Kauno technologijos universitetas

Matematikos ir gamtos mokslų fakultetas

Intelektikos pagrindai

(P176B101)

Laboratorinis darbas Nr. 3

Darbą atliko:

MGTM – 9/1 gr. studentas

Tauras Gaulia

Darbą priėmė:

lekt. **Germanas Budnikas**

doc. **Agnė Paulauskaitė-Tarasevičienė**

Kaunas, 2021

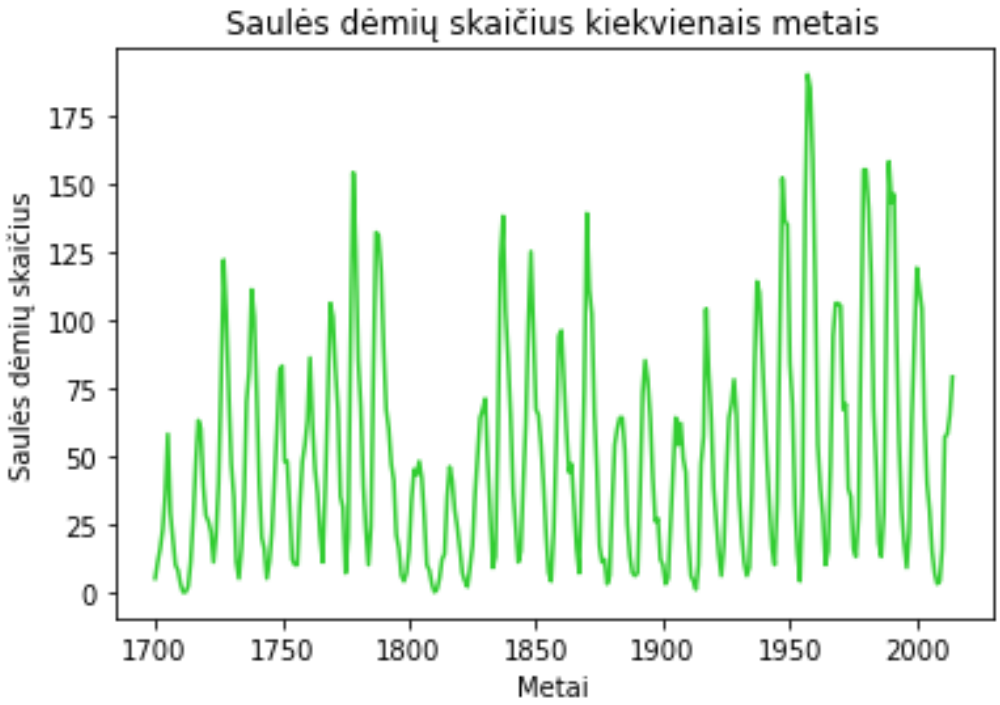
Užduotis

- Susipažinti su prognozavimo uždavinio sprendimu panaudojant tiesinį dirbtinį neuroną
- Susipažinti su neuroninio tinklo mokymosi, testavimo ir jų panaudojimo uždaviniais
- Pritaikyti įgytas žinias kuriant modelį prognozavimo ar klasifikacijos uždaviniui spręsti naudojant pasirinktą duomenų rinkinį

Darbas atliktas naudojantis Python aplinka.

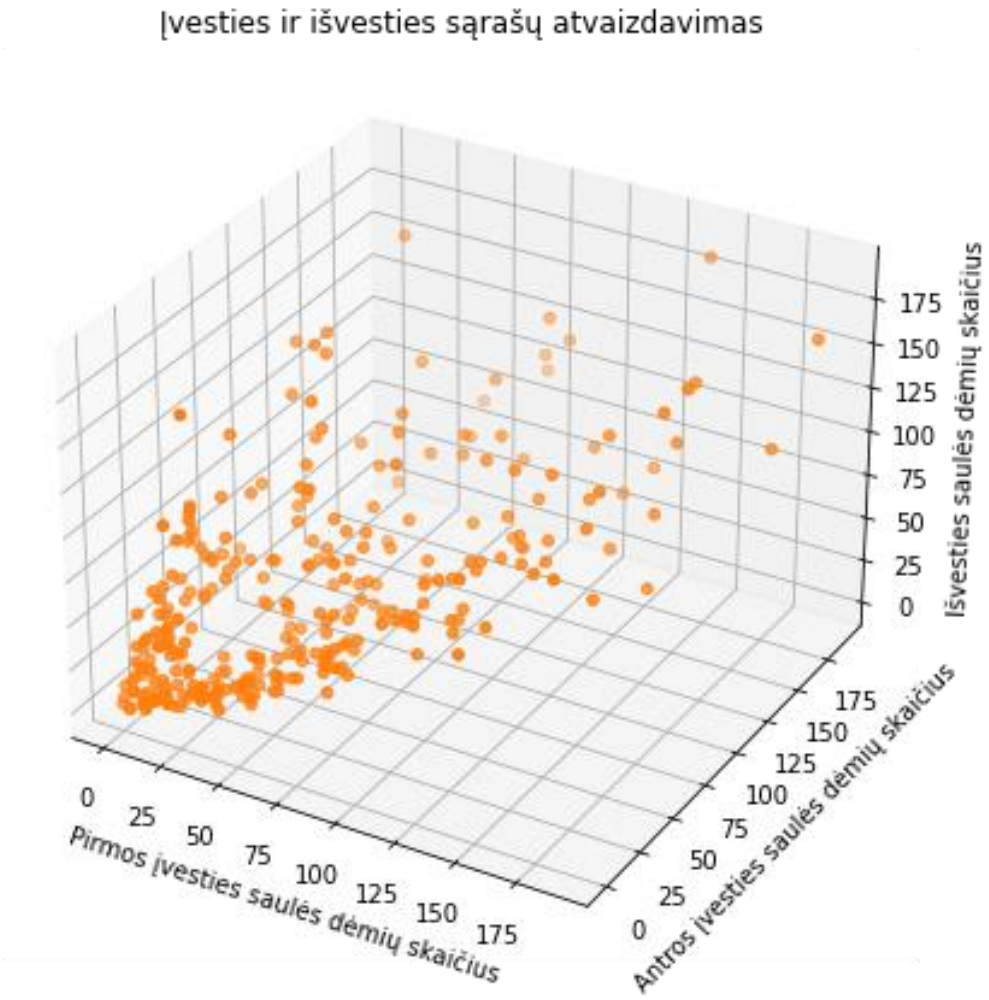
Pirma dalis

Saulės dėmių aktyvumo už 1700-2014 metus grafikas



Įvesties matricos ir išvesties vektoriaus trimatis grafikas

Kadangi autoregresinio modelio eilė yra lygi 2, įvestį ir išvestį galima pavaizduoti trimatyje grafike.



Dirbtinio neurono kūrimas

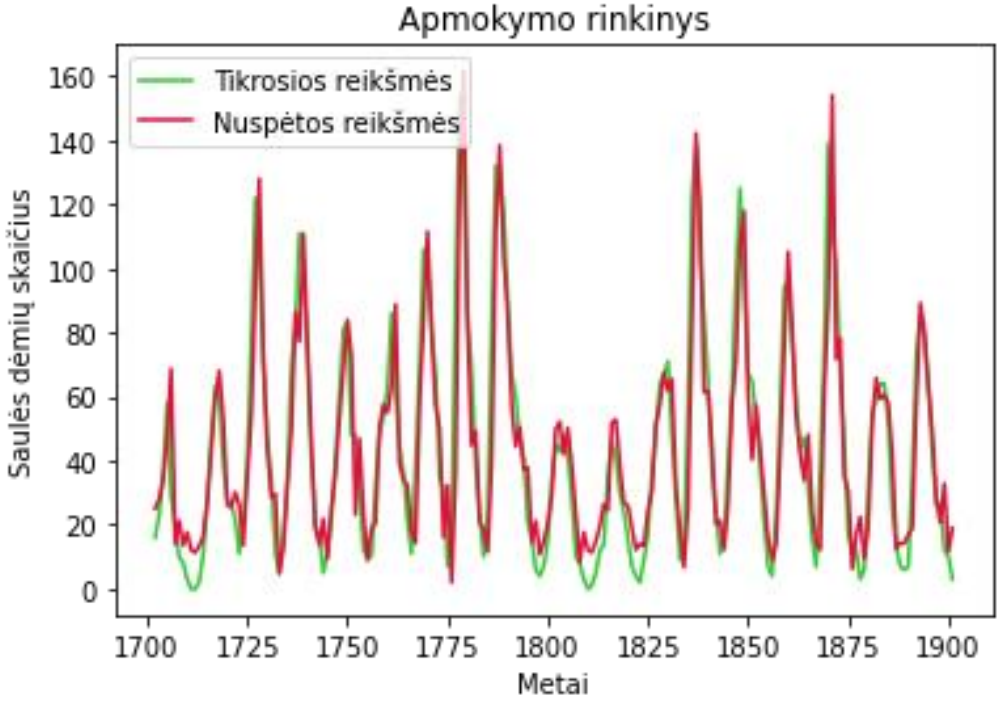
Neurono apmokymo tikrinimui duomenys yra išskiriami į dvi dalis. Apmokymo rinkiniui yra priskirti pirmieji 200 duomenų, o likusieji skirti patikrinti, ar neuronas gerai atspėjo reikšmes remdamasis mokymosi duomenimis.

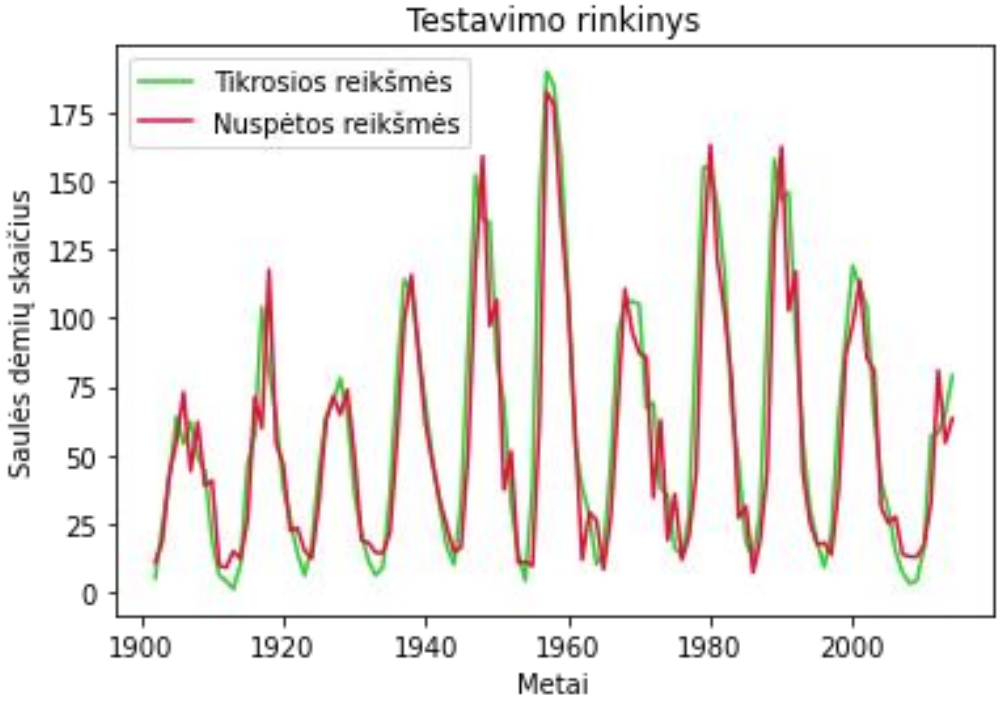
Neurono modelį sudaro dvi įvestys ir viena išvestis. Pradiniai svoriai parenkami atsitiktinai, o galutinius apskaičiuoja neuronas. Po tiesinio neurono apmokymo gaunamos tokios koeficientų reikšmės:

```
coefficient of determination: 0.8197788401382848
intercept (b): 13.403683236718116
slope: [-0.67608198  1.37150939]
```

Modelio verifikacija

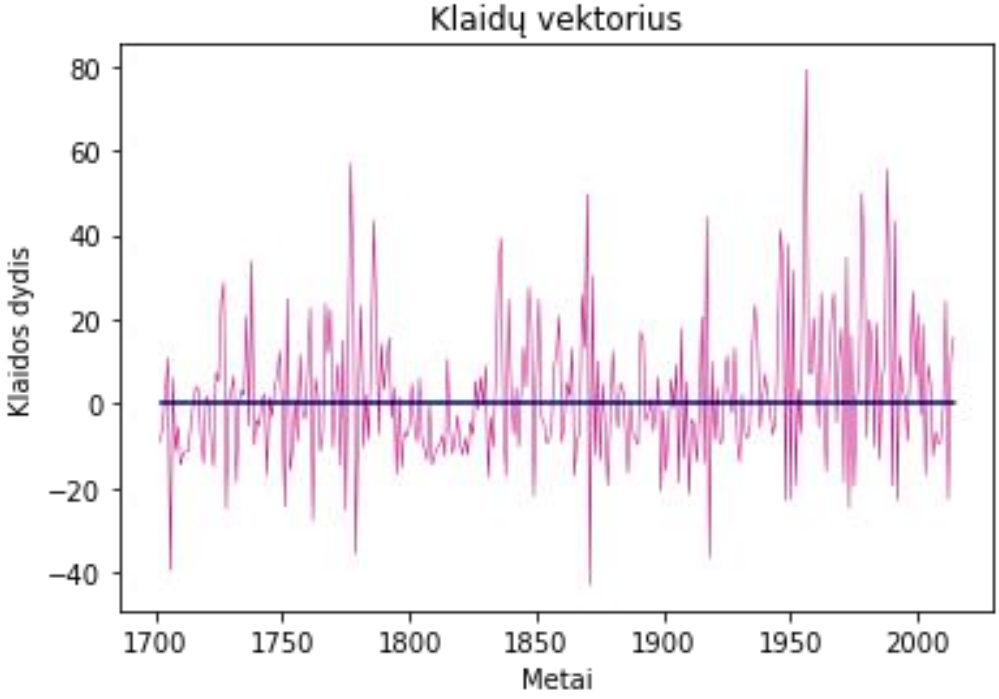
Brėžiame grafikus, jog galėtume pastebėti neurono prognozavimo kokybę. Pirmajame grafike tikriname tikrąsias dėmių aktyvumo reikšmes su prognozuojamomis reikšmėmis, o antrajame – su testavimo rinkiniu.

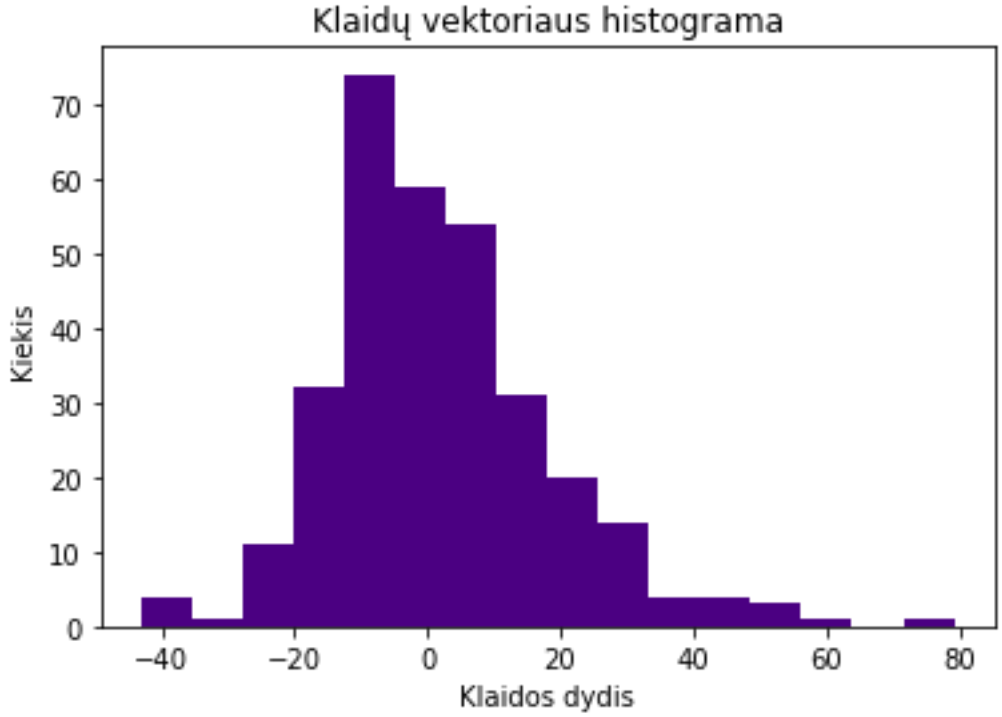




Galima pastebėti, jog antrojo grafiko nuspėtos reikšmės labiau skiriasi nuo tikrųjų nei pirmajame grafike.

Sukuriame prognozės klaidos vektorių e ir pavaizduojame jo grafiką bei histogramą.





Pagal šią histogramą galime matyti, jog didžiausias klaidų kiekis yra aplink nulį, tad neuronas nuspėja reikšmes pakankamai gerai.

MSE ir MAD skaičiavimas

Vidutinės kvadratinės prognozės klaidos reikšmei apskaičiuoti taikoma ši formulė:

$$MSE = \frac{1}{N}\sum_{k=1}^{N}\left(a(k) - \hat{a}(k)\right)^2 = \frac{1}{N}\sum_{k=1}^{N} e(k)^2$$

Gauta reikšmė $MSE = 278.26865758028515$

Prognozės absoliutaus nuokrypio medianai apskaičiuoti taikoma ši formulė:

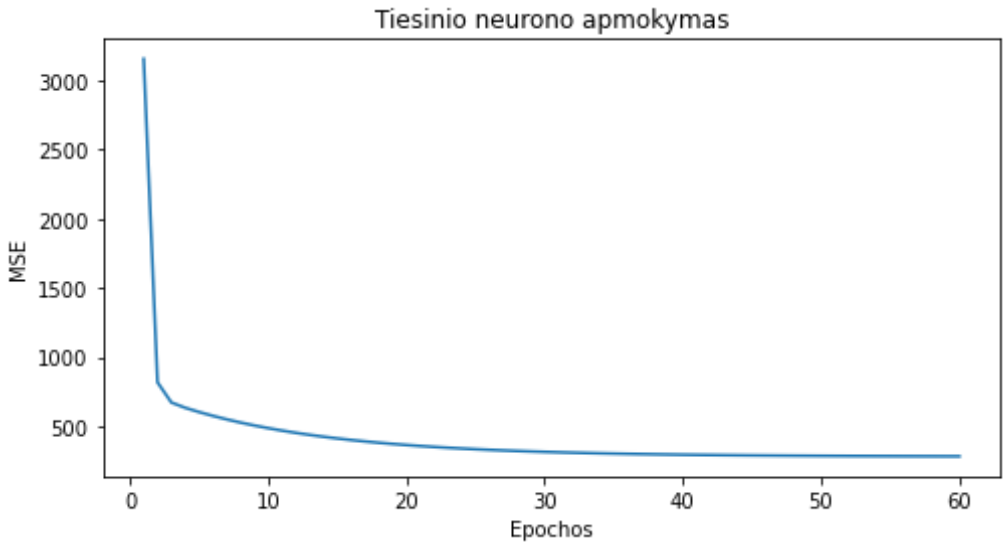
$$\mathrm{MAD} = median\ (|a(k) - \hat{a}(k)|)$$

Gauta reikšmė $MAD = 9.218889655548992$

MAD nurodo, jog vidurinė klaidos reikšmė yra maždaug 9. Skaičiuojant MAD, sudedame klaidų vektoriaus kvadratines reikšmes į bendrą sumą ir gautą reikšmę padaliname iš klaidų vektoriaus duomenų kiekio, tad atsakymas mums pateikia klaidų kvadratų vidurkį.

Neurono apmokymas iteraciniu būdu

Mokymosi greičio reikšmė keičiama, kol grafikas konverguoja.

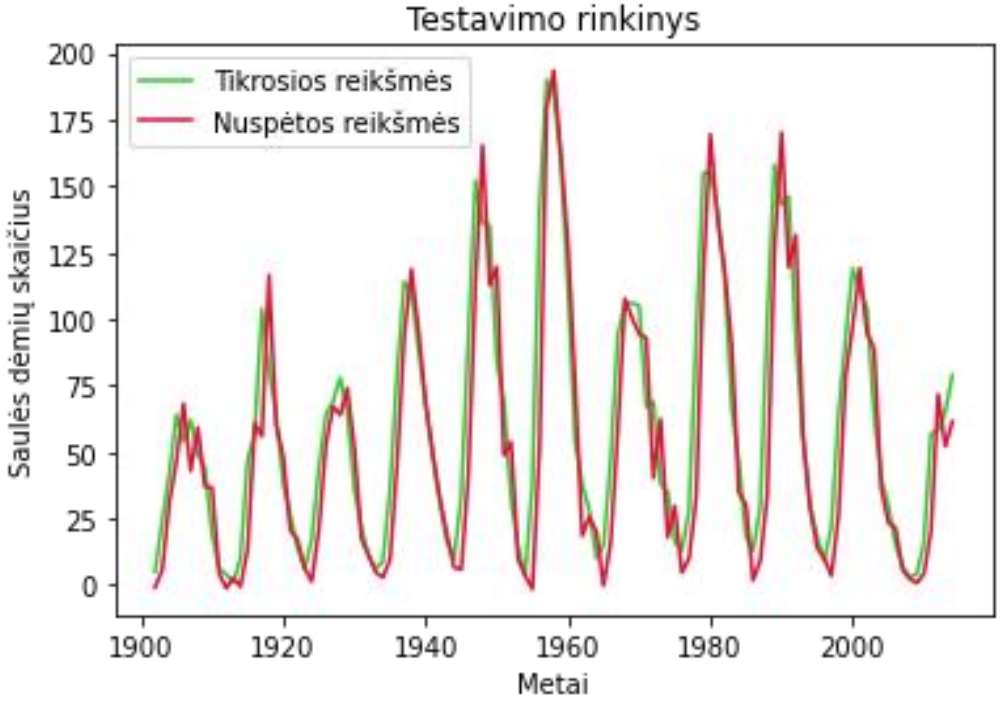


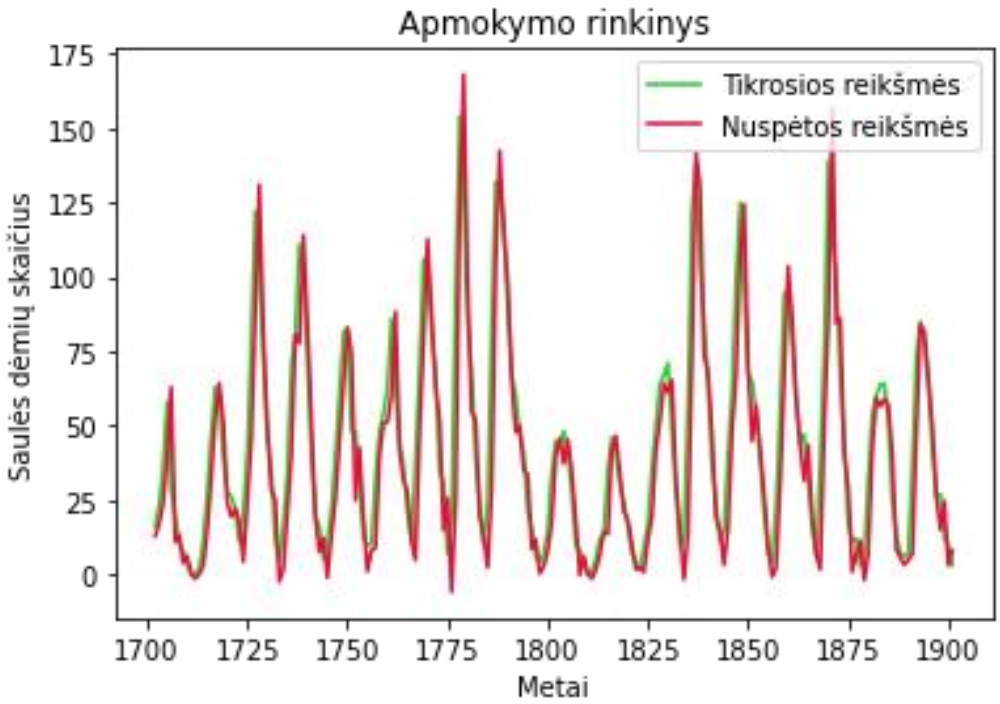
Pasirinkta mokymosi greičio reikšmė yra lygi 0.000001, o epochų kiekis – 60. Modelis įvykdomas ir gaunamos svorių koeficiento reikšmės po apmokymo:

```
array([ 0.06413375, -0.51083752,  1.39982092])
```

Neurono spėjimas su testiniais duomenimis

Neurono apmokymui buvo paskirtas 200 duomenų rinkinys, o testavimui – likusieji duomenys. Neurono rezultatų spėjimą pavaizduojame grafikuose:





Atsakymai į klausimus

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

Mokymosi procesas yra konverguojantis. Jei procesas būtų nekonverguojantis, tai būtų dėl dviejų parametrų reikšmių – mokymosi greičio bei epochų kiekio. Mokymosi greičio reikšmė turi būti tarp nulio ir vieneto. Epochų kiekis turi būti ne per didelis, nes antraip modelis „persimokys“ ir spėjamos reikšmės neturės daug naudos. Esant nekonverguojančiam procesui, keičiame šių dviejų parametrų reikšmes, kol procesas taps konverguojantis.

- Kokios yra naujos neurono svorių koeficientų reikšmės?

Naujos neurono svorių koeficientų reikšmės:

```
array([ 0.06413375, -0.51083752,  1.39982092])
```

- Kokia yra neurono darbo kokybės įverčio MSE ir MAD reikšmės?

$MSE = 278.26865758028515,$
$MAD = 9.218889655548992.$

Modelio struktūros keitimas

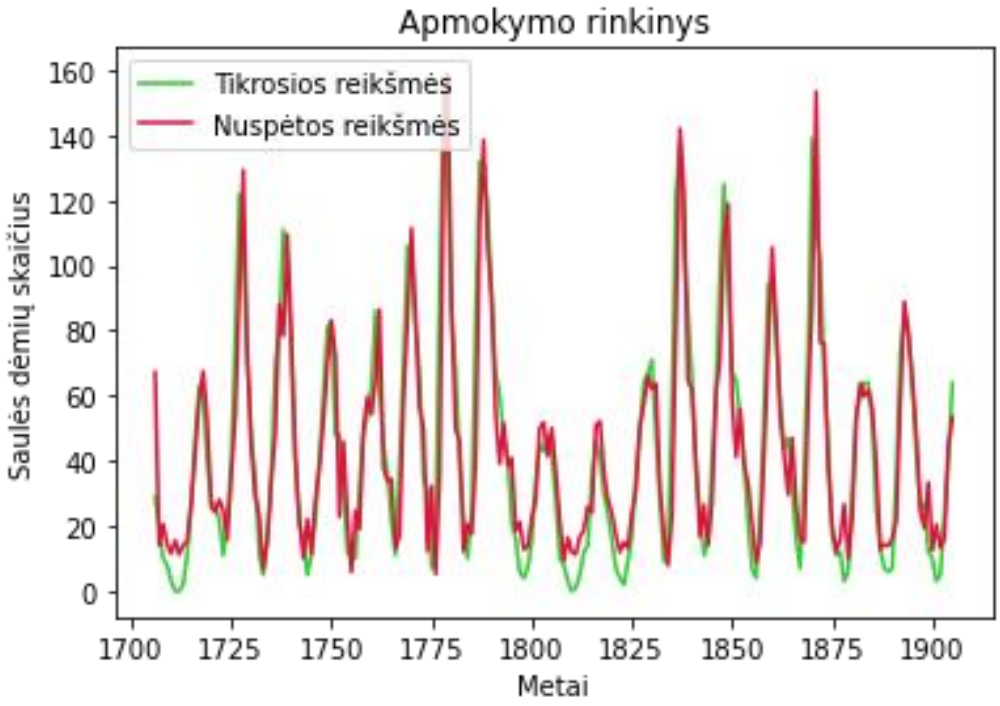
Keičiama autoregresinio modelio eilė. Uždavinio sprendimas aukščiau buvo pavaizduotas antrai eilei. Pavaizduokime modelį, kai jo eilė yra lygi 6 bei 10 ($n = 6; n = 10$).

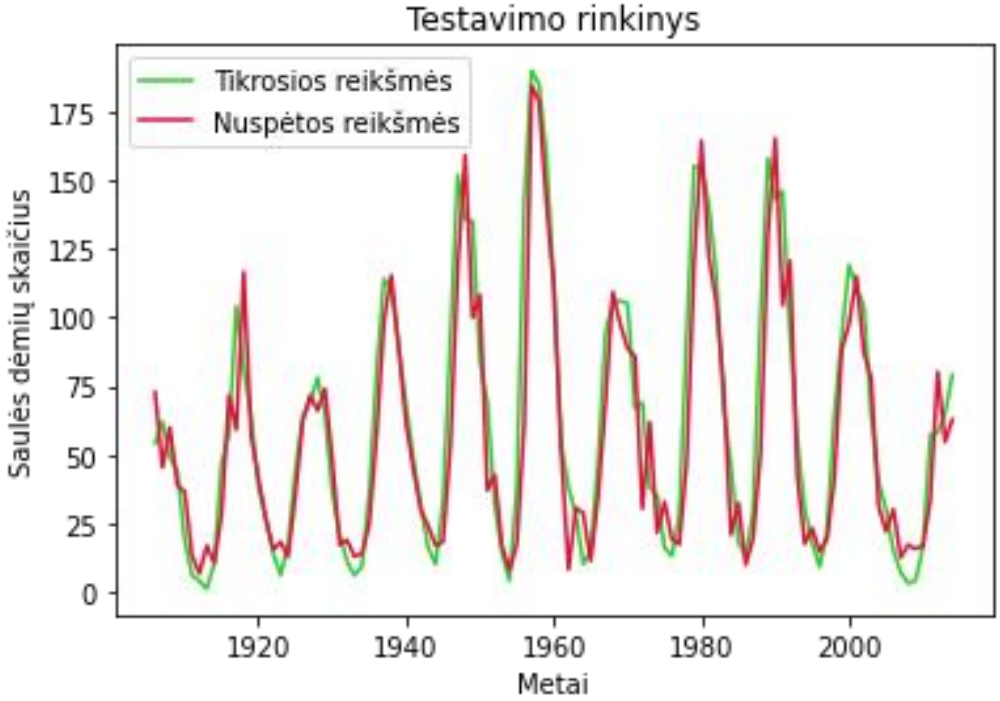
n = 6

Optimalios neurono svorių koeficientų reikšmės:

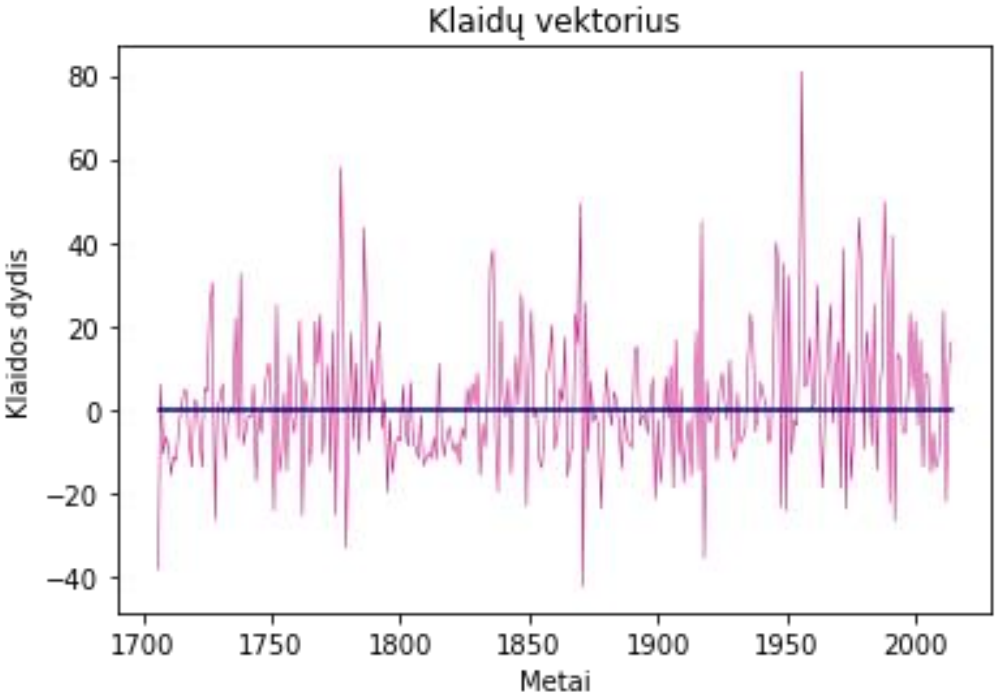
```
coefficient of determination: 0.8254213353035943
intercept: 12.487115307483776
slope: [ 0.15341921 -0.23942144  0.12574875 -0.03091404 -0.64242115  1.35102406]
```

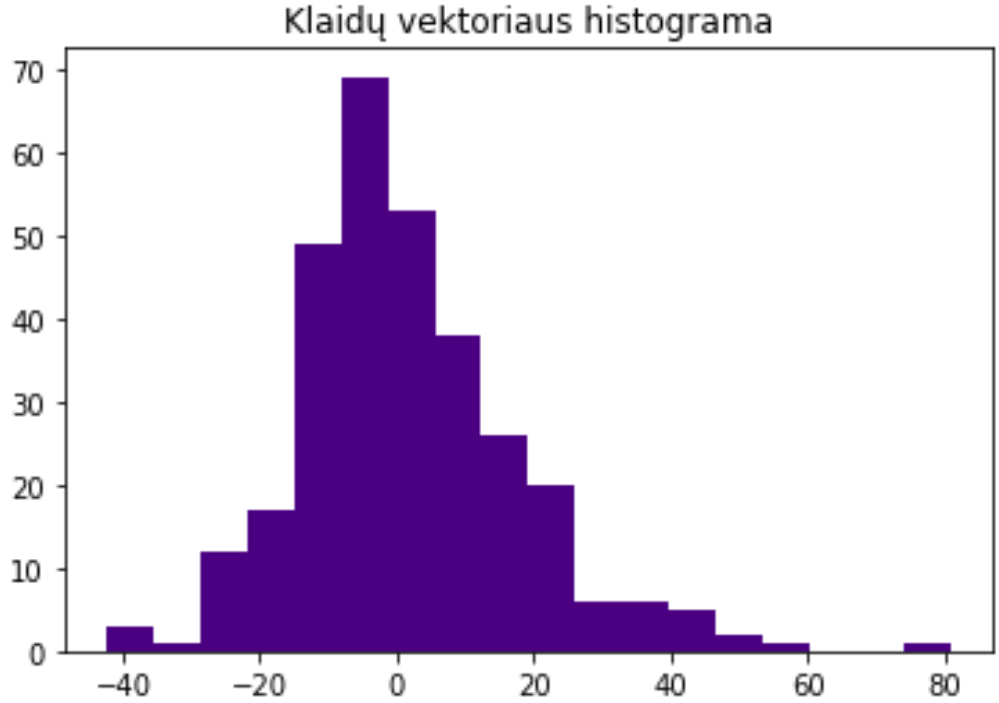
Verifikuojame modelį brėždami grafikus:



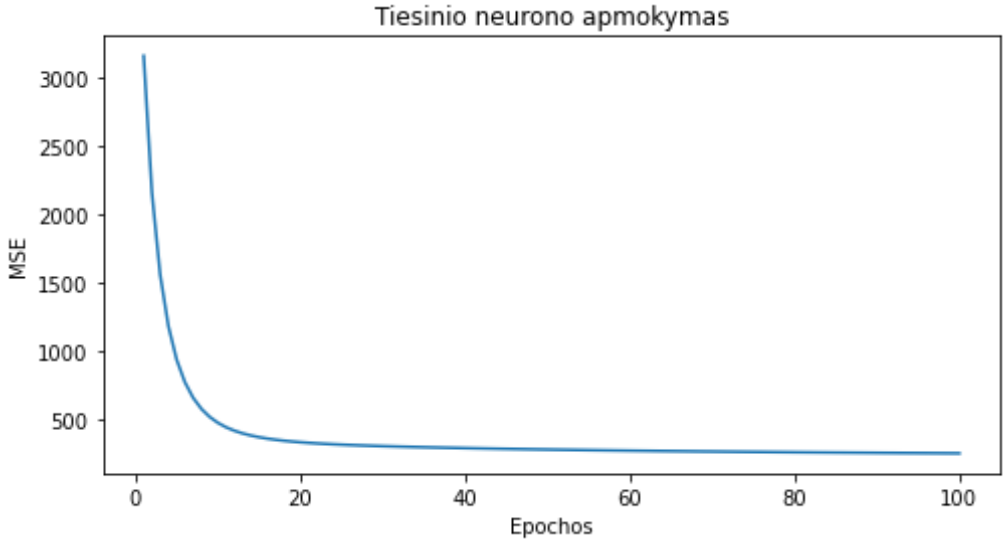


Sukuriame prognozės klaidos vektorių e ir pavaizduojame jo grafiką bei histogramą:





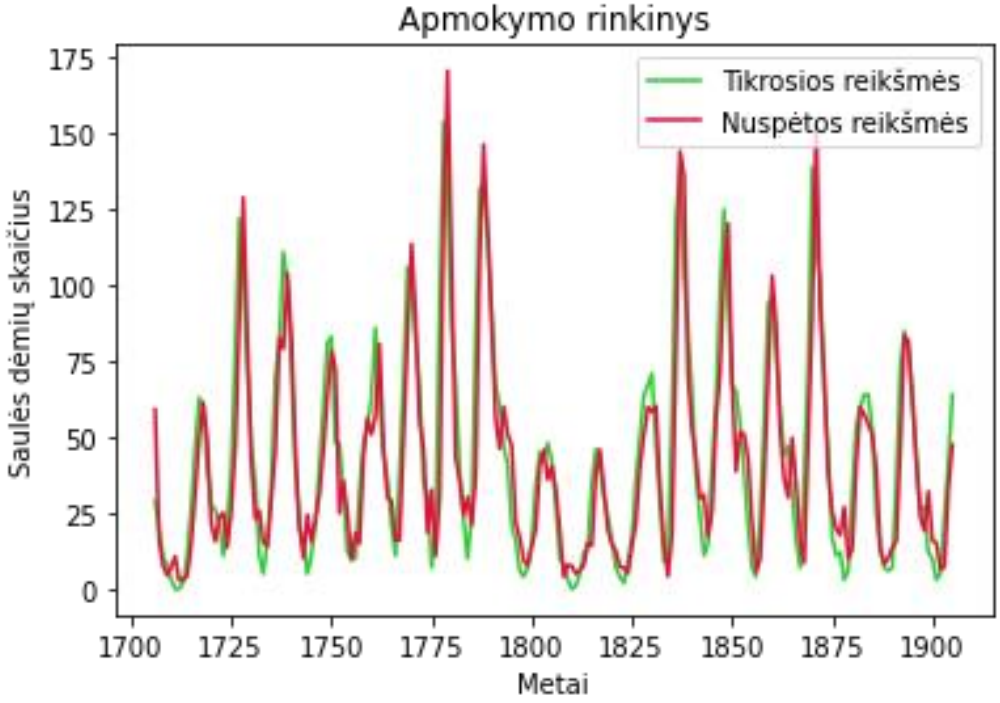
Gauta reikšmė $MSE = 271.0407045969269$
Gauta reikšmė $MAD = 8.955843546367664$
Galima pastebėti, jog šios reikšmės yra mažesnės nei antros eilės modelio. Vadinasi, šis modelis yra šiek tiek tikslesnis.

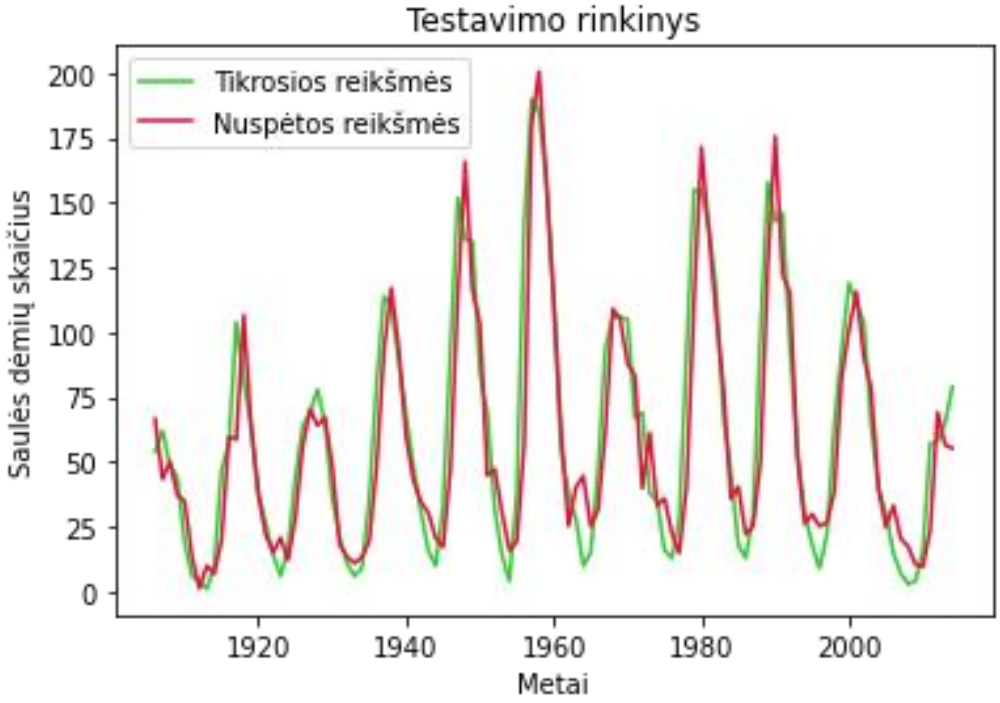


Svorių koeficiento reikšmės po apmokymo:

```
array([ 0.0392712 ,  0.22418354, -0.07615967,  0.00951886, -0.21724571,
       -0.23596389,  1.24342034])
```

Neurono rezultatų spėjimo grafikai:



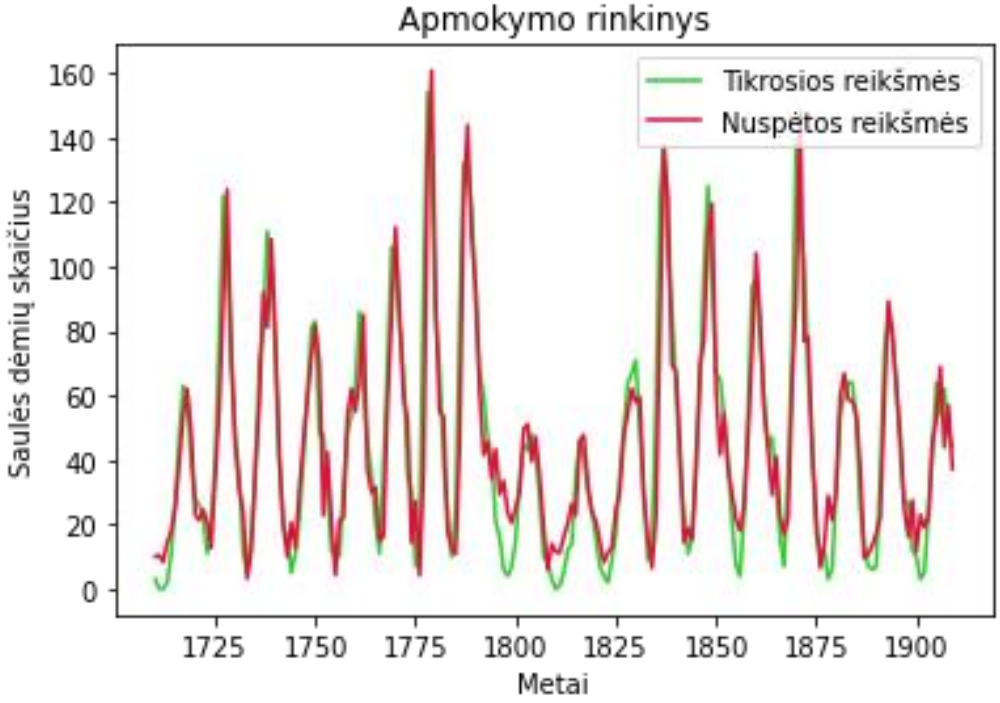


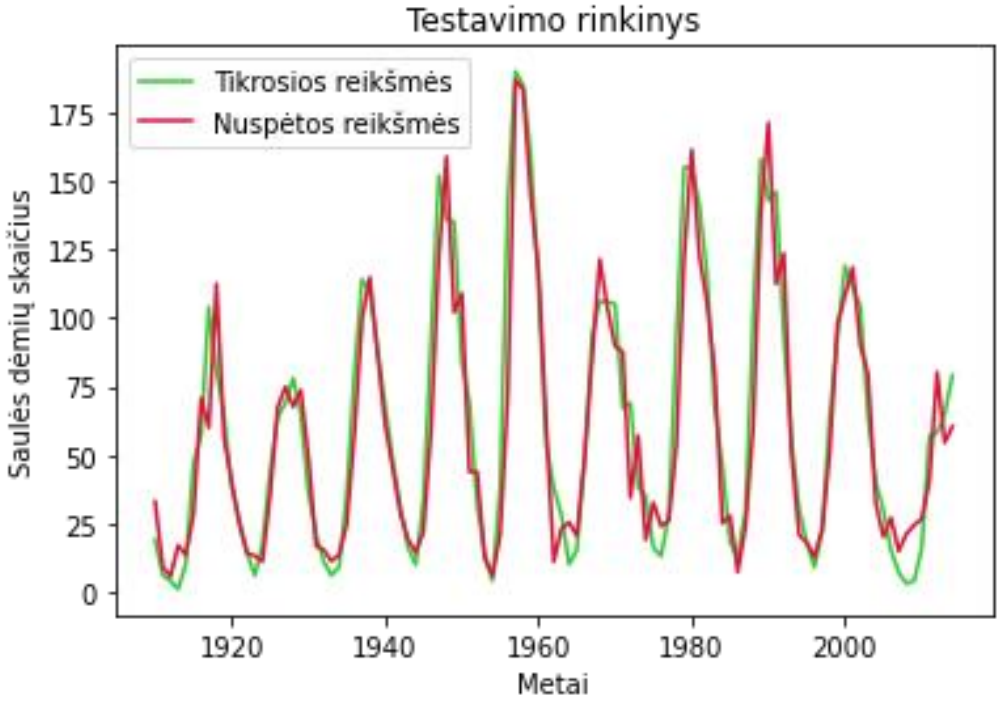
n = 10

Optimalios neurono svorių koeficientų reikšmės:

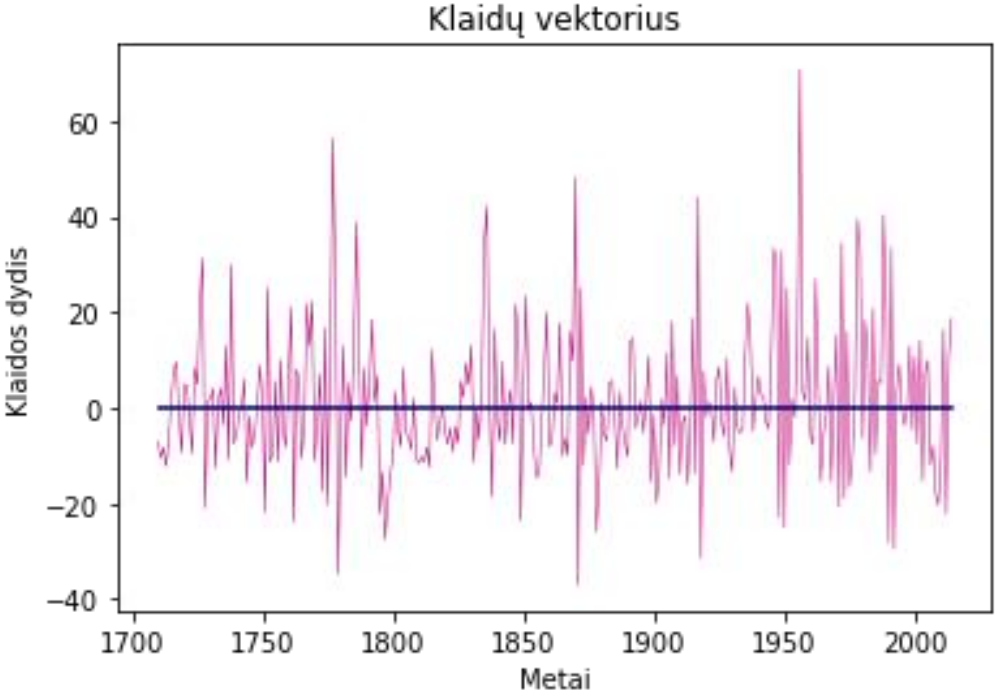
```
coefficient of determination: 0.8402037092409227
intercept: 8.11011652242965
slope: [ 0.01127462  0.11401852  0.03402009 -0.03176102  0.06298231 -0.15371419
  0.14257871 -0.05173529 -0.57435326  1.26827984]
```

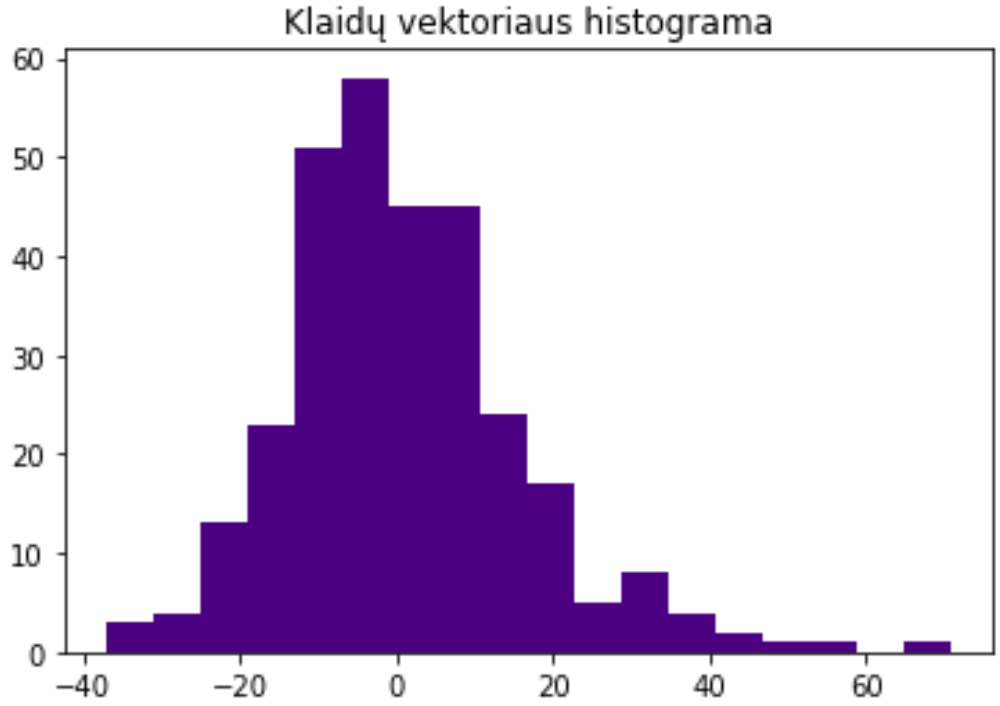
Verifikuojame modelį brėždami grafikus:



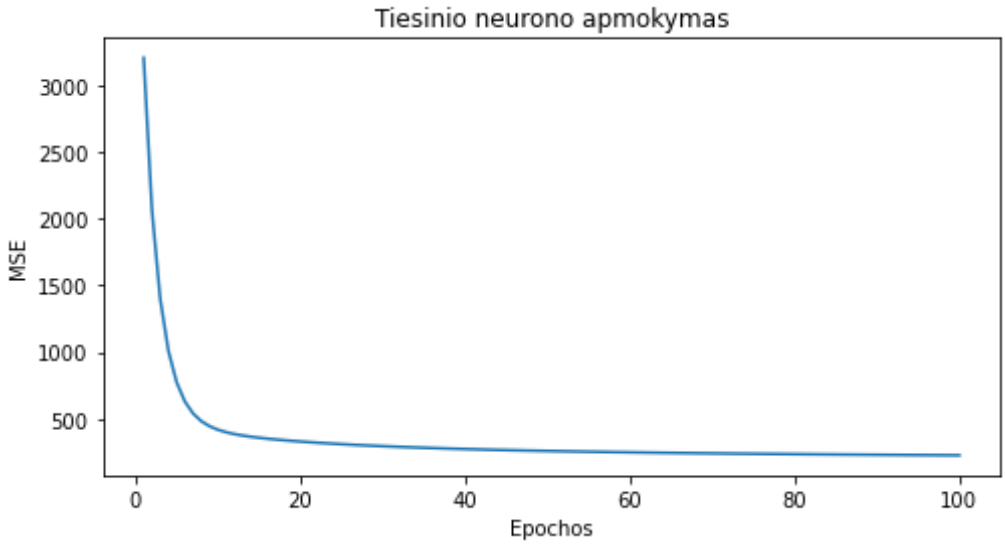


Sukuriame prognozės klaidos vektorių e ir pavaizduojame jo grafiką bei histogramą:





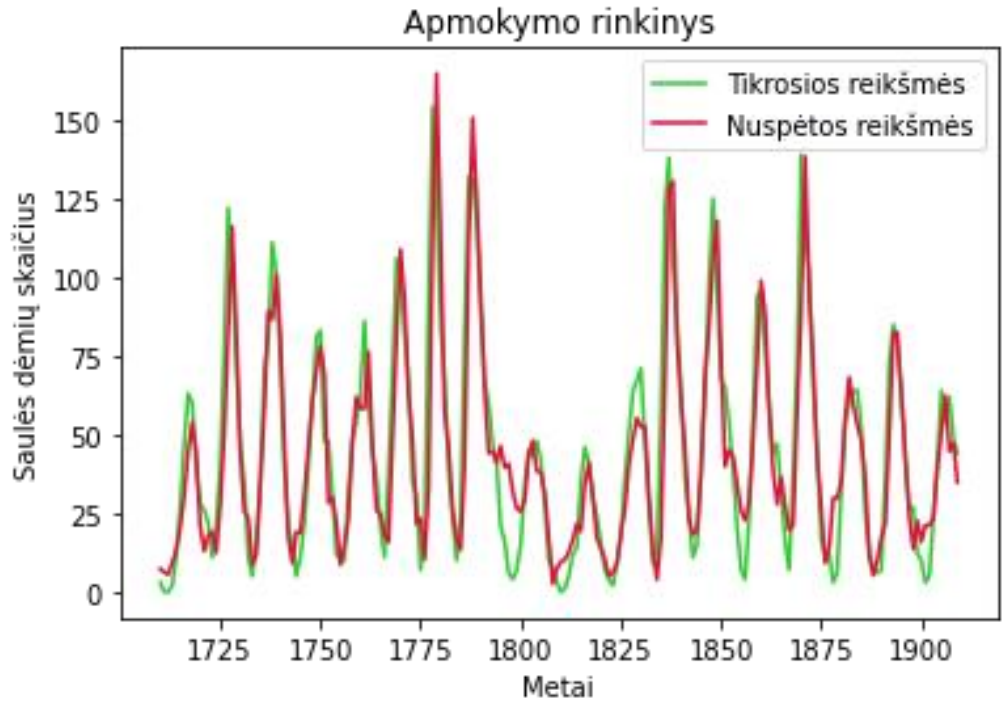
Gauta reikšmė $MSE = 232.26342820195603$
Gauta reikšmė $MAD = 8.608355711023776$
Galima pastebėti, jog šios reikšmės yra taip pat mažesnės nei antros eilės modelio. Skirtumas nėra mažas, tad galime daryti išvadą, jog kuo didesnė autoregresinio modelio eilė, tuo tikslesnis yra modelis.

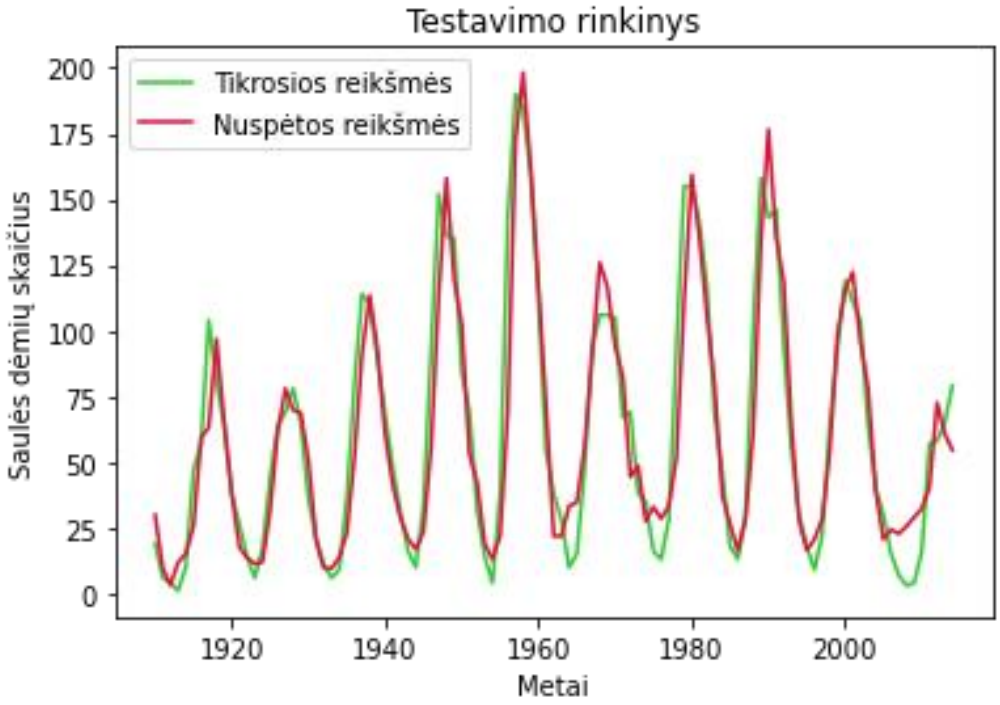


Svorių koeficiento reikšmės po apmokymo:

```
array([ 0.0125193 ,  0.10189482,  0.09364153,  0.05028534,  0.01429586,
        0.02189151,  0.02376252, -0.01344267, -0.20614669, -0.07197295,
        0.96610203])
```

Neurono rezultatų spėjimo grafikai:





Iš trijų modelių grafikų ir rezultatų reikšmių pastebime, jog modeliai per daug nesiskiria. Vis dėlto modelis, kurio eilė yra lygi 10 ($n = 10$) tiksliausiai nuspėjo duomenis.

Antra dalis

Pradinio duomenų rinkinio aprašymas

Tolydinio tipo atributai:

Atributo pavadinimas	Kiekis (Eilučių sk.)	Trūkstamos reikšmės, %	Kardinalumas	Minimali reikšmė	Maksimali reikšmė	1-asis kvartilis	3-asis kvartilis	Vidurkis	Mediana	Standartinis nuokrypis
Math score	1000	0	81	0	100	57	77	66.089	66	15.16308
Reading score	1000	0	72	17	100	59	79	69.169	70	14.600192
Writing score	1000	0	77	10	100	57.75	79	68.054	69	15.195657
Average score	1000	0	194	9	100	58.33	77.67	67.77058	68.33	14.257311

Kategorinio tipo atributai:

Atributo pavadinimas	Kiekis (Eilučių sk.)	Trūkstamos reikšmės, %	Kardinalumas	Moda	Modos dažnumas	Moda, %	2-oji Moda	2-osios Modos dažnumas	2-oji Moda, %
Gender	1000	0	2	female	518	51.8	male	482	48.2
Race/ethnicity	1000	0	5	group C	319	31.9	group D	262	26.2
Parental level of education	1000	0	6	some college	226	22.6	associate‘s degree	222	22.2

Lunch	1000	0	2	standard	645	64.5	free/reduced	355	35.5
Test preparation course	1000	0	2	none	642	64.2	completed	358	35.8

Pertvarkymų su duomenų rinkiniu atlikta nebuvo.

Pasirinktas tikslo atributas „Math score“, kuris apibūdina studento rezultatą matematikos teste. Šį atributą suklasifikuojame į 4 intervalus pagal naujai sudarytą atributą („mathLvl“):

```
df = pd.read_csv("StudentsPerformance.csv")
df["mathLvl"] = "low"
df.loc[df["math score"]>=25, 'mathLvl'] = "med"
df.loc[df["math score"]>=50, 'mathLvl'] = "high"
df.loc[df["math score"]>=75, 'mathLvl'] = "top"
`df
```

Atliksime du šio modelio eksperimentus.

Pirmasis eksperimentas

```
def baseline_model():
    model = Sequential()
    model.add(Dense(14, input_dim=2, activation='relu'))
    model.add(Dense(8, activation='tanh'))
    model.add(Dense(4, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model
```

Modelį sudaro keturi sluoksniai. Pirmajame įvesties sluoksnyje yra dvi įvestys, o jo aktyvacijos funkcija yra „ReLU“. Antrąjį „hidden“ sluoksnį sudaro 14 neuronų, o jo aktyvacijos funkcija yra „ReLU“. Trečiąjį „hidden“ sluoksnį sudaro 8 neuronai, o jo aktyvacijos funkcija yra hiperbolinis tangentas. Ketvirtąjį išvesties sluoksnį sudaro 4 neuronai, o jo aktyvacijos funkcija yra „softmax“. Epochų kiekis – 40.

10 intervalų kryžminės patikros eksperimentų rezultatai:

```
array([0.7 , 0.62, 0.66, 0.72, 0.69, 0.59, 0.72, 0.57, 0.68, 0.67])
```

Eksperimentų rezultatų vidurkis: 62,9%

Antrasis eksperimentas

Siekdami gauti geresnį rezultatą atliekame modelio pakeitimus ir eksperimentą pakartojame.

```
def baseline_model2():
    model = Sequential()
    model.add(Dense(10, input_dim=2, activation='relu'))
    model.add(Dense(8, activation='relu'))
    model.add(Dense(4, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy'])
    return model
```

Keičiame antrojo „hidden“ sluoksnio neuronų kiekį iš 14 į 10 bei trečiojo „hidden“ sluoksnio aktyvacijos funkciją iš hiperbolinio tangento į „ReLU“. Epochų kiekio nekeičiame.

10 intervalų kryžminės patikros eksperimentų rezultatai:

```
array([0.68, 0.74, 0.68, 0.72, 0.67, 0.66, 0.75, 0.74, 0.72, 0.73])
```

Eksperimentų rezultatų vidurkis: 70,9%

Išvados

Iš pirmojo laboratorinio darbo duomenų rinkinio pasirinkome atributą „Math Score“, kurį suklasifikavome ir atlikome du modelio eksperimentus. Pirmojo eksperimento metu rezultatų tikslumą gavome 62,9%. Atlikę modelio keitimus rezultatų tikslumą pakėlėme aštuoniais procentais. Geresnio rezultato siekėme keisdami sluoksnių neuronų kiekį bei aktyvacijos funkcijas.