Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Nejc Mušič

Razlaga klasifikatorjev na podlagi podkonceptov

DIPLOMSKO DELO

UNIVERZITETNI ŠTUDIJSKI PROGRAM
PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA

Mentor: prof. dr. Marko Robnik Šikonja

Ljubljana, 2023

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Kandidat: Nejc Mušič
Naslov: Razlaga klasifikacijskih problemov na podlagi podkonceptov
Vrsta naloge: Diplomska naloga na univerzitetnem programu prve stopnje Računalništvo in informatika
Mentor: prof. dr. Marko Robnik Šikonja

Opis:
Na nekaterih področjih je uporaba napovednih modelov strojnega učenja pogojena z možnostjo razlage napovedi. Večina obstoječih pristopov razlage tvori razlage posameznih napovedi v obliki faktorjev pomembnosti vhodnih atributov. Ta način ni vedno primeren, saj je prostor vhodnih atributov mnogokrat zelo velik ali človeku nerazumljiv. Preizkusite alternativno idejo razlage, ki iz učnih primerov danega razreda najprej tvori gruče, ki jih nato skuša interpretirati kot podkoncepte danega razreda z uporabo pravil in prototipov. Pristop preizkusite na nekaj umetnih in realnem problemu.

Title: Explanation of Classifiers Based on Subconcepts

Description:

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
MDEC	multidiversified ensemble clustering	Večdimenzionalno multidiversificirano združevanje v gruče
XAI	explainable artificial intelligence	razložljiva umetna inteligenca
SHAP	SHapley Additive exPlanations	Shapleyjeve aditivne razlage
DBSCAN	Density-Based Spatial Clustering of Applications with Noise	enako ime
HDBSCAN	Hierarchical Density-Based Spatial Clustering of Applications with Noise	enako ime
XGBoost	Extreme Gradient Boosting	enako ime
DBCV	Density-Based Clustering Validation	enako ime
t-SNE	t-distributed Stochastic Neighbor Embedding	enako ime
PCA	Principal component analysis	enako ime
UMAP	Uniform Manifold Approximation and Projection	enako ime
MNIST	Modified National Institute of Standards and Technology	enako ime

Povzetek

Naslov: Razlaga klasifikatorjev na podlagi podkonceptov

Avtor: Nejc Mušič

Pri nekaterih klasifikacijskih problemih je poleg natačnosti in zanesljivosti pomembna tudi razlaga odločitev (npr. v medicini). Ker dajejo ponavadi kompleksni modeli, kot so na primer nevronske boljše, boljše rezultate jih velikokrat raje uporabimo kot npr. linearno regresijo, ki je dobro razložljiva. Diplomska naloga se ukvarja z razlago kompleksnih modelov. Na podlagi gručenja podatkov iz posameznega razreda razložimo klasifikator s pomočjo odločitvenih pravil, medoidov gruč in s SHAP vrednostmi. V sklopu diplomske naloge izvedemo eksperimente za algoritme gručenja na nizko in visoko dimenzionalnih podatkovnih množicah, kjer pokažemo dobre in slabe lastnosti in ustreznost glede na podatke. Naša metoda razlage potrebuje razdelitev primerov istega razreda na več gruč, kar se je pri realnih podatkovnih množicah izkazalo za redek pojav. Z eksperimenti ugotovimo, da je naša razlaga zahtevna za interpretacijo in ni avtomatizirana. Nekatere metode razlage vrnejo vizualizirane rezultate, kar pomaga pri interpretaciji.

Ključne besede: razložljiva umetna inteligenca, MDEC, DBSCAN, HDBSCAN, KMEANS, SHAP, odločitvena pravila, medoid/prototip.

Abstract

Title: Explanation of Classifiers based on Subconcepts

Author: Nejc Mušič

In some classification problems, in addition to accuracy and reliability, the explanation of decisions is also important (e.g., in medicine). Complex models, such as neural networks, often outperform simpler models like linear regression, which are inherently interpretable. The thesis addresses the challenge of explaining such complex models. Based on data clustering, we explain the classifier using decision rules, cluster medoids, and SHAP values. We conduct experiments on clustering algorithms using low and high-dimensional datasets. These experiments aim to highlight both the strengths and weaknesses of these algorithms, as well as their suitability with respect to the specific dataset. Our explanation method relies on the division of instances from the same class into multiple clusters, a phenomenon which is infrequent in real-world datasets. Through these experiments, we discover that our explanation method is challenging to interpret and lacks full automation. Some explanation components provide visualized results, which assist in the interpretation process.

Keywords: XAI, MDEC, DBSCAN, HDBSCAN, KMEANS, SHAP, decision rules, medoid/prototype.

Poglavje 1

Uvod

V zadnjem času smo priča izjemnim dosežkom na področju strojnega učenja. Kompleksni modeli pri klasifikacijskih problemih omogočajo dobro natančnost pri napovedovanju. Slaba lastnost t.i. "black box" modelov je nerazumljivost, predvsem kako in zakaj so sprejeli določene odločitve. Za primer lahko vzamemo globoke nevronske mreže ali kompleksne ansamble (XGBoost), ki imajo na tisoče ali celo milijone parametrov. To pomeni, da so njihovi odločitveni procesi izjemno zapleteni, kar otežuje razumevanje, kako določen vhod vpliva na izhod. Z diplomsko nalogo želimo prispevati k razvoju razlage kompleksnih modelov ter omogočiti boljše razumevanje njihovega delovanja, kar bo pripomoglo k dodatni izboljšavi modelov. Prav tako je razumevanje odločitev modelov ključno za zaupanje v njihove rezultate ter za širšo sprejetost uporabe umetne inteligence v različnih kritičnih domenah (npr. medicina, varnost računalniških omrežij, letalska in avtomobilska industrija).
Postopek razlage, ki ga preizkusimo, je naslednji. Klasifikator je naučen na učni množici, na kateri poženemo algoritme za gručenje. Za avtomatski izbor parametrov uporabimo hevristiko silhuete in indeks DBCV. Pri izboru najboljšega gručenja si pomagamo z vizualizacijami podatkov v dvodimezionalnem prostoru (za to bomo uporabili algoritem t-SNE pri visokodimenzionalnih podatkih). Predpogoj razlage je ločenost podatkov na gruče, ki so medsebojno dobro ločljive in prisotnost posameznega razreda v dveh ali več

različnih gručah. Nato posamezne gruče razložimo z odločitvenimi pravili po konceptu ena gruča proti vsem ostalim. Preverimo prisotnost razredov v gruči. V primeru, da je gruča homogena (primeri pripadajo enemu razreda), lahko gručo predstavimo z medoidom. V primeru, da je gruča heterogena, znotraj gruče uporabimo odločitvena pravila, ki ločijo primere različnih razredov. Dodatno izračunamo vrednosti SHAP za vsako gručo in jih predstavimo z grafom. Vrednosti SHAP nam razložijo pomembnost atributov za naš klasifikator, kar dodatno pomaga pri interpretaciji odločitvenih pravil in medoidov.

Nov primer z algoritmom za gručenje umestimo v najbližjo gručo in s tem razložimo z zgornjimi metodami. Dodatno izračunamo vrednosti SHAP za novi primer.

V zadnjih letih so bili razviti številni pristopi za razlago kompleksnih modelov. Pristop s CBR (sklepanje iz primerov) najde pomembne značilke na podlagi lokalnih informacij in nato zbere primer, ki je bil pomemben za izgradnjo modela kot primer razlage [18]. Z odločitvenimi drevesi in odločitvenimi pravili, dobimo vpogled v odločitvene meje v podatkih in poenostavljeno predstavitev množice podatkov [1]. Vrednosti SHAP veljajo za dobro metodo razlage pomembnosti atributov, zato so bili tudi uporabljeni na različnih področjih, kot je na primer obdelava naravnega jezika (Natural Language Processing) [14]. Še nekaj sorodnih del naštejemo v podpoglavju 4.6 pri primerjavi z obstoječimi razlagami.

V diplomski nalogi bomo predstavili naslednja poglavja: za uvodom sledi predstavitev uporabljenih tehnologij in izbor parametrov za algoritme, nato opis pristopa s podskupinami, kjer bo predstavljen preprost ilustrativni primer razlage, zatem poglavje Evalvacija, kjer bodo predstavljene podatkovne množice, potem primerjava algoritmov gručenja na umetnih podatkovnih množicah in realni podatkovni množici MNIST, sledijo ocene smiselnosti prototipov (medoidov), pravil in poglavje o številu gruč, poglavje se konča z razlago realne podatkovne množice KDD99, kjer bodo podane še druge obstoječe razlage. Diplomska naloga se konča z zaključkom, kjer bomo na

kratko povzeli dobljene rezultate opisanih metod. Izpostavili bomo dobre in slabe (omejitve pristopa) lastnosti ter podali mnenje o nadaljnih raziskavah na tem področju.

Poglavje 2

Predstavitev tehnologij

V poglavju predstavimo algoritme za gručenje (podpoglavje 2.1), hevristike za oceno gručenj (podpoglavje 2.2), uporabljene klasifikatorje (podpoglavje 2.3), metode razlage (podpoglavje 2.4) in tehniko zmanjšanja dimenzionalnosti (podpoglavje 2.5). Vse naštete tehnologije uporabimo pri razlagi.

2.1 Algoritmi za gručenje

2.1.1 MDEC

Algoritem "Multidiversified Ensemble Clustering"[7] je zasnovan za gručenje visokodimenzionalnih podatkov. Združuje več algoritmov gručenja za izboljšanje splošne uspešnosti gručenja. Algoritem MDEC ustvari veliko število raznolikih metrik z naključno skalirano eksponentno funkcijo podobnosti in jih poveže z naključnimi podprostori. S tem dobimo pare metrika-podprostor. Na osnovi matrik podobnosti, pridobljenih iz metrika-podprostor parov, nato sestavi ansambel raznolikih gručenj. Zatem uporabi kriterij, zasnovan na entropiji, za oceno raznolikosti združevanj glede na gruče, na osnovi katerega uporabi tri specifične algoritme ansambelskega združevanja (angl. consensus functions). Več informacij o algoritmu je na voljo v članku . Pri algoritmu smo opredelili le parameter za število skupin.

2.1.2 KMEANS

K-means algoritem [6] je iterativni algoritem, ki poskuša razdeliti nabor podatkov v K predhodno določenih neprekrivajočih se podskupin (gruč), pri čemer vsaka podatkovna točka pripada le eni skupini. Algoritem iterativno dodeljuje primere najbližjemu centroidu in zatem premika centroide. Pri algoritmu smo opredelili le parameter za število skupin.

2.1.3 DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [8] je algoritem za gručenje, ki temelji na osnovi gostote podatkovnih točk v prostoru. Algoritem je sposoben zaznati gruče v podatkih, ki imajo različne oblike ter hkrati zaznati osamelce (šum). Pri uporabi algoritma je ključna izbira dveh parametrov - epsilon in min samples. Parameter epsilon določa največjo razdaljo med dvema podatkovnima točkama, da se štejeta kot del iste skupine, parameter min samples pa je celoštevilski parameter, ki določa najmanjše število podatkovnih točk znotraj razdalje epsilon od osrednje točke, da se ta osrednja točka šteje za skupino (skupina mora imeti vsaj min samples osrednjih točk, da se šteje za veljavno skupino). Vzamemo predlagano vrednost min samples = 2 * dim, kjer je dim dimenzija podatkovne množice. Za vrednost parametra epsilon je predlagana tehnika, ki izračuna povprečno razdaljo med vsako točko in njenimi k najbližjimi sosedi, kjer je k enak vrednosti min samples. Povprečne k-razdalje nato prikažemo v naraščajočem vrstnem redu na grafu. Optimalna vrednost za epsilon se nahaja pri točki največje ukrivljenosti (tj. kjer ima graf največji nagib) [16]. Največji nagib na grafu smo našli s pomočjo algoritma Kneed [9]. Pri obeh parametrih smo poskusili tudi okoliške vrednosti, ki se nahajajo okoli izbranih vrednosti parametrov min samples in epsilon.

2.1.4 HDBSCAN

HDBSCAN (Hierarchical Density-Based Spatial Clustering of Applications with Noise) [13] izvaja algoritem DBSCAN pri različnih vrednostih epsilon in združi rezultate, da najde gručenje, ki zagotavlja najboljšo stabilnost pri različnih vrednostih epsilon. To omogoča HDBSCAN-u, da najde gruče z različnimi gostotami (za razliko od DBSCAN-a) in je bolj odporen na izbiro parametrov [13]. Pri HDBSCAN algoritmu sta pomembna parametra min cluster size in min samples [4]. Določili smo ju kot: min cluster size = (ix1*5) in min samples = (ix2+2)*5, kjer $ix1 \in [2, 10]$ in $ix2 \in [0, 8]$ in $ix_1, ix_2 \in \mathbb{N}$.

2.2 Hevristike za oceno gručenj

2.2.1 Koeficient silhuete

Koeficient silhuete [19] je metoda za ocenjevanje kakovosti gručenja, ki se osredotoča na merjenje, kako so točke v isti gruči podobne med seboj v primerjavi z drugimi gručami. Njegova glavna naloga je ponuditi kvantitativno merilo za oceno, kako dobro so točke znotraj iste gruče povezane in kako dobro so razmejene od drugih gruč. Za oceno smo uporabili hevristiko povprečne silhuete, ki se ne osredotoča na oceno kakovosti posameznih točk, kot to počne koeficient silhuete za vsako točko, temveč daje celostno sliko o rezultatu gručenja.

2.2.2 Indeks DBCV

Indeks DBCV (Density-Based Clustering Validation) [15] je mera kakovosti gručenja, ki se uporablja za ocenjevanje učinkovitosti algoritmov za gručenje. Mera kot je koeficient silhuete je primerna za ocenjevanje gruč s sferičnimi oblikami, medtem ko algoritmi, ki temeljijo na gostoti podatkovnih točk, velikokrat zaznajo nepravilne oblike, kjer se bolje odreže indeks DBCV. Izračun indeksa DBCV tudi temelji na gostoti gruč.

2.3 Klasifikatorji

2.3.1 XGBoost

XGBoost (Extreme Gradient Boosting) [17] je zmogljiva in priljubljena metoda strojnega učenja. Gre za algoritem, ki kombinira moč odločitvenih dreves in tehnike gradientnega spusta za doseg visoke točnosti. XGBoost razložimo s konceptoma Bagging in Gradient Boosting. Pri Bagging-u so osnovni klasifikatorji naučeni na naključnih podmnožicah originalnih podatkov. Končni rezultat je pridobljen z glasovanjem ali povprečenjem. Boosting je tehnika, kjer zgradimo močen klasifikator s pomočjo veliko slabših. Pri Gradient Boost tehniki klasifikator, ki je naslednjik, popravlja napako svojega prednika. Pri XGBoost se uteži dodelijo vsem neodvisnim spremenljivkam, ki se nato vstavijo v odločitveno drevo. Nato odločitveno drevo napove rezultat. Utež spremenljivk, ki jih drevo napove napačno, se poveča. Te spremenljivke se zatem vstavijo v drugo odločitveno drevo. Posamezne klasifikatorje nato združimo, da dobimo natančnejši model.

2.3.2 Logistična regresija

Logistična regresija [3] se pogosto uporablja v strojnem učenju za reševanje klasifikacijskih problemov. Metoda napoveduje verjetnosti, da bo določen vhodni primer spadal v enega izmed razredov. Model logistične regresije temelji na linearni kombinaciji neodvisnih spremenljivk z utežmi (koeficienti) ter uporabi logistične funkcije za pretvorbo rezultata v verjetnost. Med fazo učenja se prilagajajo koeficienti, tako da optimiziramo logaritem verjetja z optimizacijsko metodo, kot je gradientni sestop.

2.4 Metode razlage

2.4.1 Odločitvena pravila

Kljub temu da so odločitvena pravila metoda strojnega učenja, smo jih uporabili za razlago gruč, saj so lahko razložljiva z obliko "IF conditions THEN response". Metoda, ki smo jo uporabili [5], iz ansambla odločitvenih dreves izvleče pravila z največjima vrednostima "Precision"in "Recall". Najboljša pravila smo nato uporabili za opis ene gruče proti ostalim. Pravila so uporabna tudi za ločitev primerov, ki pripadajo različnim razredom v eni gruči (medoid težko pojasni tako gručo).

2.4.2 Medoid

Za razlago posamezne gruče smo uporabili medoid, ki je izračunan na atributih primerov iz gruče. Gre za tipičen primer, ki enostavno opiše gručo. Pri uporabi razlage z medoidom moramo biti pazljivi na sestavo gruče. Gruča, ki je sestavljena iz dveh ali več razredov, ne bo dobro razložena z medoidom (tu si pomagamo z odločitvenimi pravili znotraj gruče).

2.4.3 SHAP

SHAP (SHapley Additive exPlanations) [11] je metoda, ki se uporablja za razumevanje in razlaganje pomembnosti posameznih značilk v modelih strojnega učenja. Njeno ime izhaja iz koncepta Shapleyevih vrednosti v teoriji iger, kjer se meri prispevek vsakega igralca k skupni vrednosti. Vrednosti SHAP bomo za vsako gručo posebaj predstavili v obliki grafa. Z metodo SHAP bomo za izbran klasifikator izvedeli pomembnost atributov, kar nam bo pomagalo pri interpretaciji pravil in medoidov in s tem pri razumevanju posamezne gruče.

2.5 Zmanjšanje dimenzionalnosti

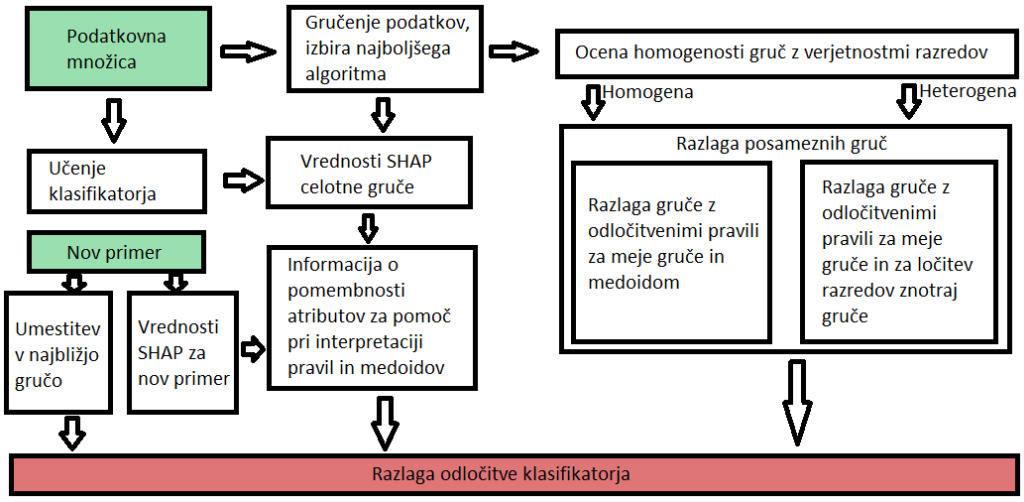
Za zmanjšanje dimenzionalnosti podatkov smo uporabili t-SNE (t-distributed Stochastic Neighbor Embedding) [22], ki se pogosto uporablja za vizualizacijo visokodimenzionalnih podatkov v nižje dimenzionalnem prostoru. Nesistematično sta bili preizkušeni še tehniki UMAP (Uniform Manifold Approximation and Projection) in PCA (Principal component analysis), a je t-SNE vrnil primernejše oblike gruč, zato smo uporabili le t-SNE. Algoritem t-SNE v visokodimenzionalnem prostoru izračuna verjetnostne porazdelitve, ki predstavljajo podobnost med primeri. Nato ustvari podobno verjetnostno porazdelitev v nižjedimenzionalnem prostoru in poskuša minimizirati razliko med tema dvema porazdelitvama. T-SNE smo uporabili tudi za predprocesiranje visokodimenzionalnih podatkov, s čimer smo dobili boljši rezultat gručenja (KMEANS, DBSCAN, HDBSCAN).

Poglavje 3

Razlaga s podskupinami

V tem poglavju predstavimo potek razlage po komponentah. Podamo diagram poteka in primer razlage umetne podatkovne množice "Homogene gruče".

Potek razlage opišemo po dveh poteh. Začnemo pri podatkovni množici, na kateri naučimo klasifikator in izvedemo algoritme gručenja. Po izvedbi algoritmov za gručenje izberemo najboljšega za trenutno podatkovno množico. Za vsako gručo ocenimo ali je homogena ali heterogena z verjetnostmi razredov v gruči. Če je gruča homogena, uporabimo za razlago gruče odločitvena pravila za meje gruče in medoid. V nasprotnem primeru uporabimo odločitvena pravila za meje gruče in odločitvena pravila za ločitev razredov znotraj gruče. S tem dobimo komponento, ki jo potrebujemo za interpretacijo odločitve klasifikatorja. Razlago posamezne gruče razširimo z grafi vrednosti SHAP, ki nam dajo informacijo o pomembnosti atributov. To informacijo uporabimo pri interpretaciji odločitvenih pravil in medoidov, tako da upoštevamo atribute, ki najbolj vplivajo na klasifikator. Nato vzamemo nov primer, ga umestimo v najbližjo gručo in izračunamo SHAP vrednosti za ta primer. Potem pogledamo, če se SHAP vrednosti primera skladajo z SHAP vrednostmi za atribute (v tem primeru je interpretacija odločitve skladna s celotno gručo). Ko se odločimo o pomembnosti atributov, sledi razlaga odločitve z inter-

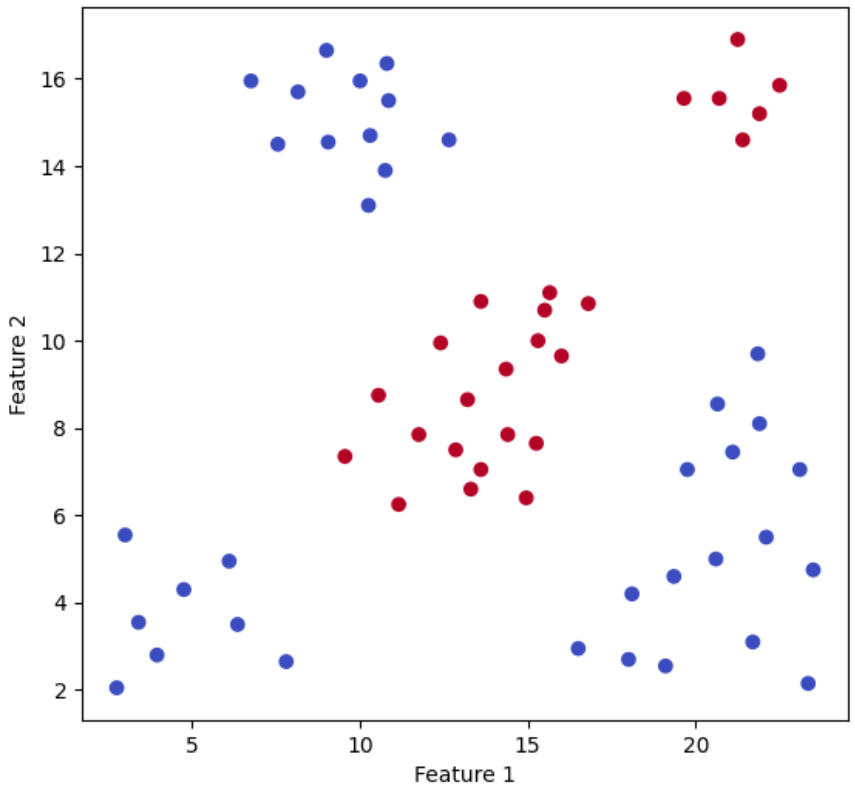


Slika 3.1: Diagram poteka razlage. Začetni stanji sta označeni z zeleno, končno z rdečo.

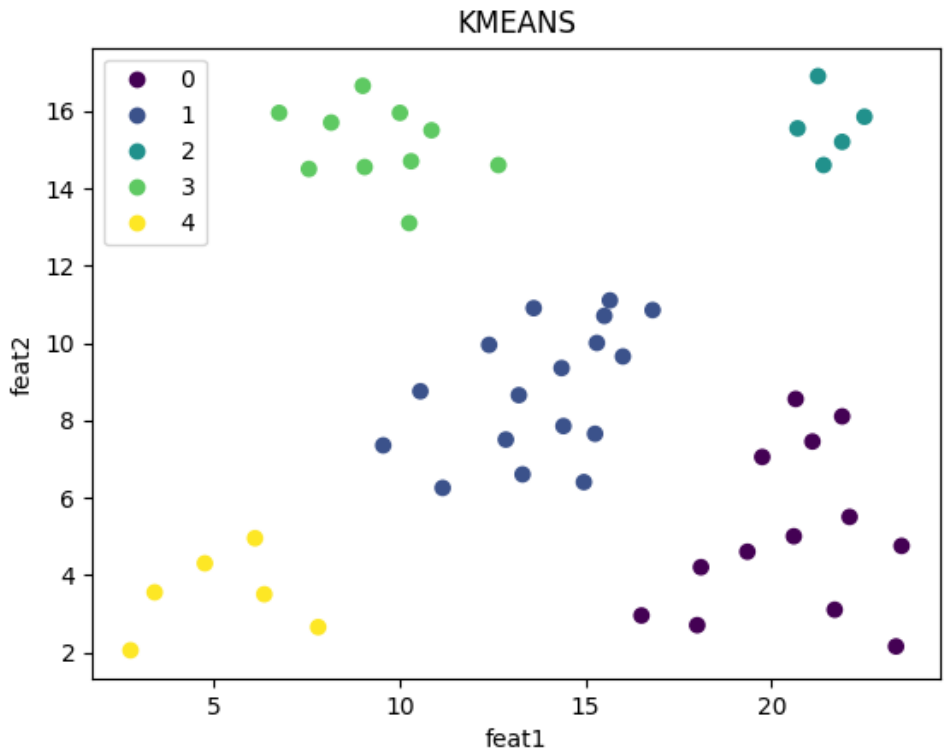
pretacijo odločitvenih pravil in medoidov (če je gruča homogena). Diagram poteka vidimo na sliki 3.1.

Za lažje razumevanje podamo preprost primer. Primer spoznamo na osnovi umetne množice podatkov, ki je prikazana na sliki 3.1. Vidimo pet gruč in vsaka od gruč v celoti vsebuje primere enega od dveh razredov, predstavljenih z modro in rdečo barvo. Podatke smo razdelili na učno in testno množico. Klasifikator, naučen na učni množici, je XGBoost in ima klasifikacijsko točnost 0.9 in F1-score 0.9 na testnih podatkih. Sledi gručenje podatkov, s katerim pridobimo podskupine podatkov. Testirali smo štiri algoritme gručenja. Njihova izbira je odvisna od posameznih podatkov. Za naš primer je bil uporabljen algoritem gručenja k-means. Algoritem kot parameter zahteva število skupin. Za avtomatizacijo izbora parametra smo uporabili mero silhuete, ki pove, kako dobro so bili gručeni podatki. Algoritem k-means smo pognali za različna števila skupin in največjo vrednost silhuete (0,601) dobili pri pet skupinah. Rezultat gručenja je viden na sliki 3.2.

Na tem mestu je potrebno omeniti, da smo za algoritme gručenja, ki temeljijo



Slika 3.2: Ilustracija umetne množice podatkov, ki vsebuje več podkonceptov. Vidimo podatke dveh razredov predstavljenih z barvami (modra, rdeča) in pet gruč. Imamo dve značilki, ki se razprostirata v dvodimenzionalnem prostoru.



Slika 3.3: Rezultat gručenja z algoritmom k-means. Največja vrednost hevristike silhuete je bila pri pet skupinah.

na gostoti podatkov uporabili hevristiko indeks DBCV, ki je analogna meri silhuete, a je namenjena za oceno gručenja teh algoritmov. V dveh dimenzijah je že na videz jasno, kje so meje gruč, a v višjih dimenzijah temu ni tako. Vizualizacije ne povedo vsega, zato uporabimo odločitvena pravila, ki povedo, kako se izbrana gruča loči od preostalih. S tem pridobimo meje posamezne gruče, ki nam kasneje, pri razlagi primerov, pomagajo pri umestitvi primerov v gruče. Vidimo tudi, katere značilke so bolj pomembne za umestitev v gručo, saj so tiste, ki najbolje ustrezajo skupini, med pogoji odločitvenih pravil. Pravila pridobljena na našem primeru so vidna v tabeli 3.1.

Tabela 3.1: Odločitvena pravila za meje posameznih gruč

Gruča	**Odločitvena pravila**	**Precision**	**Recall**
0	feat1 > 16.07 in feat2 <= 8.95	1.0	1.0
1	feat1 <= 19.35 in feat2 <= 12.10 in feat2 > 5.88	1.0	1.0
2	feat1 > 20.68 in feat2 > 11.35	1.0	1.0
3	feat1 <= 16.68 in feat2 > 12.80	1.0	1.0
4	feat1 <= 7.97 in feat2 <= 9.72	1.0	1.0

Tabela 3.2: Medoidi posameznih gruč

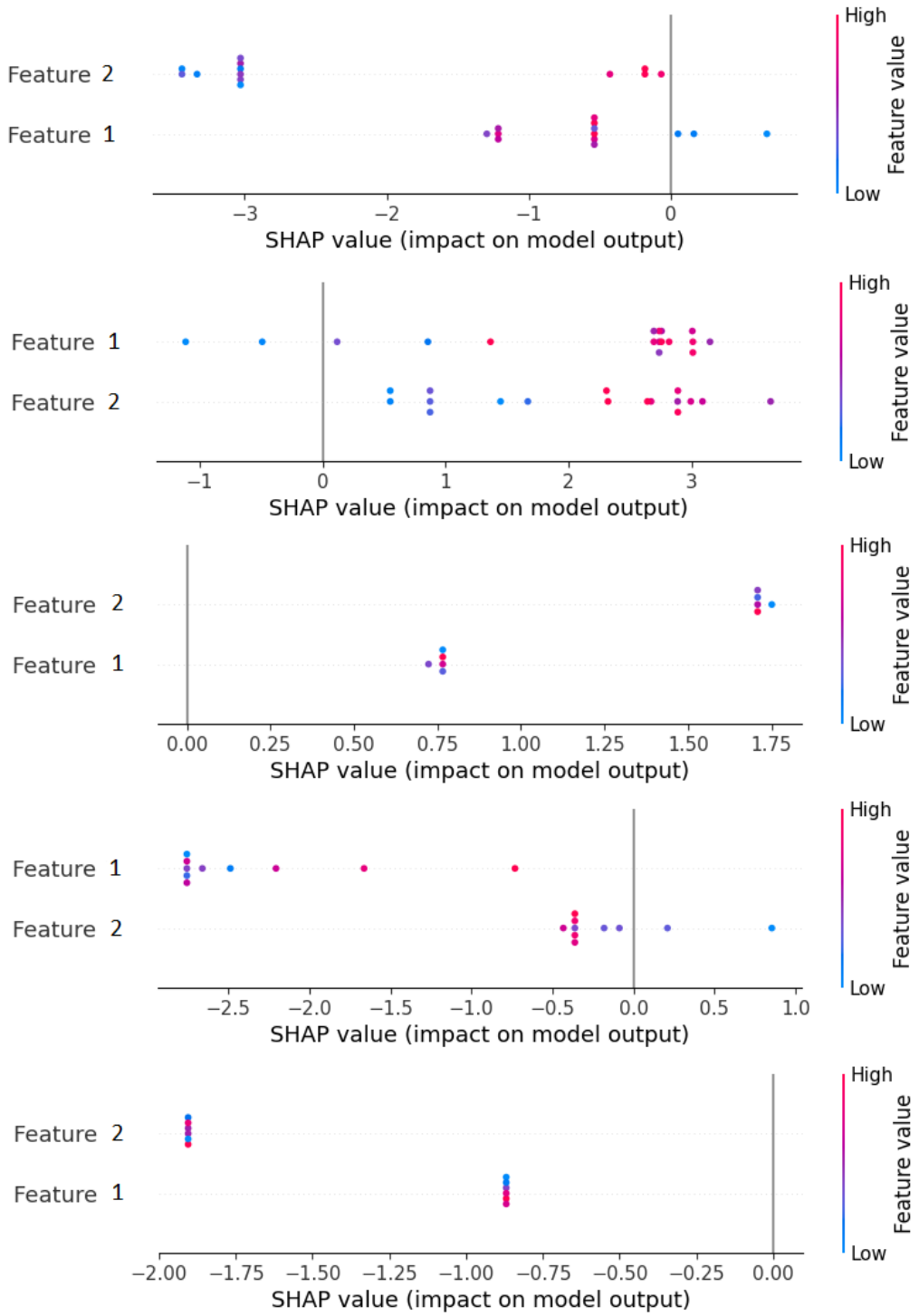
Gruča	**Feat1**	**Feat2**	**Razred**
0	20.6	5	0
1	14.35	9.35	1
2	21.9	15.2	1
3	9.05	14.55	0
4	4.75	4.3	0

Gruče razložimo tudi z medoidom, ki predstavlja tipičen primer gruče. Prav tako si bomo pogledali vrednosti SHAP celotne gruče in novega primera, ki ga vidimo v tabeli 3.3.

Tabela 3.3: Neviden primer

Feat1	**Feat2**	**Predikcija klasifikatorja**
13	9	1

Vzamimo nov primer. Klasifikator ga je uvrstil v razred ena. Primer lahko po pravilih za meje gruče razvrstimo v gručo številka ena. Tudi knjižnica za gručenje lahko napove gručo novega primera, ki ga razvrsti v gručo ena. Najprej nas zanimajo verjetnosti razredov v gruči ena. Vidimo, da je gruča v celoti homogena in vsebuje le razred ena. V takem primeru nas zanima medoid, saj dobro opiše gručo. V tabeli 3.2 imamo podane medoide za gruče. Vidimo, da medoid spada v razred ena, kar podpre odločitev klasifikatorja. Pri razlagi nam velikokrat pomaga podatek o pomembnosti značilk, ki ga bomo pridobili z vrednostmi SHAP. Na sliki 3.3 vidimo vrednosti SHAP za posamezne gruče. Nov primer spada v gručo ena, zato analiziramo drugi graf na sliki (od zgoraj navzdol). Vidimo, da oba atributa pripomoreta k klasifikaciji v razred ena (vse vrednosti so pozitivne, razen dveh nizkih v originalnem prostoru pri atributu ena). Opazimo tudi, da višje vrednosti obeh atributov v originalnem prostoru bolj pripomorejo h klasifikaciji v razred ena, kot nižje vrednosti. Nov primer ima vrednosti SHAP 3.008 in 3.091, kar pomeni, da sta obe značilki pomembni za klasifikacijo primera v razred ena.
Naša podatkovna množica (Homogene gruče) je sestavljena iz samih homogenih gruč. Če je kakšna gruča sestavljena iz več razredov, moramo uporabiti odločitvena pravila za ločitev razredov znotraj gruče.



Slika 3.4: Na grafu so prikazane vrednosti SHAP za vse gruče 0-4 od zgoraj navzdol. Vidimo, da sta za gručo ena pomembni obe značilki in da so vrednosti SHAP bolj razpršene kot pri drugih gručah. Ta gruča je sredinska in je bolj kompleksna za klasifikator kot druge.

Poglavje 4

Podatkovne množice

V poglavju predstavimo uporabljene podatkovne množice. Poglavje ima dva razdelka, kjer predstavimo umetne podatkovne množice in realne podatkovne množice. Pri realnih podatkovnih množicah opišemo tudi predprocesiranje.

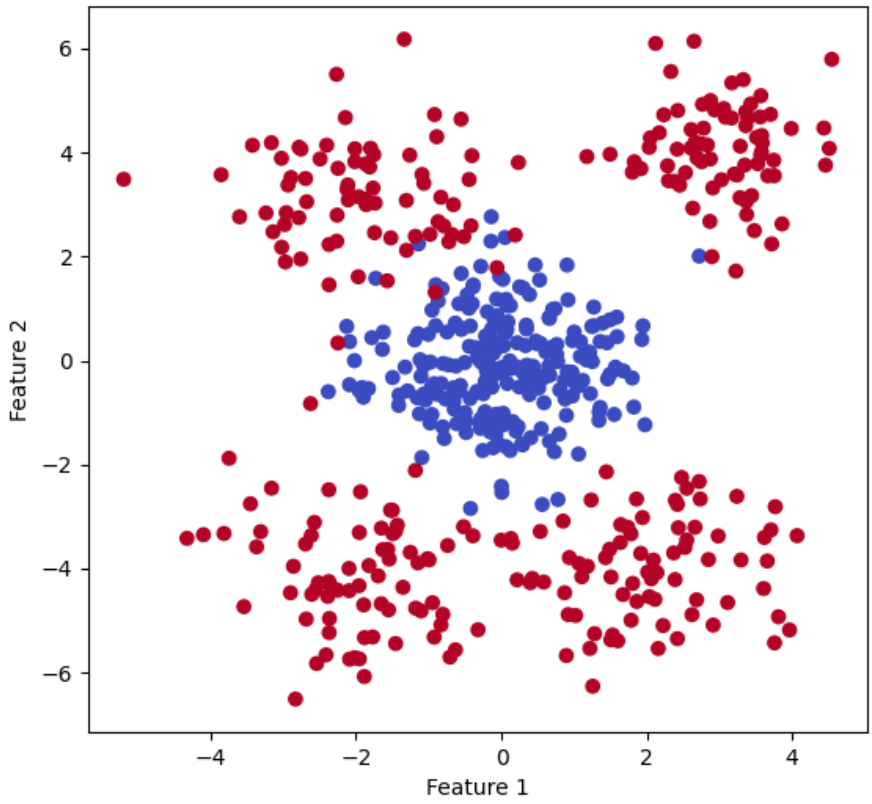
4.1 Umetne podatkovne množice

4.1.1 Umetna podatkovna množica "Krožne gruče"

Prva umetna podatkovna množica (slika 4.1) je dvodimenzionalna in ima 550 primerov. Podatkovna množica ima dva razreda, opisana sta z dvema atributoma. Zanima nas, kako dobro algoritmi najdejo pet gruč. Gruče so sferične oblike in se na mejah rahlo prekrivajo. Primeri v posameznih gručah so porazdeljeni normalno v dveh dimenzijah z različnimi standardnimi odkloni. V modri gruči je 250 primerov, v vsaki od rdečih pa 75. Podatkovna množica je bila namenjena primerjavi algoritmov in preizkusu hevristike silhuete in indeksa DBCV.

4.1.2 Umetna podatkovna množica "Trakovi 4-3"

Druga umetna podatkovna množica (slika 4.2) je dvodimenzionalna in ima 545 primerov, porazdeljenih v tri razrede in opisanimi z dvema atributoma.



Slika 4.1: Dvodimenzionalna umetna podatkovna množica "Krožne gruče" s pet gručami in dvema razredoma. Razred nič je označen z modro barvo, razred ena z rdečo.

Primeri so porazdeljeni v štiri zavite in podolgovate gruče, pozicionirane druga poleg druge. Primeri v posameznih gručah so zelo gosti do meje gruč, med in okoli gruč najdemo primere z nizko gostoto, ki predstavljajo šum. Podatkovna množica je bila namenjena primerjavi algoritmov in preizkusu hevristike silhuete in indeksa DBCV.

4.1.3 Umetna podatkovna množica "Homogene gruče"

Tretjo umetno podatkovno množico Homogene gruče (slika 3.2) opišemo v poglavju 3. Podatkovna množica je bila namenjena opisu postopka razlage klasifikatorja.

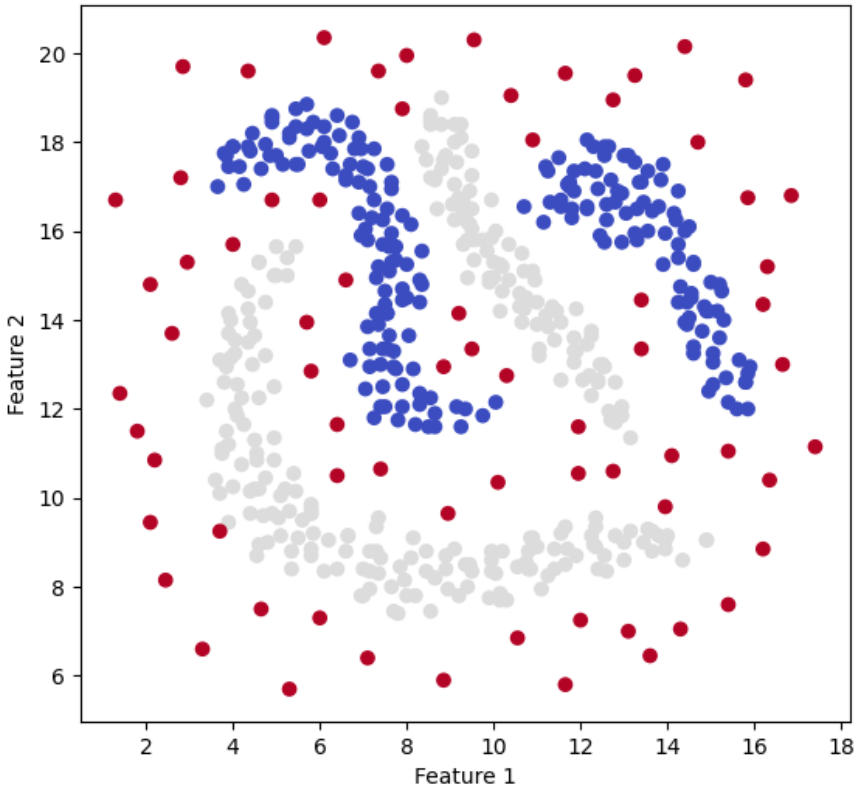
4.2 Realne podatkovne množice

4.2.1 Podatkovna množica MNIST

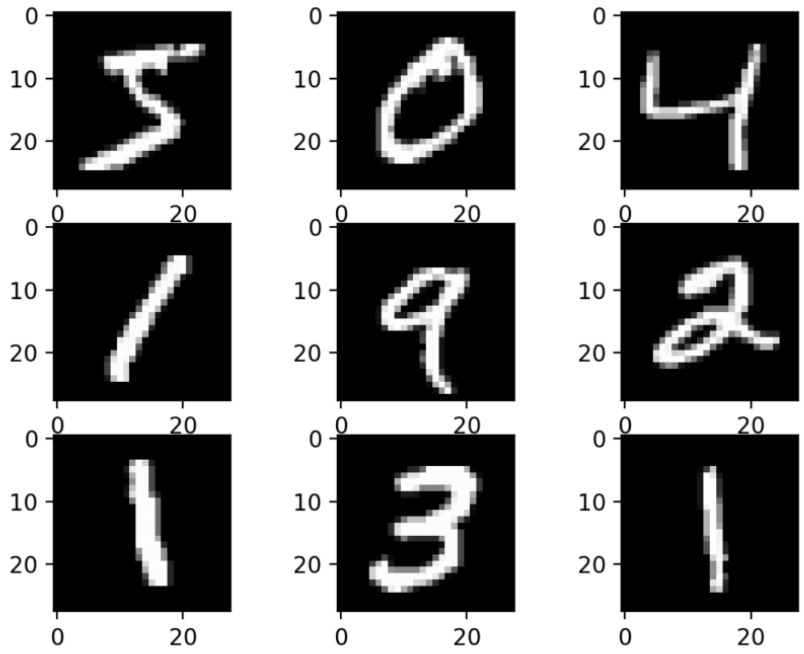
Podatkovna množica MNIST (Modified National Institute of Standards and Technology) je široko uporabljena na področju strojnega učenja in računalniškega vida. Vsebuje zbirko ročno napisanih številk (slik), ki se pogosto uporabljajo za učenje različnih algoritmov za klasifikacijo (vsebuje 10 razredov: številke 0-9). Vsaka slika je sivinska in ima resolucijo 28x28 = 784. Podatkovna množica ima 70.000 slik, ki se ponavadi delijo na učno množico (60.000) in testno množico (10.000). Podatkovno množico smo uporabili za testiranje algoritmov gručenja. Zaradi časovne kompleksnosti algoritmov smo vzeli le prvih 10.000 primerov iz učne množice in vsako sliko, ki je 2D matrika, razvili v vektor. Primere slik vidimo na sliki 4.4 [2].

4.2.2 Podatkovna množica KDD99

Podatkovna množica KDD99 [21] je široko poznana in pogosto uporabljena na področju varnosti in odkrivanja anomalij v računalniških omrežjih. Podatkovna množica je uporabljena za razvoj in ocenjevanje sistemov, ki odkrivajo vdore v računalniška omrežja, in vsebuje različne vrste omrežnega prometa,



Slika 4.2: Dvodimenzionalna umetna podatkovna množica "Trakovi 4-3". Ima 4 gruče in 3 razrede. Razreda nič in ena (modra in siva barva) predstavljata sestavo gruč, medtem ko razred dve (rdeča barva) predstavlja šum.



Slika 4.3: Primeri slik števk iz podatkovne množice MNIST, ki jih uporabljamo za strojno učenje in testiranje modelov.

vključno z običajnimi omrežnimi aktivnostmi in potencialnimi vdori. Vsebuje 23 različnih vrst napadov (ciljna spremenljivka). V našem primeru smo jih razdelili na škodljive in normalne povezave. Ima 41 atributov, kjer so tri kategorični (protocol, service in flag), ostali pa zvezni. Podatkovno množico smo (poleg ciljne spremenljivke) predprocesirali z naslednjimi koraki: odstranili smo primere, ki se ponavljajo, zatem smo uporabili tehniko eničnega kodiranje, kjer smo za vsako vrednost kategorične spremenljivke dodali nov atribut (dobimo 118 atributov) in vzeli naključnih 5.000 primerov zaradi časovne zahtevnosti algoritmov. V tabeli 4.1 vidimo imena vseh atributov (z enakimi kraticami so atributi poimenovani tudi v poglavju 6).

Tabela 4.1: Imena in kratice atributov podatkovne množice KDD99

Kratica	Ime	Kratica	Ime	Kratica	Ime
F0	duration	F40	protocol_type_b'udp'	F80	service_b'ntp_u'
F1	src_bytes	F41	service_b'IRC'	F81	service_b'other'
F2	dst_bytes	F42	service_b'X11'	F82	service_b'pm_dump'
F3	land	F43	service_b'Z39_50'	F83	service_b'pop_2'
F4	wrong_fragment	F44	service_b'auth'	F84	service_b'pop_3'
F5	urgent	F45	service_b'bgp'	F85	service_b'printer'
F6	hot	F46	service_b'courier'	F86	service_b'private'
F7	num_failed_logins	F47	service_b'csnet_ns'	F87	service_b'red_i'
F8	logged_in	F48	service_b'ctf'	F88	service_b'remote_job'
F9	num_compromised	F49	service_b'daytime'	F89	service_b'rje'
F10	root_shell	F50	service_b'discard'	F90	service_b'shell'
F11	su_attempted	F51	service_b'domain'	F91	service_b'smtp'
F12	num_root	F52	service_b'domain_u'	F92	service_b'sql_net'
F13	num_file_creations	F53	service_b'echo'	F93	service_b'ssh'
F14	num_shells	F54	service_b'eco_i'	F94	service_b'sunrpc'
F15	num_access_files	F55	service_b'ecr_i'	F95	service_b'supdup'
F16	num_outbound_cmds	F56	service_b'efs'	F96	service_b'systat'
F17	is_host_login	F57	service_b'exec'	F97	service_b'telnet'
F18	is_guest_login	F58	service_b'finger'	F98	service_b'tftp_u'
F19	count	F59	service_b'ftp'	F99	service_b'tim_i'
F20	srv_count	F60	service_b'ftp_data'	F100	service_b'time'
F21	serror_rate	F61	service_b'gopher'	F101	service_b'urh_i'
F22	srv_serror_rate	F62	service_b'hostnames'	F102	service_b'urp_i'
F23	rerror_rate	F63	service_b'http'	F103	service_b'uucp'
F24	srv_rerror_rate	F64	service_b'http_443'	F104	service_b'uucp_path'
F25	same_srv_rate	F65	service_b'imap4'	F105	service_b'vmnet'
F26	diff_srv_rate	F66	service_b'iso_tsap'	F106	service_b'whois'
F27	srv_diff_host_rate	F67	service_b'klogin'	F107	flag_b'OTH'
F28	dst_host_count	F68	service_b'kshell'	F108	flag_b'REJ'
F29	dst_host_srv_count	F69	service_b'ldap'	F109	flag_b'RSTO'
F30	dst_host_same_srv_rate	F70	service_b'link'	F110	flag_b'RSTOS0'
F31	dst_host_diff_srv_rate	F71	service_b'login'	F111	flag_b'RSTR'
F32	dst_host_same_src_port_rate	F72	service_b'mtp'	F112	flag_b'S0'
F33	dst_host_srv_diff_host_rate	F73	service_b'name'	F113	flag_b'S1'
F34	dst_host_serror_rate	F74	service_b'netbios_dgm'	F114	flag_b'S2'
F35	dst_host_srv_serror_rate	F75	service_b'netbios_ns'	F115	flag_b'S3'
F36	dst_host_rerror_rate	F76	service_b'netbios_ssn'	F116	flag_b'SF'
F37	dst_host_srv_rerror_rate	F77	service_b'netstat'	F117	flag_b'SH'
F38	protocol_type_b'icmp'	F78	service_b'nnsp'	N/A	N/A
F39	protocol_type_b'tcp'	F79	service_b'nntp'	N/A	N/A

Poglavje 5

Primerjava algoritmov za gručenje

Razlaga s podkoncepti je odvisna od gruč v podatkih. Za iskanje gruč v podatkih ne obstaja en algoritem, ki je v vseh primerih najboljši. V tem poglavju primerjamo štiri algoritme gručenja. Eksperimente izvedemo na dveh umetnih (Krožne gruče in Trakovi 4-3) in eni realni podatkovni množici (MNIST). S primerjavo se seznanimo, s kakšnimi podatki najbolje delujejo algoritmi gručenja in katere so njihove dobre in slabe lastnosti. Za ocenjevanje algoritmov gručenja bomo uporabili dve hevristiki: koeficient silhuete in indeks DBCV. Indeks silhuete je boljši za ocenjevanje algoritmov, ki delujejo na osnovi razdalj, indeks DBCV pa je boljši za algoritme, ki delujejo na osnovi gostote in del podatkov označijo kot šum [4]. DBSCAN in HDBSCAN ne razdelita vseh primerov v gruče, tako da zapišemo tudi delež označenih primerov, ostali so označeni kot osamelci oziroma šum. Ker hevristike silhuete in indeksa DBCV ne smemo primerjati (poleg tega ne povesta vseh informacij o gručenju), rezultate gručenja preučimo tudi na grafih, kjer si v visokodimenzionalnem prostoru pomagamo s t-SNE tehniko zmanjšanja dimenzij. V nadaljevanju, pri vsaki podatkovni množici, pokažemo slike le najboljših gručenj. Gručenja ostalih algoritmov najdemo v dodatku A.

5.1 Krožne gruče

Tabela 5.1: Tabela ocen in deleža označenih primerov na podatkovni množici "Krožne gruče".

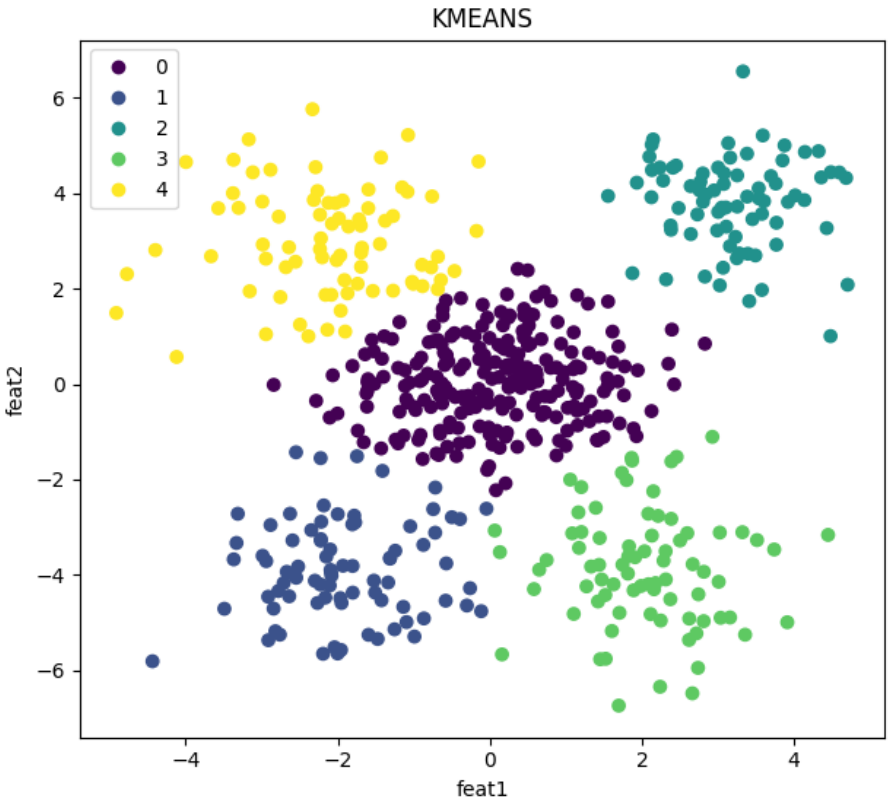
Algoritem	Silhueta	DBCV	%
MDEC	0.40	N/A	100
KMEANS	0.52	N/A	100
DBSCAN	N/A	0.34	65
HDBSCAN	N/A	0.38	69

5.1.1 Algoritem MDEC

Algoritem MDEC vrne 3 rezultate gručenja. Najboljša vrednost silhuete (tabela 5.1) je bila pri parametru za število gruč k = 3, kar ni najboljši rezultat, saj je razvidnih 5 skupin. Vredno je omeniti, da MDEC vsebuje naključne elemente, tako da se vrednosti in ocene lahko vsakič rahlo spremenijo, a so bile v večini podobne. Vzet je bil najboljši rezultat izmed izvedenih poskusov. Na sliki A.1 je bil to rezultat na skrajno desni (MDEC BG). Algoritem MDEC je pri dvodimenzionalnih podatkih ustvaril ravne pasove, kar ni bila dobra rešitev. Algoritem MDEC je namenjen uporabi v visoko dimenzionalnih prostorih, kjer se bolje odreže.

5.1.2 Algoritem K-MEANS

Algoritem k-means je vrnil zelo dober rezultat, saj so oblike gruč sferične in dovolj ločene. Najboljša vrednost silhuete (tabela 5.1) je bila pri parametru za število gruč k = 5, kar je dejansko število skupin. Silhueta k-means je v tem primeru boljša od silhuete MDEC algoritma, kar nam potruje boljše grupiranje k-means algoritma. Na sliki 5.1 je razviden rezultat gručenja.



Slika 5.1: Rezultat K-MEANS algoritma pri številu gruč k = 5 na podatkovni množici "Krožne gruče".

5.1.3 Algoritem DBSCAN

Algoritem DBSCAN nima parametra za nastavitev števila gruč, ima pa dva druga parametra - min samples in epsilon. Izbira parametrov je opisana v razdelku 2, tukaj predstavimo le najboljši rezultat. Algoritem je zelo veliko primerov označil kot osamelce in jih ni dodelil v nobeno gručo. Oznako je dobilo 65 % primerov pri najboljšem rezultatu. Omeniti je potrebno, da DBCV ocene ni smiselno primerjati z silhueto, saj delujeta na drugačnih principih. DBSCAN je našel devet gruč, kar pomeni, da je našel štiri dodatne gruče, ki niso razvidne iz podatkov. Rezultati gručenja za DBSCAN so na sliki A.2 in v tabeli 5.1.

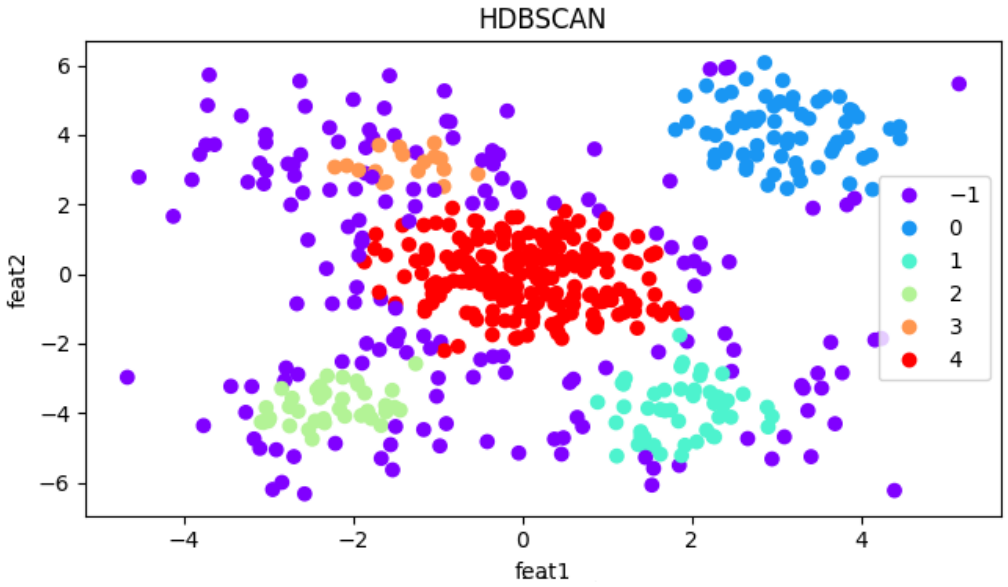
5.1.4 Algoritem HDBSCAN

Algoritem HDBSCAN samodejno določi epsilon. Namesto njega smo izbrali parametra min cluster size in min samples. Oznako je dobilo 69 % primerov pri najboljšem rezultatu. HDBSCAN je našel 5 gruč, kar pomeni, da je našel pravilno število gruč, enako kot k-means. Vidimo, da je indeks DBCV boljši kot pri DBSCAN algoritmu, kar dodatno potrjuje boljše gručenje. Rezultati gručenja za HDBSCAN so na sliki 5.2 in v tabeli 5.1.

5.2 Trakovi 4-3

Tabela 5.2: Tabela ocen in deleža označenih primerov na podatkovni množici "Trakovi 4-3".

Algoritem	Silhueta	DBCV	%
MDEC	0.25	N/A	100
KMEANS	0.44	N/A	100
DBSCAN	N/A	0.0034	87
HDBSCAN	N/A	0.095	81



Slika 5.2: Rezultat HDBSCAN algoritma na podatkovni množici "Krožne gruče".

5.2.1 Algoritem MDEC

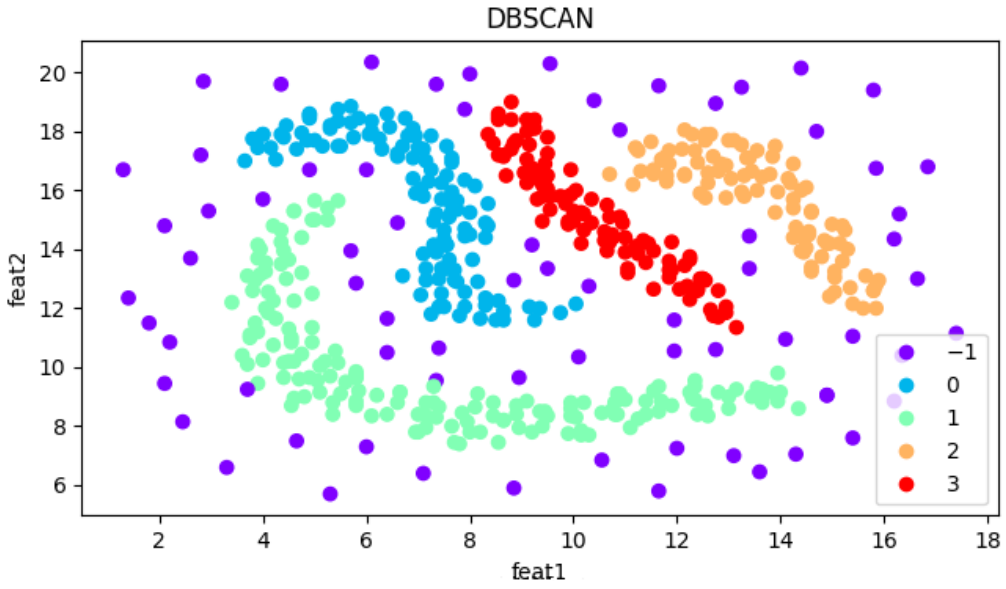
Ponovno vidimo, da je algoritem MDEC prostor razdelil na pasove. Rezultat je na sliki A.3. Najboljši rezultat (tabela 5.2) je dosegel MDEC_HC (skrajno levo) pri parametru za število gruč k = 3.

5.2.2 Algoritem K-MEANS

Algoritem k-means je vrnil najboljši rezultat pri številu gruč k = 10. Rezultat gručenja je viden v tabeli 5.2 in na sliki A.4. Vidimo, da se vsak centroid umesti v prostor in zavzame približno sferično obliko.

5.2.3 Algoritem DBSCAN

Algoritem DBSCAN je uporaben pri gručenju naravnih podolgovatih oblik, kar je razvidno tudi iz rezultata gručenja na sliki 5.3 in v tabeli 5.2. DBSCAN je našel štiri gruče kar pomeni, da je našel vse gruče in hkrati izločil šum iz



Slika 5.3: Rezultat DBSCAN algoritma na podatkovni množici "Trakovi 4-3". Na sliki se vidi označbe vseh štirih skupin in izločitev šuma iz podatkov.

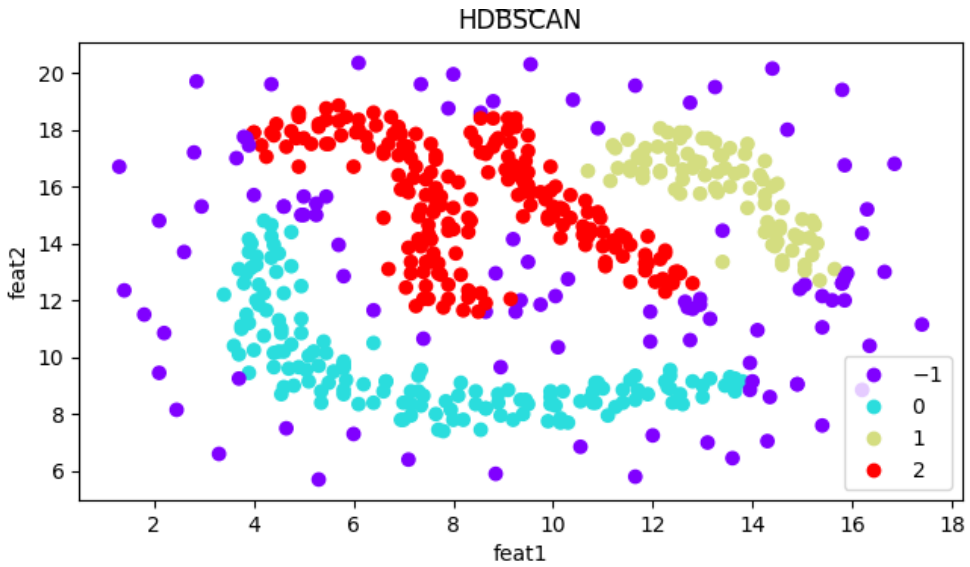
podatkov.

5.2.4 Algoritem HDBSCAN

Algoritem HDBSCAN je hierarhična različica DBSCAN algoritma, a vrne slabši rezultat kot DBSCAN, kot je razvidno iz slike 5.4 in tabele 5.2. HDBSCAN je našel tri gruče, kar pomeni, da je dve ločeni gruči združil v eno. Najboljša DBCV vrednost je 0.095 kar je presenetljivo boljši indeks DBCV od algoritma DBSCAN glede na to, da vidimo slabše gručenje algoritma HDBSCAN. Šum je dobro ločen od skupin.

5.3 MNIST

Podatkovna množica MNIST ima 784 atributov, zato smo za prikaz v dveh dimenzijah uporabili vložitev T-SNE. Zaradi časovne zahtevnosti algoritmov smo uporabili le prvih 10.000 primerov. Prav tako smo pri KMEANS, DB-

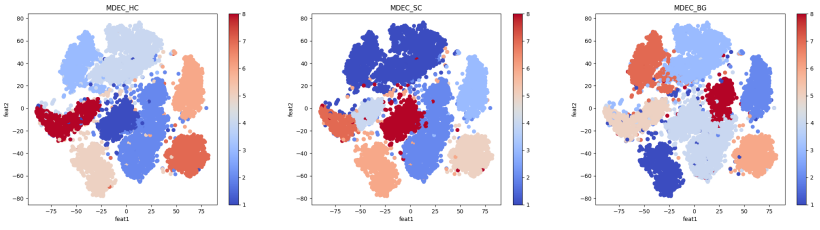


Slika 5.4: Rezultat HDBSCAN algoritma na podatkovni množici "Trakovi 4-3". Na sliki se vidi označbe 3 skupin in izločitev šuma iz podatkov.

SCAN in HDBSCAN algoritmih za gručenje vhodne podatke najprej pretvorili v dve dimenziji. Izjema je algoritem MDEC, ki je bolje deloval v originalnem, visokodimenzionalnem prostoru. V tabeli 5.3 vidimo rezultate. Hevristika silhuete pri KMEANS algoritmu je višja, kot pri MDEC, vendar na slikah vidimo, da je bolje gručil algoritem MDEC (pri MDEC algoritmu je izračunana v višjih dimenzijah). Enako velja za HDBSCAN in DBSCAN, kjer ima HDBSCAN nižji indeks DBCV, ampak je gručil bolje.

Tabela 5.3: Rezultati algoritmov za gručenje.

Algoritem	Silhueta	DBCV	%	Vizualizacija
MDEC	0.0471	N/A	100	Slika 5.5
KMEANS	0.4738	N/A	100	Slika A.5
DBSCAN	N/A	0.282	63	Slika A.6
HDBSCAN	N/A	-0.162	95	Slika A.7



Slika 5.5: Rezultat MDEC algoritma. Algoritem MDEC_HC (najbolj leva slika) je dosegel najboljšo hevristiko silhuete pri številu gruč k = 8. Našel je dve gruči manj, najdene gruče so smiselne.

Tabela 5.4: Najboljši algoritmi za posamezno podatkovno množico.

Podatkovna množica	Najboljši algoritem
Krožne gruče	KMEANS, HDBSCAN
Trakovi 4-3	DBSCAN
MNIST	MDEC

Z eksperimenti vidimo nekatere lastnosti algoritmov. Algoritem KMEANS vrne dobre rezultate s sferičnimi oblikami, kar se je izkazalo za neuporabno pri gručenju realne podatkovne množice MNIST. Algoritem t-SNE vrne tudi podolgovate oblike gruč, ki jih algoritem KMEANS razdeli na več delov. Na umetni podatkovni množici "Krožne gruče"je vrnil najboljši rezultat, saj so bile gruče krožne oblike. Algoritem MDEC je v nizkih dimenzijah vrnil nesmiselne rezultate. Prostor je delil na ravne pasove. Pri visokodimenzionalni množici MNIST je vrnil najboljši rezultat brez uporabe tehnike t-SNE. Algoritem DBSCAN je odvisen od izbire pravih parametrov. Pri umetni množici "Trakovi 4-3"je vrnil odličen rezultat, pri drugih dveh podatkovnih množicah je gručil slabše (še posebej pri podatkovni množici MNIST, kjer je vrnil nesmiseln rezultat). Za bolj stabilnega se je izkazal algoritem HDBSCAN, ki je pri vseh treh podatkovnih množicah vrnil smiseln rezultat. S testiranjem algoritmov in hevristik ugotovimo, da je pomemben del ocene gručenja pregled grafa gručenja, saj hevristike v nekaterih primerih odpovejo. V tabeli

5.4 vidimo algoritme z najboljšim rezultatom za vsako podatkovno množico, ki smo jo preizkusili.

Poglavje 6

Razlaga realne podatkovne množice

V tem poglavju bomo s predlagano metodo razlage s podkoncepti razložili klasifikacijo. Najprej gručimo podatke in izberemo najprimernejši algoritem gručenja. Zatem ocenimo homogenost gruč z verjetnostjo razredov v posamezni gruči. Če je gruča homogena jo razložimo z odločitvenimi pravili za meje gruče in medoidom, v nasprotnem primeri z odločitvenimi pravili za meje gruče in odločitvenimi pravili za ločitev razredov znotraj gruče. Nato izračunamo vrednosti SHAP za posamezno gručo, s katerimi si pomagamo pri interpretaciji odločitvenih pravil in medoidov. Potem izberemo naključni primer, ga umestimo v najbližjo gručo, izračunamo vrednosti SHAP za ta primer in interpretiramo odločitev klasifikatorja.

Za predlagano metodo razlage uporabimo podatkovno množico KDD99, saj ima strukturo, ki jo pričakuje naša metoda. Večina preostalih podatkovnih množic nima tako jasno primerov enega razreda razčlenjenega v več gruč ali pa ima slabo ločene gruče. Predprocesiranje podatkovne množice in imena atributov so na voljo v poglavju 4.2.2. Pri gručenju smo si pomagali s tehniko T-SNE, saj je pri algoritmih na osnovi gostote (v našem primeru na sliki 6.1 je to HDBSCAN) pomagala pri primerih, ki so bili označeni za šum zaradi visoke dimenzionalnosti. S tem smo pridobili sedem gruč, ki so na

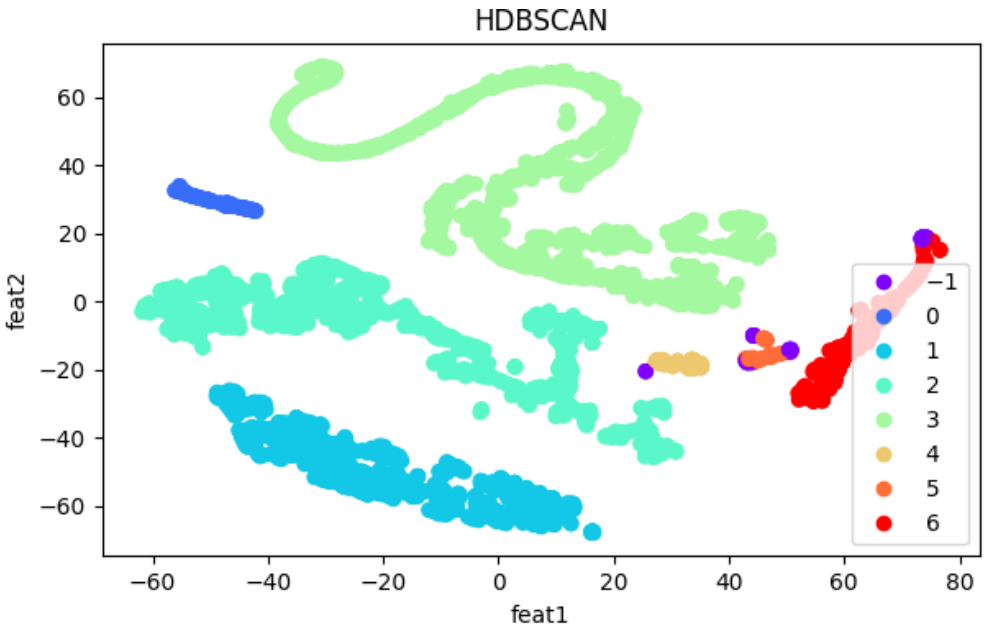
prvi pogled smiselno opredeljene, saj vidimo goste gruče, ki so dobro ločene med seboj. Poleg tega je vrednost indeksa DBCV 0.27, kar dodatno potrjuje smiselno gručenje. V tabeli 6.1 vidimo verjetnosti razredov za vsako gručo. Gruči ena in dve vsebujeta škodljive povezave, ostale normalne. Opazimo, da sta gruči dve in pet bolj mešani kot preostale gruče, kar bomo kasneje omenili pri razlagi z medoidi, kjer si bomo pomagali še z odločitvenimi pravili znotraj gruče. Za klasifikacijo povezav smo uporabili logistično regresijo, ki je imela na testnih podatkih klasifikacijsko točnost 0.9587 in F1 score 0.9587. V tabeli 6.2 imamo podana odločitvena pravila, ki med seboj ločijo gruče. Vidimo, da imamo za vsako gručo več pravil, saj so podatki visokodimenzionalni in se jih lahko dobro loči po različnih atributih. Vsi atributi niso enako pomembni za klasifikator, kar pomeni, da ni smiselno obravnavati vseh pravil enako. Pri večini pravil ene gruče vidimo majhno podmnožico atributov, ki se ponavljajo in so s tem verjetno bolj pomembni za meje kot drugi. Z odločitvenimi pravili odkrijemo dodatno znanje iz podatkov, ki nam pomaga predvsem, ko je gruča homogena, saj del razreda ločimo od preostalih primerov. Pri interpretaciji pravil je pomembno pogledati meri Precision in Recall, saj so nekatera pravila manj točna ali pa ne zajamejo vseh primerov. Ko imamo občutek, kako so skupine medsebojno ločene, vsako lahko opišemo z medoidom, ki je tipičen primer gruče. Medoide vidimo v tabeli 6.3. Vidimo, da sta medoida M1 in M2 normalni povezavi, ostali medoidi predstavljajo škodljive. Opazimo, da imata edina atribut F1 (src_bytes) enak 0. Močno se razlikujeta tudi po atributu F19 (count), ki ima večjo vrednost od preostalih medoidov. Pri atributu F25 (same_srv_rate) imata nizke vrednosti, ostali ima vrednost 1. Prav tako je razlika pri F32 (dst_host_same_src_port_rate), kjer imata edina vrednost 0. Pri atributih F34 (dst_host_serror_rate), F35 (dst_host_srv_serror_rate), F86 (service_private) in F112 (flag_S0) imata vrednost 1, ostali imajo 0 (binarno kodirane vrednosti). Enako analizo lahko naredimo za vsak medoid posebaj proti ostalim. Zaradi velikega števila atributov smo obdržali v tabeli le tiste, ki se razlikujejo vsaj pri enem izmed medoidov. Poleg medoidov nas zanima tudi, kako so ločeni razredi znotaj gruče dve in pet, kar lahko vidimo v tabeli

6.4. S tabelo 6.4 si pomagamo v primeru, da je nov primer umestimo v gruči dve ali pet, saj je medoid lahko zavajujoč. Zanima nas tudi, kateri atributi so pomembni za naš klasifikator (logistično regresijo), kar vidimo na sliki 6.2 za gručo 1, saj si bomo s tem pomagali pri interpretaciji odločitve. Vrednosti SHAP za preostale gruče najdemo v dodatku B. Na sliki 6.2 vidimo, da so pomembni atributi F19, F29, F20 in F28, ostali so skoraj ničelni. Atributa F19 in F29 pripomoreta h klasifikaciji v razred 1 (normalna povezava). Atribut 20 z visokimi vrednostmi pripomore h klasifikaciji v razred 0 (škodljive povezave), nizke vrednosti pa h klasifikaciji v razred 1. Atribut 28 je nepomemben napram prejšnim trem in pripomore h klasifikaciji v razred 0. Vidimo, da je veliko atributov nepomembnih. Pomembnost atributov nato upoštevamo pri odločitvenih pravilih in medoidih, kjer nam pravila in medoidi sestavljeni iz pomembnih atributov podajo informacijo o ločitvenih mejah in o sami gruči. S to informacijo lahko pogledamo nov primer in na podlagi njegovih atributov razložimo, zakaj spada v gručo (odločitvena pravila) in v kakšne vrste gručo spada (medoid).

Sedaj vzamemo naključen primer iz testne množice. Zaradi visoke dimenzionalnosti niso podane vse vrednosti atributov (pomembni atributi glede na vrednosti SHAP: F19 = 218, F20 = 6, F28 = 255, F29 = 6). Primer je logistična regresija opredelila v razred ena, algoritem za gručenje pa ga je razvrstil v gručo ena. Pogledamo tabelo verjetnosti razredov in vidimo, da je gruča ena v celoti homogena s 100% primerov, ki pripadajo razredu ena (škodljiva povezava).Nov primer je dobil SHAP vrednosti F19: 13.9, F29: 2.4, F20: 0.34, F28: -0.23, vrednosti ostalih atributov so zanemarljivo majhne. Vrednosti SHAP novega primera potrjujejo pomembnost zgornjih atributov v skupini ena. V tabeli z odločitvenimi pravili 6.2 vidimo devet pravil, ki z visoko natančnostjo opišejo gručo z majhno podmnožico atributov. Iz odločitvenih pravil za meje gruče razberemo, da primer zadostuje vsem pogojem, kar primer z gotovostjo postavi v gručo. Poleg tega imamo še veliko pogojev z atributi, ki nas v tem trenutki ne zanimajo, saj niso pomembni za odločitev klasifikatorja. Skupina je homogena, zato jo dobro opiše

medoid M1 v tabeli 6.3.Medoid je iz razreda ena, kar pomeni, da je gruča sestavljena iz normalnih povezav (100% povezav). Medoid ima izmed vseh največjo vrednost atributa F19, kar nam iz zgornje razlage vrednosti SHAP za gručo potrjuje, da je bil klasificiran v razred ena. Analizo za M1 in M2 proti ostalim smo naredili zgoraj, sedaj si pogledamo v čem se razlikujeta M1 in M2, saj oba pripadata razredu ena. Očitna razlika se vidi pri atributu 29, kjer je medoid M1 bolj podoben M0, čeprav je M0 iz razreda nič. Enako velja za atribut F19. Pri atributu F116 se razlikujeta, saj ima le medoid M2 vrednost 0. V primeru, da bi bil nov primer iz gruče dve ali pet (nečisti gruči), bi si ogledali še pravila znotraj gruče, ki ločijo razrede.

Na podatkovni množici KDD99 že obstajajo poskusi razlage, [12], kjer je bila podatkovna množica razložena z odločitvenim drevesom. Atributi so bili glede na pomembnost ocenjeni z mero entropije. Pri odločitvenem drevesu sta bila dva izmed pomembnih atributov enaka (najpomembnejši: Count in pa Dst_host_count). Njihov pristop omogoča razlago celotne podatkovne množice z enim drevesom, kar olajša interpretacijo medtem ko se naš pristop bolj poglobi v prostorsko povezanost primerov in iz tega pridobi informacije na podlagi odločitvenih pravil in medoidov. Remah Younisse in sod. [23] so različne modele za strojno učenje razložili s SHAP vrednostmi in jih potrdili na podlagi ocenjevanja gostote verjetnosti z jedrom KDE (kernel density estimation) grafov. Večina klasifikatorjev je imela med najpomembnejšimi značilkami značilko count, dst_host_srv_count in dst_host_count. Razlika je bila pri porazdelitvi pomembnosti med atributi. Kot pomembne je bilo označenih večje število značilk kot pri naši razlagi. Vrednosti SHAP so bile upravičene, kar pri nas ne drži, a poleg pomembnosti atributov nismo izvedeli nič o podatkovni množici. Z algoritmom KMEANS so na podlagi gruč [20] iskali povezavo med tipom napada in uporabljenim protokolom. Osredotočili so se na analizo podatkov v posamezni gruči kot pri naši razlagi, a niso pojasnevali klasifikatorjev. Ugotovili so, da je protokol TCP najbolj deležen napadov. Naši medoidi imajo vsi TCP protokol, razen medoida M0,



Slika 6.1: Rezultat T-SNE + HDBSCAN algoritma na učni množici. Algoritem je našel 7 gruč. DBSV indeks je 0.5617.

kar ga naredi zanimivega za nadaljno analizo. Podatkovno množico so opisali tudi z odločitvenimi pravili [10]. Za razliko od naše razlage so odločitvena pravila uporabili na negručenih podatkih, kar se je rezultiralo v drugačnih pravilih. Uporabili so odločitvena pravila za ločitev razredov, kot smo mi v nečistih gručah. V pogojih odločitvenih so se pojavili atributi, ki tvorijo tudi naša odločitvena pravila, a z drugačnimi vrednostmi. Njihova razlaga, kot pri razlagi z odločitvenemim drevesom, ne gruči podatkov pred uporabo odločitvenih pravil, zato je o podobnih zaključih težko sklepati.

Tabela 6.1: Verjetnosti razredov v gručah

Razred	**G0**	**G1**	**G2**	**G3**	**G4**	**G5**	**G6**
0	0.966	0	0.344	0.993	0.787	0.641	0.972
1	0.334	1	0.656	0.007	0.213	0.359	0.028

Tabela 6.2: Odločitvena pravila za meje posameznih gruč

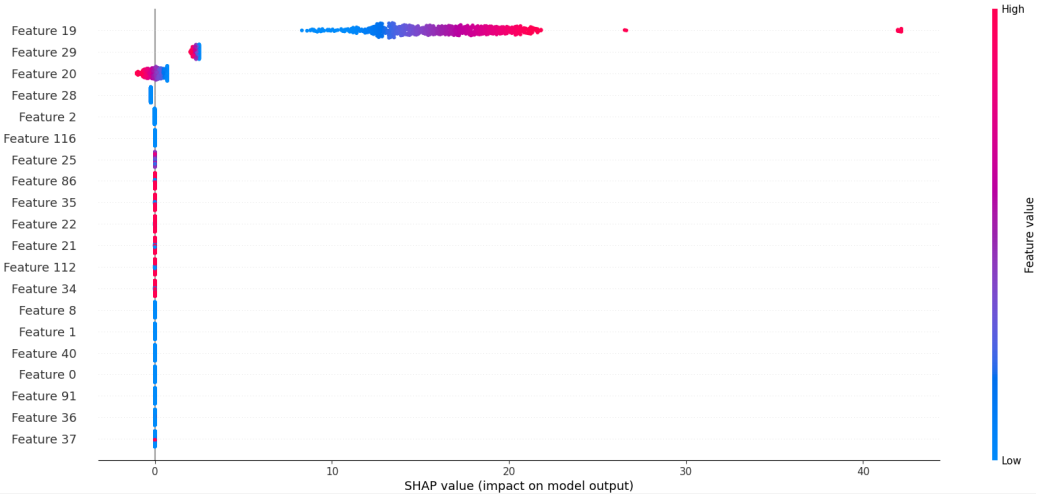
Gruča	**Odločitvena pravila**	**Precision**	**Recall**
0	F0 > 640.5 in F2 > 52.5	1.0	1.0
0	F0 > 629.5 in F30 <= 0.7755	1.0	1.0
0	F0 > 724.5 in F31 > 0.01	1.0	1.0
0	F0 > 640.5 in F1 <= 2567830.0	1.0	1.0
0	F0 > 640.5 in F28 > 11.5	1.0	1.0
0	F0 > 640.5 in F33 <= 0.02	1.0	1.0
0	F0 > 640.5 in F60 <= 0.5	1.0	1.0
1	F116 <= 0.5 in F19 > 162.0 in F32 <= 0.005	1.0	1.0
1	F19 > 162.0 in F25 <= 0.265 in F28 > 185.0	1.0	1.0
1	F19 > 162.0 in F30 <= 0.115 in F32 <= 0.005	1.0	0.9991
1	F19 > 162.0 in F28 > 185.0 in F30 <= 0.115	1.0	0.9987
1	F19 > 162.0 in F20 <= 49.0 in F32 <= 0.005	1.0	0.9985
1	F19 > 162.0 in F28 > 185.0 in F29 <= 48.5	1.0	0.9985
1	F19 > 162.0 in F32 <= 0.005 in F39 > 0.5	1.0	0.9985
1	F1 <= 14.0 in F19 > 162.0 in F32 <= 0.005	1.0	0.9985
1	F116 <= 0.5 in F19 > 162.0 in F28 > 185.0	1.0	0.9985
2	F1 <= 137.0 in F19 <= 162.0 in F2 <= 1269.5	1.0	0.991
3	F1 <= 27526.0 in F63 > 0.5 in F8 > 0.5	0.999	0.984
3	F1 <= 27540.5 in F1 > 71.0 in F63 > 0.5	1.0	0.981
3	F1 <= 27540.5 in F2 > 36.5 in F63 > 0.5	1.0	0.981
3	F23 <= 0.135 in F63 > 0.5 in F9 <= 0.5	0.998	0.982
3	F1 > 70.5 in F63 > 0.5 in F9 <= 0.5	0.999	0.979
3	F63 > 0.5 in F8 > 0.5 in F9 <= 0.5	0.999	0.978
4	F1 <= 475.0 in F1 > 143.5 in F102 <= 0.5 in F60 > 0.5	1.0	0.66
5	F1 <= 1482.5 in F1 > 475.0 in F20 <= 279.0 in F60 > 0.5	1.0	0.77
6	F1 > 598.0 in F91 > 0.5	1.0	0.812
6	F1 > 644.5 in F12 <= 4.5 in F32 <= 0.315 in F33 <= 0.11 in F6 <= 2.5 in F8 > 0.5 in F97 <= 0.5	1.0	0.812

Tabela 6.3: Medoidi z atributi, ki se razlikujejo.

Atribut	**M0**	**M1**	**M2**	**M3**	**M4**	**M5**	**M6**
F0	3058	0	0	0	0	0	5
F1	147	0	0	286	201	854	1536
F2	105	0	0	1768	0	0	328
F8	0	0	0	1	1	0	1
F19	1	240	109	24	6	2	1
F20	1	1	9	35	6	2	1
F21	0	1	1	0	0	0	0
F22	0	1	1	0	0	0	0
F25	1	0.04	0.08	1	1	1	1
F26	0	0.06	0.06	0	0	0	0
F27	0	0	0	0.11	0	0	0
F28	255	255	255	143	170	106	80
F29	1	1	15	255	28	100	154
F30	0	0.04	0.06	1	0.09	0.47	0.98
F31	0.6	0.07	0.07	0	0.02	0.04	0.03
F32	0.93	0	0	0.01	0.09	0.47	0.01
F33	0	0	0	0.01	0.07	0.02	0.01
F34	0	1	1	0	0	0	0
F35	0	1	1	0	0	0	0
F39	0	1	1	1	1	1	1
F40	1	0	0	0	0	0	0
F60	0	0	0	0	1	1	0
F63	0	0	0	1	0	0	0
F81	1	0	0	0	0	0	0
F86	0	1	1	0	0	0	0
F91	0	0	0	0	0	0	1
F112	0	1	1	0	0	0	0
F116	1	1	0	1	1	1	1
Class	0	1	1	0	0	0	0

Tabela 6.4: Pravila za primere različnih razredov znotraj gruče. V oklepajih so zapisane vrednosti Precision in Recall.

Gruča	**Pravila za razred 0**	**Pravila za razred 1**
2	F111 <= 0.5 in F25 > 0.49 in F29 > 1.5 in F31 <= 0.845 in F33 <= 0.44 in F35 <= 0.03 in F4 <= 1.5 in F54 <= 0.5 (1,0.96)	F1 <= 74.5 in F25 <= 0.49 (0.9992,0.93)
5	F30 <= 0.665 (1,1)	F30 > 0.665 (1,1)



Slika 6.2: Vrednosti SHAP za gručo 1

Poglavje 7

Evalvacija

V tem poglavju podamo mnenja o izbiri števila gruč (podpoglavje 7.1), smiselnosti prototipov (podpoglavje 7.2) in smiselnosti odločitvenih pravil (podpoglavje 7.3). Mnenja podamo na podlagi eksperimentov razlage s podkoncepti na podatkovni množici "Homogene gruče"in KDD99. Na koncu poglavja podamo še diskusijo o ugotovitvah, uspešnosti razlage (preprostost, interpretabilnost, vizualizacija, prilagodljivost) in avtomatizaciji postopka.

7.1 Število gruč

Število gruč je pri algoritmih, kot so k-means in MDEC, igra pomembno vlogo, in zato tudi pri razlagi. Pri umetnih podatkih smo vedeli število gruč, tako da je bila najboljša ocena gručenja takrat, ko smo za število gruč uporabili dejansko vrednost. Težje je ta parameter najti v realnih podatkih, ko ne vemo koliko gruč se skriva v njih. Zato je smiselno preučiti podatke v nižjedimenzionalnih prostorih, preizkusiti različne vrednosti števila gruč in iskati najboljšo oceno gručenja. Veliko število gruč se pozna tudi pri razlagi, kjer se nestrnjenost gruč kaže v slabo razložljivih prototipih in pravilih, ki bi morala podajati isto razlago. Premajhno število gruč pa preveč posploši razlago in poda razlage, ki bi morale biti ločene na več pravil in prototipov (dobimo nečiste skupine). V obeh primerih je lahko razlaga neverodostojna.

7.2 Smiselnost prototipov

Prototipi, kot so medoidi, so smiselni takrat, ko je gruča homogena in jo lahko predstavimo z reprezentativnim primerom. Pomemben je tudi razpored primerov z različnimi razredi v gruči, saj lahko medoid predstavlja manjšinski razred gruče, če so primeri iz manjšinjskega razreda razporejeni v središču sferične gruče. Prav tako nam medoid ne pove, katere značilke so pomembne. Če vzamemo podolgovato gručo v dvodimenzionalnem prostoru v obliki linije, je pri medoidu pomembna le ena dimenzija, kar ne bo razvidno iz prototipa. Pri takšni težavi si lahko pomagamo s vrednostmi SHAP. V visokih dimenzijah so medoidi neintuitivni in nepregledni, kar ponovno rešujejo vrednosti SHAP za pomembnost atributov.

7.3 Smiselnost pravil

Pri odločitvenih pravilih se lahko zgodi, da so gruče dobro ločene v eni dimenziji in slabše v neki drugi. Tako je smiselno gledati samo pogoj v pravilu, ki dobro loči skupine. V visokih dimenzijah se zgodi, da več različnih podmnožic pravil dobro loči skupino. Tu si lahko pomagamo z vrednostmi SHAP za gručo ter za posamezen primer, saj nam ti povedo, katere značilke so pomembne in to upoštevamo v pogojih pravil. Treba je omeniti, da pravila delijo prostor s hiperpravokotniki, kar pomeni, da ne zajamejo v celoti kompleksnih oblik, kar tudi razlago. Smiselnost pravil se da oceniti tudi z vrednostmi Precision in Recall, ki jih izračunamo za pravila.

Sam postopek razlage je težek za interpretacijo in v nekaterih primerih nepopoln. Večina realnih podatkovnih množic nima pričakovane strukture in zato naše metode razlage ne moremo uporabiti. Rezultati posameznih metod zahtevajo človeško oceno. Zaradi tega postopek ni v celoti avtomatiziran, saj je izbor pravega algoritma gručenja ključen. Naše hevristike za oceno gručenja niso zadostne za izbor. Po izvedbi posameznih komponent dobimo rezultate metod, ki jih je potrebno interpretirati, kar zahteva osebo s pravimi znanji.

Dodatno oviro predstavlja visoka dimenzionalnost podatkov, ki dodatno oteži razlago in vmesne človeške ocene. Dobra lastnost naše razlage je razčlenitev podatkov na gruče, saj s tem pridobimo dodatne informacije o podkonceptih, ki so drugače metodam razlage, kot na primer odločitvenim pravilom in medoidom, skrite. Rezultati nekaterih komponent (gručenje, SHAP) so predstavljeni vizualno, kar olajša razumevanje in omogoča lažje sklepanje.

Poglavje 8

Zaključek

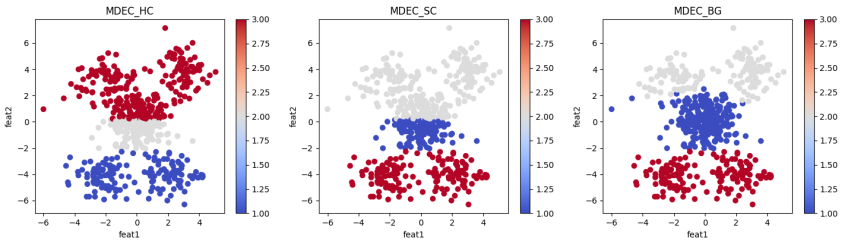
V diplomski nalogi smo predstavili metodo razlage klasifikatorja na podlagi podkonceptov (gruče primerov istega razreda) z uporabo algoritmov gručenja, odločitvenih pravil, medoidov in vrednostimi SHAP. Izvedli smo eksperimente s štirimi algoritmi za gručenje na dveh umetnih in dveh realnih podatkovnih množicah. Za izbor parametrov in oceno kvalitete gručenja smo uporabili hevristiki silhuete in indeks DBCV. Pri visokodimenzionalnih podatkih je bila uporabljena tudi tehnika za zmanjšanje dimenzij T-SNE.
Primerjava algoritmov gručenja pokaže, da se algoritem MDEC slabo odreže v nizkodimenzionalnih prostorih (2D), medtem ko se je dobro odrezal v visokodimenzionalnem prostoru. Algoritem KMEANS se je odrezal dobro, na sferičnih gručah, tudi če so si bile gruče blizu. V visokodimenzionalnem prostoru se je bolje odrezal v kombinaciji s T-SNE, a vseeno ne tako dobro kot algoritem HDBSCAN. Algoritma DBSCAN in HDBSCAN sta brez zmanjšanja dimenzionalnosti s T-SNE veliko primerov klasificirala kot šum. Algoritem DBSCAN je v kombinaciji s T-SNE v splošnem slabo gručil, razen na umetni podatkovni množici "Trakovi 4-3". Algoritem HDBSCAN se je izkazal za bolj stabilnega (od DBSCAN) v kombinaciji s T-SNE, zato je bil uporabljen tudi pri razlagi realne podatkovne množice KDD99.
Naš postopek razlage s podkoncepti se je v večini primerov realnih podatkovnih množic izkazal za neuporabnega, saj imajo le redke podatkovne množice

primere enega razreda razdeljene v dve ali več gruči (premajhna razčlenjenost primerov istega razreda na gruče). Predstavljena metoda razlage razdeli podatkovno množico na manjše dele, kar naredi razlago bolj razumljivo in bolj podrobno, kot če bi metode uporabili na celotni podatkovni množici. Pri razlagi z medoidi so se pojavile težave v visokodimenzionalnem prostoru, saj je medoid z veliko atributi dokaj neintuitiven in težko razumljiv, na tem mestu smo si pomagali z vrednostmi SHAP. Ogledali smo si le pomembne atribute in prikazali le atribute, v katerih so se medoidih različnih gruč razlikovali. Odločitvena pravila so v nekaterih primerih vsebovala nepomembne atribute, a so vseeno vsebovala dodatno informacijo o gruči. Težavo predstavljajo gruče sestavljene iz dveh ali več razredov, saj je medoid lahko zavajojoč. Uporabili smo tudi odločitvena pravila, tokrat znotraj gruče, da so pomagala ločiti primere enega razreda od preostalih.

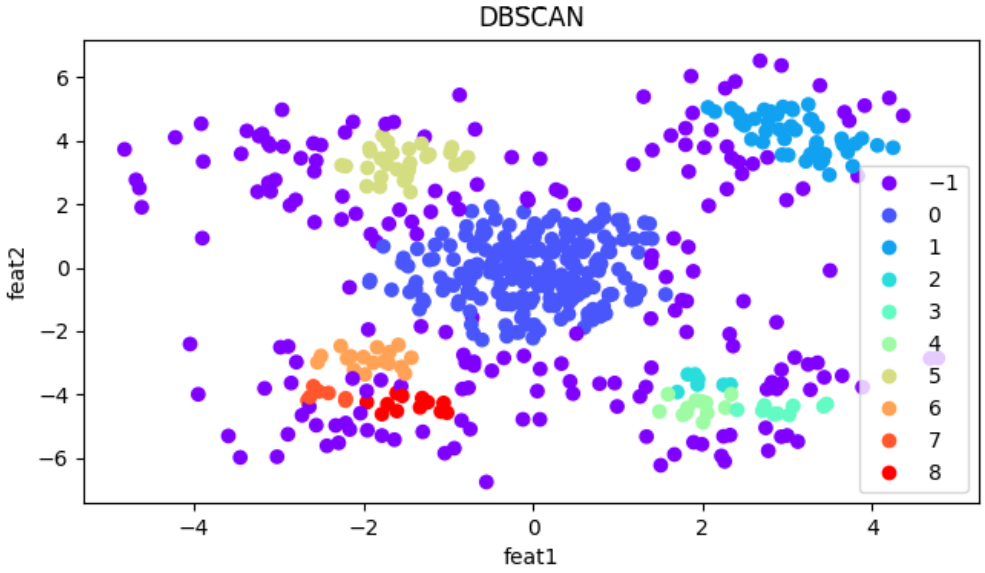
Raziskave se lahko nadaljujejo v smeri dodatnih razlag za posamezno gručo. V to je lahko vključena vizualizacija dodatnih primerov (ne samo medoida) iz različnih delov gruče in dodatne analize nečistih gruč, predvsem, kako naj bo gruča predstavljena. Atributi bi lahko opisali z statističnimi metodami, saj o njih naše razlage ne povedo dosti. Uporabno bi bilo tudi analizirati primernost podatkovnih množic za metodo s podkoncepti, za kar bi lahko razvili avtomatsko analizo primernosti.

Dodatek A

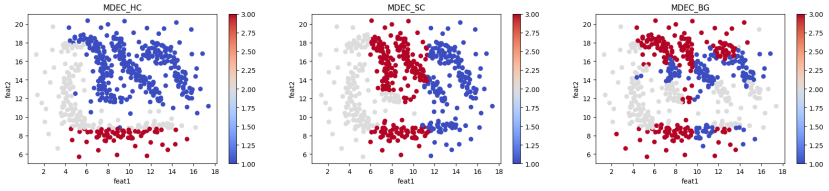
Celotna primerjava gručenj



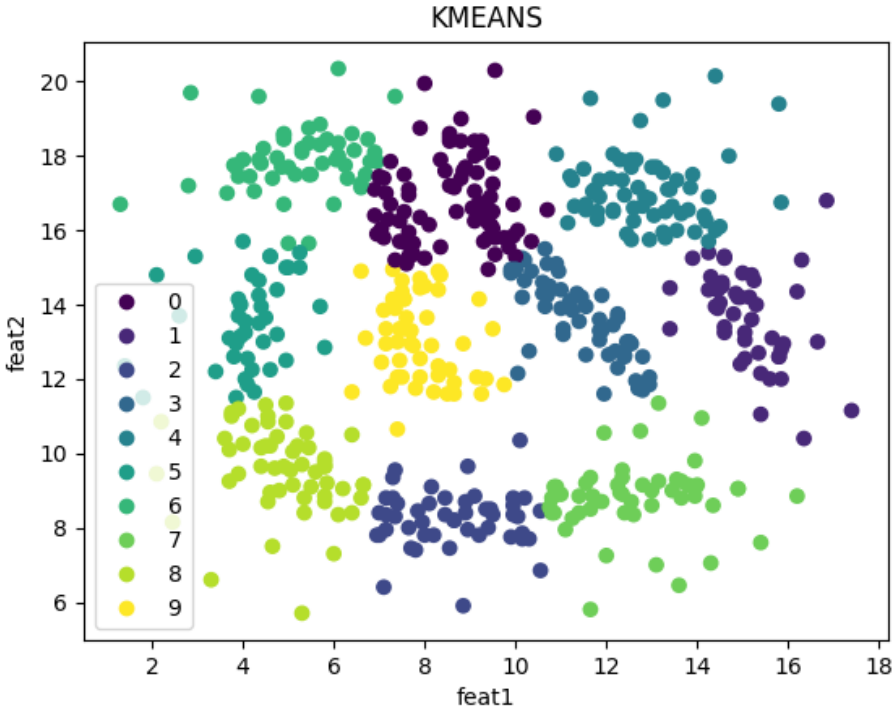
Slika A.1: Tri rezultati MDEC algoritma pri številu gruč k = 3 na podatkovni množici "Krožne gruče". Zanima nas najbolj desni rezultat z najboljšo hevristiko.



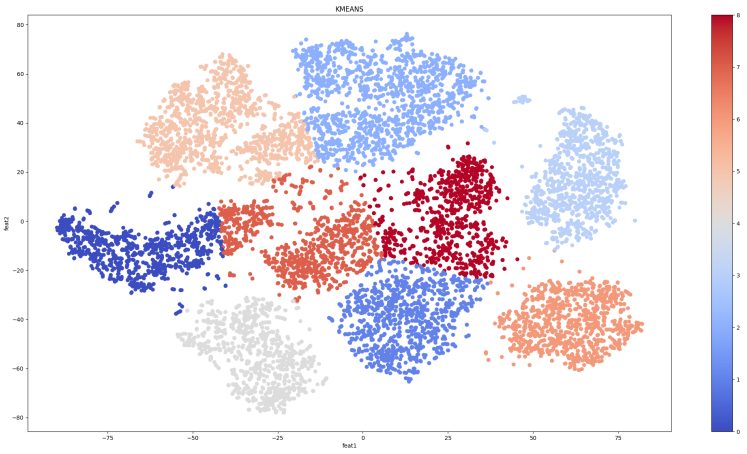
Slika A.2: Rezultat DBSCAN algoritma na podatkovni množici "Krožne gruče".



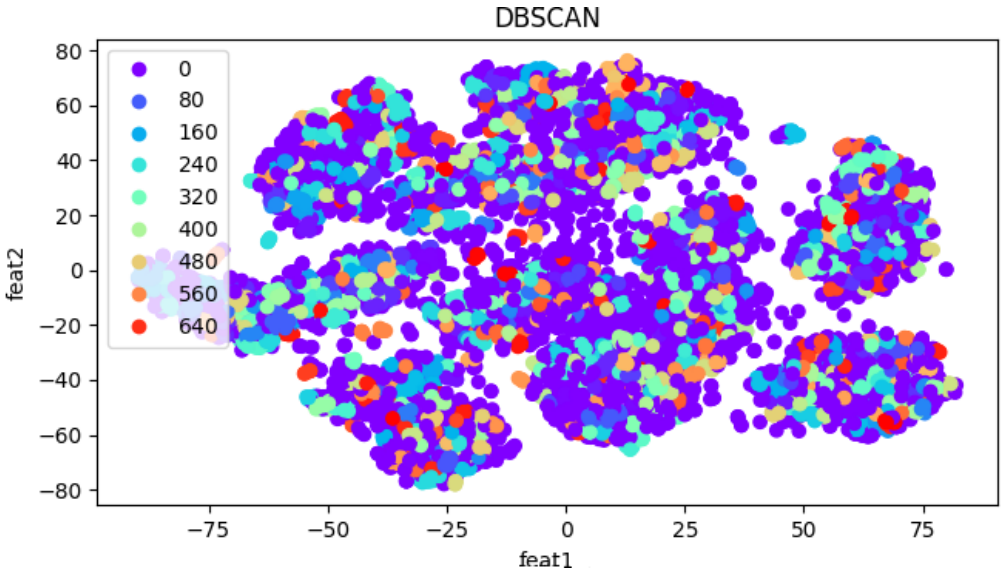
Slika A.3: Tri rezultati MDEC algoritma pri številu gruč k = 3 na podatkovni množici "Trakovi 4-3". Najbolj levi je dobil najboljšo oceno.



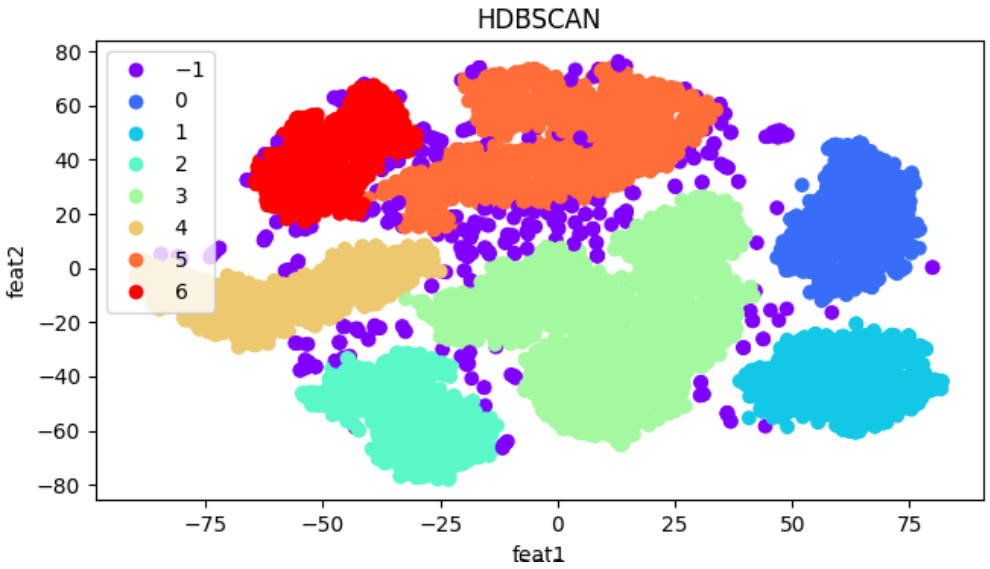
Slika A.4: Rezultat K-MEANS algoritma pri številu gruč k = 10 na podatkovni množici "Trakovi 4-3".



Slika A.5: Rezultat KMEANS algoritma na podatkovni množici MNIST. Vidimo, da je algoritem dosegel najboljšo hevristiko silhuete pri številu gruč k = 9. Vse gruče niso smiselne, kar se vidi pri temno modri gruči.



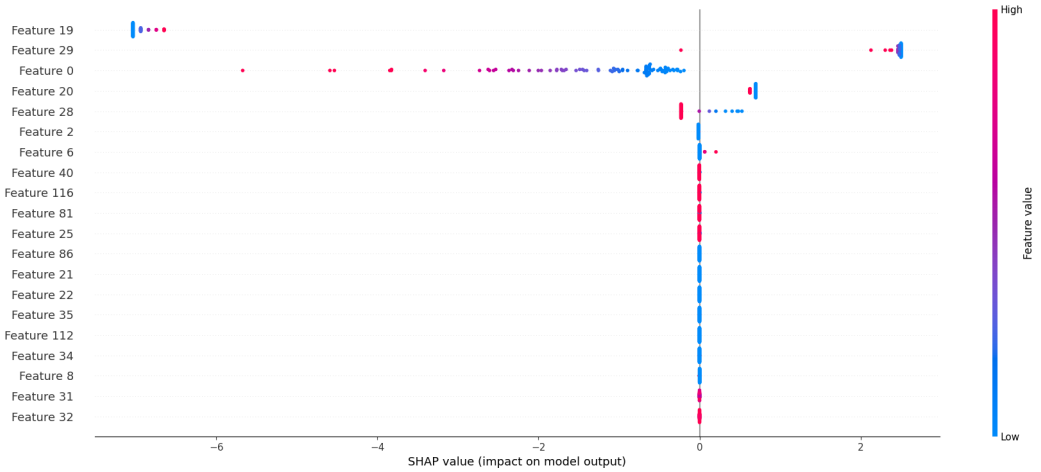
Slika A.6: Rezultat DBSCAN algoritma na podatkovni množici MNIST. Algoritem ni deloval po pričakovanjih, kar se vidi po nerazumnlivih oznakah. Ločene gruče so neobstoječe. Naša izbira parametrov se v tem primeru ne obrestuje, saj je algoritem našel 640 gruč.



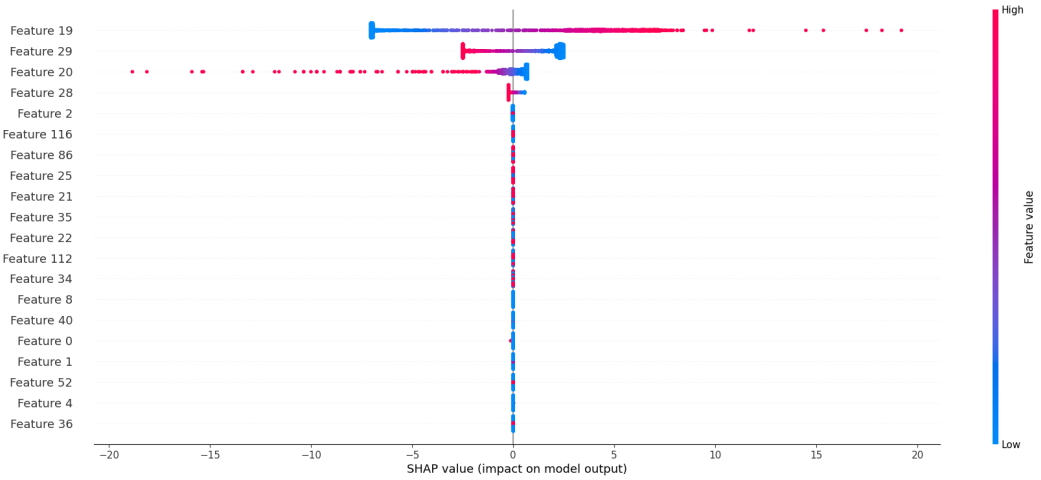
Slika A.7: Rezultat HDBSCAN algoritma na podatkovni množici MNIST. Algoritem je našel smiselne gruče (7), a ne vseh. Na sliki se vidi, da bi morali biti oranžna in zelena gruča razdeljeni na dodatne podgruče.

Dodatek B

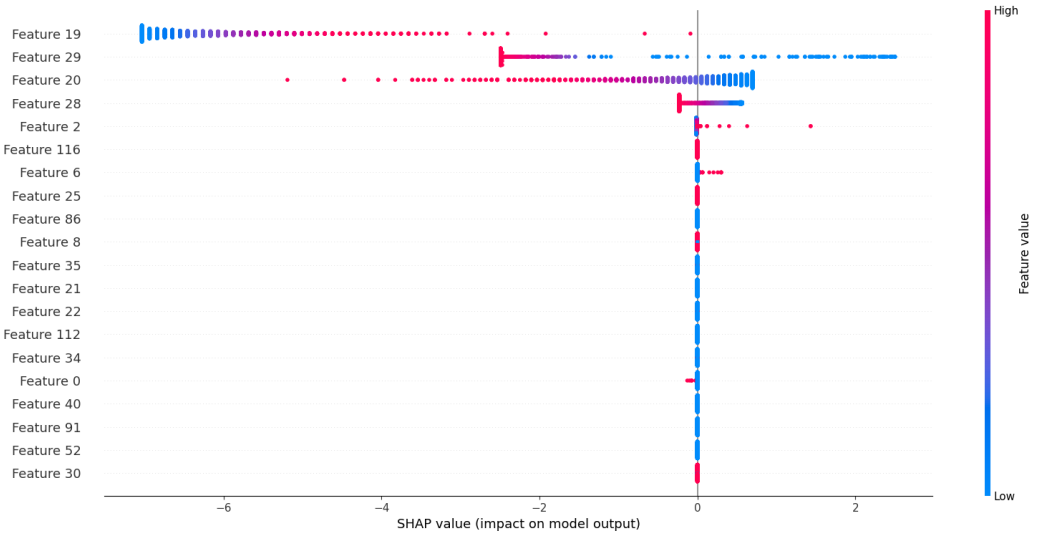
Vrednosti SHAP preostalih gruč iz poglavja 6



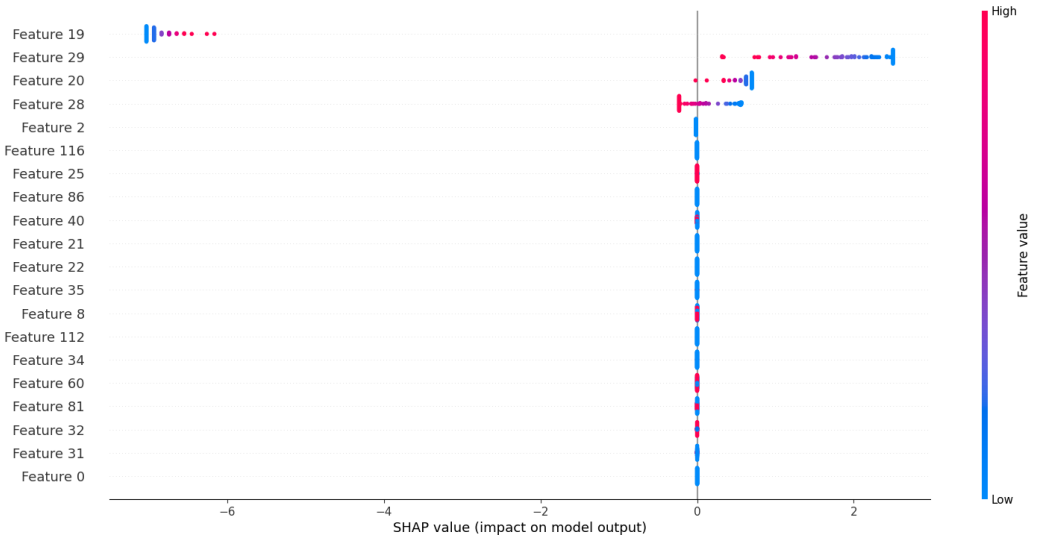
Slika B.1: Vrednosti SHAP za gručo nič na podatkovni množici KDD99.



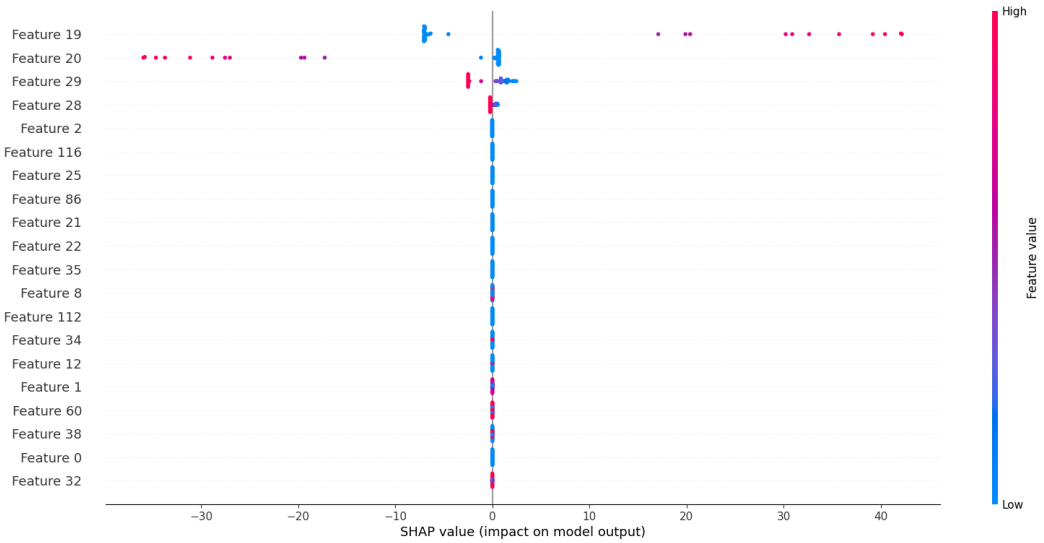
Slika B.2: Vrednosti SHAP za gručo dve na podatkovni množici KDD99.



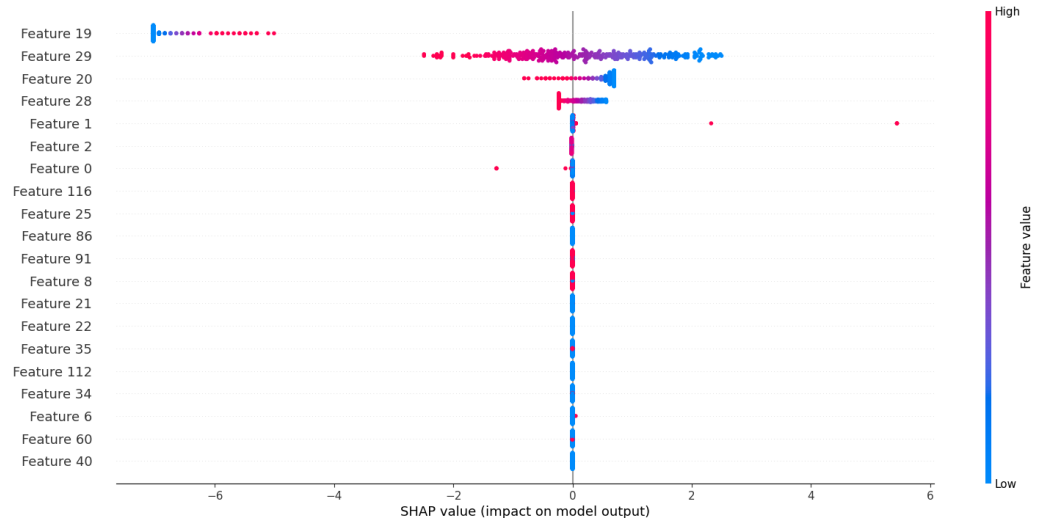
Slika B.3: Vrednosti SHAP za gručo tri na podatkovni množici KDD99.



Slika B.4: Vrednosti SHAP za gručo štiri na podatkovni množici KDD99.



Slika B.5: Vrednosti SHAP za gručo pet na podatkovni množici KDD99.



Slika B.6: Vrednosti SHAP za gručo šest na podatkovni množici KDD99.

Literatura

[1] Chidanand Apté in Sholom Weiss. "Data mining with decision trees and decision rules". V: *Future generation computer systems* 13.2-3 (1997), str. 197–210.

[2] Jason Brownlee. "How to Develop a CNN for MNIST Handwritten Digit Classification". V: *Machine Learning Mastery* (2019). Datum dosega 11.8.2023. URL: https://machinelearningmastery.com/how-to-develop-a-convolutional-neural-network-from-scratch-for-mnist-handwritten-digit-classification/.

[3] Alfred DeMaris. "A tutorial in logistic regression". V: *Journal of Marriage and the Family* (1995), str. 956–968.

[4] Charles Frenzel. "How To Tune HDBSCAN". V: *Towards Data Science* (2021). Datum dosega 11.8.2023. URL: https://towardsdatascience.com/tuning-with-hdbscan-149865ac2970.

[5] Jerome H. Friedman in Bogdan E. Popescu. "Predictive learning via rule ensembles". V: *The Annals of Applied Statistics* 2.3 (2008), str. 916–954. DOI: 10.1214/07-AOAS148.

[6] John A Hartigan in Manchek A Wong. "Algorithm AS 136: A k-means clustering algorithm". V: *Journal of the royal statistical society. series c (applied statistics)* 28.1 (1979), str. 100–108.

[7] Dong Huang, Chang-Dong Wang, Jian-Huang Lai in Chee-Keong Kwoh. "Toward multidiversified ensemble clustering of high-dimensional data: From subspaces to metrics and beyond". V: *IEEE Transactions on Cybernetics* 52.11 (2021), str. 12231–12244.

[8] Kamran Khan, Saif Ur Rehman, Kamran Aziz, Simon Fong in Sababady Sarasvady. "DBSCAN: Past, present and future". V: *The fifth international conference on the applications of digital information and web technologies (ICADIWT 2014)*. IEEE. 2014, str. 232–238.

[9] Daniel Kleine. "Detecting knee- / elbow points in a graph of a function". V: *Towards Data Science* (2021). Datum dosega 20.7.2023. URL: https://towardsdatascience.com/detecting-knee-elbow-points-in-a-graph-d13fc517a63c.

[10] Shih-Wei Lin, Kuo-Ching Ying, Chou-Yuan Lee in Zne-Jung Lee. "An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection". V: *Applied Soft Computing* 12.10 (2012), str. 3285–3290.

[11] Scott M Lundberg in Su-In Lee. "A unified approach to interpreting model predictions". V: *Advances in neural information processing systems* 30 (2017).

[12] Basim Mahbooba, Mohan Timilsina, Radhya Sahal in Martin Serrano. "Explainable artificial intelligence (XAI) to enhance trust management in intrusion detection systems using decision tree model". V: *Complexity* 2021 (2021), str. 1–11.

[13] Leland McInnes, John Healy in Steve Astels. "hdbscan: Hierarchical density based clustering." V: *J. Open Source Softw.* 2.11 (2017), str. 205. DOI: 10.21105/joss.00205.

[14] Edoardo Mosca, Ferenc Szigeti, Stella Tragianni, Daniel Gallagher in Georg Groh. "SHAP-based explanation methods: a review for NLP interpretability". V: *Proceedings of the 29th International Conference on Computational Linguistics*. 2022, str. 4593–4603.

[15] Davoud Moulavi, Pablo A Jaskowiak, Ricardo JGB Campello, Arthur Zimek in Jörg Sander. "Density-based clustering validation". V: *Proceedings of the 2014 SIAM international conference on data mining*. SIAM. 2014, str. 839–847.

[16] Tara Mullin. "DBSCAN Parameter Estimation Using Python". V: *Medium* (2020). Datum dosega 8.7.2023. URL: https://medium.com/@tarammullin/dbscan-parameter-estimation-ff8330e3a3bd.

[17] Ashutosh Nayak. "XGBoost: An Intuitive Explanation". V: *Towards Data Science* (2019). Datum dosega 15.8.2023. URL: https://towardsdatascience.com/xgboost-an-intuitive-explanation-88eb32a48eff.

[18] Conor Nugent in Pádraig Cunningham. "A case-based explanation system for black-box systems". V: *Artificial Intelligence Review* 24 (2005), str. 163–178.

[19] Peter J. Rousseeuw. "Silhouettes: A graphical aid to the interpretation and validation of cluster analysis". V: *Journal of Computational and Applied Mathematics* 20 (1987), str. 53–65. DOI: https://doi.org/10.1016/0377-0427(87)90125-7.

[20] Mohammad Khubeb Siddiqui in Shams Naahid. "Analysis of KDD CUP 99 dataset using clustering based data mining". V: *International Journal of Database Theory and Application* 6.5 (2013), str. 23–34.

[21] Saurabh Singh. "Building an Intrusion Detection System using KDD Cup'99 Dataset". V: *Medium* (2020). Datum dosega 16.8.2023. URL: https://medium.com/analytics-vidhya/building-an-intrusion-detection-model-using-kdd-cup99-dataset-fb4cba4189ed.

[22] Laurens Van der Maaten in Geoffrey Hinton. "Visualizing data using t-SNE." V: *Journal of machine learning research* 9.11 (2008).

[23] Remah Younisse, Ashraf Ahmad in Qasem Abu Al-Haija. "Explaining Intrusion Detection-Based Convolutional Neural Networks Using Shapley Additive Explanations (SHAP)". V: *Big Data and Cognitive Computing* 6.4 (2022), str. 126.