

**RĪGAS TEHNISKĀ UNIVERSITĀTE**  
**Datorzinātnes un informācijas tehnoloģijas fakultāte**

**Priekšmeta “Mākslīga intelekta pamati(1) 21/22”**  
**Otrā praktiskā darba atskaite**

Darbā izmantota datu kopa:  
<https://archive.ics.uci.edu/ml/datasets/seeds>

Projekta atrašanās vieta tīmeklī:  
[https://github.com/nikolajeff/2\\_pd\\_ai](https://github.com/nikolajeff/2_pd_ai)

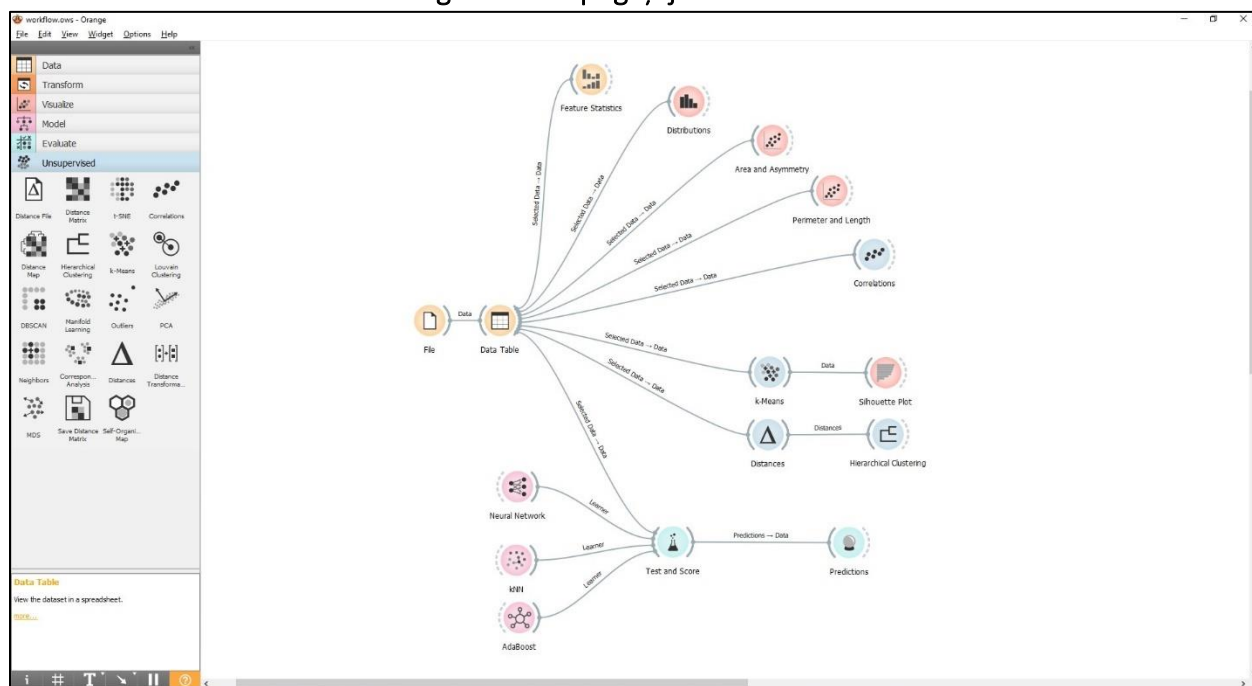
Izstrādāja:  
Informācijas tehnoloģijas students  
**Nikita Nikolajevs**  
1.grupa  
201RDB058

**Rīga, 2022**

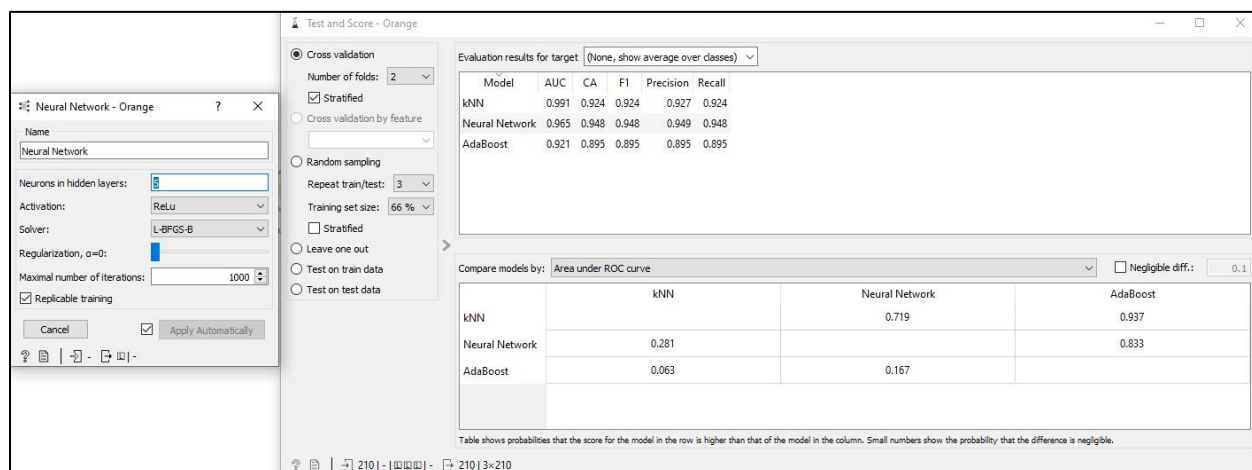
## Saturs

<b>“Orange” rīka atspoguļojums visam darbam .....</b>	<b>3</b>
<b>I daļa - Datu pirmāpstrāde/izpēte .....</b>	<b>5</b>
1. Datu kopas apraksts .....	5
2. Datu kopas satura apraksts .....	5
3. Datu apstrādes rezultāti “Orange” rīkā .....	7
4. Pirmās daļas secinājumi .....	9
<b>II daļa - Nepārraudzītā mašīnmācīšanās .....</b>	<b>10</b>
1) Hierarhiskās klasterizācijas algoritms .....	10
2) K-Vidējo algoritms .....	13
3) Otrās daļas secinājumi .....	16
<b>III daļa - Pārraudzītā mašīnmācīšanās .....</b>	<b>17</b>
<b>Secinājumi .....</b>	<b>18</b>

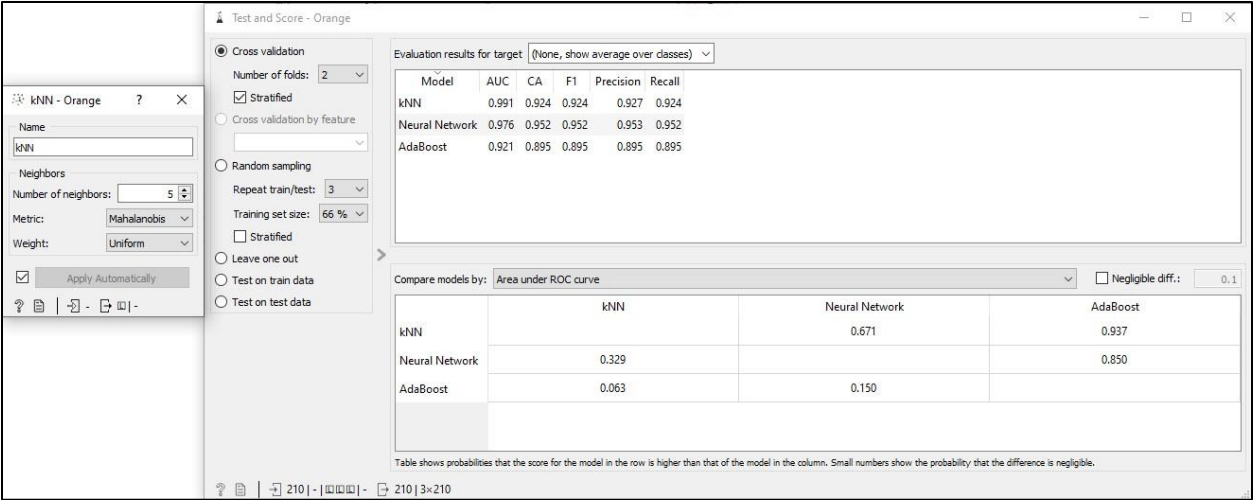
## “Orange” rīka atspoguļojums visam darbam



Att. 0.1. “Orange” programmatūra darbplūsmas ekrānuzņēmums

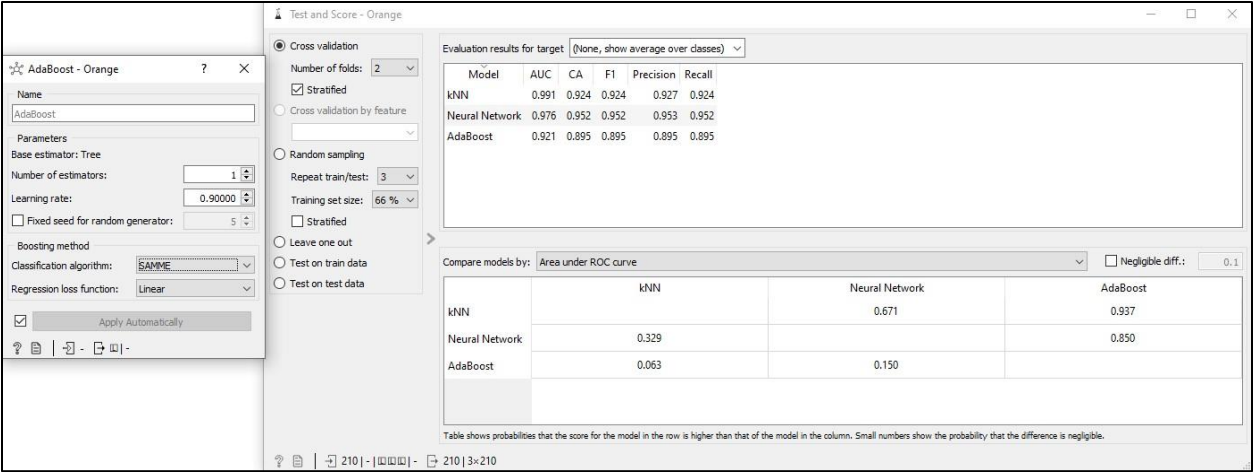


Att. 0.2. Neironu tīklas algoritma mainīgo vērtības un rezultāti.



Att. 0.3. kNN algoritma mainīgo vērtības un rezultāti.

Att. 0.4. AdaBoost algoritma mainīgo vērtības un rezultāti.



Att. 0.3. kNN algoritma mainīgo vērtības un rezultāti.

## I daļa - Datu pirmapstrāde/izpēte

### 1. Datu kopas apraksts

Datu kopas nosaukums ir – “Trīs dažādām kviešu šķirnēm piederošo kodolu ģeometrisko īpašību mērījumi”. Kopa ir ņemta no grāmatas “Informācijas tehnoloģijas biomedicīnā, 2 daļa”, no “Pilnīgs gradientu klasterizācijas algoritms rentgena attēlu funkciju analīzei” nodaļas.

Šo datu kopu apkopoja Małgorzata Charytanowicz un Jerzy Niewczas ar “Matemātikas un Datorzinātņu Institūta” atbalstu Ļublinā

Datu savākšanas veids bija sekojošs – kviešu sēklas fotogrāfijas tika uzņemtas, izmantojot 13 \* 18 cm rentgena “KODAK” plāksnes. Fotogrāfijas vēlāk tika ievadītas speciāla programma “Grains”, kas analizēja bildes un izvadīja katras sēkles septiņas ģeometriskas pazīmes.<sup>1</sup>

### 2. Datu kopas satura apraksts

Datu kopa satur 210 ierakstus par trīs dažādiem sēklu veidiem. Katram sēklu veidam ir pa 70 ierakstiem mūsu tabulā. Katrs ieraksts ietvēr sevī septiņas dažādas ģeometriskas parametrus. Šos parametrus var redzēt Tabulā 1.1.

№	Atribūta apzīmējums	Skaidrojums	Vērtību tips	Vērtību diapazons vai kopa
1.	ID	Kolonnai izmantotais unikālais identifikators (netiek ņemts aprēķinos)	Vesels skaitlis	[1; 210]
2.	Seed type	Sēklas šķirne. Līdz ar to, ka pašā zinātniskajā darbā nav minēti sēklu tipa numuri, bet datu kopā pie šīm atribūtiem ir uzrakstīti cipari, tie ir attiecīgi atstāti, ka šī atribūta vērtības.		{1, 2, 3}
3.	Perimeter	Kviešu sēklas perimetrs	Reālais skaitlis	[12.41; 17.25]
4.	Kompaktums	Šī atribūta vērtība tiek aprēķināta, izmantojot formulu $C = \frac{4 \times \pi \times \text{Area}}{\text{Perimeter}^2}$		[0.8081; 0.9183]
5.	Length of kernel	Sēklas garums		[4.899; 6.675]
6.	Width of kernel	Sēklas platums		[2.63; 4.033]
7.	Asymmetry coefficient	Asimetrijas koeficients		[0.7651; 8.456]
8.	Length of kernel groove	Sēklas rievas garums		[4.519; 6.55]
9.	Area	Laukums, ko aizņem kviešu šēkla		[10.59; 21.18]

Tabula 1.1. Datu kopas atribūtu apraksts.

<sup>1</sup> [https://www.researchgate.net/publication/226738117\\_Complete\\_Gradient\\_Clustering\\_Algorithm\\_for\\_Features\\_Analysis\\_of\\_X-Ray\\_Images](https://www.researchgate.net/publication/226738117_Complete_Gradient_Clustering_Algorithm_for_Features_Analysis_of_X-Ray_Images)

Attēlā 1.1. vār redzēt tabulas augšējo daļu, atribūtus un 38 datu ierakstus.

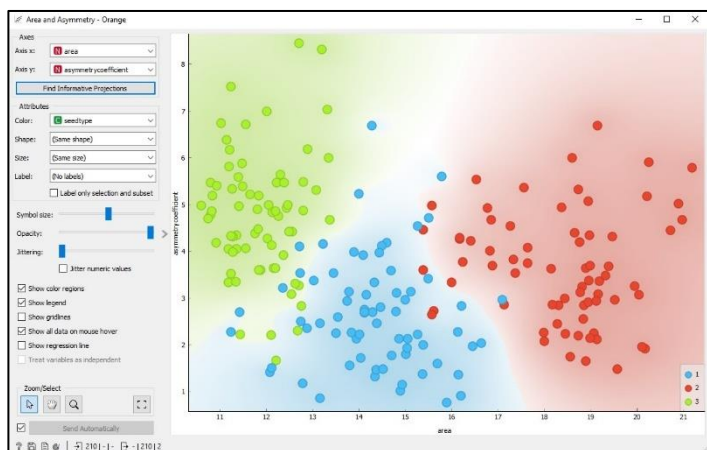
Data Table - Orange								
<div>Info</div> <div>210 instances (no missing data)</div> <div>7 features</div> <div>Target with 3 values</div> <div>No meta attributes</div> <div>Variables</div> <div><input checked="" type="checkbox"/> Show variable labels (if present)</div> <div><input type="checkbox"/> Visualize numeric values</div> <div><input checked="" type="checkbox"/> Color by instance classes</div> <div>Selection</div> <div><input checked="" type="checkbox"/> Select full rows</div> <div>Restore Original Order</div> <div><input checked="" type="checkbox"/> Send Automatically</div>								
	seedtype	perimeter	compactness	lengthofkernel	widthofkernel	symmetrycoefficient	lengthofkernelgroove	area
1	1	14.84	0.8710	5.763	3.312	2.2210	5.220	15.26
2	1	14.57	0.8811	5.554	3.333	1.0180	4.956	14.88
3	1	14.09	0.9050	5.291	3.337	2.6990	4.825	14.29
4	1	13.94	0.8955	5.324	3.379	2.2590	4.805	13.84
5	1	14.99	0.9034	5.658	3.562	1.3550	5.175	16.14
6	1	14.21	0.8951	5.386	3.312	2.4620	4.956	14.38
7	1	14.49	0.8799	5.563	3.259	3.5860	5.219	14.69
8	1	14.10	0.8911	5.420	3.302	2.7000	5.000	14.11
9	1	15.46	0.8747	6.053	3.465	2.0400	5.877	16.63
10	1	15.25	0.8880	5.884	3.505	1.9690	5.533	16.44
11	1	14.85	0.8696	5.714	3.242	4.5430	5.314	15.26
12	1	14.16	0.8796	5.438	3.201	1.7170	5.001	14.03
13	1	14.02	0.8880	5.439	3.199	3.9860	4.738	13.89
14	1	14.06	0.8759	5.479	3.156	3.1360	4.872	13.78
15	1	14.05	0.8744	5.482	3.114	2.9320	4.825	13.74
16	1	14.28	0.8993	5.351	3.333	4.1850	4.781	14.59
17	1	13.83	0.9183	5.119	3.383	5.2340	4.781	13.99
18	1	14.75	0.9058	5.527	3.514	1.5990	5.046	15.69
19	1	14.21	0.9153	5.205	3.466	1.7670	4.649	14.70
20	1	13.57	0.8686	5.226	3.049	4.1020	4.914	12.72
21	1	14.40	0.8584	5.658	3.129	3.0720	5.176	14.16
22	1	14.26	0.8722	5.520	3.168	2.6880	5.219	14.11
23	1	14.90	0.8988	5.618	3.507	0.7651	5.091	15.88
24	1	13.23	0.8664	5.099	2.936	1.4150	4.961	12.08
25	1	14.76	0.8657	5.789	3.245	1.7910	5.001	15.01
26	1	15.16	0.8849	5.833	3.421	0.9030	5.307	16.19
27	1	13.76	0.8641	5.395	3.026	3.3730	4.825	13.02
28	1	13.67	0.8564	5.395	2.956	2.5040	4.869	12.74
29	1	14.18	0.8820	5.541	3.221	2.7540	5.038	14.11
30	1	14.02	0.8604	5.516	3.065	3.5310	5.097	13.45
31	1	13.82	0.8662	5.454	2.975	0.8551	5.056	13.16
32	1	14.94	0.8724	5.757	3.371	3.4120	5.228	15.49
33	1	14.41	0.8529	5.717	3.186	3.9200	5.299	14.09
34	1	14.17	0.8728	5.585	3.150	2.1240	5.012	13.94
35	1	14.68	0.8779	5.712	3.328	2.1290	5.360	15.05
36	1	15.00	0.9000	5.709	3.485	2.2700	5.443	16.12
37	1	15.27	0.8734	5.826	3.464	2.8230	5.527	16.20
38	1	15.38	0.9079	5.832	3.683	2.9560	5.484	17.08

Att.1.1. Datus kopas tabulas daļa.

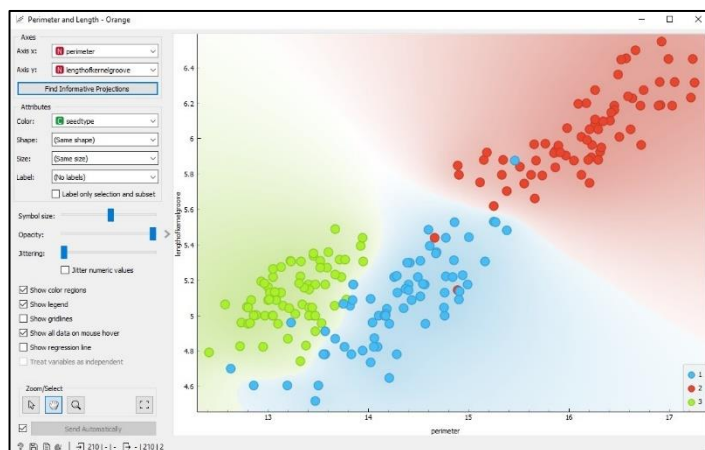
### 3. Datu apstrādes rezultāti “Orange” rīkā

Izmantojot datu kopu, “Orange” rīkā tiek izveidotas dažādas infografikas elementi (t.i. histogrammas, izkliedes diagrammas(scatter plot) un statistikas elements ar informāciju par datu kopu).

Lai pamatotu klases atdalīšanu, tiek izveidotas divas izkliedes diagrammas, kuri ir redzāmi attēlā 1.2. un 1.3. Abas diagrammas labi ilustrē trīs dažādu sēkļu veidu klasifikācijas iespēju. Lai gan klases nedaudz saplūst malās, lielāka sēkļu daļa atrodas apmērām savā diapazonā. Tas ir vislabāk redzāms att. 1.3., kuras diagrammā tikai 3-4 objekti atrodas “nepareizās” vietās.

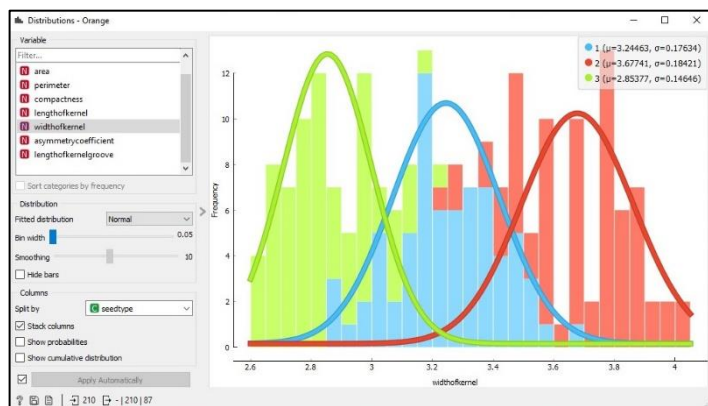


Att. 1.2. Izkliedes diagramma  
(asimetrijas koeficients un laukums)

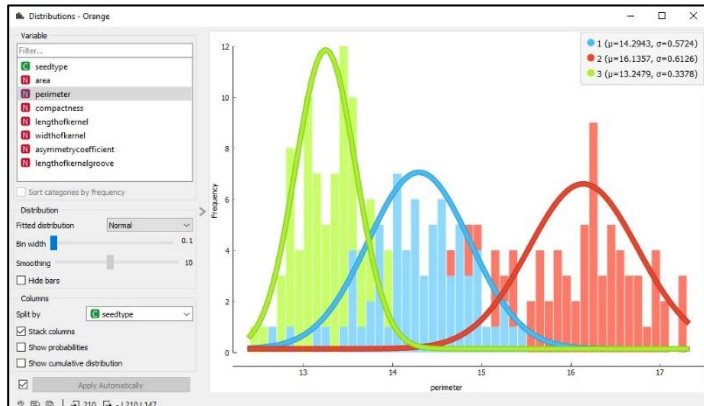


Att. 1.3. Izkliedes diagramma  
(sēkla garums un perimetrs)

Klases atdalāmība var arī būt atspoguļota izmantot histogrammas ar standarta novirzes grafiku. Piemēram, attēlos 1.4. un 1.5. vār redzēt plātuma un perimētra sadalījumu attiecīgi.



Att. 1.4. Plātuma sadalījums



Att. 1.5. Perimetra sadalījums

Attēlā 1.6. var redzēt dažus statistikas elementus, piemēram, katra atribūta sadalījumu, vidējo vērtību, mediānu, dispersiju, minimumu un maksimumu.



Att. 1.6. Datu kopas atribūtu statistika.

#### 4. Pirmās daļas secinājumi

Pirms datu analīzes ir svarīgi pieminēt, ka datu kopas klases ir pilnīgi līdzsvarotas (atsaucoties uz datu kopa aprakstu, tiek ņemti pa 70 katra sēklu veida).

Ņemot verā visus infografikas elementus, ir viegli salasīt objektu klases atalāmību. Skatoties uz izkļedes diagrammam attēlos 1.2. un 1.3., varu secināt, ka visi trīs sēkļu veidi ir savietoti savās sektoros un nesaplūst, izņemot pāris gadījumus, kas nav kritisks. Att. 1.2. rāda, ka, ņemot verā asimetrijas koeficientu un sēklas laukumu, trīs klases atrodas diezgan tālu viens no otra, atkal – neskatoties uz dažiem izņēmumiem, kad objekts atrodas svēšā diapazonā.

Apskatot sadalījumu diagrammas attēlos 1.4. un 1.5, kuri atspoguļo tikai viena kritērija sadali, ir redzama diezgan skaidra atalāmība. Protams, skatoties uz histogrammu tikai ar vienu parametru, dažādu sēkļu veidu parametru vērtības iekļūt vienā diapazonā kas ir diezgan neizbēgami. Joprojam, pievēršot uzmanību uz standarta novirzes līnijas, to virsotnes atrodas pietiekami tālu viens no otra. Tas attēlo skadrāku atdalāmību.

Skatoties uz statistikas elementiem (att. 1.6.), varu veikt sekojošus secinājumus. Skatoties uz “Distribution” diagrammam, ir iespējams redzēt īpašības, kuri arī atspoguļo klases sadalāmību. Piemēram, paskatīsimies uz “widthofkernel” sēkla plātuma histogrammu. Katrs sēkļu veids dominē savā vērtību regionā. Pirmos trīs stābiņos ir visvairāk zālas krāsas objektu, divos nākamajos stabos dominē zila krāsa, sestajā stabā ir tikpat daudz sarkano un zilo objektu, pēc tam dominē pedējais sēkļu veids.

## II daļa - Nepārraudzītā mašīnmācīšanās

Šī daļā tiek apskatīti nepārraudzītās mašīnmācīšanās iespējas un metodes. Precīzāk – Hierarhiskās klasterizācijas algoritms un K-Vidējo algoritms.

### 1) Hierarhiskās klasterizācijas algoritms

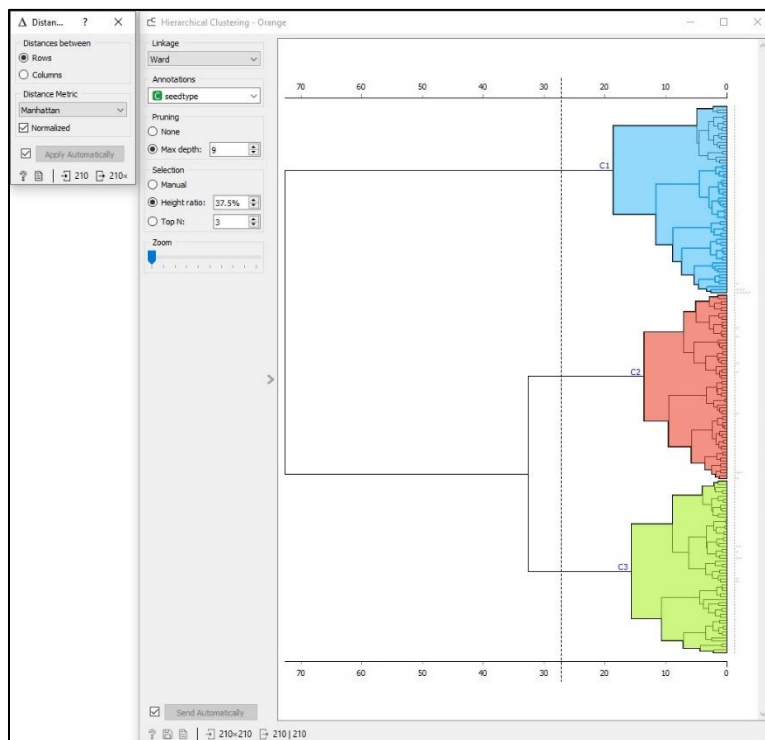
Hierarhiskās klasterizācijas algoritmam ir daži hiperparametri, kuri nodrošina atšķirīgus algoritma rezultātus. Neņemot vērā parametru “Distances between”, kur divas vērtības ir “Rindas” un “Kolonnas”. Mainot to ar mūsu datu kopu nav nekādas jēgas, jo distance starp atribūtiem mums neko noderīgu nedarīs.

Savukārt, “Distance metric” parametrs ir daudz interesantāks. Tās nodrošina dažādus objektu savstarpēja attāluma aprēķināšanas veidus(formulas). Piemēram, “Euclidean”, “Manhattan” vai “Jaccard”.

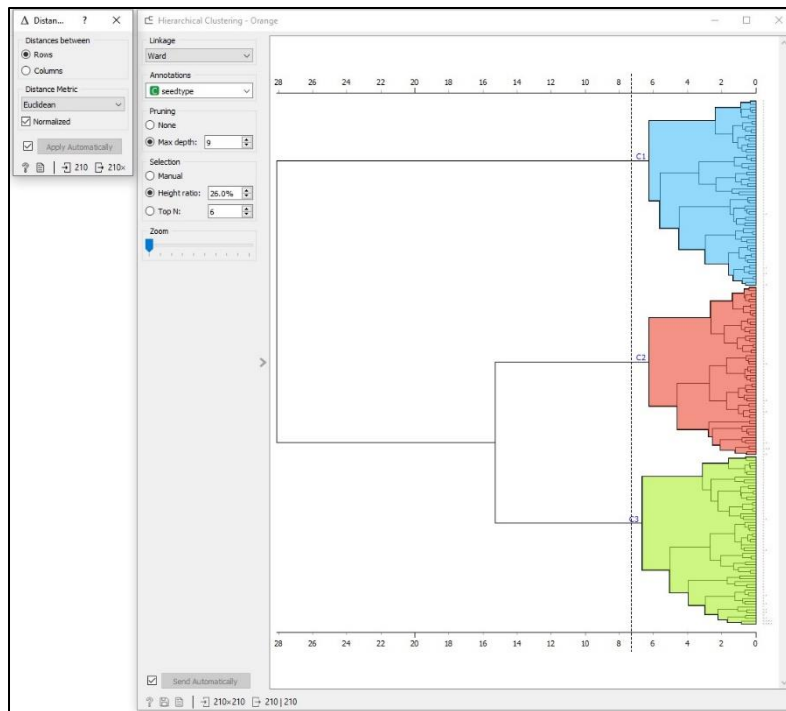
“Pruning” opcija ir atbildīga par parādītas klasterizācijas dziļumu. Tas neietekmē algoritmu, taču maina, kā izskatās hierarhijas koks.

Visvarīgākais parametrs – “Selection”. Tās kontrolē cik daudz klasteru ir vizuāli redzams laika momentā. To var izdarīt manāli nospiežot uz klastera ar opciju “Manual”, pārvietojot augstuma līniju ar opciju “Height ration” vai arī norādot, tieši cik daudz augšējo klasteru lietotājs grib redzēt ar opciju “Top N”.

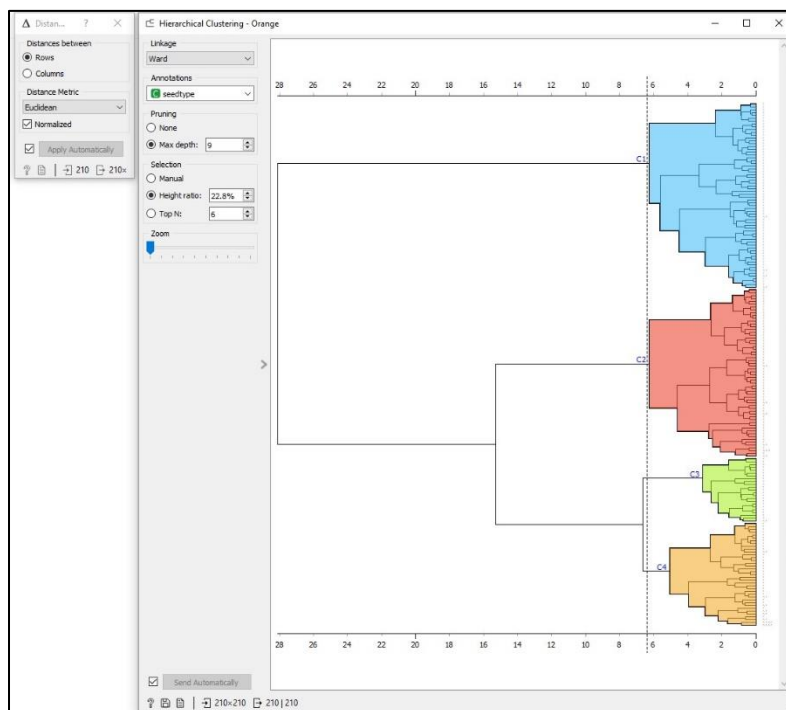
Attēlos no 2.1. līdz 2.4. ir redzamas hierarhijās skati ar dažādiem parametriem.



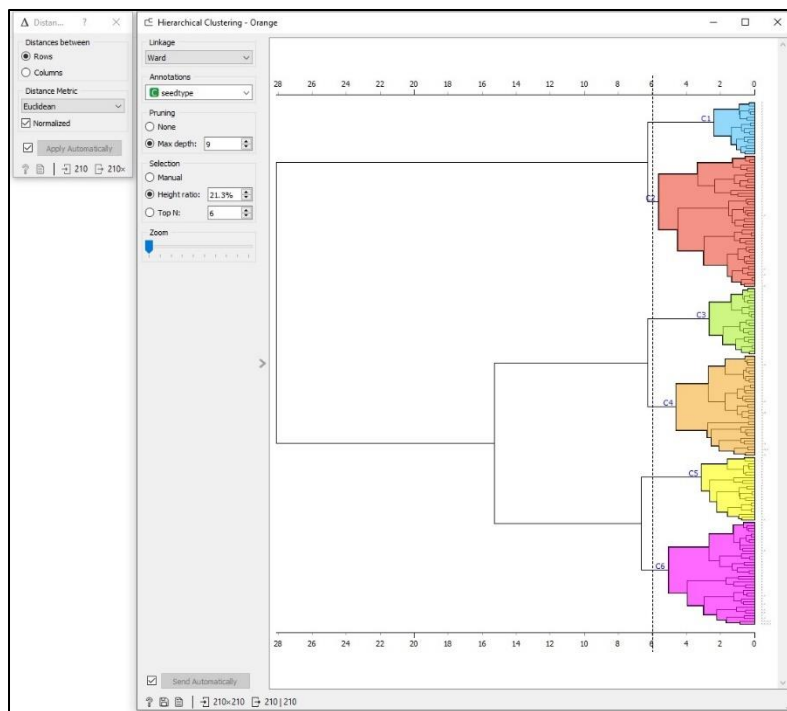
Att. 2.1. Hierarhiskā klasterizācijas algoritma darba demonstrēšana  
(Distance Metric = Manhattan; Selection.TopN = 3)



Att. 2.2. Hierarhiskā klasterizācijas algoritma darba demonstrēšana  
(Distance Metric = Euclidean; Klasteru skaits = 3)



Att. 2.3. Hierarhiskā klasterizācijas algoritma darba demonstrēšana  
(Distance Metric = Euclidean; Klasteru skaits = 4)



Att. 2.4. Hierarhiskā klasterizācijas algoritma darba demonstrēšana  
(Distance Metric = Euclidean; Klasteru skaits = 6)

## 2) K-Vidējo algoritms

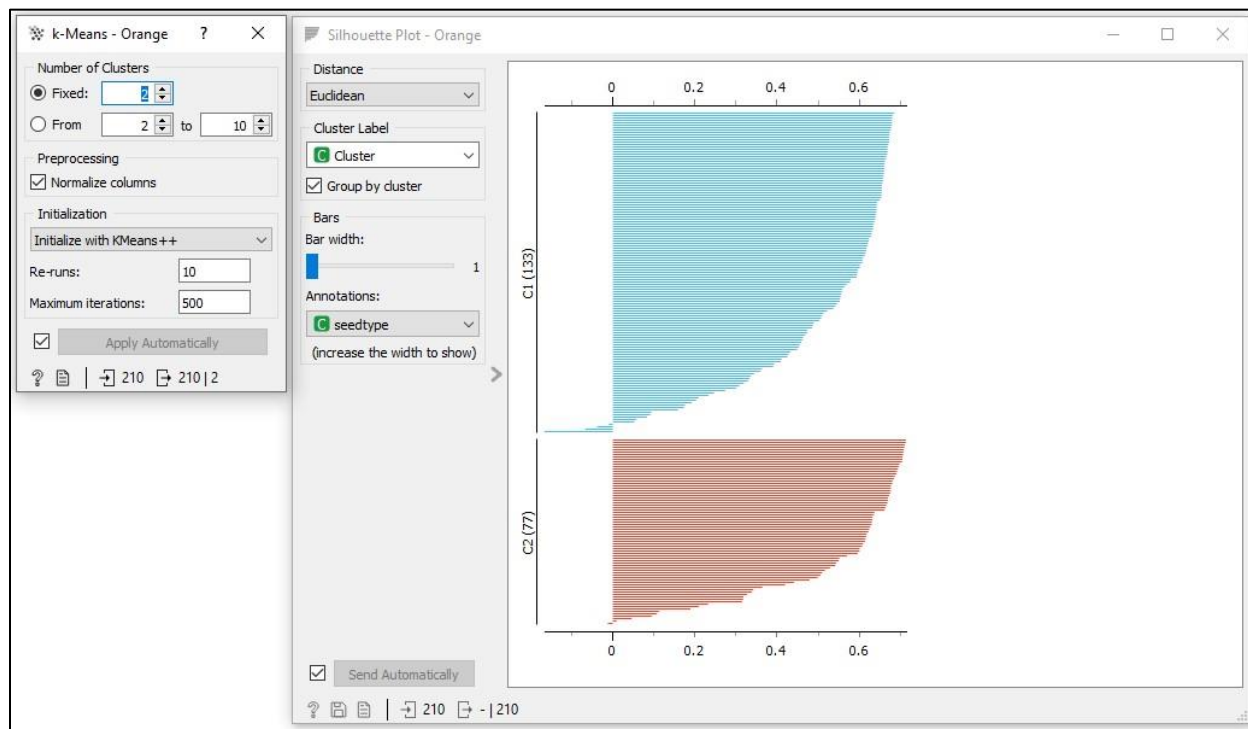
Klasterizējot ar K-Vidējo algoritmu ir dažādas iespējas saņemt rezultātu, mainot hiperparametrus.

“Number of clusters” – klasteru daudzums, kuru mēs gribam redzēt tālāk “Silhouette Plot” diagrammā. Ir iespēja uzstādīt fiksēto klāsteru skaitu, vai arī varām izvēlēties klasteru skaitu diapazonu, kas arī radīs “Silhouette Score” vērtības dažādam klasteru skaitam, kas atspoguļo, cik labi objekti ir atdālīti pa klasteriem.

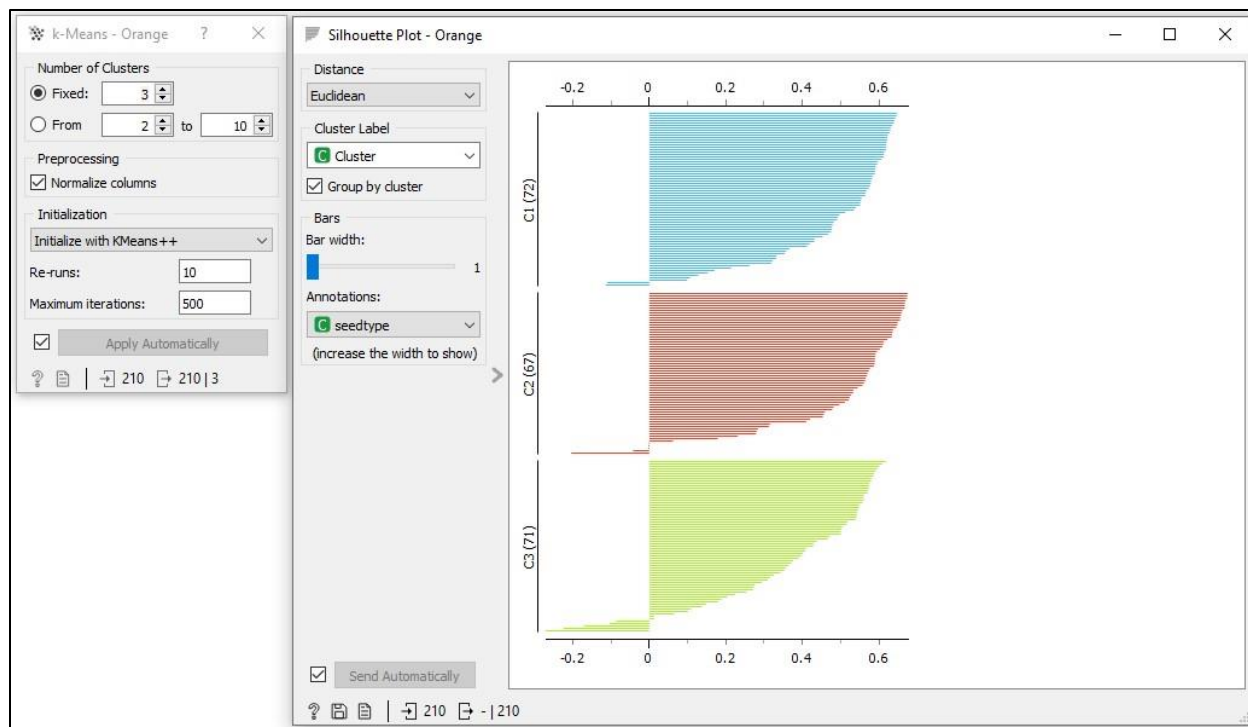
“Initialization” vietā varam mainīt iekšējās algoritma mainīgus, kuri ietekmēs algoritma precizitāti, taču var pieprasīt vairāk daora resursu aprēķiniem.

“Distance” – līdzīgi Hierarhiskās klasterizācijas algoritmam – ir iespēja izvēlēties dažādus objektu savstarpēja attāluma aprēķināšanas veidus. Tomēr, ir tikai trīs iespējas – “Euclidean”, “Manhattan” un “Cosine”.

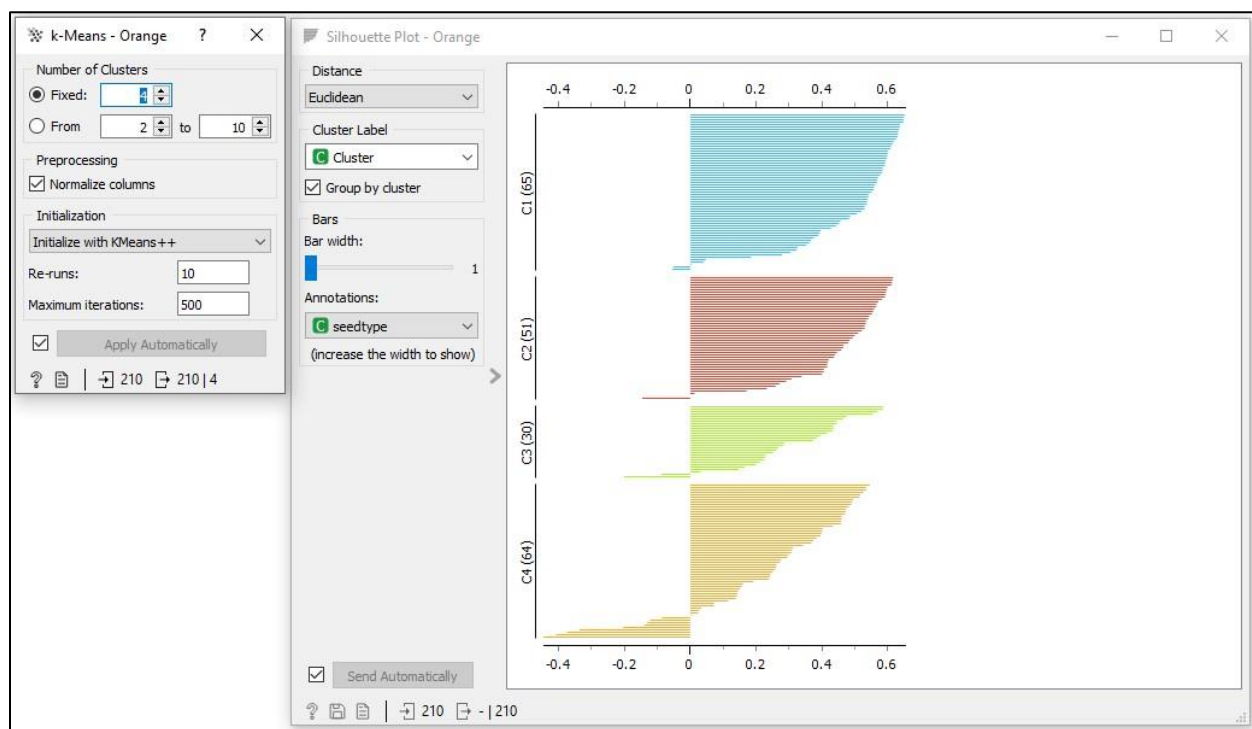
Attēlos no 2.5. līdz 2.9. ir parādīti dažādi klasterizēšanas rezultāti ar K-Vidējo algoritmu, mainot klasteru daudzumu parametru k diapazonā [2; 6].



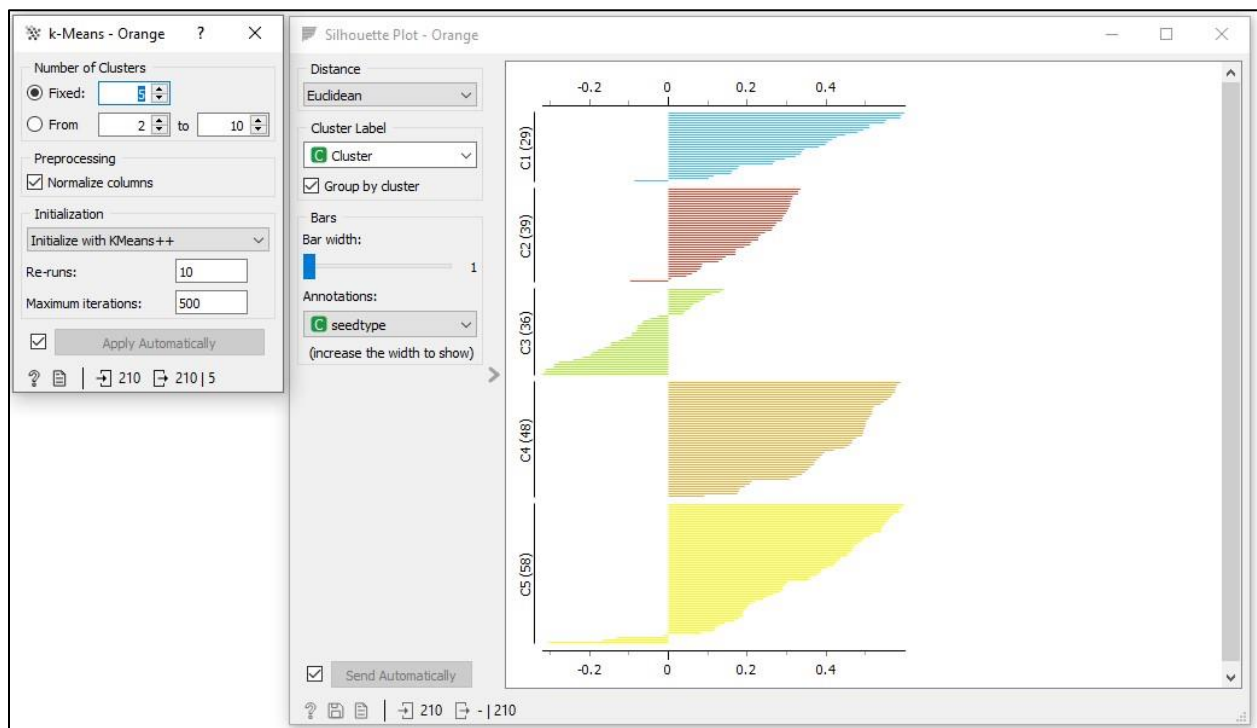
Att. 2.5. K-Vidējo algoritma darba demonstrācija pie  $k = 2$



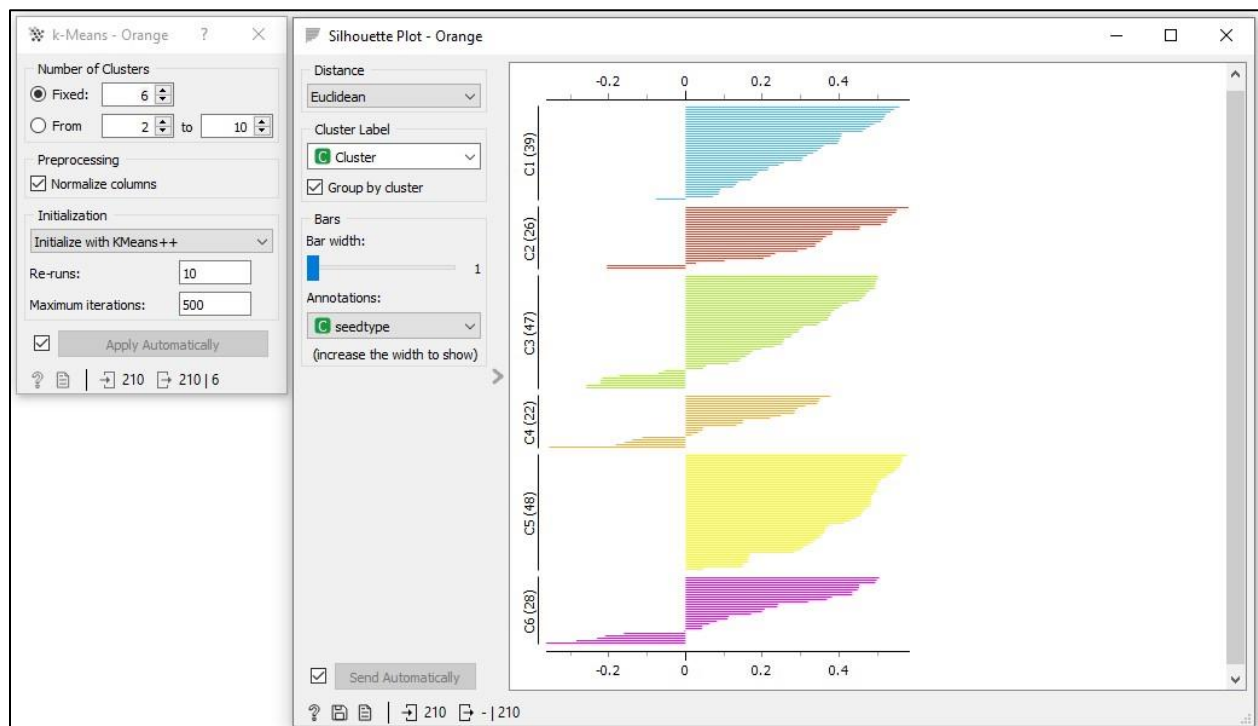
Att. 2.6. K-Vidējo algoritma darba demonstrācija pie  $k = 3$



Att. 2.7. K-Vidējo algoritma darba demonstrācija pie  $k = 4$



Att. 2.8. K-Vidējo algoritma darba demonstrācija pie  $k = 5$



Att. 2.9. K-Vidējo algoritma darba demonstrācija pie  $k = 6$

### 3) Otrās daļas secinājumi

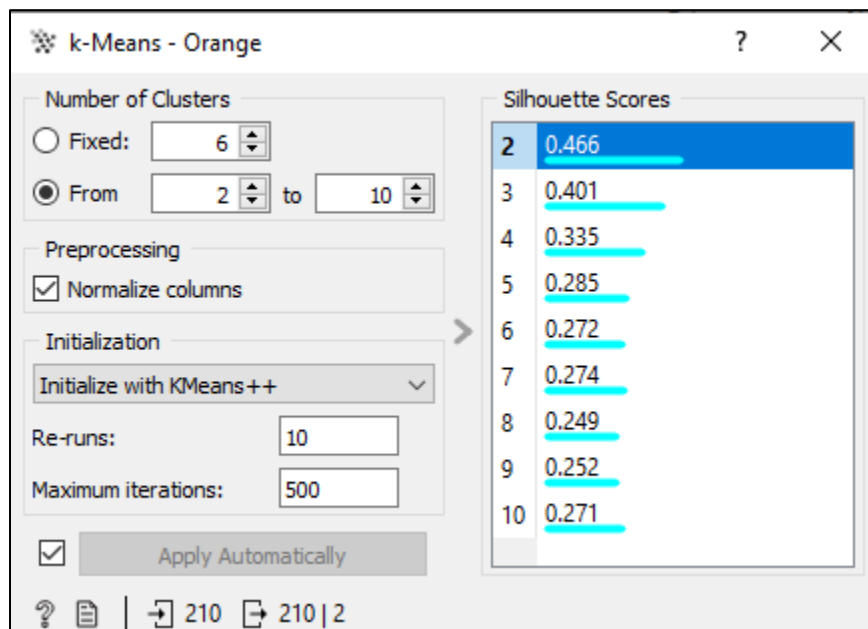
#### 1. Hierarhiskās klasterizācijas algoritma rezultātu analīze:

Pirmkārt, attēli 2.1. un 2.2. parāda atšķirību starp Eiklīda un Manhetenas attāluma aprēķina veidu izmantošanu. Tas tikai nedaudz ietekmē rezultātus. Katrs klasteris zaudē tikai vienu vai divus objektus, kuri nonāk citā klasterī.

Otrkārt, attēli 2.2., 2.3. un 2.4. attēlo dažādas klastera daudzumus. Neskatoties uz to, ka mūsu datu kopā ir trīs dažādi objektu tipi, šis algoritms var piedāvāt sadalīt mūsu kopu četros vai arī sešos klasteros. Šī algoritma uzvedību var saistīt ar mazo dispersiju (dažos atribūtos dažādi objekti gandrīz neatšķirās).

#### 2. K-Vidējo algoritma rezultātu analīze:

Neskatoties uz to, ka mūsu datu kopai ir piemēroti trīs klases, “Silhouette Score” rāda, ka vislabāka klasterizācija ir pie tikai diviem klasteriem (att. 2.10.). Tas arī norāda uz nelielu atšķirību starp dažiem mainīgajiem.



Att. 2.10. K-Vidējo algoritma piedāvāts “Silhouette Score”

Tomēr, skatoties uz attēliem no 2.5. līdz 2.9. varām redzēt, ka, jo lielāks ir “k” koeficients vai klasteru skaits, jo vairāk objektiem programma nevar piešķirt kategoriju, kāmēr pie  $k = 3$ , aptuveni 10 objekti nav kategorizēti vai ir grūtības ar kategorijas piešķiršanu, kas ir pieteikāmi, salīdzinot ar rezultātiem pie  $k > 3$ .

Mainot distances aprēķinu veidu atkāļ gandrīz neietēkme rezultātus.

### III daļa - Pārraudzītā mašīnmācīšanās

Pedējā daļā ir aprakstīti pārraudzītā mašīnmācīšanās algoritmu darbība.

Tika izvēlēti trīs dažādi algoritmi – “Neironu tīkli”, “kNN” un “AdaBoost”. Visiem trim algoritmiem tiek iedota pilna datu kopa ar 210 ierakstiem. Precizitātes, kā arī citu algoritmu rezultātu vērtības var redzēt 3.1.

#### 1) kNN

Vienkāršākais algoritms no trim ir kNN, kas meklē "k" tuvākos kaimiņus un izvēlas objekta tipu, pamatojoties uz visbiežāk sastopamo no tiem. Pamatvērtība ir  $k = 5$ . Vērtības mainīšana ne vienmēr uzlabo algoritma precizitāti. Tomēr precizitāte mēdz palikt diapazonā no aptuveni 0.890 līdz 0.915, kas ir ļoti augsta vispārējā precizitāte. Mainot attāluma aprēķina veidu, precizitāte būtiski nemainās. Tomēr es atklāju, ka, izmantojot "Mahalanobis" aprēķina metodi ar  $k = 5$ , precizitāte vislabāk uzlabojas līdz 0.952, kas ir vislielākā vērtība, kuru es redzēju precizitātes, kolonna.

#### 2) Neironu tīkli

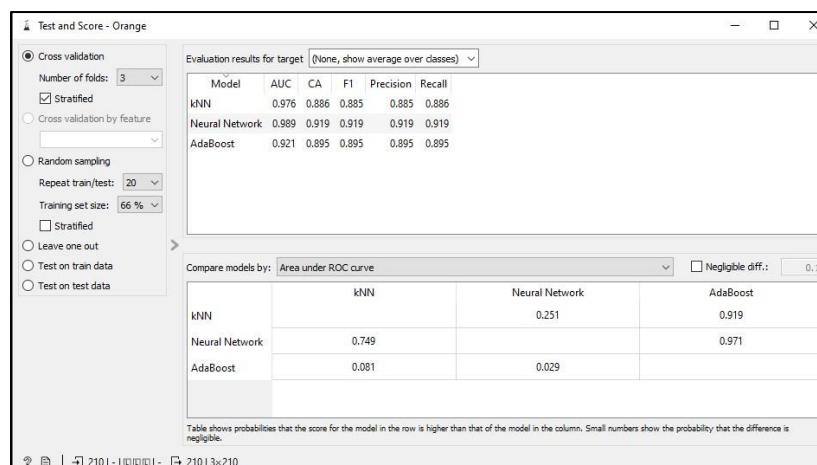
Neironu tīkla algoritms parāda aptuveni tādus pašus rezultātus ar precizitāti aptuveni 0.9-0.93. Pietiek, ja maksimālais iterāciju skaits ir 100. Vairāk iterāciju precizitāti īsti nepalielina. Tomēr 0-100 diapazonā, jo vairāk ir iterāciju, jo labāk.

Opcija "Neironi slēptajos slāņos" darbojas nedaudz neparedzami. Ar vērtību 100 (kas ir sākotnējā vērtība), precizitāte ir visaugstākā ~ 0.91. Mainot to uz 10, vērtība samazinās līdz 0.367. Tomēr, ja šis mainīgais ir 5, precizitāte ir 0.9, kas ir gandrīz visaugstākajā līmenī.

#### 3) AdaBoost

AdaBoost (adaptīvā pastiprināšana) – īsi sakot – algoritms, kas mācās no savām kļūdām. Katra nākamā paaudze (izņemot pašu pirmo, kura nejauši izvēlas atbildi) mācās no iepriekšējās, pārvēršot savas kļūdas par pareiziem atbildem.<sup>2</sup>

AdaBoost algoritmam ir vissliktākā starta precizitāte (kas ir tikai precizitāte ar sākotnējiem parametriem) 0,9, kas joprojām ir ļoti apmierinošs. Kas ir interesanti, kad es mēģināju mainīt mainīgos, precizitāte nedaudz nesvārstās, paliekot pie vērtībām 0,895 vai 0,9.



The screenshot shows the 'Test and Score' window in Orange3. It displays evaluation results for three models: kNN, Neural Network, and AdaBoost. The metrics shown are AUC, CA, F1, Precision, and Recall. Below the main table, there is a 'Compare models by' section showing the Area under ROC curve for each model.

Model	AUC	CA	F1	Precision	Recall
kNN	0.976	0.886	0.885	0.885	0.886
Neural Network	0.989	0.919	0.919	0.919	0.919
AdaBoost	0.921	0.895	0.895	0.895	0.895

	kNN	Neural Network	AdaBoost
kNN		0.251	0.919
Neural Network	0.749		0.971
AdaBoost	0.081	0.029	

Table shows probabilities that the score for the model in the row is higher than that of the model in the column. Small numbers show the probability that the difference is negligible.

Att. 3.1. Trīs algoritmu precizitātes rezultāti.

<sup>2</sup> <https://blog.paperspace.com/adaboost-optimizer/>

## Secinājumi

Spriežot par visiem algoritmu rezultātiem, varu teikt, ka labi sagatavota datu kopa pamatā garantē labus rezultātus jebkuram mašīnmācīšanās algoritmam. Es jau no paša sākuma varēju pateikt, ka trīs klasēs ir viegli atdalāmas, skatoties tikai uz izkliedes diagrammam.

Izmantojot trīs no minētajiem algoritmiem, tas tika pierādīts. Ar vidējo precizitāti  $> 0.9$  (9 no 10 objektiem) algoritmi pareizi uzminēja sēklas veidu.