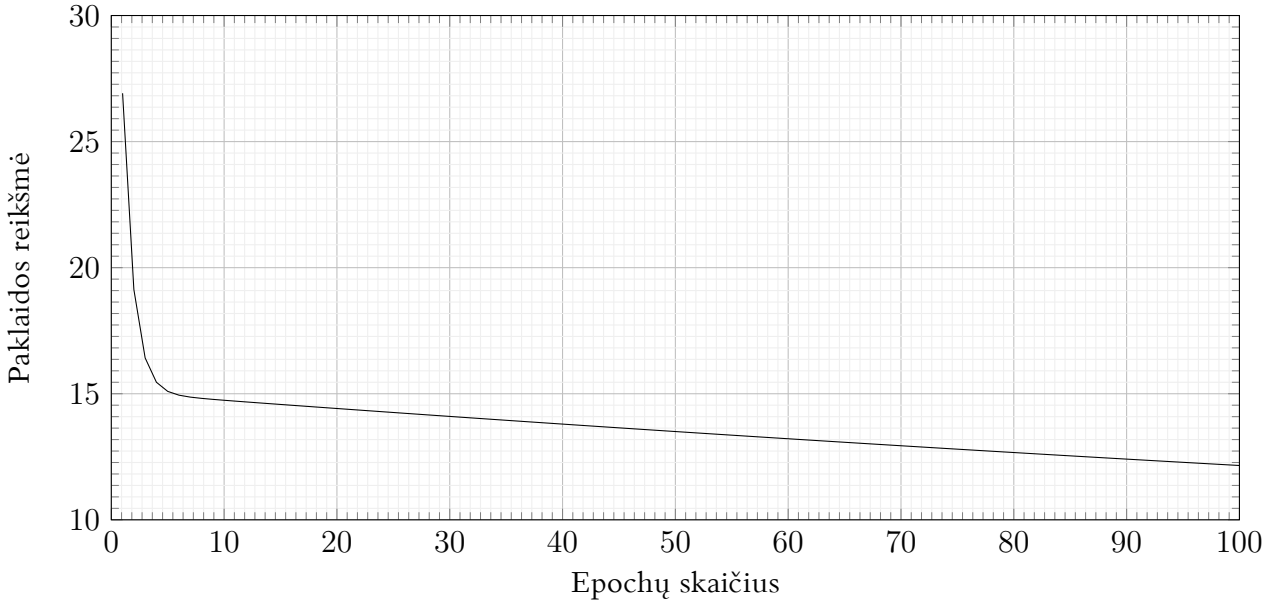
arba

$$w_k \coloneqq w_k + \eta(t_i - y_i)(-x_{ik})$$

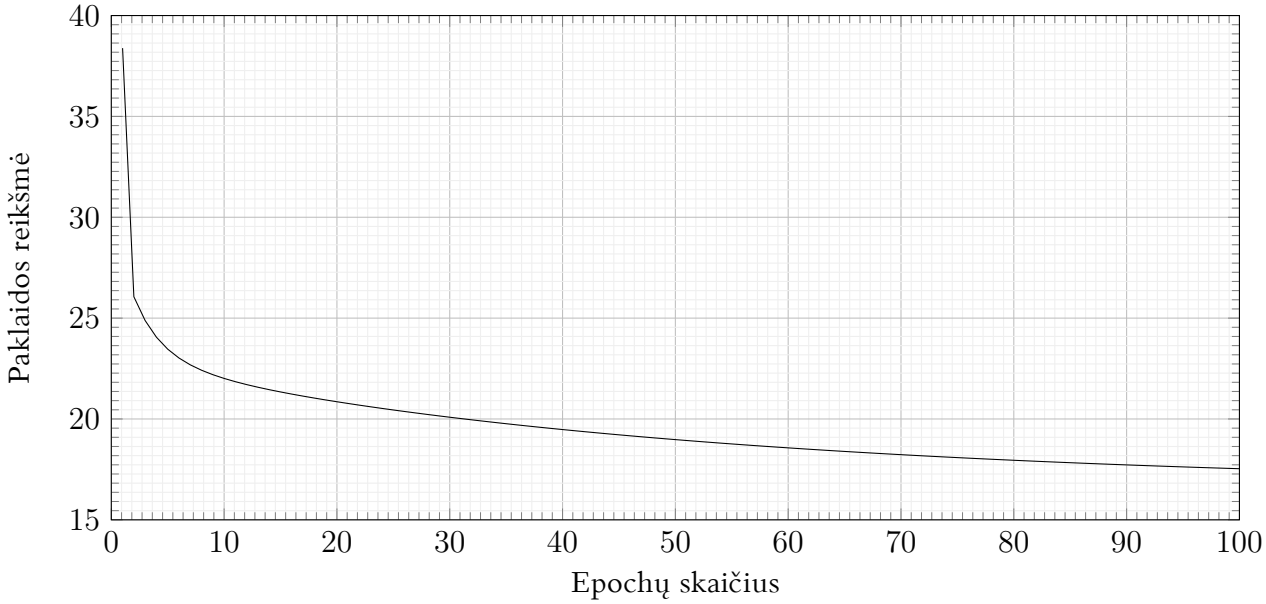
5. Tyrimo rezultatai

5.1. Paklaidos reikšmių priklausomybė nuo epochų skaičiaus

Ir irisų, ir krūties vėžio navikų paklaidos reikšmė greit nukrenta, o po to ir toliau lėtai, logaritmiškai mažėja didėjant epochų skaičiui.



1 pav. Paklaidos reikšmės priklausomybė nuo epochų skaičiaus irisus klasifikuojančiam neuronui

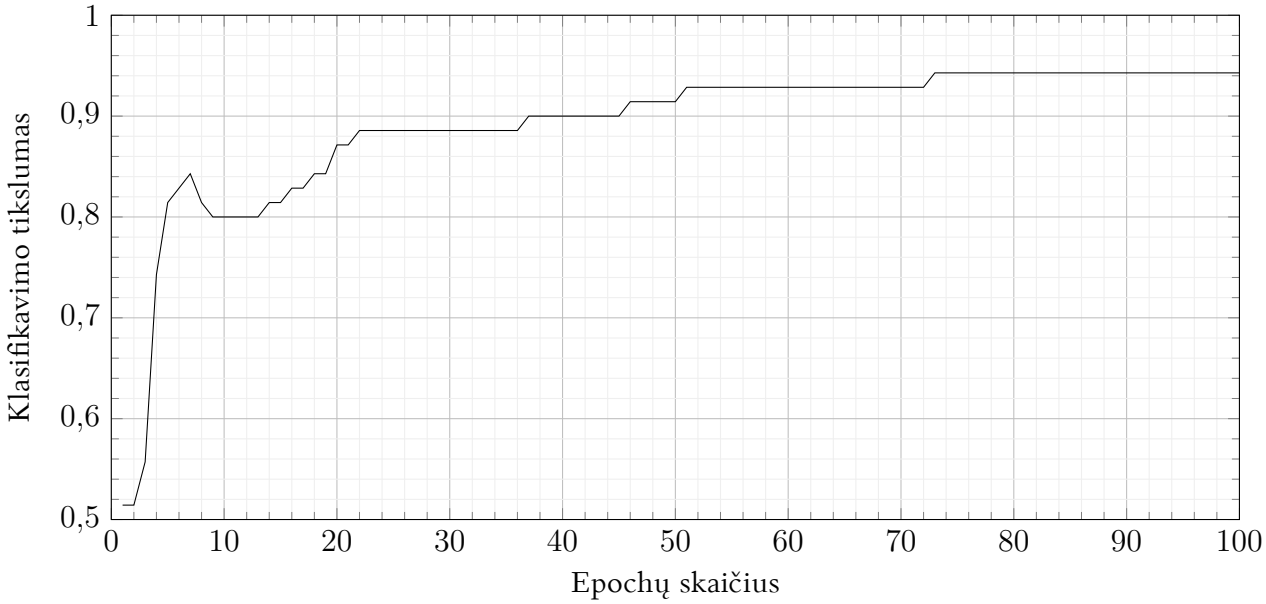


2 pav. Paklaidos reikšmės priklausomybė nuo epochų skaičiaus krūties vėžio navikus klasifikuojančiam neuronui

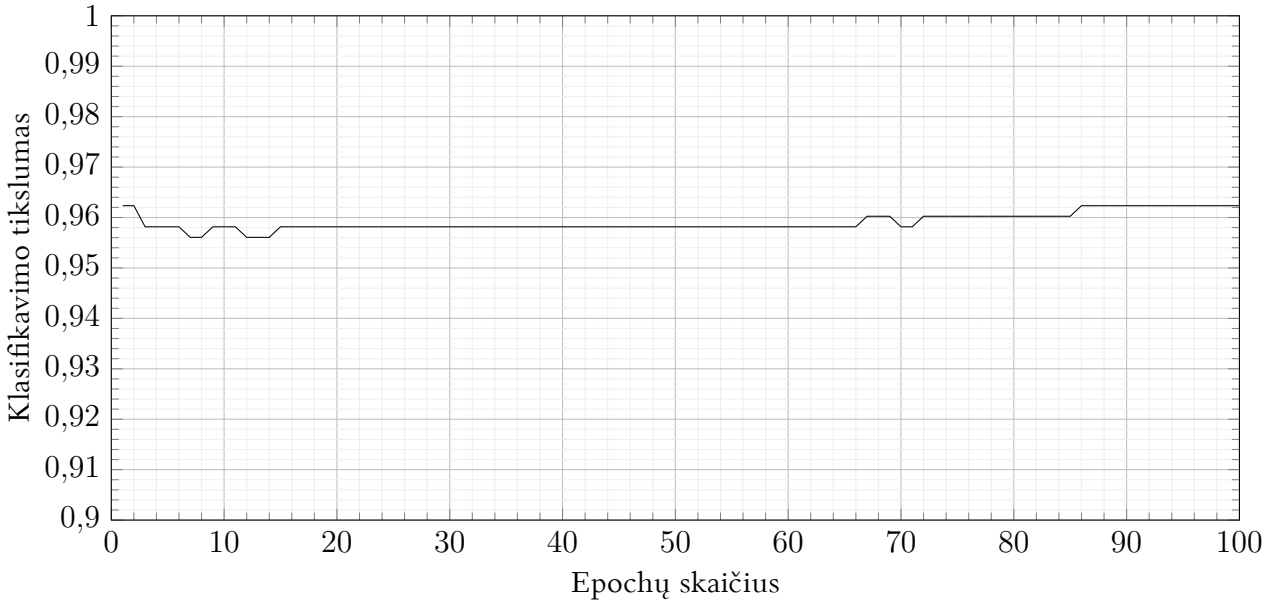
5.2. Klasifikavimo tikslumo priklausomybė nuo epochų skaičiaus

Irisų klasifikavimo tikslumas labiausiai auga pirmomis epochomis, vėlesnėmis epochomis auga ir toliau, tačiau lėčiau.

Krūties vėžio navikų klasifikavimo tikslumas visada lieka aukštas ir beveik nepriklauso nuo epochų skaičiaus.



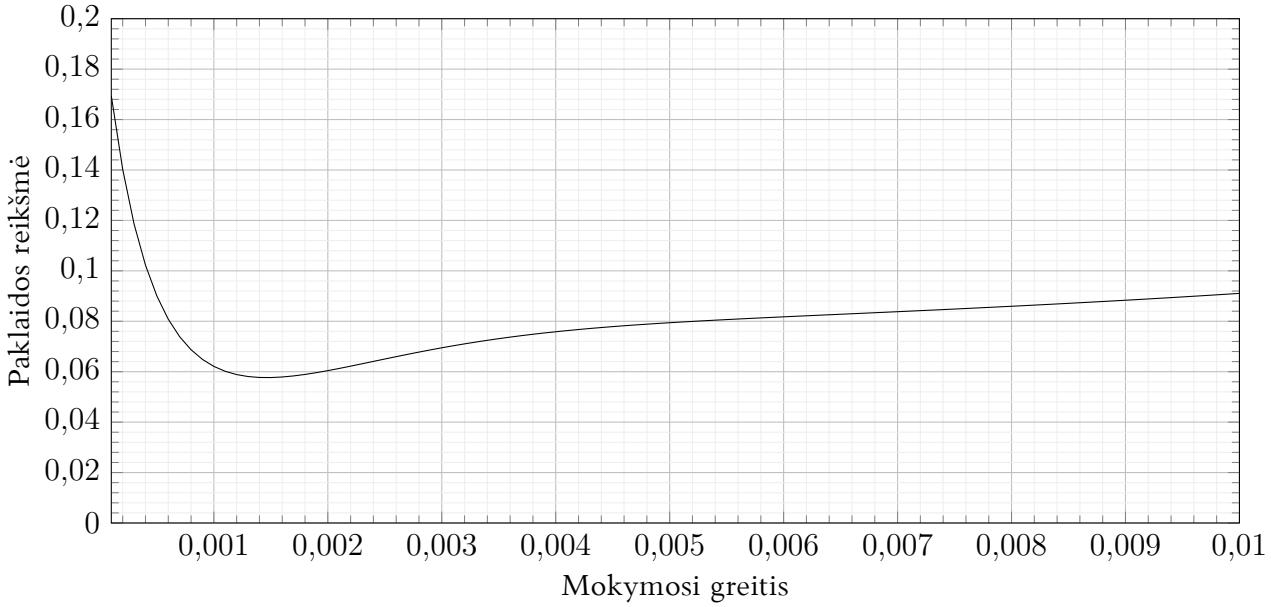
3 pav. Klasifikavimo tikslumo priklausomybė nuo epochų skaičiaus irisus klasifikuojančiam neuronui



4 pav. Klasifikavimo tikslumo priklausomybė nuo epochų skaičiaus krūties vėžio navikus klasifikuojančiam neuronui

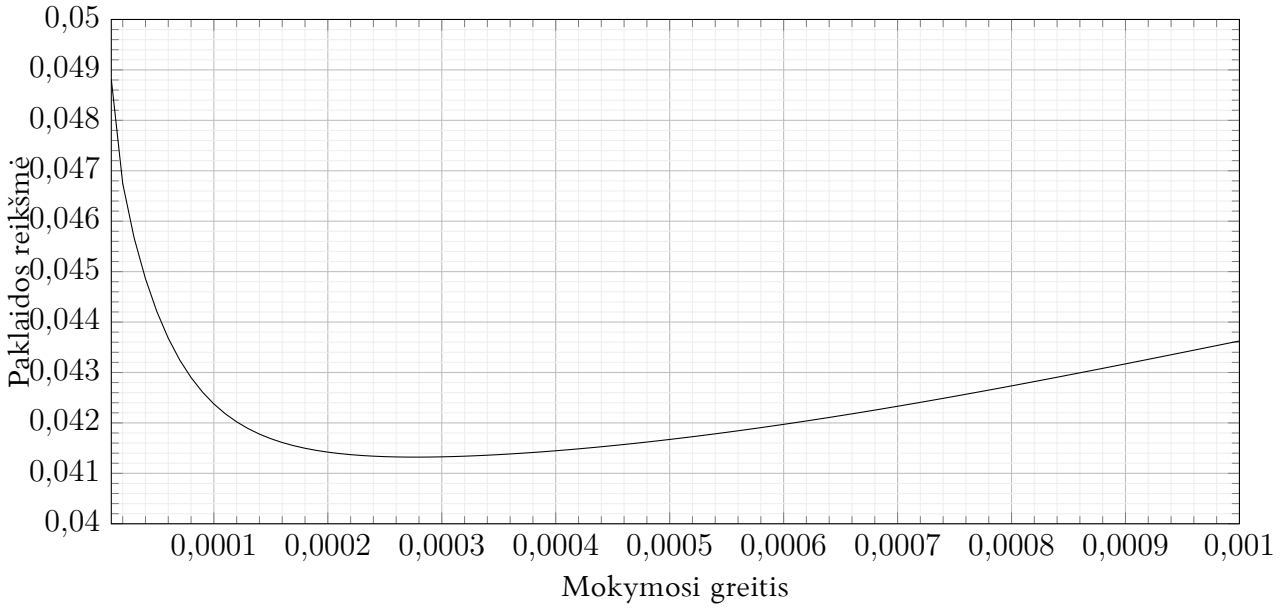
5.3. Rezultatų priklausomybė nuo skirtingų mokymosi greičio reikšmių

Irisus klasifikuojančio neurono paklaidos reikšmė esant mažam mokymosi greičiui yra aukšta, tada nukrenta iki minimumo, kai mokymosi greitis yra 0,0015.



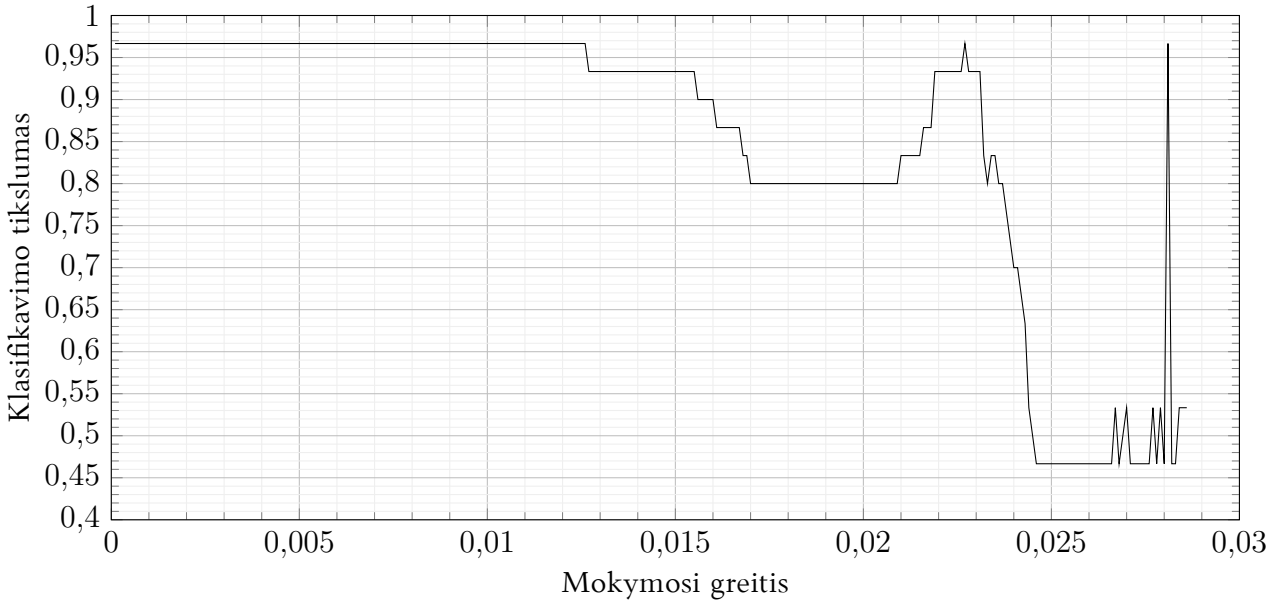
5 pav. Paklaidos reikšmės priklausomybė nuo mokymosi greičio irisus klasifikuojančiam neuronui

Krūties vėžio navikus klasifikuojančio neurono paklaidos reikšmė esant mažam mokymosi greičiui yra aukšta, tačiau nukrenta iki minimumo, kai mokymosi greitis yra 0,00028.



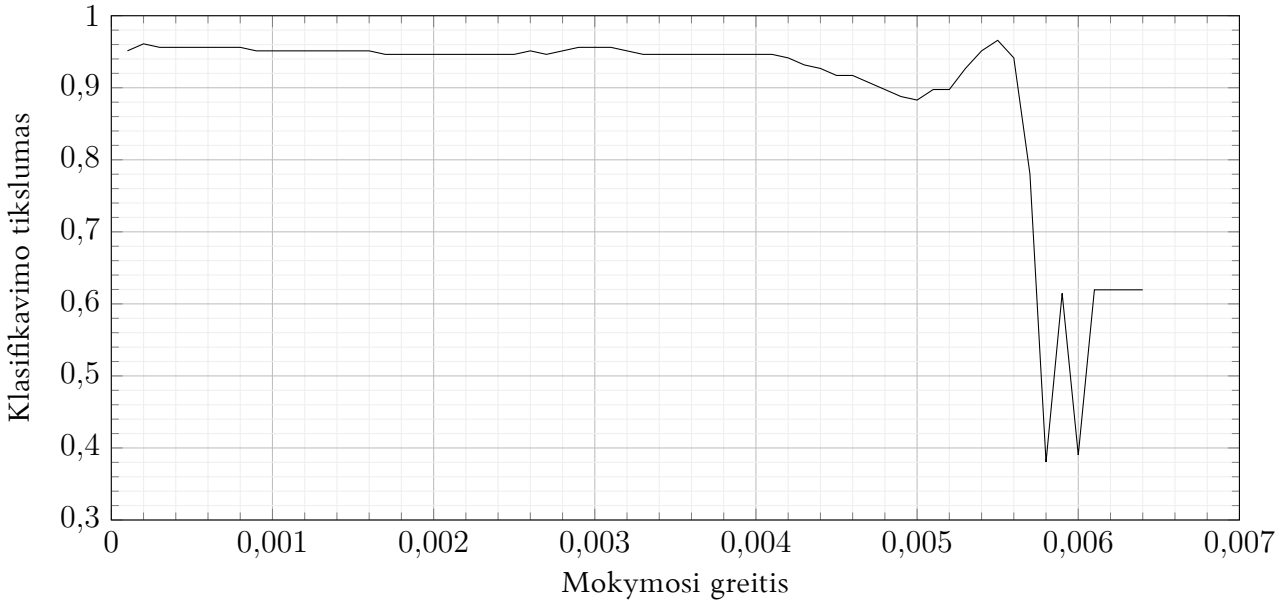
6 pav. Paklaidos reikšmės priklausomybė nuo mokymosi greičio krūties vėžio navikus klasifikuojančiam neuronui

Aukščiausias irisų klasifikavimo tikslumas gaunamas, kai mokymosi greitis yra nuo 0,0001 iki 0,0126.



7 pav. Klasifikavimo tikslumo priklausomybė nuo mokymosi greičio irisus klasifikuojančiam neuronui

Aukščiausias krūties vėžio navikų klasifikavimo tikslumas pasiekiamas, kai mokymosi greitis lygus 0,0055.



8 pav. Klasifikavimo tikslumo priklausomybė nuo mokymosi greičio krūties vėžio navikus klasifikuojančiam neuronui

6. Gauti duomenys

Ir irisus, ir krūties vėžio navikus klasifikuojančių dirbtinių neuronų mokymui naudota 100 epochų, kadangi tada klasifikavimo tikslumas jau būna pasiekęs aukščiausią galimą reikšmę.

Irisus klasifikuojančiam neurono klasifikavimo tikslumas aukščiausias, o paklaidos reikšmė mažiausia, kai mokymosi greitis lygus 0,0015.

Krūties vėžio navikus klasifikuojančiam neurono paklaidos reikšmė mažiausia, kai mokymosi greitis lygus 0,0055. Tačiau, paklaidos reikšmė mažiausia, kai mokymosi greitis yra 0,00028. Atsižvelgiant į tai, kad klasifikavimo tikslumas su mokymosi greičiu lygiu 0,00028 mažesnis tik per 0,009756, optimalus santykis tarp mažiausios paklaidos reikšmės ir aukščiausio klasifikavimo tikslumo gaunamas su mokymosi greičiu, kuris lygus 0,00028.

1 lentelė. Irisus klasifikuojančio dirbtinio neurono rezultatai

Gauti svoriai	-0,0956; -0,2692; -0,196; 0,3945; 0,6613
Epochų skaičius	100
Paklaida paskutinėje epochoje mokymo duomenims	4,3894
Klasifikavimo tikslumas paskutinėje epochoje mokymo duomenims	0,9143
Paklaida testavimo duomenims	0,0807
Klasifikavimo tikslumas testavimo duomenims	0,9667

2 lentelė. Krūtų vėžio navikus klasifikuojančio dirbtinio neurono rezultatai

Gauti svoriai	-0,2409; 0,032; 0,0247; 0,0158; 0,011; 0,0113; 0,0423; 0,0171; 0,009; 0,0093
Epochų skaičius	100
Paklaida paskutinėje epochoje mokymo duomenims	17,0885
Klasifikavimo tikslumas paskutinėje epochoje mokymo duomenims	0,9623
Paklaida testavimo duomenims	0,0413
Klasifikavimo tikslumas testavimo duomenims	0,9561

7. Rezultatai ir išvados

Apmokius neuroną klasifikuoti irisus ir krūties vėžio navikus pasiektas 95 % klasifikavimo tikslumas, apskaičiuoti svoriai, su kuriais pasiekiamas aukščiausias tikslumo ir žemiausios paklaidos rezultatas. Dirbtinis neuronas naudoja stochastinį gradientinį nusileidimą ir ADALINE mokymosi taisyklę. Paklaidos reikšmė krenta didėjant epochų skaičiui. Paklaidos reikšmė įgija minimumą tik su tam tikru mokymosi greičiu, o toliau didėja. Klasifikavimo tikslumas taip pat priklauso nuo mokymosi greičio ir krenta mokymosi greičio reikšmei kylant.

Šaltiniai

[Fis36] R. A. Fisher. *Iris* [UCI Machine Learning Repository]. 1936. Pasiekiamas per DOI: 10.24432/C56C76.

[Kur21] O. Kurasova. *Skaitmeninis intelektas ir sprendimų priėmimas*. 2021.

[Wol92] William Wolberg. *Breast Cancer Wisconsin (Original)* [UCI Machine Learning Repository]. 1992. Pasiekiamas per DOI: 10.24432/C5HP4Z.