KAUNO TECHNOLOGIJOS UNIVERSITETAS
INFORMATIKOS FAKULTETAS

INTELEKTIKOS PAGRINDAI (P176B101)

3 laboratorinio darbo ataskaita.

Atliko:
IFF-1/6 gr. studentas
Lukas Kuzmickas
Priėmė:
jaun.asist. Nakrošis Arnas

KAUNAS 2024

Turinys

PAVEIKSLŲ SĄRAŠAS

1. ĮVADAS

Laboratorinio darbo užduotis susideda iš dviejų dalių.

Pirmoje dalyje reikia sukurti neuroną su tiesine aktyvavimo funkcija, jį apmokinti naudojant 1700 m. – 2014 m. saulės dėmių aktyvumo duomenis, ištirti gautas neurono prognozės paklaidas ir atsakyti į klausimus.

Antroje dalyje reikia pasirinkti duomenų rinkinį bei tikslo atributą sukurti reikšmės prognozavimo, ar klasifikacijos modelį, įvertinti sukurto modelio tikslumą, naudojant 10 intervalų kryžminės patikros metodą, pabandyti padidinti modelio tikslumą taikant nurodytus metodus.

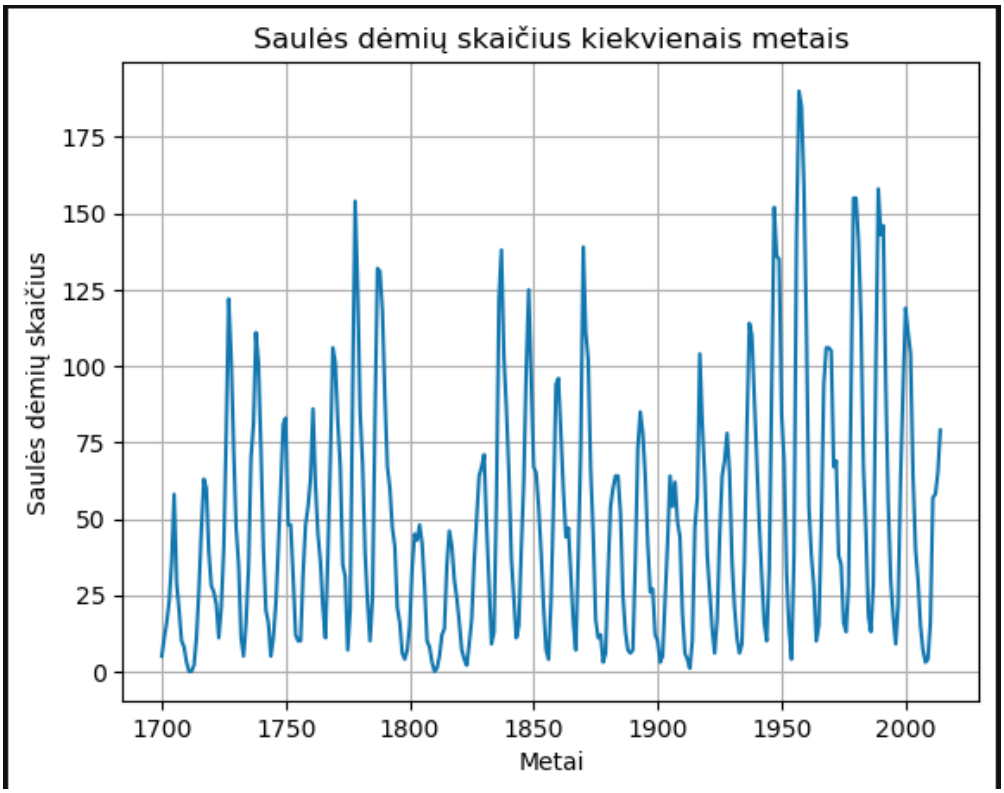
2. DUOMENŲ PARUOŠIMAS

Duomenų apie saulės dėmių aktyvumą nuskaitymui iš failo sunspot.txt naudotas kodas pateiktas žemiau, suskirstome matricoje, pirmas stulpelis – metai, antras – saulės dienų aktyvumas (1 pav.):

```
class InOut:
    @staticmethod
    def read_file(filename):
        f = open(filename,"r")
        arr = {"year": [], "sunspot": []}
        for line in f:
            s_line = line.split("\t")
            arr["year"].append(float(s_line[0]))
            arr["sunspot"].append(float(s_line[1].strip()))
        f.close()
        return arr
```

1 pav. Nuskaitymo metodo kodo fragmentas.

Naudojantis gautais duomenimis nubraižome saulės dėmių aktyvumo už 1700- 2014 metus grafiką, grafikas matomas (2 pav.):



2 pav. Saulės dėmių skaičius kiekvienais metais grafikas..

Toliau suskirstome nuskaitytus duomenis į įvesties ir išvesties matricas (3 pav.):

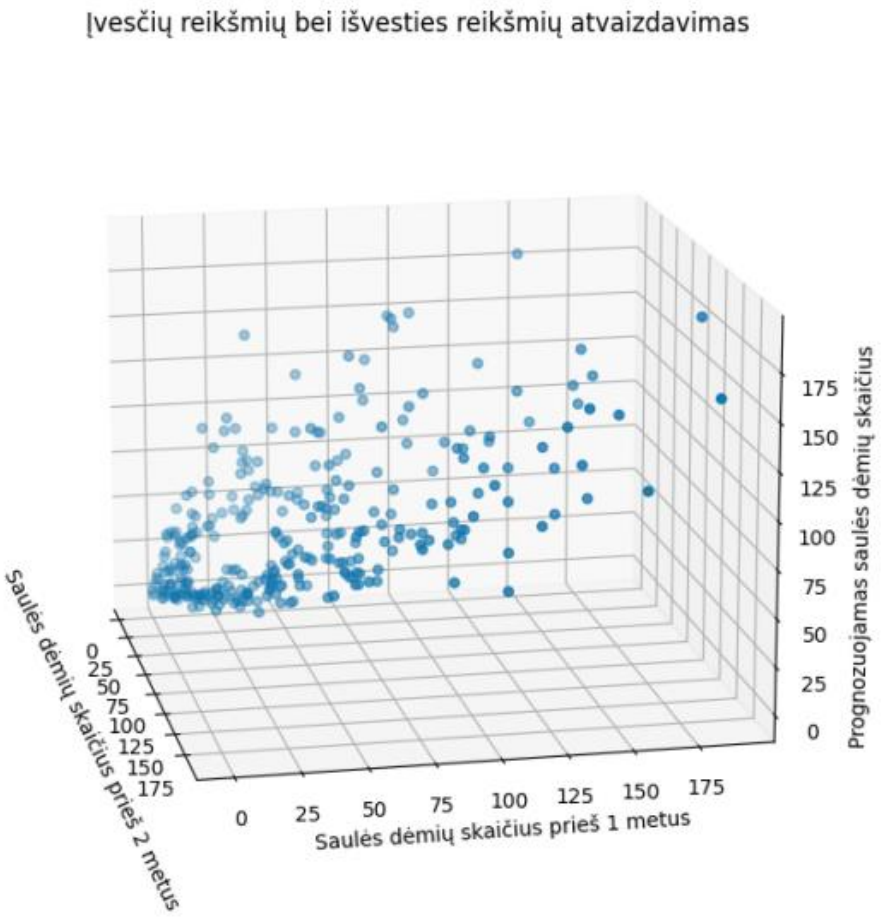
```
@staticmethod
def prepare_data(data, n):
    return np.array([[data[j] for j in range(i, i + n)] for i in range(0, len(data) - n)]), np.array(data[n:len(data)])
```

3 pav. Duomenų suskirstymas į matricas.

Data – duomenys;

N – skaičius nurodantis pagal kelių metų saulės dėmių skaičių bus prognozuojama sekančių metų saulės dėmių skaičius.

Toliau nubrėžiame trimate diagramą, kurioje pastebime, kaip nuo įvesties reikšmės kinta išvestys (4 pav.).



4 pav. Įvesčių reikšmių bei išvesties reikšmių atvaizdavimas trimatėje erdvėje.

Iš 4 pav. galima pastebėti, kad neurono svoriniai koeficientai turėtų būti parinkti taip, kad juos naudojant nubraižyta plokštuma būtų kuo arčiau kiekvieno trimatėje erdvėje matomo taško.

Toliau suskirstome duomenys į apmokymo ir testavimo aibes (5 pav.):

```
@staticmethod
def split_data(P, T, count):
    return P[:count], T[:count], P[count:], T[count:]
```

5 pav. Duomenų suskirstymas į apmokymo ir testavimo aibes.

Šio laboratorinio darbo atveju 200 duomenų įrašų buvo įtraukti į apmokymo duomenų rinkinį, o visi kiti – į testavimo duomenų rinkinį.

3. TIESINĖS AUTOREGRESIJOS MODELIO KŪRIMAS IR TESTAVIMAS

Toliau naudojantis apmokymo duomenų rinkiniu sukuriamas tiesinės autoregresijos modelis. Modelis sukurtas naudojantis „sklearn“ biblioteka (6 pav.).

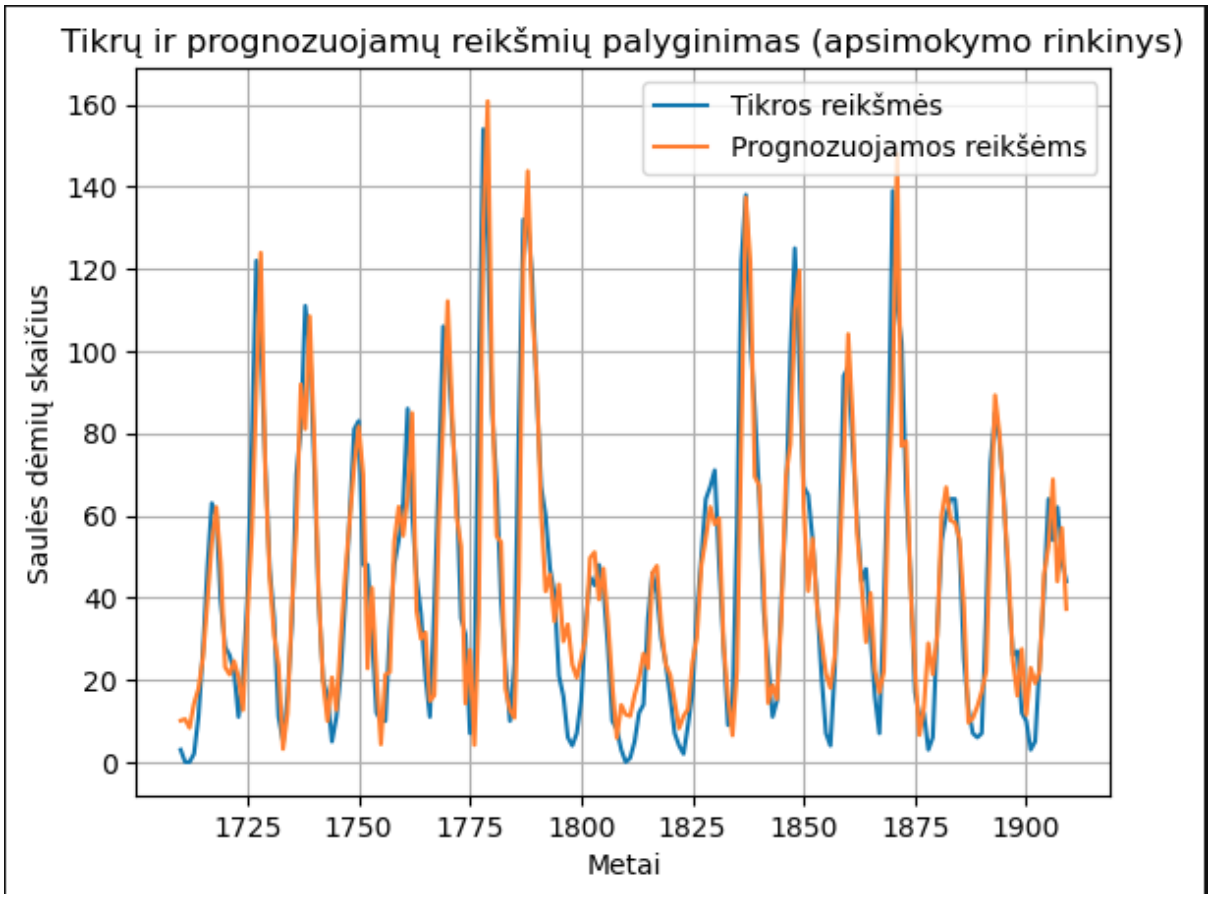
```
Pu, Tu, Pt, Tt = Tasks.split_data(P,T,TEST)

model = LinearRegression().fit(Pu, Tu)
```

6 pav. Modelio sukūrimas naudojant sklearn.

Naudojant tiesinės autoregresijos modelį gautos tokios svorių koeficientų reikšmės:

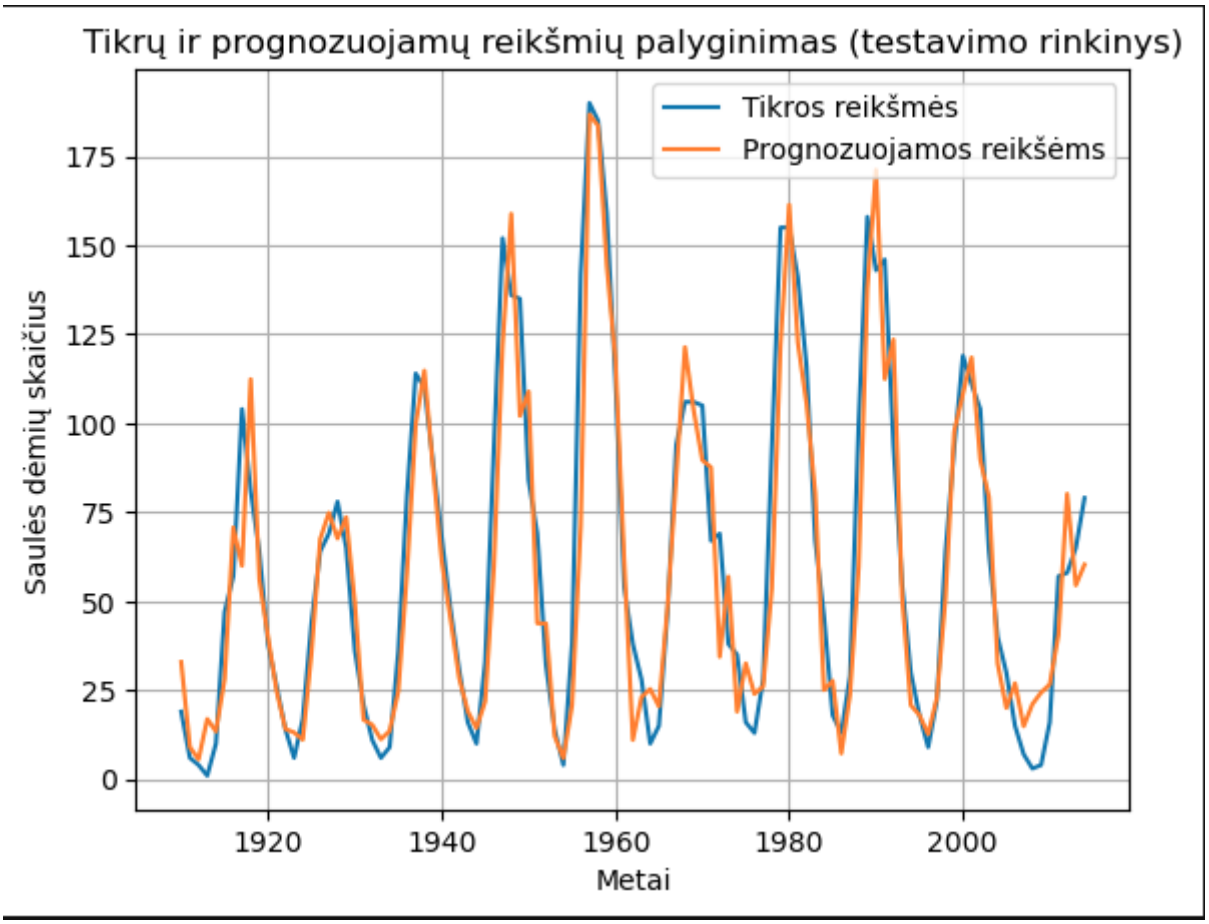
- b = 13.403683236718116;
- $w1$ = −0.6760819763970695;
- $w2$ = 1.3715093938395846.

Galime patikrinti modelio prognozavimo kokybę atliekant modelio veikimo imitaciją. Pirmiausia tam buvo panaudotas apmokymo duomenų rinkinys (rezultatai 7 pav.):



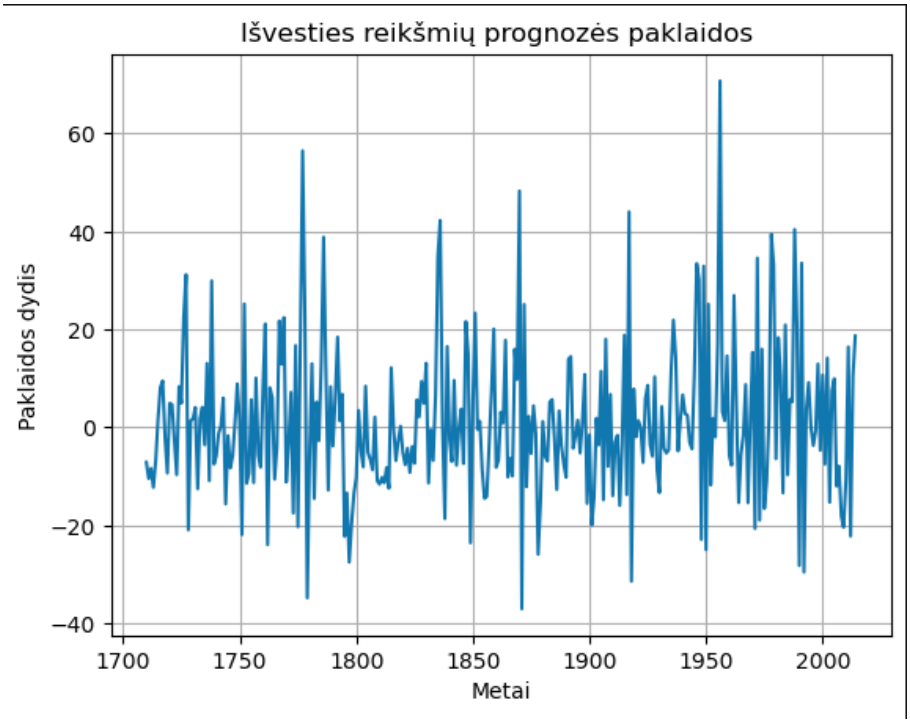
7 pav. Tikrų ir prognozuojamų reikšmių palyginimo grafikas.

Naudojant testavimo rinkinį (rezultatai 8 pav.):



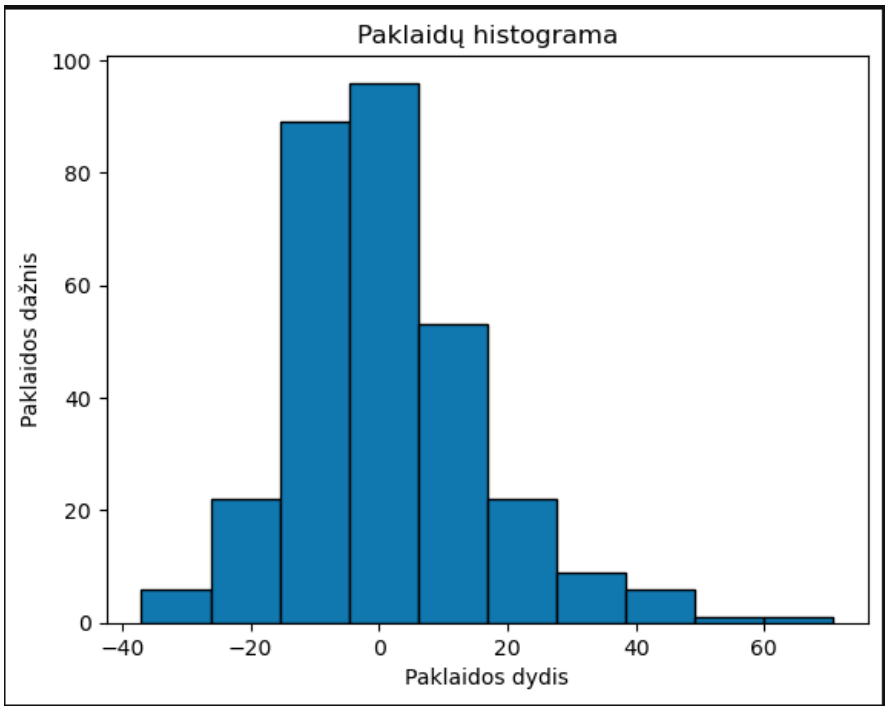
7 pav. Tikrų ir prognozuojamų reikšmių palyginimo grafikas.

Nubraižome išvesties reikšmių prognozės paklaidų grafiką (8 pav.):



8 pav. Išvesties reikšmių prognozės paklaidų grafikas.

Nubraižome išvesties reikšmių prognozės paklaidų histogramą (9 pav.):



8 pav. Išvesties reikšmių prognozės paklaidų histograma..

Iš 8 pav. galima pastebėti, kad dauguma paklaidų yra mažos (dydis tarp 0), tačiau egzistuoja ir kelios labai didelės paklaidos.

Norint įsitikinti, kad sukurto modelio prognozė yra pakankamai tiksli galima apskaičiuoti MSE bei MAD reikšmes pateiktas (9 pav.) ir (10 pav.)

čia N – duomenų rinkinio dydis,

$e(k)$ – tam tikros prognozės paklaida

$$MSE = \frac{1}{N} \cdot \sum_{k=1}^{N} e(k)^2$$

9 pav. MSE apskaičiavimo formulė.

$$MAD = median(|e(k)|), k \in [1; N]$$

10 pav. MAD apskaičiavimo formulė.

Atlikus skaičiavimus, gautos tokios įverčių reikšmės:

• MSE = 232.2686575802852;

• MAD = 8.618889655548999.

MSE reikšmė yra pakankamai didelė (šiame darbe MSE įvertis neturi viršyti 300). MSE yra jautresnis paklaidoms, nes prognozės paklaidos yra pakeliamos kvadratu. Tačiau MAD įvertis yra ganėtinai mažas, nes apskaičiuojamas gaunant vidurinės pagal dydį prognozės paklaidos reikšmės modulį.

4. DIRBTINIO NEURONO KŪRIMAS IR TESTAVIMAS

Toliau modelį modifikuojame, jis bus apmokomas iteraciniu būdu.

Dirbtinio neurono apmokymui parinkti tokie parametrai:

• Mokymosi greitis lr = 0.00015;

• Siekiama mokymosi paklaidos įverčio MSE reikšmė $goal$ = 200;

• Maksimalus epochų skaičius $epsc$ = 750.

Gautos svorio koeficientų reikšmės apmokius neuroną:

• b = 1.1875839979343173;

• $w1$ = −0.3487975057866066;

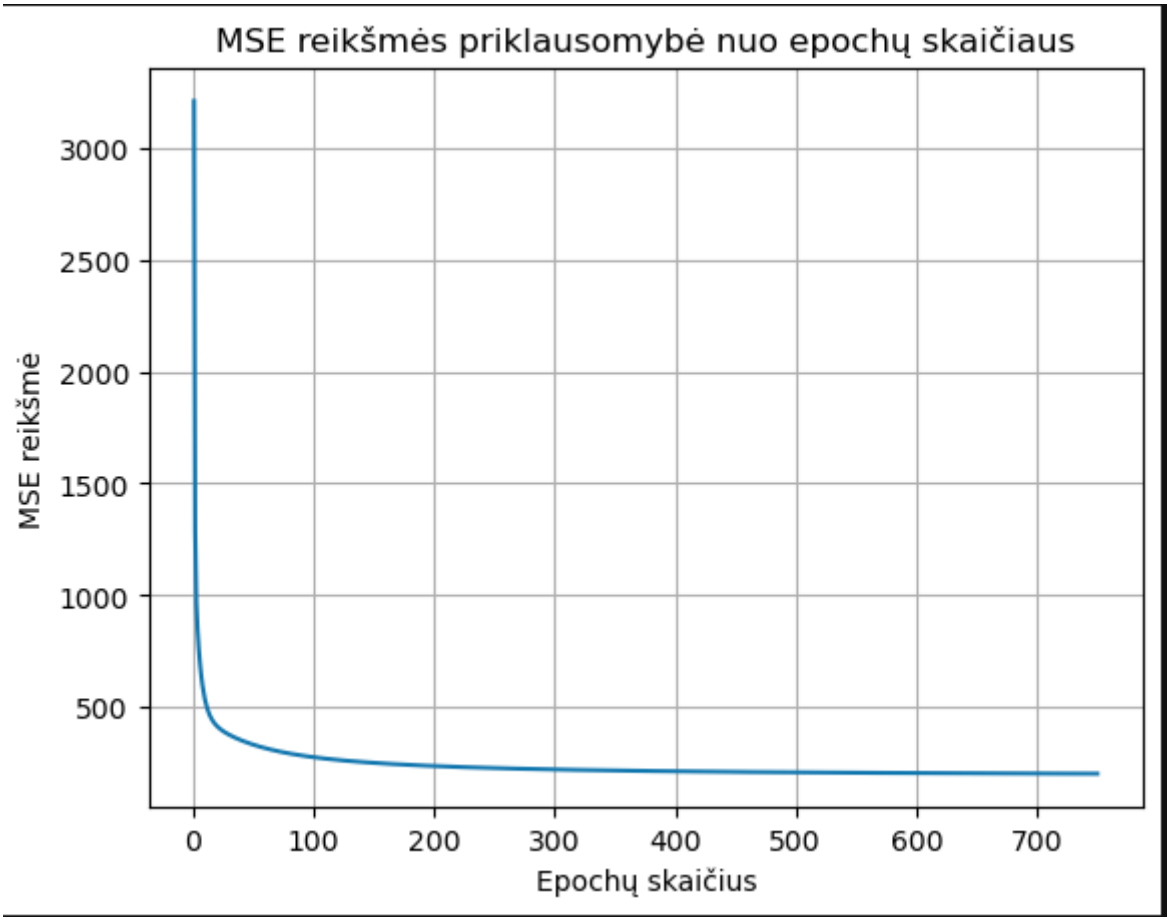
• $w2$ = 1.3600247049298067.

Aukščiau gautos svorio koeficientų reikšmės yra artimos tiesinės autoregresijos modelio svorio koeficientų reikšmėms. Taip yra todėl, nes tiesinės autoregresijos modelio svorio koeficientų reikšmės yra optimalios naudojant vieną neuroną, o iteraciniu būdų apmokomo neurono svorio koeficientų reikšmės turėtų artėti prie optimalių reikšmių.

5. ATSAKYMAI Į KLAUSIMUS PATEIKTUS LABORATORINIO DARBO APRAŠE

Ar mokymosi procesas yra konverguojantis? Jeigu ne, pamąstyti kas gali būti priežastimi ir pakeisti atitinkamą parametrą.

Taip, mokymosi procesas yra konverguojantis, nes vykdant jį mažėja MSE įverčio reikšmė (11 pav.):



11 pav. MSE priklausomybės nuo epochų skaičiaus grafikas.

Kokia yra neurono darbo kokybės įverčio MSE ir MAD reikšmės ?

MSE = 234.3211785207986

MAD = 8.542483083183384

Pastebėta, kad didinant maksimalų epochų skaičių tikslumas nežymiai padidėja. Mažinant bei didinant mokymosi greitį tikslumas mažėja.

Maksimali mokymosi greičio lr reikšmė, kuri užtikrina konvergavimą yra 0.00015.

6. MODELIO STRUKTŪROS KEITIMO ĮTAKA PROGNOZAVIMO TIKSLUMUI

Keičiame modelio struktūra ir tiriami gauti rezultatai, kai modelio eilė n = 6 bei n = 10.

Kai n = 6:

Po tiesinės autoregresijos modelio apmokymo gautos MSE ir MAD įverčių reikšmės:

MSE = 271.0407,

MAD = 8.955.

Po dirbtinio neurono apmokymo iteraciniu būdu su parametrais:

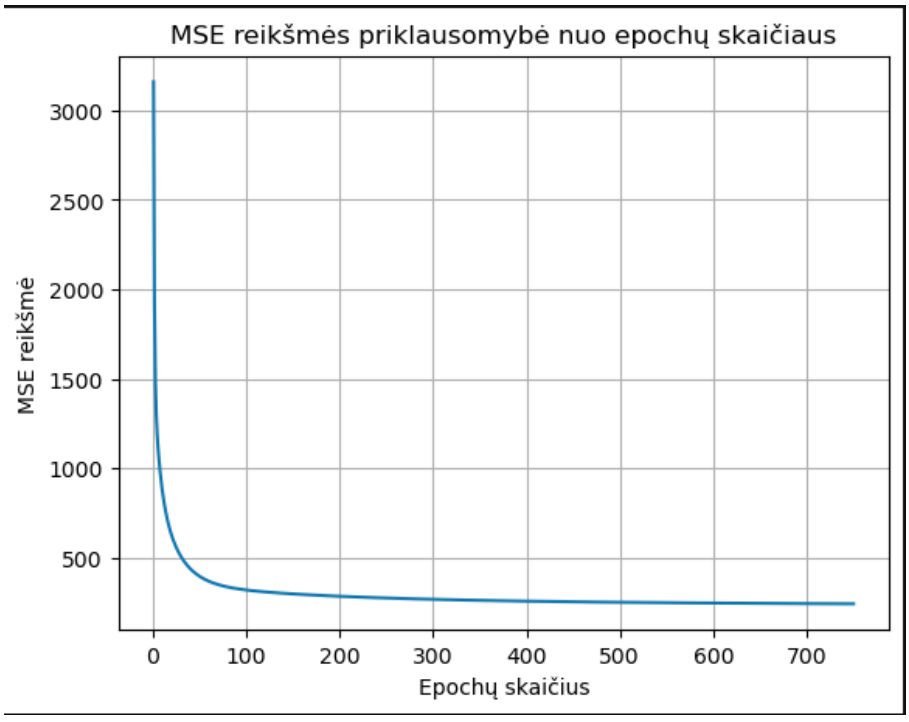
lr = 0.00015,

$epcs$ = 750,

$goal$ = 200.

gautos MSE ir MAD įverčių reikšmės: MSE = 295.84, MAD = 9.34.

Pastebime, kad išsilaiko mažesnė MSE įverčio reikšmė (12 pav.):



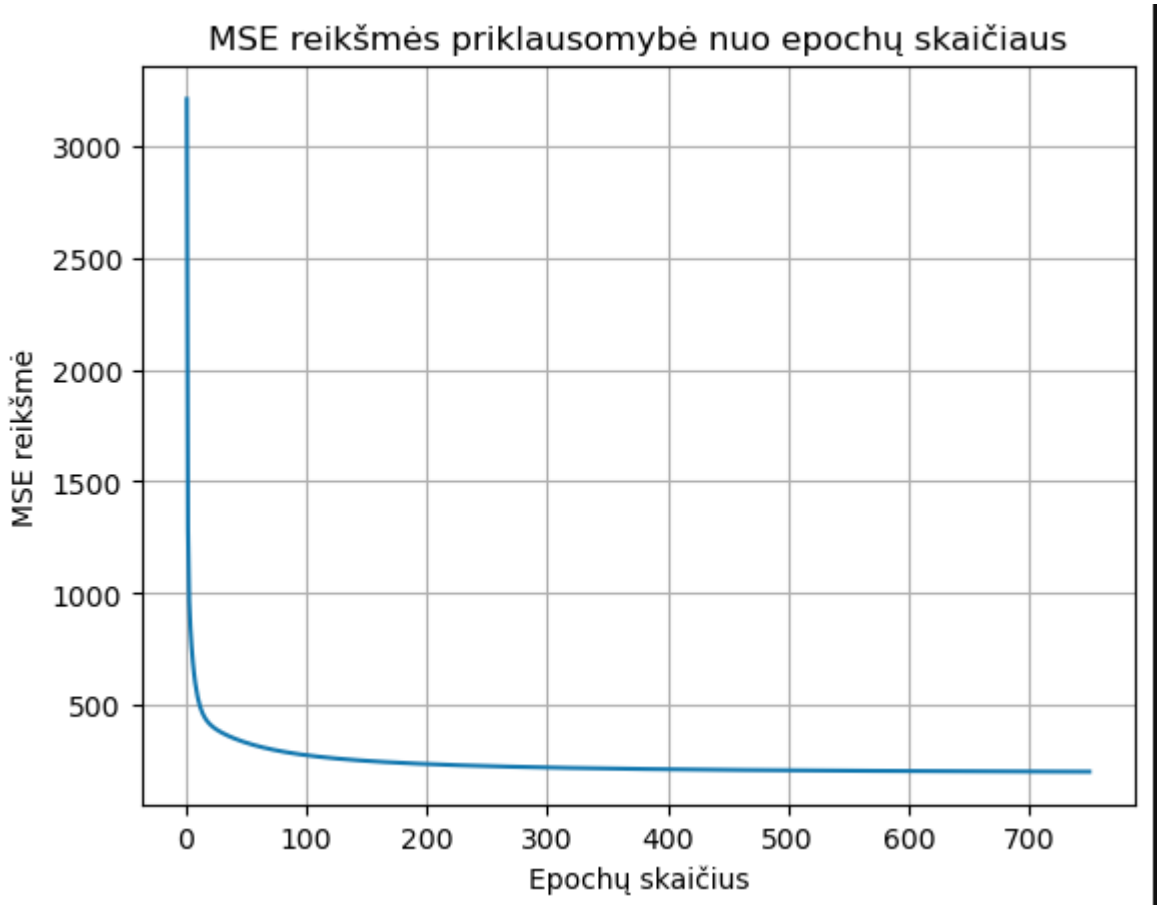
12 pav. MSE priklausomybės nuo epochų skaičiaus grafikas.

Padidinus modelio įvesčių kiekį iki 6, modelio tikslumas šiek tiek pagerėjo.

Kai n = 10:

Po tiesinės autoregresijos modelio apmokymo gautos MSE ir MAD įverčių reikšmės: MSE = 232.26, MAD = 8.6.

Po dirbtinio neurono apmokymo iteraciniu būdu su parametrais lr = 0.000015, $epcs$ = 750, $goal$ = 200 gautos MSE ir MAD įverčių reikšmės: MSE = 234.35, MAD = 8.577. Su didesniu modelio įvesčių skaičiumi, nusistovi mažesnė MSE įverčio reikšmė.



13 pav. MSE priklausomybės nuo epochų skaičiaus grafikas.

Padidinus modelio įvesčių kiekį iki 10 gautas dar geresnis tikslumas.

7. DUOMENŲ RINKINIO APRAŠYMAS IR PERTVARKYMAI

Šiame laboratoriniame darbe naudojamas kitoks duomenų rinkinys. Šiame duomenų rinkinyje pateikiamos „Vinho Verde“ vyno kokybės duomenų variantas. Duomenų rinkinys apibūdina įvairių vyne esančių cheminių medžiagų kiekį ir jų poveikį jo kokybei.

Išskirti tolydiniai duomenų rinkinio atributai: fixed acidity, volatile acidity, citric acid, residual sugar, chlorides, free sulfur dioxide, total sulfur dioxide, density, pH, sulphates, alcohol. Išmetame atributą id.

Išskirti kategoriniai duomenų rinkinio atributai: quality.

Kiekvienam tolydinio tipo atributui reikia apskaičiuoti:

• bendrą reikšmių skaičių;

• trūkstamų reikšmių procentą;

• kardinalumą;

• minimalią ir maksimalią reikšmes;

• 1-ąjį ir 3-ąjį kvartilius;

• vidurkį;

• medianą;

• standartinį nuokrypį.

Rezultatai matomi (14 pav.):

Tolydinio tipo duomenys:

Atributo pavadinimas	Kiekis (Eilučių sk.)	Trūkstamos reikšmės, %	Kardinalumas	Minimali reikšmė	Maksimali reikšmė	1-asis kvartilis	3-asis kvartilis	Vidurkis	Mediana	Standartinis nuokrypis
fixed acidity	1599	0%	96	4.6	15.9	7.1	9.2	8.319637	7.9	1.741096
volatile acidity	1599	0%	143	0.12	1.58	0.39	0.64	0.527821	0.52	0.17906
citric acid	1599	0%	80	0	1	0.09	0.42	0.270976	0.26	0.194801
residual sugar	1599	0%	91	0.9	15.5	1.9	2.6	2.538806	2.2	1.409928
chlorides	1599	0%	153	0.012	0.611	0.07	0.09	0.087467	0.079	0.047065
free sulfur dioxide	1599	0%	60	1	72	7	21	15.874922	14	10.460157
total sulfur dioxide	1599	0%	144	6	289	22	62	46.467792	38	32.895324
density	1599	0%	436	0.99007	1.00369	0.9956	0.997835	0.996747	0.99675	0.001887
pH	1599	0%	89	2.74	4.01	3.21	3.4	3.311113	3.31	0.154386
sulphates	1599	0%	96	0.33	2	0.55	0.73	0.658149	0.62	0.169507
alcohol	1599	0%	65	8.4	14.9	9.5	11.1	10.422983	10.2	1.065668

14 pav. Tolydinio tipo atributų reikšmės.

Kiekvienam kategorinio tipo atributui reikia apskaičiuoti:

• bendrą reikšmių skaičių;

• trūkstamų reikšmių procentą;

• kardinalumą;

• modą;

• modos dažnumo reikšmę;

• modos procentinę reikšmę;

• 2-ąją modą;

• 2-osios modos dažnumo reikšmę;

• 2-osios modos procentinę reikšmę.

Skaičiavimų rezultatai matomi (15 pav.):

Kategorinio tipo duomenys									
Atributo pavadinimas	Kiekis (Eilučių sk.)	Trūkstamos reikšmės, %	Kardinalumas	Moda	Modos dažnumas	Moda, %	2-oji Moda	2-osios Modos dažnumas	2-oji Moda, %
quality	1599	0%	6	5	681	43%	6	638	40%

15 pav. Kategorinio tipo atributų reikšmės.

Iš 1 ir 2 lentelių galima pastebėti, kad atributai neturi tuščių reikšmių, todėl nieko tvarkyti nereikia. Taip pat galima matyti, kad nėra atributų, kurie turėtų didelius kardinalumus, o tai reiškia, kad nei vienas atributas unikaliai neidentifikuoja duomenų. Vadinasi iš duomenų rinkinio nereikia šalinti nei atributų, nei duomenų įrašų. Be to visi tolydiniai atributai bus normalizuoti.

Galiausiai atributo quality reikšmės bus pernumeruotos nuo 0 einančiais sveikaisiais skaičiais įskaitant, kaip ir antrame laboratoriniame darbe.

• 3 → 0;

• 4 → 1;

• 5 → 2;

- 6 → 3;
- 7 → 4;
- 8 → 5.

8. DIRBTINIO NEURONŲ TINKLO ARCHITEKTŪRA

Dirbtinis neuronų tinklas buvo kuriamas naudojantis python programavimo kalbos biblioteka „tensorflow“ (16 pav.):

```
model = tf.keras.Sequential([
    tf.keras.layers.Dense(16, activation='sigmoid', input_dim=11),
    tf.keras.layers.Dense(32, activation='sigmoid'),
    tf.keras.layers.Dense(32, activation='sigmoid'),
    tf.keras.layers.Dense(1, activation='linear')
])
model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001), loss=tf.keras.losses.MeanSquaredError())
model.fit(data_in_train,data_out_train,epochs=200,batch_size=50)
data_out_pred = np.around(model.predict(data_in_test))
accuracy.append(accuracy_score(data_out_test,data_out_pred))
```

16 pav. Tensorflow neuronų tinklas.

Šiuo kodu aprašytas neuroninis tinklas yra sudarytas iš 4 sluoksnių:

Pirmame sluoksnyje yra 16 neuronų, į kiekvieną kurių paduodami 11 įvesties signalų.

Antrame sluoksnyje yra 32 neuronai, į kiekvieną kurių paduodami visi pirmo sluoksnio išvesčių signalai.

Trečiame sluoksnyje yra 32 neuronai, į kiekvieną kurių paduodami visi antro sluoksnio išvesčių signalai. Visų šių sluoksnių neuronų aktyvacijos funkcija yra pateikta (17 pav.):

$$f(x) = \max(0, x)$$

17 pav. Aktyvacijos funkcija.

Ketvirtame sluoksnyje yra 1 neuronas, kuris gauna signalus iš visų 3 sluoksnio neuronų. Ketvirto sluoksnio neurono aktyvacijos funkcija yra pateikta (18 pav.):

$$f(x) = x$$

18 pav. Aktyvacijos funkcija.

Parinktas sukurto modelio mokymosi greitis $lr = 0.1$, pasirinktas klaidos įvertis, kuris bus optimizuojamas – MSE.

9. SUKURTO DIRBTINIO NEURONŲ TINKLO TIKSLUMO ĮVERTINIMAS

Sukurto dirbtinio neuronų tinklo prognozavimo tikslumui įvertinti naudojant 10 intervalų kryžminės patikros metodą (19 pav.):

```
cv = KFold(n_splits=10)
accuracy = []
for train, test in cv.split(data_in):
    data_in_train, data_in_test, data_out_train, data_out_test = data_in[train], data_in[test], data_ordinal_out[train], data_ordinal_out[test]
    model = tf.keras.Sequential([
        tf.keras.layers.Dense(16, activation='relu', input_dim=11),
        tf.keras.layers.Dense(32, activation='relu'),
        tf.keras.layers.Dense(32, activation='relu'),
        tf.keras.layers.Dense(1, activation='linear')
    ])
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.1), loss=tf.keras.losses.MeanSquaredError())
    model.fit(data_in_train,data_out_train,epochs=200,batch_size=50)
    data_out_pred = np.around(model.predict(data_in_test))
    accuracy.append(accuracy_score(data_out_test,data_out_pred))

print(accuracy)
print("Tikslumas: {0:2.2%}".format(np.mean(accuracy)))
print("Standartinis nuokrypis: {0:2.2%}".format(np.std(accuracy)))
```

19 pav. Kryžminės patikros metodas.

Naudojant 10 intervalų kryžminės patikros metodą, duomenų rinkinys padalinamas į 10 dalių.

9 iš gautų dalių yra naudojamos apmokyti dirbtini neuronų tinklą (pasirinktas apmokymo epochų skaičius lygus 200), o likusi dalis naudojama apmokyto modelio testavimui.

Ištestavus modelį įsimenamas teisingų prognozių procentas ir viena iš mokymo dalių sukeičiama su testavimo dalimi. Šis procesas kartojamas, kol visos 10 duomenų rinkinio dalių yra panaudojamos modelio testavimui. Gautiems tikslumams apibendrinti skaičiuojamas jų vidurkis bei standartinis nuokrypis.

Su kiekvienu kryžminės patikros intervalu gauti prognozių tikslumai: 66.02%, 65.18%, 63.37%, 45.85%, 39.07%, 40.07%, 40.94%, 42.14%, 47.47%, 40.07%

Sukurto dirbtinio neuronų tinklo vidutinis prognozių tikslumas yra 51.84%, prognozių tikslumo standartinis nuokrypis yra 12.69%.

10. DIRBTINIO NEURONŲ TINKLO ARCHITEKTŪROS PATOBULINIMAS

Patobuliname mūsų neuronų tinklą (20 pav.):

```
cv = KFold(n_splits=10)
accuracy = []
for train, test in cv.split(data_in):
    data_in_train, data_in_test, data_out_train, data_out_test = data_in[train], data_in[test], data_ordinal_out[train], data_ordinal_out[test]
    model = tf.keras.Sequential([
        tf.keras.layers.Dense(16, activation='sigmoid', input_dim=11),
        tf.keras.layers.Dense(32, activation='sigmoid'),
        tf.keras.layers.Dense(32, activation='sigmoid'),
        tf.keras.layers.Dense(32, activation='sigmoid'),
        tf.keras.layers.Dense(32, activation='sigmoid'),
        tf.keras.layers.Dense(1, activation='linear')
    ])
    model.compile(optimizer=tf.keras.optimizers.Adam(learning_rate=0.001), loss=tf.keras.losses.MeanSquaredError())
    model.fit(data_in_train,data_out_train,epochs=200,batch_size=50)
    data_out_pred = np.around(model.predict(data_in_test))
    accuracy.append(accuracy_score(data_out_test,data_out_pred))
```

20 pav. Patobulintas neuronų tinklas.

Iš 20 pav. kodo fragmento galima pastebėti tokius modelio pakeitimus:

• Pridėti dar 2 neuronų sluoksniai, kuriuose yra po 32 neuronus;

• Visų sluoksnių, išskyrus paskutinį aktyvacijos funkcija pakeista;

• Pakeistas modelio mokymosi greitis lr = 0.001.

Su kiekvienu kryžminės patikros intervalu gauti prognozių tikslumai: 60.08%, 55.65%, 64.88%, 62.75%, 58.75%, 55.2%, 60.63%, 66.25%, 59.86%

Patobulinto dirbtinio neuronų tinklo vidutinis prognozių tikslumas yra 59.59%, prognozių tikslumo standartinis nuokrypis yra 4.12%.

Tai reiškia, kad patobulintas modelis pasiekia 7.75% geresnį tikslumą.

11. IŠVADOS

1. Dirbtinių neuronų tinklo prognozavimo kokybė labai priklauso nuo pačio tinklo architektūros;

2. Ieškant artimos optimaliai dirbtinių neuronų tinklo struktūros, gali reikėti išbandyti kelis, o kartais ir daugiau, skirtingų tinklo variantų.