VILNIAUS UNIVERSITETAS

MATEMATIKOS IR INFORMATIKOS FAKULTETAS

DUOMENŲ MOKSLAS. BAKALAURAS

Tiesioginio sklidimo DNT naudojant sistemą WEKA

Praktinė užduotis Nr. 3

Užduotį atliko: Ugnius Vilimas 3 kursas, 2 grupė

VU el. paštas: ugnius.vilimas@mif.stud.vu.lt

2023

Turinys

Tikslai

- Išmokyti neuroninį tinklą teisingai klasifikuoti duomenis naudojant sistemą WEKA.
- Sukonstruoti neuroninį tinklą „Excel“ programoje.

Uždaviniai

- Paruošti duomenis programai „WEKA“, paskirstyti juos į du failus mokymui ir testavimui.
- Programoje „WEKA“ sukonstruoti reikiamas užduočių sekas.
- Gautus duomenis sudėti į lenteles.
- „Excel“ programoje sukonstruoti neuroninį tinklą.
- Gauti rezultatus „Excel“ programoje ir palyginti juos su „WEKA“ programos rezultatais.

Duomenys

Užduočiai atlikti buvo naudojami irisų duomenys, kurie buvo išsaugoti kaip *.arff* failas įdiegus sistemą „WEKA“. Tada iš failo pasidariau 2 failus, kur viename buvo testavimo duomenys, kitame duomenys mokymui. Mokymo duomenys turėjo 80 % visų duomenų, tai buvo 120 duomenų eilučių, po 40 duomenų visoms klasėms. Likę duomenys buvo testavimui, juose buvo 30 duomenų eilučių, po 10 duomenų visoms klasėms. **Gautas užduoties variantas – 1,** todėl buvo panaikintas pirmasis stulpelis duomenų „sepallength“. Duomenys buvo pilni, nebuvo trūkstamų reikšmių.

Svarbios formulės

1. Įėjimo reikšmių ir svorių sandaugų sumos formulė: $a_i = \sum_{k=0}^{n} w_{ik} x_k$.

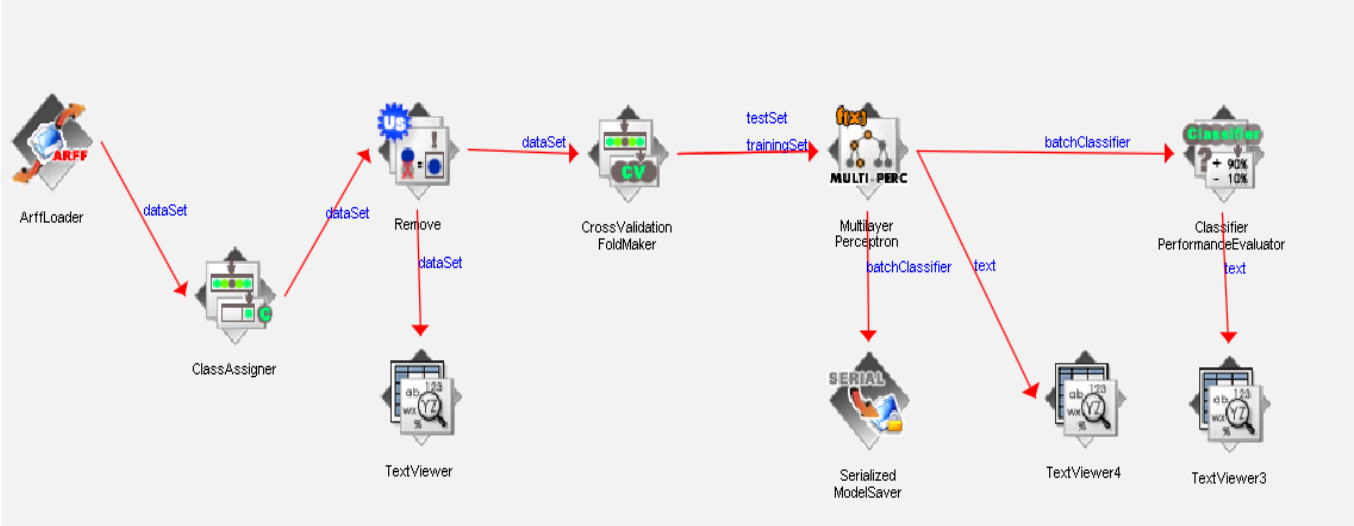
2. Sigmoidinė funkcija: $f(a) = \frac{1}{1+e^{-a}}$.

3. Normalizavimo formulė (tam, kad gauti svorius nuo -1 iki 1):

$$x_{ij} \leftarrow \frac{2x_{ij} - min_{\left(x_1, x_2, \ldots, x_{mj}\right)} - max_{(x_1, x_2, \ldots, x_{mj})}}{max_{(x_1, x_2, \ldots, x_{mj})} - min_{\left(x_1, x_2, \ldots, x_{mj}\right)}}.$$

Programos „Weka“ pirmoji užduočių seka

1 pav. yra pavaizduota užduočių seka, kuri buvo įvykdyta naudojant iris mokymo duomenis pavadinimu *iris_train_test.arff.* su *ArffLoader* pagalba, mes įkeliame mokymo duomenis arff formatu. Tada naudojant *ClassAssigner* nurodysime, kad mūsų klasė yra nurodyta paskutiniame duomenų stulpelyje. *Remove* komponentė mums padeda pašalinti nereikalingus stulpelius iš duomenų, mūsų atveju šalinome pirmąjį stulpelį. *CrossValidationFoldMaker* yra naudojamas atlikti kryžminę patikrą. *Multilayer Perceptron* komponentjje, mes keičiame parametrus, kurie padės geriausiai išmokti klasifikuoti duomenis. Juose keisime tokius parametrus kaip: mokymo greitis, neuronų skaičius ar *momentum*. *Serialized ModelSaver* mums leidžia išsaugoti gautą modelį, tam, kad panaudotumėm jį antrajai sekai. *Classifier PerformanceEvaluator* mums leidžia pamatyti klasifikavimo rezultatų tikslumą, tačiau, tam, kad juos pamatyti, reikia pridėti komponentes *TextViewer* kuriuos paspaude, ekrane matysime visą reikiamą informaciją, apie mokymą.



1 pav. Tinklo mokymo seka WEKA.

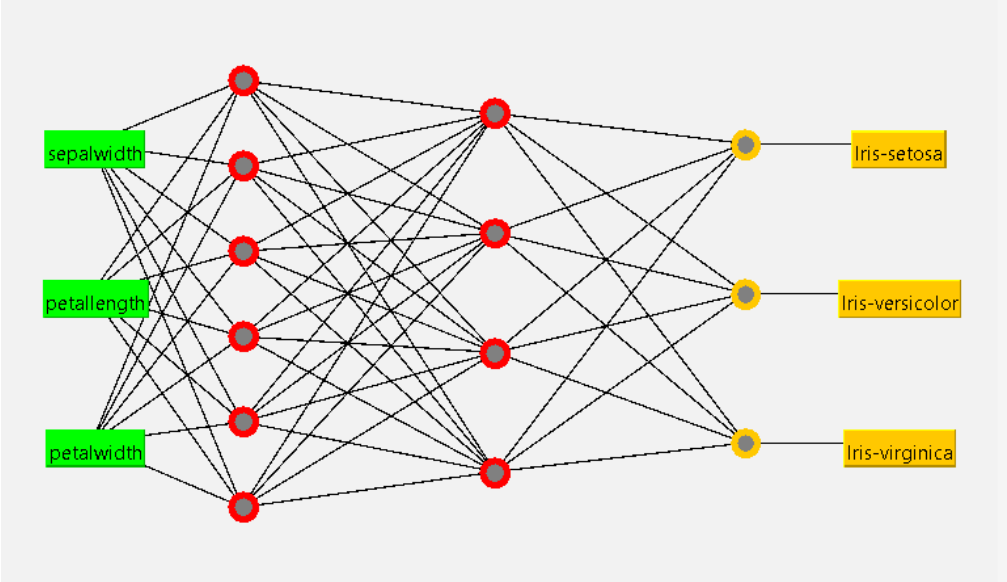
1 lentelėje yra pateikiami skirtingai parametrai, tokie kaip sluoksnių skaičius, perceptronų skaičius, mokymo greitis ir *momentum* pagal kuriuos vėliau yra gaunamas teisingai klasifikuotų duomenų skaičius iš 120. Pasirinkau rezultatus nagrinėti pagal 1 arba 2 paslėptus sluoksnius. Vėliau juos skirsčiau į perceptronų skaičių, viename sluoksnyje ėmiau 4 arba 6, dviejuose 8 (5 ir 3) arba 10 (6 ir 4). Mokymo greičius pasirinkau nagrinėti 3, jie buvo: 0.1, 0.3 ir 0.5. O reikšmė *momentum* buvo pasirinkta 0.2 ir 0.4. Tikslumas galėjo būti nuo 0 iki 120.

1 lentelė. Klasifikavimo tikslumas su vienu ir dviem paslėptais neuronų sluoksniais.

Paslėpti sluoksniai	Perceptronų skaičius	Mokymo greitis	*Momentum*	Teisingai klasifikuotų duomenų skaičius (iš 120)
1	4	0,1	0,2	115
		0,3	0,2	114
		0,5	0,2	113
		0,1	0,4	115
		0,3	0,4	113
		0,5	0,4	113
	6	0,1	0,2	115
		0,3	0,2	114
		0,5	0,2	112
		0,1	0,4	115
		0,3	0,4	113
		0,5	0,4	113
2	8 (5 ir 3)	0,1	0,2	114
		0,3	0,2	114
		0,5	0,2	115
		0,1	0,4	113
		0,3	0,4	114
		0,5	0,4	114
	10 (6 ir 4)	0,1	0,2	113
		0,3	0,2	113
		0,5	0,2	114
		0,1	0,4	114
		0,3	0,4	114
		0,5	0,4	113

Iš lentelės matome, kad beveik visi geriausi rezultatai buvo ties vieno paslėpto sluoksnio dalimi, geriausi klasifikavimo rezultatai gavosi 115 iš 120, kad dar tikrai nėra tobula, tačiau jau arti maksimumo. Galime aiškiai pastebėti, kad buvus bet kokiems perceptronų skaičiams ir *momentum* reikšmėms, visada geriausi rezultatai gavosi su 0.1 mokymo greičiu, tad galime teigti, jog *momentum* parametras mūsų atveju darė mažiau įtakos, negu, kad mokymo greitis. Vis keliant mokymo greitį, pastebime ir blogiausią klasifikavimo rezultatą lentelėje, kuris buvo pasiektas tada, kad mokymo greitis buvo 0.5, o *momentum* – 0.2. Esant dviem paslėptiems sluoksniams, rezultatai buvo gana vienodi, dažniausiai 113 arba 114, tačiau išsiskyrė vienas atvejis, kai mokymo greitis buvo 0.5, o *momentum* – 0.2 mes gauname geriausią klasifikavimo rezultatą, nors su tokiais parametrais, esant vienam sluoksniui, jis buvo blogiausias. Galime padaryti trumpą išvadą, kad esant dviem sluoksniams, mūsų rezultatai gavosi stabilesni, tai yra, jų amplitudė buvo mažesnė.

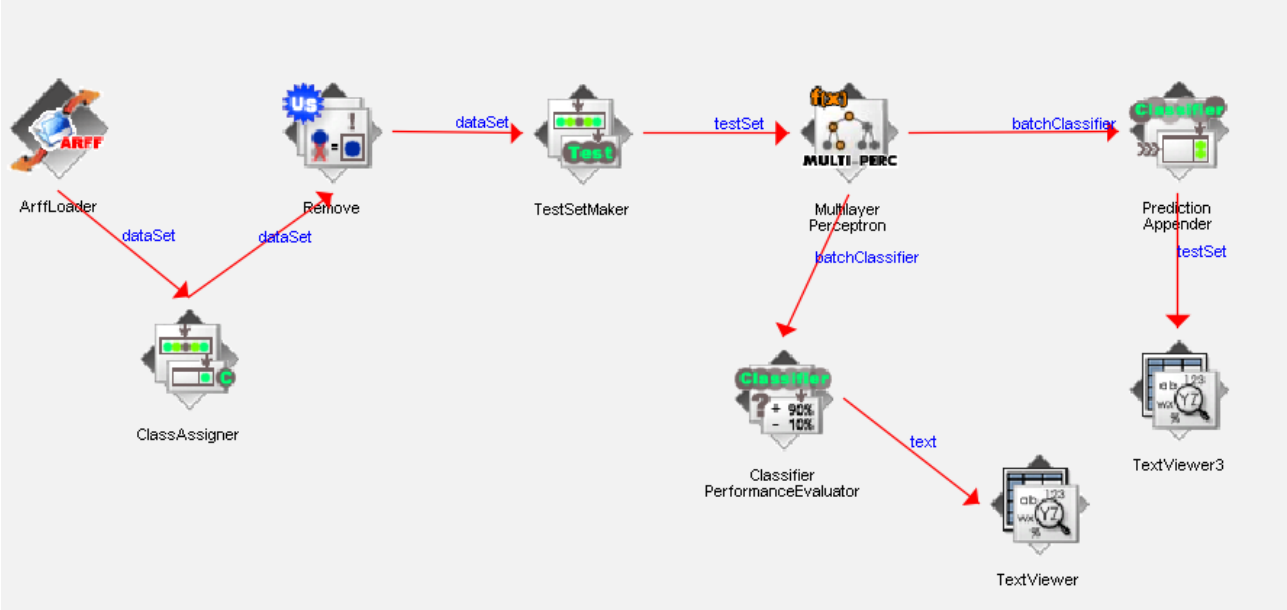
2 pav. yra pavaizduota mūsų atveju naudojamas dirbtinio neuroninio tinklo modelis, kuris turi 2 sluoksnius, kurie turi po 6 ir 4 perceptronus atitinkamai. Matome, kad duomenų parametras „sepallength“ yra išmestas, pagal priskirtą užduoties sąlygą.



2 pav. Dirbtinio neuroninio tinklo modelis.

Programos „Weka“ antroji užduočių seka

Antroje užduočių sekoje yra naudojami testavimo duomenys, kurie vadinosi *iris_new.arff.* Ši seka ir užduotis buvo sudaryta sukurtu tinklo modeliu iš pirmosios užduoties. 3 pav. yra pateikta visa seka. Vienodai kaip ir pirmoje sekoje, galime matyti *ArffLoader, ClassAssigner, Remove, Multilayer Perceptron, Classifier PerformanceEvaluator, TextViewer* užduotis. *TestSet Maker* duomenis priskiria testavimui. Tada turime nurodyti jau išsaugotą klasifikatoriaus modelį. Jį reikia įkelti į *Multilayer Perceptron*. Tada *Prediction Appender* galima nurodyti, ar bus rodomos klasių priskyrimo tikimybės ar iškarto spėjimas.



3 pav. Tinklo testavimo seka WEKA.

```
@data
3.5,1.3,0.3,Iris-setosa,Iris-setosa
2.3,1.3,0.3,Iris-setosa,Iris-setosa
3.2,1.3,0.2,Iris-setosa,Iris-setosa
3.5,1.6,0.6,Iris-setosa,Iris-setosa
3.8,1.9,0.4,Iris-setosa,Iris-setosa
3,1.4,0.3,Iris-setosa,Iris-setosa
3.8,1.6,0.2,Iris-setosa,Iris-setosa
3.2,1.4,0.2,Iris-setosa,Iris-setosa
3.7,1.5,0.2,Iris-setosa,Iris-setosa
3.3,1.4,0.2,Iris-setosa,Iris-setosa
2.6,4.4,1.2,Iris-versicolor,Iris-versicolor
3,4.6,1.4,Iris-versicolor,Iris-versicolor
2.6,4,1.2,Iris-versicolor,Iris-versicolor
2.3,3.3,1,Iris-versicolor,Iris-versicolor
2.7,4.2,1.3,Iris-versicolor,Iris-versicolor
3,4.2,1.2,Iris-versicolor,Iris-versicolor
2.9,4.2,1.3,Iris-versicolor,Iris-versicolor
2.9,4.3,1.3,Iris-versicolor,Iris-versicolor
2.5,3,1.1,Iris-versicolor,Iris-versicolor
2.8,4.1,1.3,Iris-versicolor,Iris-versicolor
3.1,5.6,2.4,Iris-virginica,Iris-virginica
3.1,5.1,2.3,Iris-virginica,Iris-virginica
2.7,5.1,1.9,Iris-virginica,Iris-virginica
3.2,5.9,2.3,Iris-virginica,Iris-virginica
3.3,5.7,2.5,Iris-virginica,Iris-virginica
3,5.2,2.3,Iris-virginica,Iris-virginica
2.5,5,1.9,Iris-virginica,Iris-virginica
3,5.2,2,Iris-virginica,Iris-virginica
3.4,5.4,2.3,Iris-virginica,Iris-virginica
3,5.1,1.8,Iris-virginica,Iris-virginica
```

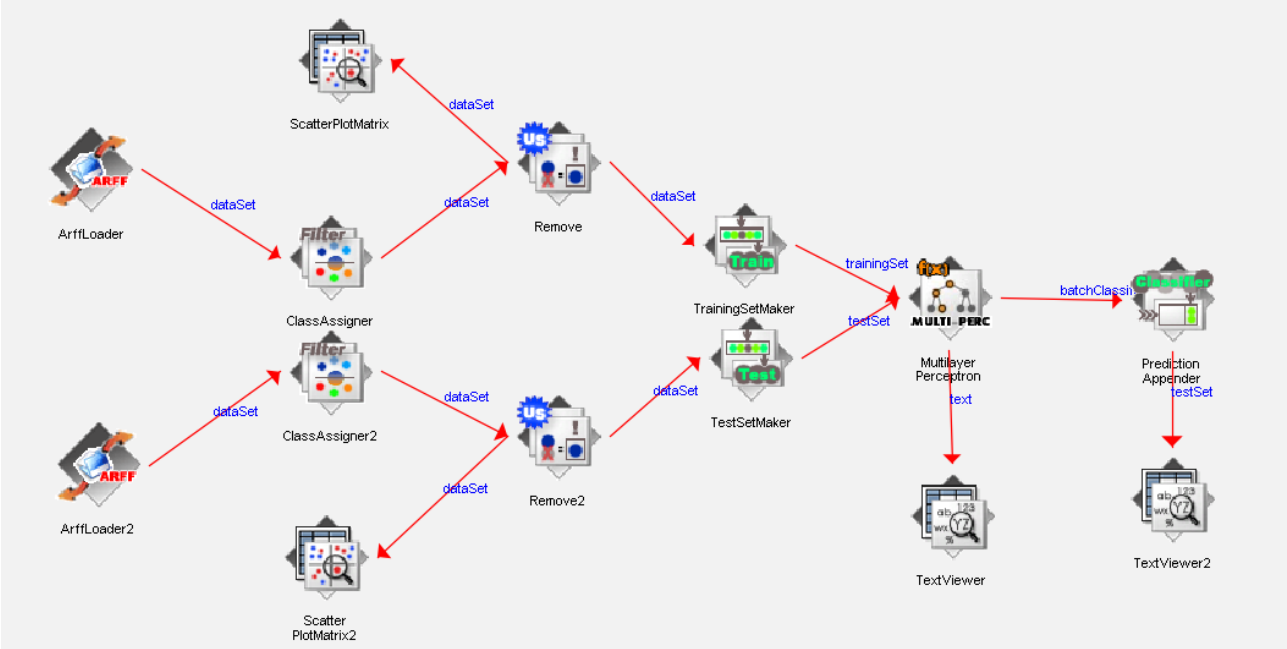
```
@data
3.5,1.3,0.3,Iris-setosa,0.983807,0.01613,0.000062
2.3,1.3,0.3,Iris-setosa,0.953083,0.046827,0.00009
3.2,1.3,0.2,Iris-setosa,0.983254,0.016683,0.000063
3.5,1.6,0.6,Iris-setosa,0.98032,0.019614,0.000067
3.8,1.9,0.4,Iris-setosa,0.983179,0.016758,0.000063
3,1.4,0.3,Iris-setosa,0.980419,0.019514,0.000067
3.8,1.6,0.2,Iris-setosa,0.984278,0.01566,0.000062
3.2,1.4,0.2,Iris-setosa,0.982992,0.016945,0.000063
3.7,1.5,0.2,Iris-setosa,0.984221,0.015717,0.000062
3.3,1.4,0.2,Iris-setosa,0.983408,0.016529,0.000063
2.6,4.4,1.2,Iris-versicolor,0.017688,0.980076,0.002236
3,4.6,1.4,Iris-versicolor,0.016004,0.981178,0.002818
2.6,4,1.2,Iris-versicolor,0.019935,0.978178,0.001886
2.3,3.3,1,Iris-versicolor,0.025046,0.973399,0.001555
2.7,4.2,1.3,Iris-versicolor,0.018087,0.979717,0.002195
3,4.2,1.2,Iris-versicolor,0.022783,0.975504,0.001713
2.9,4.2,1.3,Iris-versicolor,0.019785,0.978251,0.001964
2.9,4.3,1.3,Iris-versicolor,0.019106,0.97884,0.002054
2.5,3,1.1,Iris-versicolor,0.031664,0.966947,0.001389
2.8,4.1,1.3,Iris-versicolor,0.019535,0.978485,0.00198
3.1,5.6,2.4,Iris-virginica,0.000396,0.001343,0.998261
3.1,5.1,2.3,Iris-virginica,0.000465,0.00188,0.997654
2.7,5.1,1.9,Iris-virginica,0.000742,0.006826,0.992433
3.2,5.9,2.3,Iris-virginica,0.000399,0.001371,0.99823
3.3,5.7,2.5,Iris-virginica,0.00039,0.001307,0.998304
3,5.2,2.3,Iris-virginica,0.000436,0.001632,0.997932
2.5,5,1.9,Iris-virginica,0.000659,0.004589,0.994752
3,5.2,2,Iris-virginica,0.000702,0.005975,0.993324
3.4,5.4,2.3,Iris-virginica,0.000469,0.001956,0.997575
3,5.1,1.8,Iris-virginica,0.001949,0.199209,0.798842
```

4 pav. Klasifikavimo rezultatai (spėjimas kairėje ir tikimybės dešinėje).

4 pav. matome klasifikavimo rezultatus. Iš šių paveikslų galime matyti, kad gautos klasės atitinka norimas klases. Reiškia neuroninis tinklas klasifikavo duomenis visiškai teisingai.

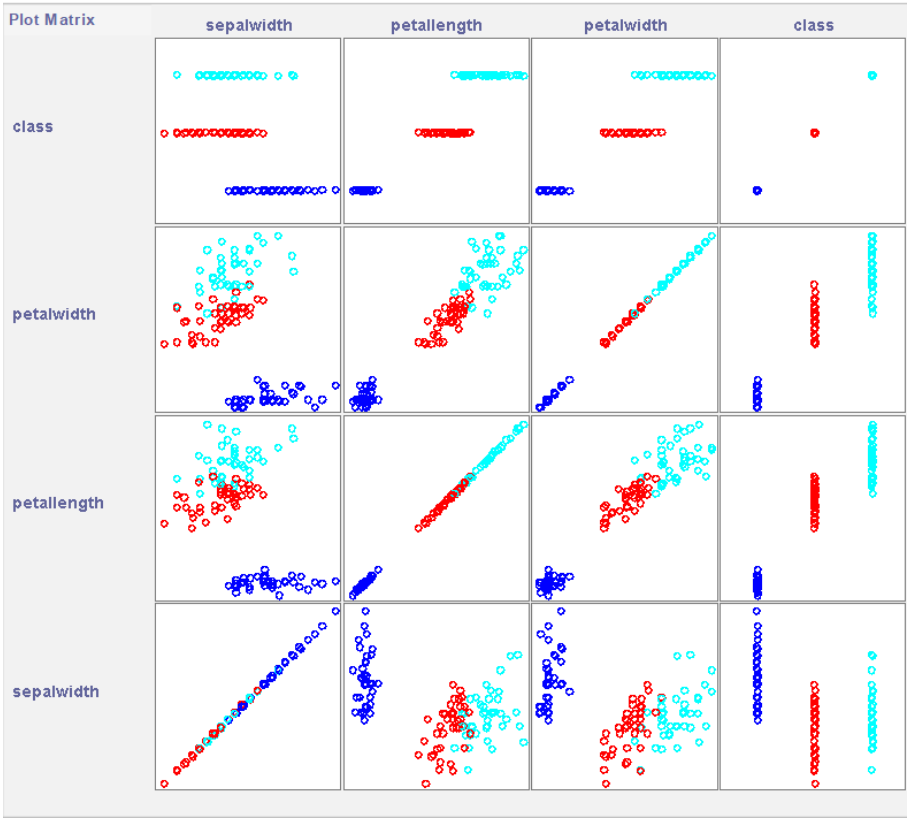
Programos „Weka“ trečioji užduočių seka

5 pav. yra duomenų testavimo ir mokymo seka WEKA, kurioje jau buvo naudojami abu failai *iris_new.arff* ir *iris_train_test.arff*. Čia yra visiškai visos tos pačios dalys *ArffLoader, ClassAssigner, Remove, TrainingSetMaker, TestSetMaker, Multilayer Perceptron, PredictionAppender ir TextViewer.* Vienintelė nauja dalis yra *Scatter PlotMatrix,* kuri parodo dekarto koordinačių sistemą jau atrinktiems duomenims. Tokios sistemos bus dvi, nes reikėjo atskirai pažiūrėti kaip atrodo testavimo ir mokymo duomenys. Dar vienas naujas dalykas, kurio nebuvo ankstesnėse sekose yra sujungimas *TrainingSetMaker* ir *TestSetMaker* į vieną komponentę *Multilayer Perceptron,* kas mums leidžia iškarto apmokyti neuroną ir naudoti gautą modelį spėjimams.

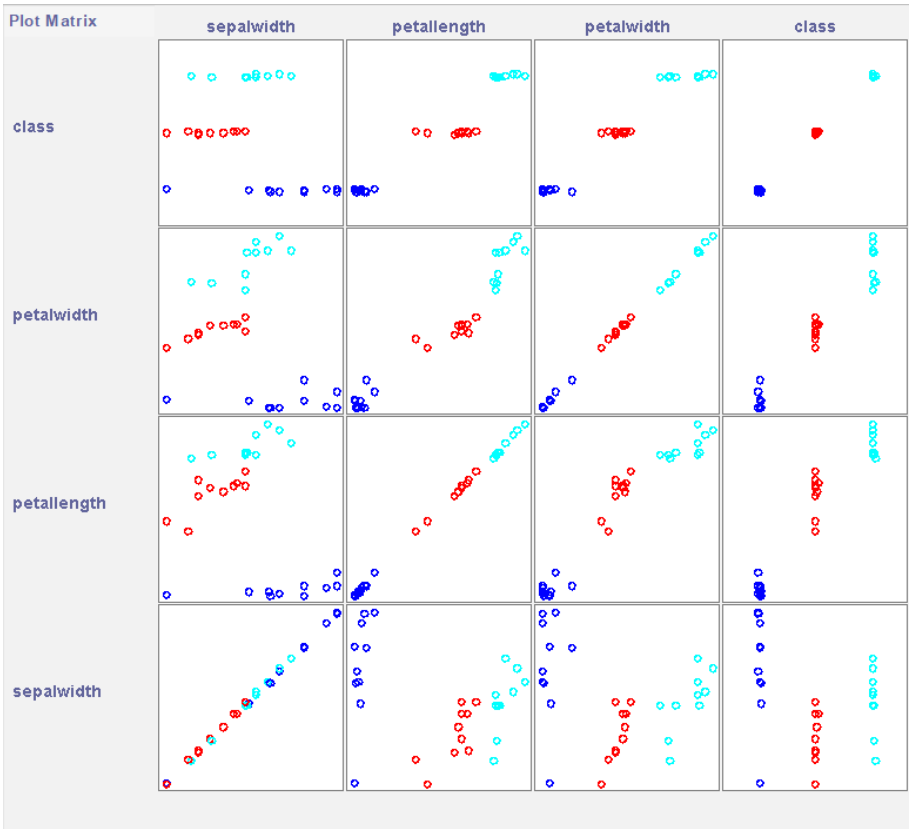


5 pav. Tinklo testavimo ir mokymo seka WEKA.

6 ir 7 pav. pateiktas duomenų požymių porų vaizdas Dekarto koordinačių sistemose. 6 pav. yra pavaizduota mokymo duomenys, o 7 pav. – testavimo duomenys, kurių yra žymiai mažiau. Raudona spalva žymi iris-versicolor, tamsiai mėlyna – iris-setosa, šviesiai mėlyna – iris-virginica duomenis. Matuoti parametrai: taurėlapio plotis – sepalwidth, žiedlapio ilgis – petallength ir žiedlapio plotis – petalwidth. Galime aiškiai pamatyti, kaip duomenys yra pasidaline į atskirus plotus ir labai nedaug maišosi ir nepersidengia.



6 pav. Mokymo duomenų dekarto koordinačių sistema.



7 pav. Testavimo duomenų dekarto koordinačių sistema.

2 lentelė. Gautos svorių reikšmės WEKA.

Neuronai	w0	w1	w2	w3
3	-6.542241935427199	-3.5545488427883423	6.587360261326346	9.324202861289333
4	5.89783377274894	3.1313933650887398	-5.7536696643857645	-8.97803706011309
5	-2.975735561823562	2.8180064239067493	-4.140459414594477	-3.9162118052179014
6	-0.8565350430297103	-0.4203309002635567	0.2893313178284408	-0.09096595825657808
7	-0.27765365200122116	-2.28123442720022	1.8971508739379348	2.0796744003700716

3 lentelė. Gautos svorių reikšmės paslėptame sluoksnyje WEKA.

	treshold	node 3	node 4	node 5	node 6	node 7
Neuronai	w0	w1	w2	w3	w4	w5
0	-1.9019111006502343	-2.320738441479091	0.7970493833568829	6.1709560657302855	-1.2410381292926143	-4.752514851225004
1	-0.29557222556934426	-7.490049630513156	5.398893040380996	-9.514315919589333	-0.17480893209200915	-0.030325200773895247
2	-1.3041128218288869	6.199212299454869	-6.964035133082515	-3.389188177342466	-0.28886507944707956	3.788534300999628

Trečioje lentelėje pateikiami gauti svorių rinkiniai 0, 1 ir 2 neurone. Ketvirtoje lentelėje pateikti paslėptame sluoksnyje esančių penkių neuronų svorių rinkiniai. Mokymosi greitis dirbtiniam neuroniniam tinklui buvo 0,3, o *momentum* reikšmė - 0,4, paslėptame neuronų sluoksnių yra vienas su penkiais paslėptais neuronais.

Skaičiavimai programoje „Excel“

Pradiniai duomenys, su kuriais buvo atliekamas testavimas Excel programoje, yra sunormalizuojami pagal formulę intervale nuo -1 iki 1. Neuronų svorių lentelės, gautos WEKA programoje buvo perrašomos į Excel programą. Pateikiamos dvi lentelės, 2 lentelė ir 3 lentelė, kadangi naudojame tik vieną paslėptą neuronų sluoksnį. Sumuojami duomenų įėjimo vektoriai su paslėptų neuronų svorių vektorių sandaugomis. Šiuo atveju paslėptų neuronų yra penki, todėl lentelėje yra penki stulpeliai ir 30 eilučių, tiek kiek yra testavimo duomenų mūsų duomenų faile. Tada yra skaičiuojami paslėpto sluoksnio įėjimai, t. y., skaičiuojamos sigmoidinės funkcijos reikšmės nuo sumų, kurios jau buvo apskaičiuojamos vienu žingsniu anksčiau. Toliau sumuojamos jau gautų funkcijų reikšmių vektorių ir paslėptų neuronų svorių vektorių sandaugos, šias sandaugas skaičiuojame visiems duomenų įėjimo vektoriams. Galiausiai skaičiuojame neuroninio tinklo išėjimus. Excel faile „Darbas 3“ yra 3 aplankai, kur pirmajame yra atlikti visi skaičiavimai. Sekantys 2 buvo naudojami lentelių įkėlimui į aprašą.

Rezultatai

4 lentelė. Klasifikavimo tikslumo palyginimas abiem programoms.

Setona EXCEL	*Versicolor* EXCEL	*Virginica* EXCEL	*Setona* WEKA	*Versicolor* WEKA	*Virginica* WEKA
0,991616081	0,011741372	8,32121E-06	0.987521	0.01247	0.000009
0,959781703	0,036912406	2,07377E-05	0.9619	0.038081	0.000019
0,990995617	0,01188964	8,44728E-06	0.987083	0.012908	0.000009
0,991063176	0,012416869	8,65352E-06	0.984063	0.015927	0.00001
0,753948892	0,290439195	7,12819E-05	0.986883	0.013109	0.000009
0,989653682	0,012877434	9,05899E-06	0.984548	0.015441	0.00001
0,991959216	0,011675168	8,26398E-06	0.987961	0.01203	0.000008
0,990899114	0,011962199	8,48583E-06	0.986853	0.013138	0.000009
0,991866898	0,011676322	8,2659E-06	0.987916	0.012076	0.000008
0,99118999	0,011836776	8,3925E-06	0.987212	0.012779	0.000009
0,003929045	0,98907405	0,008038666	0.006803	0.989054	0.004143
0,007395403	0,986814394	0,00609986	0.008964	0.986923	0.004113
0,00569905	0,991669372	0,004728918	0.009418	0.98757	0.003012
0,006210383	0,992087865	0,003840387	0.013805	0.984062	0.002133
0,005206989	0,98977716	0,006222942	0.008038	0.988224	0.003739
0,020539891	0,991514549	0,001655139	0.021436	0.977002	0.001562
0,010617187	0,991439141	0,003023244	0.012913	0.984631	0.002456
0,009467832	0,991157617	0,00343538	0.011782	0.985534	0.002684
0,015037649	0,990136496	0,002037228	0.026265	0.972399	0.001336
0,008159184	0,991309833	0,003776729	0.011064	0.986156	0.00278
0,000103167	0,000384294	0,999779394	0.00011	0.000395	0.999495
0,000123759	0,000411097	0,999734566	0.000135	0.000471	0.999394
0,000106346	0,000507107	0,999693124	0.000165	0.001439	0.998396
0,000109006	0,000386391	0,999769187	0.000115	0.000402	0.999483
0,000114775	0,000385468	0,999767154	0.000116	0.000391	0.999493
0,000108108	0,000394769	0,999764633	0.00012	0.000434	0.999446
9,15851E-05	0,000441208	0,999749766	0.000135	0.000997	0.998869
0,000137214	0,000559113	0,999614466	0.000187	0.001243	0.998569
0,000173054	0,000435408	0,999650524	0.000164	0.000496	0.99934
0,000255648	0,003221697	0,99727207	0.000556	0.040776	0.958668

4 lentelėje pateikti klasifikavimo tikslumai gauti WEKA bei Excel programose. Kaip matome, tikslumas abiejose programose labai panašus ir reikšmės skiriasi tik labai nedidele skaičiaus dalimi.

Išvados

Galime padaryti išvadas apie trečiąjį laboratorinį darbą, kuriame mes dirbome su WEKA ir Excel programomis. Pastebėjau, kad abiejų programų klasifikavimo tikslumo rezultatai buvo labai panašūs, vietomis beveik identiški. Dar pastebėjau, kad parametras *momentum* turėjo mažesnę įtaką klasifikavimo tikslumui, negu, kad mokymo greitis. Drastiškai pakėlus mokymo greitį iki reikšmės artimos vienetui, pastebėjau, kad klasifikavimo tikslumas ryškiai sumažėja, taip pat nutiko ir pakėlus parametrą *momentum.*

Paskutinis dalykas kurį pastebėjau buvo tai, kad esant vienam sluoksniui, rezultatai gavosi labai nestabilūs, o naudojant 2 paslėptus sluoksnius, beveik visi tikslumai buvo vienodi arba skyrėsi labai neryškiai, nepriklausomai nuo neuronų skaičiaus, mokymo greičio ar *momentum*.