



RĪGAS TEHNISKĀ UNIVERSITĀTE

Datorzinātnes un informācijas tehnoloģijas fakultāte

**Mašīnmācīšanās algoritmu lietojums**

Otrais praktiskais darbs

### Izstrādāja:

### Artjoms Kučerjavijs, 211RDB105

### 8. grupa, Datorsistēmas

### GitHub saite: https://github.com/TheHappyLemon/AI\_RTU\_DSP332

Rīga, 2023

Contents

[Ievads 3](#_Toc134743070)

[I daļa - Datu pirmapstrāde/izpēte 4](#_Toc134743071)

[II daļa – Nepārraudzītā mašīnmācīšanās 10](#_Toc134743072)

[K-vidējo algoritms 10](#_Toc134743073)

[Hierarhiskā klasterēšana 12](#_Toc134743074)

[III daļa – Pārraudzītā mašīnmācīšanās 15](#_Toc134743075)

[k-tuvāko kaimiņu metode 15](#_Toc134743076)

[Loģistikās regresijas metode 15](#_Toc134743077)

[Neironu tīklu algoritms 15](#_Toc134743078)

[Eksperimentu veikšanā 16](#_Toc134743079)

[Pirmā eksperimenta rezultāti 17](#_Toc134743080)

[Pirmā eksperimenta secinājumi 18](#_Toc134743081)

[Otra eksperimenta rezultāti 19](#_Toc134743082)

[Otra eksperimenta secinājumi 20](#_Toc134743083)

[Trešā eksperimenta rezultāti 21](#_Toc134743084)

[Treša eksperimenta secinājumi 22](#_Toc134743085)

[Orange rīka darbvirsmā 23](#_Toc134743086)

[Secīnājumi 24](#_Toc134743087)

[Izmantota literatūra 25](#_Toc134743088)

# Ievads

Lai atvērtu manu projektu un datu kopu, ir jāiet pēc saites uz manu Github profilu un jālejupielādē sekojoši faili: PD2\_orange\_artkuc.ows un Raisin\_data.xlsx (20. att)

A screenshot of a computer

Description automatically generated with medium confidence

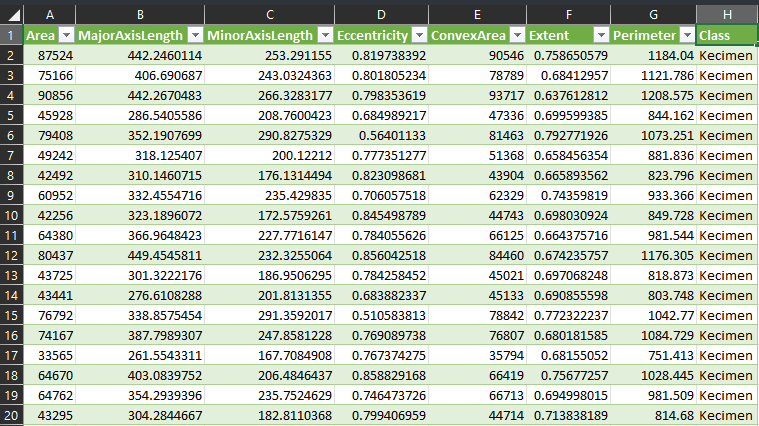
# I daļa - Datu pirmapstrāde/izpēte

Priekš praktiska darba es esmu izvēlējis kopu ar nosaukumu “Raisin Dataset”[[1]](#footnote-1). Kopas autori ir trīs cilvēki: Ilkay CINAR, Murat KOKLU un Sakir TASDEMIR, Selcuk University, Faculty Of Technology, Department Of Computer Engineering. Dota kopa tika izveidota 2020 gadā, pētijuma “Classification of Raisin Grains Using Machine Vision and Artificial Intelligence Methods” rezultātā. Bija izveidotas sistēmas, lai atšķirtu divas rozīnes veidus, kas aug Turcijā - (Kecimen and Besni).

Kopa satur sevi 900 objektus. Šī objekti attiecīgi ir sadalīti divas klasēs. Katrā klasē ietvaros ir precīzi 450 objekti. Datu kopas objektam ir 7 atribūti. Zemāk dotajā tabulā (1. tabula) ir aprakstīti objektu atribūti.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Nr.** | **Atribūta nosaukums** | **Paskaidrojums** | **Datu tips** | **Minimāla vērtība** | **Maksimālā vērtība** | **Vīdēja vērtība** |
| 1 | Area | pikseļu skaits rozīņu grauda robežas | skaitlis | 25387 | 235047 | 87804.12 |
| 2 | MajorAxisLength | Galvenās ass garums, kas ir garāka līnija, ko var uzzīmēt uz rozīnes | skaitlis | 225.63 | 997.292 | 430.93 |
| 3 | MinorAxisLength | Otras ass garums, kas ir īsāka līnijā, ko var uzzīmēt uz rozīnes | skaitlis | 143.711 | 492.275 | 254.488 |
| 4 | Eccentricity | ekscentricitātes mērījums elipsei, kurai ir tādi paši momenti kā rozīnēm. | skaitlis | 0.349 | 0.962 | 0.782 |
| 5 | ConvexArea | Norāda pikseļu skaitu mazāko izliekto čaulu apgabalā, ko veido rozīņu graudiņš. | skaitlis | 26139 | 278217 | 91186.09 |
| 6 | Extent | Sniedz attiecību starp reģionu, ko veido rozīne un kopējā pikseļu skaita ierobežojošajā apgabalā. | skaitlis | 0.38 | 0.835 | 0.7 |
| 7 | Perimeter | attālums starp rozīnes robežām un pikseļiem ap to. | skaitlis | 619.074 | 2697.753 | 1165.907 |
| 8 | Class | Rozīnes klase | kategorija | - | - | - |

1. tabula. Kopas objektu atrībuti

Zemāk ir dots ekrānšāviņš (1. att) no MS Excel ar pirmajiem 20 objektiem no izmantotas datu kopas.

1. att. Kopas objektu atrībuti

Praktisko darbu es pildu ar Orange rīka palīdzību. Pirmkārt es importēju izmantoto datu kopu un uzstādīju “Class” atribūtu kā mērķa mainīgo, visi pārējie atribūti ir parastas pazīmes. Pielietojot rīku “Continuize” es normalizēju pazīmes vērtības intervālā [0;1]. Izmantojamā datu kopā nebija trūkstošo vērtību, tādēļ rīku “Impute” es nelietoju.

Graphical user interface, chart, bar chart

Description automatically generatedPielietojot rīku “Distributions” es esmu ieguvis vairākas histogrammas izpētei. Kā bija zināms iepriekš. Objektu daudzums ir vienmērīgi sadalīts starp divām klāsem. (2. Att)

2. att. Vienmērīgs sadalījums pēc klasēm

Chart

Description automatically generated with low confidenceZemāk dotais attēls ir iegūts ar rīku “Feature statistics”, kas atspoguļo atšķirību pēc katra atribūta divās klasēs. Vislielākā atšķirība ir atrībutos: ConvexArea, Perimeter un Area. Vismazāka dispersija ir atribūtam Extent (0.167). Visliellāka – ConvexArea (0.626) (3.att)

3. att. Feature statistics

3. att. Feature statistics

Chart, scatter chart

Description automatically generatedLai vizuāli labāk atšķirtu klases pēc atribūtiem, es pielietoju rīku Scatter plot. Objekti ir novietotu ļoti tuvu viens otram un visefektīvāk tos var atdalīt pēc atribūtiem ConvexArea un Perimeter. (4. att) kā arī Eccentricity un Perimeter (5. att)

Chart, scatter chart

Description automatically generated

5. att. Grupējums pēc Eccentricity un Perimeter

4. att. Grupējums pēc ConvexArea un Perimeter

**Graphical user interface

Description automatically generated**Pielietojot rīku Correlatiions esmu ieguvis sekojošo skatu (6. att). No tā ir redzams, ka vairāki atribūti korelē. Piemēram, ConvexArea, kurā netiks izmantotā nepārraudzītas mašīnmācīšanās

6. att. Atribūtu korelācija

Chart, histogram

Description automatically generatedKlasēm ir gandrīz normāls sadalījums pēc atribūtiem Extent un Perimeter. 7. attēla ir paradīta rezultāts ar rīku Distributions pēc atribūta Extent.

7. att. Sadalījums pēc Extent

# II daļa – Nepārraudzītā mašīnmācīšanās

## K-vidējo algoritms

K vidējo algoritms ir pirmais darbā lietojams algoritms, kas balstās uz diviem parametriem grupē objektus K klasteros. K vidējo algoritmam ir viens hiperparametrs K – klasteru skaits, uz kuru tiks dalīti objekti.

Graphical user interface

Description automatically generatedLietojot Orange rīka eksistējošo algoritmu K-means, esmu sadalījis savus objektus klasteros. Orange rikā ir izmantojams Sillhouette Score, kurš paradīja, kā manā gadījuma vislabākā sadalīšana notiks divos klasteros (8. att) .

8. att. K-vidējo rezultāts

Labākais rezultāts tika iegūts, sadalot kopas objektus divos klasteros. Svarīgi ari ir tas, ka es jau esmu normalizējis savus datus darba sakumā, tāpēc neizvēlējos opciju “Normalize columns”. Zemāk dotie dati tika iegūt pielietojot Orange rīkā Silhouette plot.

Pielietojot parametra K vērtība kā 2 K-vidējo algoritms sadalīja manus objektus 2 klasteros ar Silhouette vērtībām 0.350 un 0.482 attiecīgi. Kopējais Sillhouette Score ir 0.445. C1 Bija tikai dažādi Kecimen klases objekti. C2 bija tikai daži Besni objekti.

Pielietojot parametra K vērtība kā 3 K-vidējo algoritms sadalīja manus objektus 3 klasteros ar Silhouette vērtībām 0.330 un 0.299, un 0.312 attiecīgi. Kopējais Sillhouette Score ir 0.313. C1 Bija ap 1/6 Kecimen klases objekti. C2 bija ap 1/6 Besni objekti. C3 bija tikai daži Kecimen objekti.

Pielietojot parametra K vērtība kā 4 K-vidējo algoritms sadalīja manus objektus 4 klasteros ar Silhouette vērtībām 0.305 un 0.243, un 0.366, un 0.296 attiecīgi. Kopējais Sillhouette Score ir 0.311. C1 Bija tikai daži Kecimen objekti. C2 bija tikai daži Besni objekti. C3 bija pārsvara Kecimen objekti. C4 bija pārsvarā Besni objekti.

Pielietojot parametra K vērtība kā 5 K-vidējo algoritms sadalīja manus objektus 5 klasteros ar Silhouette vērtībām 0.236 un 0.229, un 0.282, un 0.225, un 0.344 attiecīgi. Kopējais Sillhouette Score ir 0.275. C1 Bija tikai daži Kecimen objekti. C2 bija tikai daži Besni objekti. C3 bija vienmērīgs sadalījums starp diviem klasēm. C4 bija pārsvarā Kecimen objekti. C5 bija pārsvarā Kecimen objekti.

Pielietojot parametra K vērtība kā 6 K-vidējo algoritms sadalīja manus objektus 6 klasteros ar Silhouette vērtībām 0.250 un 0.251, un 0.334, un 0.265, un 0.264, un 0.219. Kopējais Sillhouette Score ir 0.275. C1 Bija tikai daži Kecimen objekti. C2 bija tikai daži Besni objekti. C3 bija pārsvara Kecimen objekti. C4 bija pārsvarā Besni objekti. C5 bija gandrīz vienmērīgs sadalījums. C6 bija tikai daži Kecimen objekti.

Zemāk ir redzams objektu atdalījums pēc K-vidējo algoritma ar K = 2. (9. att)

Chart, scatter chart

Description automatically generated

9. att. K-vidējo rezultāts

## Hierarhiskā klasterēšana

Hierarhiskā klasterēšana ir mašīnmācīšanās algoritms, kas neprasa definēt klasteru skaitu pirms algoritma darbības. Tā ir galvenā atšķirība starp K-vidējo un Hierarhiskā klasterēšanas algoritmiem. Priekš klasterešanos es esmu izvēlējies Pilnīgo (Complete) saistīšanu.

Chart

Description automatically generatedPirmajā eksperimentā esmu izvietojis atdalošo līniju tā, lai iegūtu divus klasterus. Sanāca, ka pirmajā klasteri C1 bija dažas reizes vairāk objektu nekā klasterī C2. Tomēr C1 klasterī bija apmēram vienādi abu klašu pārstāvji, bet C2 klasterī bija tikai Besni klases objekti (10.att)

10. att. 2 klasteri

Otrajā eksperimentā esmu izvietojis atdalošo līniju tā, lai iegūtu četrus klasterus. Sanāca, ka klasteros C1 un C2 bija abu klašu pārstāvji, tomēr klasterī C1 bija vairāk Besni objektu un klasterī C2 pretēji bija vairāk Kecimen objektu. Klasteris C3 sanāca mazs no 6 objektiem 2 – Kecimen, 4 – Besni. Un klasterī C4 bija tikai Besni objekti(11. att)

Chart, box and whisker chart

Description automatically generated

11. att. 4 klasteri

Chart, box and whisker chart

Description automatically generatedTrešajā eksperimentā esmu izvietojis atdalošo līniju tā, lai iegūtu sešus klasterus. Sanāca, ka klasterī C1 bija abu klašu pārstāvji, tomēr vairāk Besni objektu. C2 klasterī jau bija tikai divi Besni objekti. C3 Klasterī bija abu klašu pārstāvji, bet pārsvarā Kecimen objekti. C4 un C5 klasterī sanāca mazi. C4 bija 3 Besni objektu un 2 Kecimen. C5 tikai trīs Besni objekti. C6 klasterī bija tikai Besni objekti. (12. att)

12. att. 6 klasteri

Eksperimentu rezultātā es varu secināt, ka neskatoties uz to, ka klases, protams, ir ļoti līdzīgas, ko varēja redzēt darba sākumā attēlos numurs 4 un 5, tos tomēr var veiksmīgi atdalīt viens no otra ar nelielām kļūdām.

# III daļa – Pārraudzītā mašīnmācīšanās

## k-tuvāko kaimiņu metode

Pirmais algoritms, kas ir izvēlēts pārraudzītām mašīnmācīšanām manā darbā ir k-tuvāko kaimiņu metode jeb kNN algoritms. Tas ir mūsdienās plaši izmantojams algoritms. Tas ir populārs, jo nav pārāk sarežģīts un to var pielietot, gan ja ir klasificēšanas uzdevums, gan ja, piemēram ir, regresijas uzdevums. Tā būtība īsumā ir sekojoša: Ja vajag uzzināt kāda objekta klasi, mēs apskatīsim visus zināmos mums objektus un atradīsim to, kas ir pēc zināmiem atribūtiem vislīdzīgākais. Lai saprastu, ka objekts ir līdzīgs citam objektam ir lietojams kāds specifisks noteikšanas likums, piemēram, attālums (tuvums). Piemēram ir Eiklīda attālums un Manhetenas attālums. Gadījumā, ja objektam ir kategoriskas pazīmes, nevis, piemēram skaitliskas, tas ir pielietojams Heminga Attālums.

kNN algoritmam ir viens hiperparametrs. Tas ir kaimiņu skaits(K), ar kuriem tiks veikts salīdzinājums. Protams, svarīga arī ir metrika, kura ir lietojamā, lai atrastu attālumu.

Esmu izvēlējies šo algoritmu, tāpēc kā tas bija apskatīts lekcijas laikā un ir daudz informācijas lekcijas materiālos, es ar pirmo reizi esmu sapratis tā būtību un tas ir plaši izmantojams algoritms.

## Loģistikās regresijas metode

Otrais algoritms, kas ir izvēlēts pārraudzītām mašīnmācīšanām manā darbā ir loģistiskā regresija. To plaši izmanto, ja uzdevumā ir divi klases varianti – to izmanto binārajā klasifikācijā. Ko tas nozīme ir tas, ka tas nosaka datu objekta piederību vienai no divām opcijām, klasēm, iespējam. To var salīdzināt ar Boolean vērtību: True/False. Šī vērtības izslēdz otro variantu: labi/slikti, 1/0 un tā tālāk. Lineārā regresija kā ievades datus saņems kāda objekta pazīmes (tā vērtības) un atpakaļ kā rezultātu izdos kādu reālo skaitļu. Loģistiskā regresija atrod arī matemātisko funkciju, kas saista objekta pazīmes ar to klasi.

Metodei ir viens hiperparametrs – Strength, kas var būt diapazonā no 1000 līdz 0.001. 1000 ir vājš, 0.001 ir stiprs.

Esmu izvēlējies šo algoritmu, tāpēc kā tas bija apskatīts lekcijas laikā un ir daudz informācijas lekcijas materiālos ar attēliem, grafikiem un funkcijām. Es uzskatu, kas tas ir labs variants manā gadījuma, jo man ir bināra klasifikācijas.

## Neironu tīklu algoritms

Trešais algoritms, kas ir pielietots manā darba ir Neironu tīklu algoritms. Šis algoritms ietver sevi 4 hiperparametrus, kuri ir sekojoši: 1) Neironu skaits slēptajos slānos, tas atbild par tīkla struktūru. 2) Funkcija, kura nosāka ka neironam būtu jāreaģē uz saņemtiem datiem. 3) Regulārizācija - neirona tīkla mācīšanas ātrums. 4) Maksimālas iterāciju skaits, tas ir šis metodes galvenais ierobežojums.

## Eksperimentu veikšanā

Priekš apmācības darbā es esmu pielietojis Cross validation metode Orange rīka, kurā pēc noklusējumā 66% procenti no visiem kopas objektiem, kas ir tieši 594 objekti.

Ar katru no metodēm es paveicis trīs eksperimentus. Eksperimentu apraksts ar pielietotiem hiperparametriem ir atrodams zemāk, 2. tabulā.

|  |  |  |  |
| --- | --- | --- | --- |
| **Eksperimenta Nr.** | **kNN** | **Loģistiskā regresija** | **Neironu tikli** |
| **1** | K = 5  Euclidean attālums | C = 1 | Neironi: 100,100  activation: logistic  a = 0.1  iterāciju ierobežojums = 1000 |
| **2** | K = 3  Manhetena attālums | C = 0.1 | Neironi: 10,10,5  activation: logistic  a = 0.1  iterāciju ierobežojums = 500 |
| **3** | K = 8  Euclidean attālums | C = 40 | Neironi: 5,6,7,8  activation: logistic  a = 1  iterāciju ierobežojums = 200 |

2. tabula. Eksperimentu apraksts.

## Pirmā eksperimenta rezultāti

Zemāk ir pirmā eksperimenta rezultāti (13,14 attēli):

Chart

Description automatically generated

13.att ROC Analysis 1-ājā eksperimentā

A screenshot of a computer

Description automatically generated with medium confidence

14.att. Kļūdu matricas 1-ājā eksperimentā

## Pirmā eksperimenta secinājumi

Pirma eksperimenta (2. tab) rezultātā, pielietojot visus trīs algoritmus, ar aprakstītiem hiperparametriem, es ieguvus sekojošas F1 vertības. Neironu tīkli – 0.853, kNN – 0.821 un Loģistiska regresija deva rezultātu vienādu ar 0.842. Balstoties uz šiem rezultātiem, es varu secināt, ka pēc F1 parametra vislabāko rezultātu es esmu ieguvis pielietojot Neironu tīklu algoritmu. Tomēr pēc 13. attēla ir spilgti redzāms, ka kNN un logistiska regresija dod diezgan lidzīgus rezultātus. Jo visu trīs algoritmu ROC analysis grafiki, gandriz pārklājas un veido gandrīz taisno stūri, uz ko ir jātiecas.

Spriežot pēc esoša eskperimenta kļūdu matricam, es varu secināt sekojošo: Neironu tīklu algoritma bija vismazāk kļūdu, tas kļūdaini aprēķināja tikai 75 objektus, kas ir vienāds apmēram 12 procenti no visiem 630 datu objektiem, kas ir labs rezultāts. kNN un loģistiska regresija deva nedaudz sliktātkus rezultātus un kļūdaini aprēķināja 84 un 82 objektus, kas ir 13.33 procenti un 13 procenti attiecīgi. Nedaudz sliktāk, bet atkal, ļoti tuvu.

Visi trīs algoritmi parādiju labu efektīvitāti ar uzdotiem hiperparamētriem, izpilde ari neprasīja daudz laika. Var būt rezultāts bija citādāks, ja man būtu vairākas klašu iezimes, tomēr ar divām klasēm rezultāts ir apmierinošs.

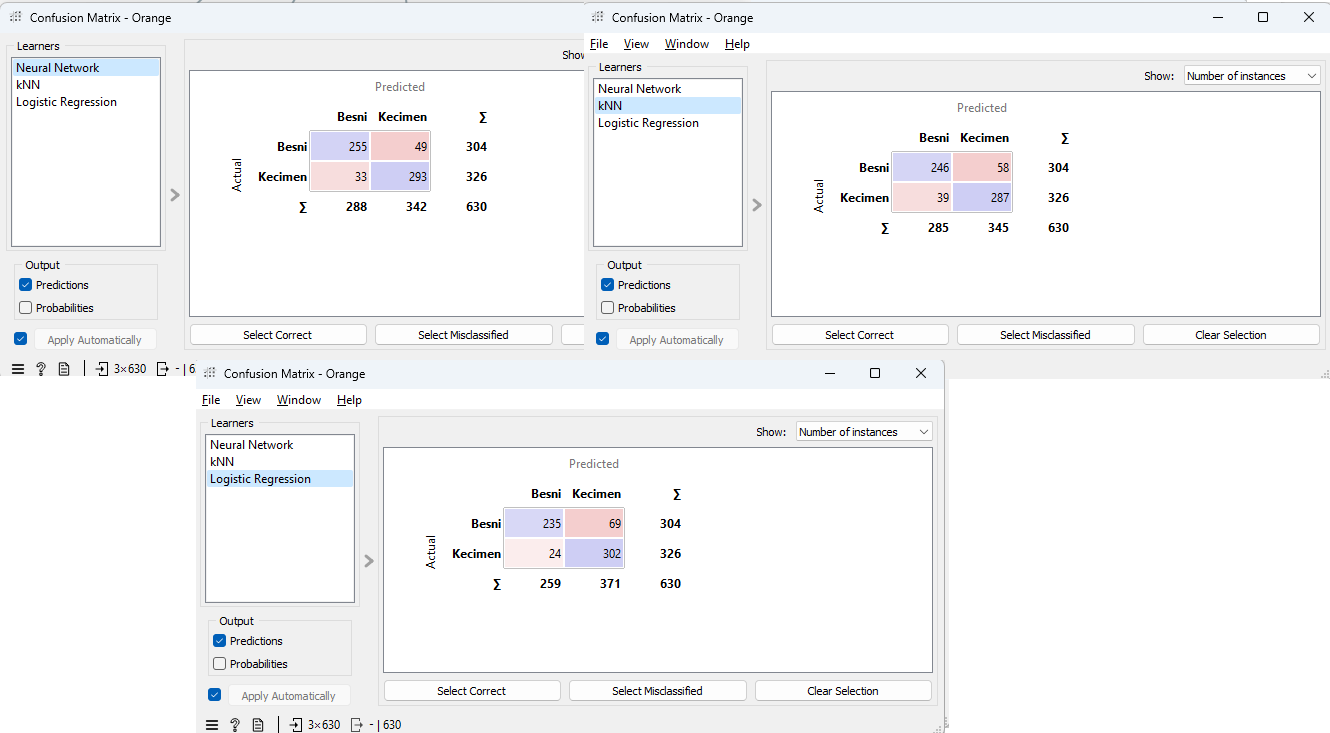
## Otra eksperimenta rezultāti

Zemāk ir otra eksperimenta rezultāti (15,16 attēli):

Chart, line chart

Description automatically generated

15.att ROC Analysis 2-ājā eksperimentā.



16.att. Kļūdu matricas 2-ājā eksperimentā

## Otra eksperimenta secinājumi

Otra eksperimenta (2. tab) rezultātā, pielietojot visus trīs algoritmus, ar aprakstītiem hiperparametriem, es ieguvus sekojošas F1 vertības. Neironu tīkli – 0.863, kNN – 0.807 un Loģistiska regresija deva rezultātu vienādu ar 0.871. Balstoties uz šiem rezultātiem, es varu secināt, ka pēc F1 parametra vislabāko rezultātu es esmu ieguvis pielietojot Loģistiskas algoritmu. Tomēr pēc 13. attēla ir spilgti redzāms, ka Neironu tiklus algoritms dod diezgan lidzīgus rezultātus. Jo visu šō algoritmu ROC analysis grafiki, gandriz pārklājas un veido gandrīz taisno stūri, uz ko ir jātiecas. kNN algoritms ir nedaudz sliktāk un tā grafiks ir ari tālāķ

Spriežot pēc esoša eskperimenta kļūdu matricam, es varu secināt sekojošo: Loģistiskas regresijas algoritma bija vismazāk kļūdu, tas kļūdaini aprēķināja tikai 82 objektus, kas ir vienāds apmēram 13 procenti no visiem 630 datu objektiem, kas ir labs rezultāts. kNN un loģistiska regresija deva nedaudz sliktātkus rezultātus un kļūdaini aprēķināja 97 un 91 objektus, kas ir 13539 procenti un 14.44 procenti attiecīgi. Nedaudz sliktāk, bet atkal, diezgani tuvu.

Visi trīs algoritmi parādiju labu efektīvitāt, izņemot kNN, ar uzdotiem hiperparamētriem, izpilde ari neprasīja daudz laika. Var būt rezultāts bija citādāks, ja man būtu vairākas klašu iezimes, tomēr ar divām klasēm rezultāts ir apmierinošs.

## Trešā eksperimenta rezultāti

Zemāk ir trešā eksperimenta rezultāti (17,18 attēli):

Chart

Description automatically generated

17.att ROC Analysis 3-ājā eksperimentā.

A screenshot of a computer

Description automatically generated with medium confidence

18.att. Kļūdu matricas 3-ājā eksperimentā

## Treša eksperimenta secinājumi

Treša eksperimenta (2. tab) rezultātā, pielietojot visus trīs algoritmus, ar aprakstītiem hiperparametriem, es ieguvus sekojošas F1 vertības. Neironu tīkli – 0.325, kNN – 0.874 un Loģistiska regresija deva rezultātu vienādu ar 0.857. Balstoties uz šiem rezultātiem, es varu secināt, ka pēc F1 parametra vislabāko rezultātu es esmu ieguvis pielietojot Loģistiskas regresijas algoritmu. Tomēr pēc 13. attēla ir spilgti redzāms, ka kNN algoritms dod diezgan lidzīgus rezultātus. Jo šo divu algoritmu ROC analysis grafiki, gandriz pārklājas un veido gandrīz taisno stūri, uz ko ir jātiecas. Tomēr Neironu tīklu algoritms ar tik māzajiem parametriem deva ļoti slikto rezultātu salidzinot ar citiem algoritmiem.

Spriežot pēc esoša eskperimenta kļūdu matricam, es varu secināt sekojošo: Neironu tīklu algoritma bija visvairāk kļūdu, tas kļūdaini aprēķināja pat 304 objektus, kas ir vienāds apmēram 46 procenti no visiem 630 datu objektiem, kas ir slikts rezultāts. kNN un loģistiska regresija deva lidzīgus pirmājas eskperimentam rezultātus un kļūdaini aprēķināja 84 un 78 objektus, kas ir 13.33 procenti un 12.8 procenti attiecīgi. Nedaudz labāk.

Visi trīs algoritmi izņemot Neironu tīklu algoritmu parādiju labu efektīvitāti ar uzdotiem hiperparamētriem, izpilde ari neprasīja daudz laika. Var būt rezultāts bija citādāks, ja man būtu vairākas klašu iezimes, tomēr ar divām klasēm rezultāts ir apmierinošs.

# Orange rīka darbvirsmā

Zemāk ir redzāma mana darbvirsmā Orange rīkā, beidzot visu otro praktisko darbu (19. att). Augšā ir vecākie objekti (priekš analizes un izpētes), lejā ir jaunākie objekti (priekš mašinmacīšanām)

Map

Description automatically generated

19.att. Darbā gaitā izveidots Orange projekts

# Secīnājumi

Eksperimentu rezultāta varu pateikt, kā abas klases no datu kopas ir atdalāmi ar klasterizāciju, kā ari ir atpazīti, pielietojot dažadus algoritmus. Kļudas bija, bet te nebija parāk lielas. palielinot hiperparamtrus, var samazināt šo kļudu un iegūst precīzākus rezultātus. Gadījumā, kad hiperparametri bija speciāli uzstādīti ne parāk labie, rezulāts arī bija pārsvāra kļūdains. Man personiski darbs patīka, jo ļava paspēleties ar daudziem datiem, izpētit dažadus algoritmus diezgan ātri un vienkārši. Rezultāts, manā izpratnē ir sasniegts.

# Izmantota literatūra

1. Vietne ar publiski pieejamām datu kopām. *Izmantots 11. maijā 2023. gadā. Pieejams:* <https://www.muratkoklu.com/datasets/>
2. *CINAR I., KOKLU M. and TASDEMIR S., (2020). Classification of Raisin Grains Using Machine Vision and Artificial Intelligence Methods, Gazi Journal of Engineering Sciences.*

*Izmantots 11. maijā 2023. gadā. Pieejams:* <https://dergipark.org.tr/tr/download/article-file/1227592>

1. *Vietne ar vairākam publiskām datu kopām, satur visparīgo aprakstu par manu izvēlēto kopu. Izmantots 11. maijā 2023. gadā. Pieejams:* <https://www.kaggle.com/datasets/muratkokludataset/raisin-dataset>
2. *Krievijas zinātņu akadēmijas filiāle - volgas baseina ekoloģijas institūts. Raksts par sadalījumiem. Izmantots 11. maijā 2023. gadā. Pieejams:* <http://www.ievbras.ru/ecostat/Kiril/Library/Book1/Content350/Content351.htm>
3. Dr.sc.ing., Dr.paed. asoc.prof. Alla Anohina-Naumeca Mākslīgā intelekta un sistēmu inženierijas katedra Datorzinātnes un informācijas tehnoloģijas fakultāte Rīgas Tehniskā universitāte. Kurss mākslīga intelekta pamati, 4.3. TĒMA: NEPĀRRAUDZĪTĀ MAŠĪNMĀCĪŠANĀS. Pieejams ORTUS vidē.
4. Dr.sc.ing., Dr.paed. asoc.prof. Alla Anohina-Naumeca Mākslīgā intelekta un sistēmu inženierijas katedra Datorzinātnes un informācijas tehnoloģijas fakultāte Rīgas Tehniskā universitāte. Kurss mākslīga intelekta pamati, 4.2. TĒMA: PĀRRAUDZĪTĀS MAŠĪNMĀCĪŠANĀS ALGORITMI. Pieejams ORTUS vidē.

1. CINAR I., KOKLU M. and TASDEMIR S., (2020). Classification of Raisin Grains Using Machine Vision and Artificial Intelligence Methods, Gazi Journal of Engineering Sciences, vol. 6, no. 3, pp. 200-209, December, 2020, DOI: https://doi.org/10.30855/gmbd.2020.03.03 [↑](#footnote-ref-1)