Istraživanje asteroida primjenom algoritama grupiranja u strojnom učenju

Đunđek, Vedrana

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
FIZIČKI ODSJEK

Vedrana Đunđek

Istraživanje asteroida primjenom algoritama grupiranja u strojnom učenju

Diplomski rad

Zagreb, 2017.

SVEUČILIŠTE U ZAGREBU
PRIRODOSLOVNO-MATEMATIČKI FAKULTET
FIZIČKI ODSJEK

INTEGRIRANI PREDDIPLOMSKI I DIPLOMSKI SVEUČILIŠNI STUDIJ
FIZIKA I INFORMATIKA; SMJER NASTAVNIČKI

Vedrana Đunđek

Diplomski rad

Istraživanje asteroida primjenom algoritama grupiranja u strojnom učenju

Voditelj diplomskog rada: izv. prof. dr. sc. Goranka Bilalbegović

Ocjena diplomskog rada: _____

Povjerenstvo: 1. _____

2. _____

3. _____

Datum polaganja: _____

Zagreb, 2017.

Hvala mentorici, roditeljima, sestri i prijateljima. Također se zahvaljujem NASA SpaceRocks ekipi za ”NASA developer key” koji mi je omogućio brži rad na njihovoj bazi NeoWs (Near Earth Object Web Service).

The Doctor: ”And when you go back to the stars and tell others of this planet, when you tell them of its riches, its people, its potential, when you talk of the Earth, then make sure that you tell them this...*[shouts]*...It. Is. Defended!”

Sažetak

S napretkom teleskopa i razvojem algoritama strojnog učenja povećava se mogućnost uočavanja novih asteroida i određivanja njihovih osobina. U ovom diplomskom radu analizirani su asteroidi u blizini Zemlje pomoću nenadziranog strojnog učenja. Koristili smo dva algoritama grupiranja u programskom jeziku Python: KMEANS i DBSCAN. Pokušali smo grupirati asteroide u blizini Zemlje koristeći podatke američke državne uprave za zrakoplovna i svemirska istraživanja (NASA). Usporedili smo rezultate s poznatim potencijalno opasnim asteroidima. Trenutne verzije korištenih programa dostupne su na adresi https://github.com/lib686/avenge-the-dinosaurs.

Opisan je projekt znanosti za građanstvo Asteroids@home kao metodički dio diplomskog rada. Diskutiran je značaj ovog projekta za popularizaciju astronomije, informatike te znanosti općenito.

Ključne riječi: asteroidi, potencijalno opasni asteroidi, strojno učenje, algoritmi grupiranja, algoritam *k*-srednjih vrijednosti, DBSCAN algoritam

The Application of Machine Learning Clustering Algorithms to the Study of Asteroids

Abstract

With the advancement of telescopes and the development of machine learning algorithms we increase the ability to detect new asteroids and determine their characteristics. In this master's degree thesis, Near-Earth Asteroids (NEAs) were analyzed using unsupervised machine learning. We were using two clustering algorithms in programming language Python: KMEANS and DBSCAN. We attempted to cluster NEAs using data from National Aeronautics and Space Administration (NASA). We compared the results with known Potentially Hazardous Asteroids (PHA). Current versions of the programs used can be found at https://github.com/lib686/avenge-the-dinosaurs.

We described the citizen science project Asteroids@home as a teaching method part of the thesis. The importance of this project for the popularization of astronomy, computer science, and science in general is discussed.

Keywords: asteroids, potentially hazardous asteroids, machine learning, clustering algorithms, *k*-means algorithm, DBSCAN algorithm

Sadržaj

1 Uvod

Znanstvenici su ukazali na važan utjecaj asteroida na druge svemirske objekte, uključujući i Zemlju. Postoje teorije po kojima su sudari s asteroidima i kometima omogućili život na Zemlji, ali i uzrokovali izumiranje dinosaura i drugih vrsta.

U 20. stoljeću asteroidi su počeli okupirati i maštu umjetnika te su pronašli svoje mjesto u popularnoj kulturi - literaturi, filmovima, serijama, računalnim igricama. U knjizi Mali princ autor je glavni lik smjestio na asteroid naziva B-612. Osim potencijalnog naseljavanja, razvija se i ideja rudarenja asteroida kako bi se iskoristili njihovi resursi. Popularna arkadna računalna igra "Asteroids" (1979.) temeljila se na uništavanju asteroida od strane svemirskog broda te izbjegavanju sudara. Navedena igrica je sudjelovala u razvoju pogrešne ideje o postojanju gustih asteroidnih polja unutar kojih svemirski brodovi imaju malu mogućnost manevriranja u svrhu izbjegavanja sudara. Znanost je pokazala da su asterodi mali u promjeru te da je njihova međusobna udaljenost velika. Zbog toga bi prolazak kroz područje gdje se nalaze asteroidi bio relativno siguran.

Uz moguće koristi od asteroida poput kolonizacije i rudarenja, pojavila se i zabrinutost vezana uz mogućnost udara asteroida u Zemlju. Ne možemo ne spomenuti popularni film Armageddon u kojemu ljudi nastoje spriječiti udar asteroida na Zemlju i kataklizmične posljedice. Predloženo je bušenje u unutrašnjost asteroida te postavljanje nuklearne bombe koja bi razdijelila asteroid te bi tako bio izbjegnut izravan sudar s našim planetom.

Zbog otkrivanja velikog broja asteroida, kao i pada meteora u Chelyabinsku 2013. godine, razvile su se spoznaje o opasnosti od udara asteroida i kometa na Zemlju. Razvija se i interes za korištenjem računalnih metoda kako bi se asteroidi mogli brže i efikasnije klasificirati. To bi moglo omogućiti i razvoj boljih metoda zaštite planeta od potencijalnog udara asteroida. Novi asteroidi se otkrivaju svakodnevno. Zbog toga se istražuje mogućnost iskorištavanja metoda strojnog učenja u obradi novih podataka i pronalaženju korelacija u gibanju i osobinama asteroida.

U ovom radu korišteni su podaci o asteroidima i kometima u blizini Zemlje (Near Earth Objects, NEO) koje je uredila Američka državna uprava za zrakoplovna i svemirska istraživanja (NASA, National Aeronautics and Space Administration). Informacije o tim objektima smo prikupljali sa web servisa NeoWs (Near Earth Object

Web Service). Korištenjem KMEANS i DBSCAN algoritama grupiranja provjeravali smo postoje li ovisnosti unutar podataka o asteroidima u blizini Zemlje. Specijalno smo analizirali potencijalno opasne asteroide (PHA, Potentially Hazardous Asteroids) za Zemlju, te one koji to nisu.

U drugom poglavlju uvodimo osnovne pojmove o asteroidima i njihovim osobinama. Naglasak se stavlja na asteroide u blizini Zemlje. U trećem poglavlju napisan je pregled osnovnih ideja strojnog učenja, algoritama grupiranja (KMEANS, DBSCAN) te osnovnih biblioteka i metoda korištenih u radu. U četvrtom poglavlju opisani su rezultati grupiranja podataka o asteroidima.

U Dodatku A prikazano je distribuirano, volontersko računarstvo kao način poticanja interesa za znanost preko projekta znanosti za građanstvo (engl. citizen science). Opisan je projekt Asteroids@home koji korisnicima omogućava sudjelovanje u izračunima karakteristika asteroida.

2 Asteroidi

Asteroidi su mali, najčešće stjenoviti ili metalni, objekti koji se gibaju eliptičnim putanjama oko Sunca. Rotiraju oko vlastite osi koja ovisi o njihovim karakteristikama. Naziv im je dao britanski astronom William Herschel. Na grčkom jeziku *asteroeidēs* korijen vuče iz riječi *astēr* koja označava zvijezdu, planet. Asteroidi su ostatci koji prilikom formiranja Sunčevog sustava prije 4.6 milijardi godina nisu sudjelovali u stvaranju većih tijela te se od onda nisu promijenili. Zbog toga proučavanjem asteroida dobivamo uvid u procese koji su se događali za vrijeme stvaranja Sunčevog sustava. Promjeri su im od nekoliko metara, do 530 kilometara kod Veste koja je najveći uočen asteroid Sunčevog sustava. Radi se o čvrstim objektima nepravilnih oblika koji su rezultat sudara i stapanja s drugim tijelima. Nekoliko najvećih asteroida su približno sferičnog oblika. Asteroidi nemaju vlastitu atmosferu i ne mogu podržavati oblike života koji su nam danas poznati.

Asteroide možemo podijeliti prema njihovim orbitama odnosno, lokacijama unutar Sunčevog sustava. Za potrebe ovog rada ćemo spomenuti nekoliko grupacija [1].

- *Glavni asteroidni pojas*

  Glavni asteroidni pojas (engl. *Main asteroid belt*) se prostire između 2 i 3.5 *au*[1] odnosno između Marsa i Jupitera.

  U području glavnog asteroidnog pojasa orbitira većina poznatih asteroida. Procjenjuje se da ovaj pojas sadrži između 1.1 i 1.9 milijuna asteroida većih od jednog kilometra u promjeru, te milijune manjih. Prilikom formiranja Sunčevog sustava gravitacija Jupitera onemogućila je stvaranje drugog planeta u tom području te su se mali objekti sudarali međusobno i stvarali objekte koje danas promatramo kao asteroide tog pojasa. Stvaranje Jupitera je sprječavalo grupiranje manjih objekata u veće u tom prostoru.

- *Trojanski asteroidi*

  Trojanski asteroidi dijele putanju s nekim planetom, no ne sudaraju se s njim jer se skupljaju oko $L4$ ili $L5$ Langrangeove točke[2].

[1] engl. *Astronomical unit*, jedinica duljine koja predstavlja prosječnu udaljenost Zemlje od Sunca; danas je 1 *au* definiran kao 149 597 870 700 metara

[2] Langrangeove točke su lokacije u blizini dva objekta velikih masa koji rotiraju oko zajedničkog centra mase u kojima treći objekt (u odnosu na njih zanemarivo male mase) ostaje u orbiti na približno stalnoj udaljenosti od ta dva tijela

S obzirom na to da je na tim lokacijama gravitacijsko privlačenje Sunca i planeta izjednačeno s nastojanjem asteroida da promjene orbitu, asteroidi se gibaju stabilno ispred ili iza planeta pod kutom od približno 60 stupnjeva. Postoje npr. Jupiterovi trojanci (najbrojniji), trojanci Neptuna i Marsa, a prije nekoliko godina NASA je otkrila i Zemljinog trojanca.

- *Asteroidi u blizini Zemlje* (engl. *Near-Earth Asteroids, NEA*)

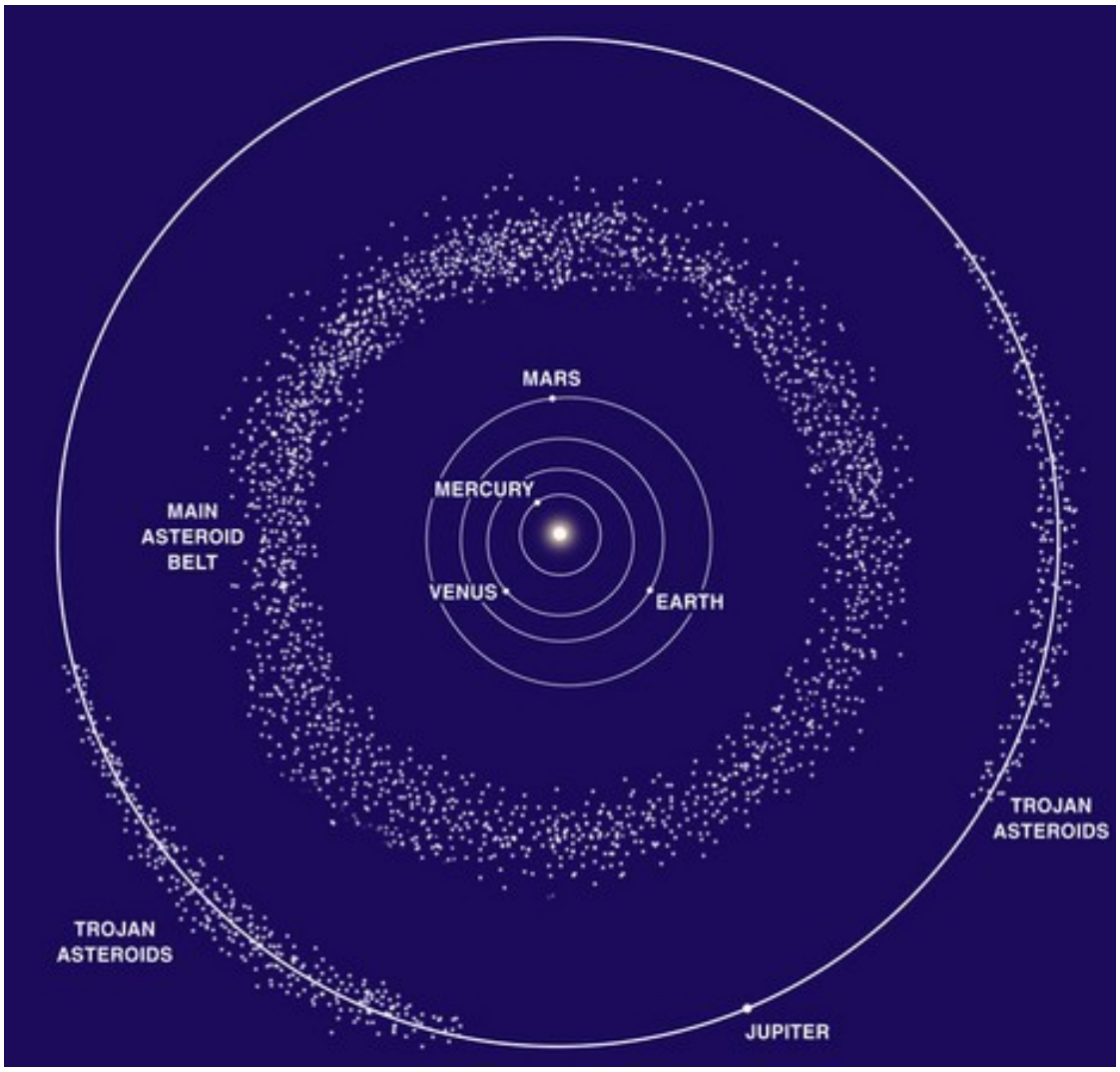
  Asteroidi u blizini Zemlje su oni čija orbita ih dovodi na $1.3\,au$ (195 milijuna km) od Sunca. Ti asteroidi se približe Zemljinoj orbiti na 44 milijuna km. Za usporedbu, prosječna udaljenost Mjeseca od Zemlje iznosi 0.00257 *au* što znači da koncept blizine asteroida shvaćamo jako široko. Vjeruje se kako se radi o objektima koji su, ili zbog sudara s drugim asteroidima, ili zbog utjecaja Jupitera, gurnuti iz područja glavnog asteroidnog pojasa. Do sada nam je poznato više od 700 000 asteroida od čega je u šestom mjesecu 2017. godine, 16 209 asteroida pripadalo skupini asteroida u blizini Zemlje.

Vizualizaciju raspodjele navedenih asteroida možemo vidjeti na Slici 2.1. U Poglavlju 2.1 predstavljena je detaljnije NEA grupacija.

Ceres (otkriven 1801.) se do 2006. godine smatrao najvećim asteroidom. Ima promjer od 945 kilometara. Zbog nove klasifikacije Plutona i Ceres je postao patuljasti planet[3]. Ceres ima masu koja sačinjava jednu trećinu ukupne mase svih objekata asteroidnog pojasa [1]. Ukupna masa svih asteroida je manja od mase Mjeseca. Na Slici 2.2 uspoređena je veličina nekoliko objekata.

Asteroid Vesta je poznat i kao 4 Vesta. Dobio je oznaku 4 zbog toga što je četvrti otkriven, dok je naziv Vesta dobio po rimskoj božici vatre, ognjišta i doma. Danas, s povećanjem broja otkrivenih asteroida i nemogućnosti određivanja redoslijeda njihova otkrivanja, postoje drugačija pravila njihovih imenovanja [2]. Nakon što asteroid dobije svoju službenu oznaku koja označava vrijeme kada je otkriven (npr. 1983 TE1) te kada se dovoljno dugo promatra kako bi se njegova orbita mogla pouzdano predvidjeti, dobije brojčanu oznaku poput Veste. Tek onda, osoba koja ga je otkrila ima pravo prva predložiti njegov naziv. Trojanski asteroidi se nazivaju prema herojima Trojanskog rata dok se asteroidima u blizini Zemlje, u pravilu, daju imena iz

[3]Objekti koji su skoro sferičnog oblika, imaju eliptičnu putanju oko Sunca na čijem putu se nalaze drugi objekti poput asteroida



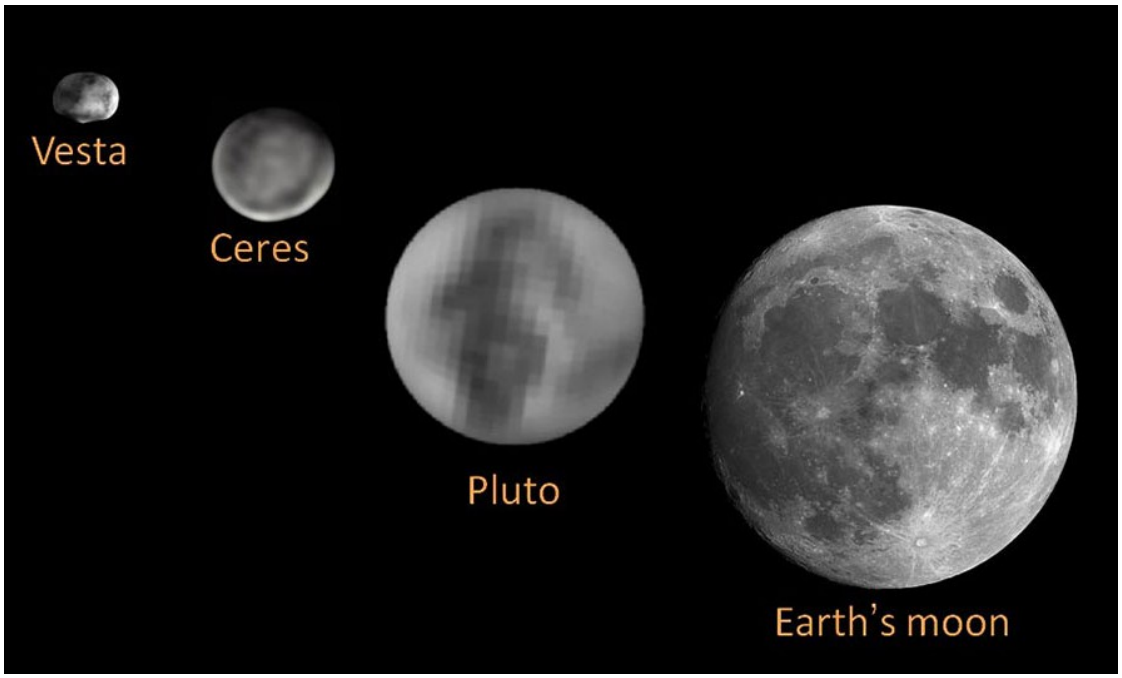
Slika 2.1: Grupacije asteroida [3]

mitologije. Bez obzira na stroga pravila za imenovanje, danas u glavnom asteroidnom pojasu orbitiraju asteroidi 9007 James Bond, 12818 Tomhanks, 8353 Megryan, 2309 Mr. Spock te 13681 Monty Python. Asteroidi postoje i u drugim područjima Sunčevog sustava no važno je naglasiti kako u ovom radu analiziramo podatke samo asteroida u blizini Zemlje.

U nastavku slijedi pregled osnovnih parametara asteroida. Pomoću njih opisujemo ponašanje i karakteristike asteroida u blizini Zemlje.

Apsolutna magnituda (engl. *Absolute magnitude,* H) je jedan od rijetkih parametara koji je određen za većinu poznatih asteroida. S tim parametrom se može procijeniti veličina asteroida. H je sjaj kojeg objekt ima na udaljenosti 1 *au* od Sunca i 1 *au* od Zemlje (promatrača) uz idealni slučaj kada bi fazni kut iznosio nula.

Uz apsolutnu magnitudu, potrebno je poznavati i parametar *albedo* objekta kako



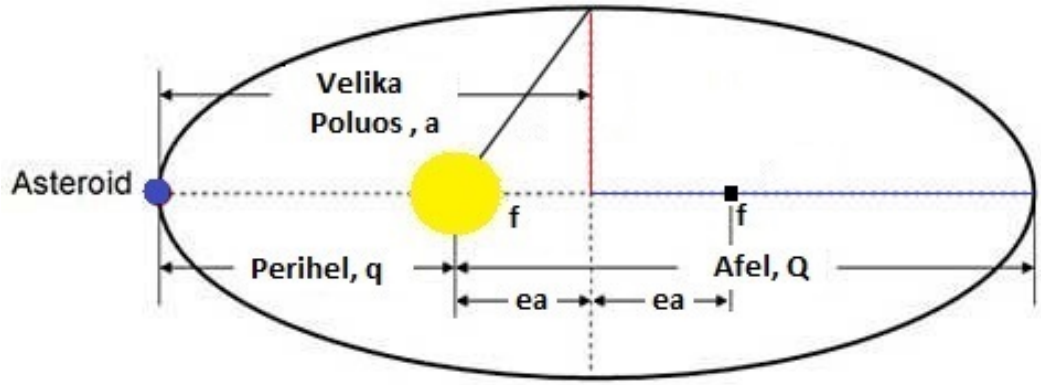
Slika 2.2: Usporedba veličina Mjeseca, Plutona, Ceresa i Veste [3]

bi se mogla točnije procijeniti prava veličina objekta. On govori o reflektivnim karakteristikama objekta. Radi se o odnosu upadne svjetlosti na objekt i reflektirane svjetlosti s objekta. Vrijednost za svaki objekt se nalazi unutar intervala $[0, 1]$. Albedo vrijednosti jedan predstavlja idealne refleksivne karakteristike. Npr. Venera ima najveću vrijednost albeda od svih planeta Sunčevog sustava te je zbog toga jako dobro vidljiva sa Zemlje. Vrijednost albeda joj je blizu 0.7 što znači da reflektira gotovo 70% svjetlosti koja na nju padne.

MOID (engl. *Minimum Orbital Intersection Distance*) je veličina koja predstavlja minimalnu udaljenost između orbita dva objekta. Vrijednosti *MOID*-a korištene u ovom radu predstavljaju odnos orbite određenog asteroida i orbite Zemlje. Niska *MOID* vrijednost može ukazivati na mogućnost sudara asteroida sa Zemljom te putanje takvih asteroida treba pozorno pratiti. S obzirom na to kako se orbita asteroida mijenja s vremenom, mijenja se i njegov parametar *MOID*.

Velika poluos orbite (engl. *Semi-major axis*, a) označava udaljenost koja iznosi polovicu vrijednosti velike osi elipse. Perihel (q) predstavlja udaljenost asteroida kada je najbliži Suncu dok afel (Q) kada je najudaljeniji od Sunca.

Ekscentricitet orbite (engl. *Eccentricity*, e) govori o mjeri kojom se orbita objekta oko drugog tijela razlikuje od kružne orbite. Definiramo ga kao odnos udaljenosti između dva fokusa (f) elipse (koja iznosi $2ea$) i duljine velike osi ($2a$). Kružna orbita



Slika 2.3: Prikaz parametara povezanih s eliptičnom putanjom [4]

ima $e = 0$ dok eliptična može imati vrijednosti: $0 < e < 1$ (Slika 2.3).

Inklinacija orbite (engl. *Inclination*, i) je kut između vektora normale na ravninu orbite tijela i neke referentne ravnine.

Tisserandov parametar [5] predstavlja vrijednost koja je približno očuvana između planeta i objekta prije i poslije njihovog susreta. Ovaj parametar omogućava određivanje različitih tipova orbita. Uzima u obzir ekscentricitet, inklinaciju i veliku poluos objekta te veliku poluos planeta što je vidljivo u Formuli 2.1 kojom se određuje T_J u odnosu na Jupiter. Tisserandov parametar u odnosu na Jupiter se isto tako koristi kako bi se odredila razlika između asteroida i kometa. Uzima se da je $T_J > 3$ za asteroide, dok je $2 < T_J < 3$ za komete. Spomenuto je točno u pravilu, no postoje iznimke.

$$T_J = \frac{a_J}{a} + 2[(1 - e^2)\frac{a}{a_J}]^{1/2} \cos(i) \tag{2.1}$$

Kada se otkrije novi objekt, izračunava se njegova vrijednost T_J. Ako se u narednim promatranjima otkrije još jedan objekt s istim Tisserandovim parametrom, a različitim drugim karakteristikama, navjerojatnije se radi o istom objektu čiji su se orbitalni parametri izmijenili s vremenom zbog susreta s Jupiterom.

2.1 Asteroidi u blizini zemlje

Asteroidi u blizini Zemlje i kometi u blizini Zemlje (engl. *Near-Earth Comets*, NEC) predstavljaju tzv. objekate u blizini Zemlje (engl. *Near-Earth Objects*, NEO). Spomenuti asteroidi i kometi pripadaju NEO kategoriji ako im je perihelna udaljenost q

manja od 1.3 *au* od Sunca. Većina NEO su upravo asteroidi koji imaju, zbog gravitacijskog privlačenja s drugim objektima u svemiru ili zbog sudara s istima, odgovarajuću orbitu koja ih dovodi u blizinu Zemlje. NEA je dakle pojam koji uključuje više kategorija asteroida te je naglasak upravo na odnosu njihove orbite sa Zemljinom [6]. NEA dijelimo u grupe prema udaljenosti perihela q, udaljenosti afela Q te njihovim velikim poluosima a [7].

- Asteroidi *Atirine grupe* imaju orbitu manju od Zemljine i ne križaju se s njom. U teoriji ne predstavljaju opasnost za Zemlju, no njihove orbite se mogu izmijeniti međudjelovanjem s drugim objektima.
- Asteroidi *Atenove grupe* isto tako prelaze putanju Zemlje, no njihov afel se nalazi unutar Zemljine putanje te im je orbita nešto manja od Zemljine.
- Asteroidi *Apollove grupe* prelaze putanju Zemlje odnosno njihovi se periheli nalaze unutar putanje Zemlje. Zbog toga postoji mogućnost ulaska asteroida ove grupe u Zemljinu atmosferu s potencijalnim padom na površinu Zemlje. Radi se o najbrojnijoj grupi NEA asteroida.
- Asteroidi *Amorove grupe* imaju veću orbitu od Zemljine i takav perihel da ne nailaze na Zemljinu orbitu. Zbog toga možemo reći da asteroidi iz Amorove grupe nisu prijetnja za Zemlju. No s vremenom, njihove orbite se mogu promijeniti te je od velikog značaja kontinuirano praćenje putanja svih asteroida. Amorovoj grupi asteroida pripada i asteroid 433 Eros čiji nepravilan izgled i strukturu možemo vidjeti na Slici 2.4. Eros je prvi otkriveni NEA, no drugi je po veličini. Najveći NEA naziva se 1036 Ganymed i isto tako pripada Amorovoj grupaciji asteroida.

Ove grupe su dobile nazive prema poznatim otkrivenim asteroidima navedenih osobina. Radi se redom o asteroidima: 163693 Atira, 2062 Aten, 1862 Apollo, 1221 Amor. Osobine i orbite ovih grupa u odnosu na Zemlju su prikazane u Tablici 2.1.

2.2 Potencijalno opasni asteroidi

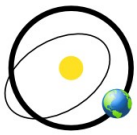
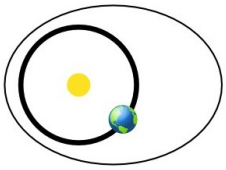
Asteroidi čija je minimalna udaljenost njihove i Zemljine orbite (*MOID*) manja ili jednaka 0.05 *au*, a apsolutna magnituda manja ili jednaka 22, smatraju se potenci-

Slika 2.4: Mozaik slika asteroida 433 Eros dobivenih za vrijeme misije NEAR Shoemaker [3]

jalno opasnim asteroidima (engl. *Potentially Hazardous Asteroids*, PHA). Od 16 tisuća poznatih NEA, za čak 1 803 se smatra da su PHA [8].

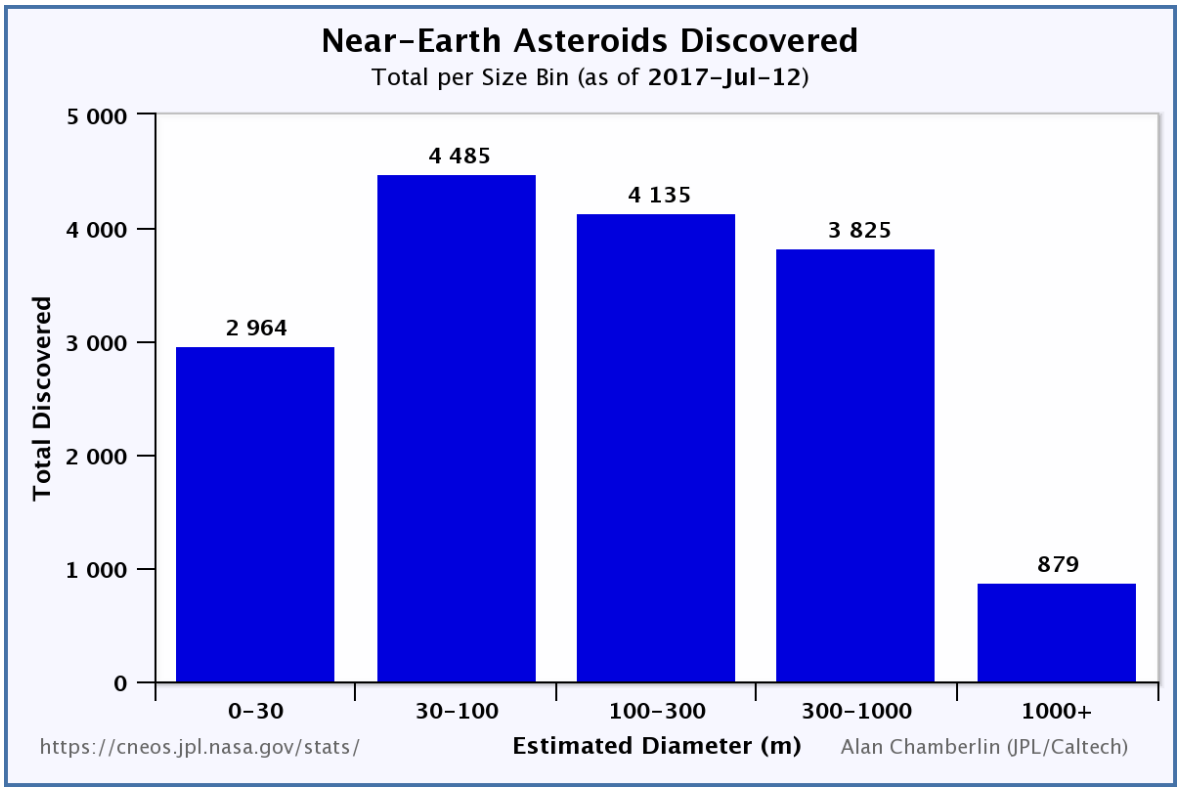
Još od vremena nastanka Zemlje, asteroidi se stalno sudaraju s njom. Potencijalno opasni asteroidi s razarajućim posljedicama su jako rijetki. Na Slici 2.5 možemo vidjeti statistiku broja otkrivenih NEA u odnosu na veličinu njihovih promjera. Možemo uočiti kako postoji skoro 900 NEA čiji je promjer veći od jednog kilometra što ih stavlja u rizičnu skupinu onih asteroida koji mogu imati kobne posljedice za život na Zemlji. Manji asteroidi mogu uzrokovati razarajući tsunami, ili uništiti veći grad. Prosječno jedanput godišnje manji asteroidi od oko 2 metra ulaze u Zemljinu atmosferu no raspadaju se prije dolaska do površine. Prema NASA-i, asteroidi manji od 25 metara će najvjerojatnije izgorjeti u atmosferi Zemlje. Asteroid ili njegov dio koji se sudari sa Zemljom naziva se meteorit. Poznata teorija tvrdi kako je upravo asteroid (ili komet) promjera od najmanje 10 kilometara uzrokovao izumiranje dinosaura i mnogih drugih vrsta prije 65 milijuna godina. Druge teorije predviđaju mogućnost da su građevni elementi života te velike količine vode koje su danas na Zemlji stigli na naš planet upravo preko sudara s asteroidima ili kometima u prošlosti.

Nedavan slučaj pada asteroida dogodio se u Chelyabinsku, gradu u Rusiji (2013.

Grupe NEO			
Grupa	Obilježja	Opis	Vizualizacija
NEA	$q < 1.3$ *au*	Asteroidi u blizini Zemlje	-
Atira	$a < 1.0$ *au* $Q < 0.983$ *au*	Asteroidi u blizini Zemlje čije orbite su u potpunosti sadržane unutar orbite Zemlje	
Aten	$a < 1.0$ *au* $Q > 0.983$ *au*	Asteroidi u blizini Zemlje s orbitom koja se križa sa Zemljinom te čija je velika poluos orbite manja od Zemljine	
Apollo	$a > 1.0$ *au* $q < 1.017$ *au*	Asteroidi u blizini Zemlje s orbitom koja se križa sa Zemljinom te čija je velika poluos orbite veća od Zemljine	
Amor	$a > 1.0$ *au* $q > 1.017$ *au* $q < 1.3$ *au*	Asteroidi u blizini Zemlje koji se približavaju Zemlji s orbitom izvan orbite Zemlje no unutar orbite Marsa	
PHA	MOID≤ 0.05 *au* $H \leq 22$	Asteroidi u blizini Zemlje kojima je minimalna udaljenost orbite od Zemljine 0.05 *au* ili manja i čija je apsolutna magnituda 22 ili sjajnija	-

Tablica 2.1: Podjela asteroida u blizini Zemlje [7]

godine). Imamo mogućnost vidjeti mnoge snimke tog događaja koji je uzrokovao udarni val i ozlijedio više od tisuću ljudi. Poznati slučaj pada objekta dogodio se i u Sibiru 30. 6. 1908. godine (Tunguska). Iako relativno mali, od oko 100 metara u promjeru, imao je razarajući utjecaj na skoro 2000 km^2 površine. U spomen na ovaj događaj 30. lipnja je proglašen Međunarodnim danom asteroida.



Slika 2.5: Broj otkrivenih NEA prema njihovom promjeru [9]

NEO objekti mogu predstavljati veliku opasnost za Zemlju zbog čega je NASA 2016. godine objavila otvaranje Ureda za koordinaciju obrane planeta (engl. Planetary Defense Coordination Office, PDCO) čiji je zadatak praćenje i rana detekcija potencijalno opasnih objekata. Za sada, samo je za nekoliko asteroida predviđen njihov pad na površinu Zemlje prije samog događaja. Prvi takav slučaj je asteroid 2008 TC3 otkriven otprilike 20 sati prije udara.

3 Strojno učenje

Strojno učenje (engl. *Machine Learning*, ML) primjenom računalnog programiranja omogućava izvođenje novog znanja iz prijašnjih iskustava ili primjera podataka. U tipičnom problemu kojeg rješavamo strojnim učenjem imamo n uzoraka podataka s njihovim karakteristikama (atributima, engl. *features*) na temelju kojih se predviđaju svojstva nepoznatih uzoraka. Iz podataka se dolazi do određenih pravilnosti pomoću različitih metoda. Korištenje strojnog učenja postalo je dijelom svakodnevnog života, iako možda toga nismo svjesni. Koristi se prilikom poboljšanja kvalitete pretraživanja Interneta pomoću tražilica, sigurnosti u računarstvu (antivirusni programi i detekcija novih malicioznih programa), u ciljanom marketingu, preporukama koje se pojavljuju na društvenim mrežama i online sustavima zabave koji prema prijašnjim pretraživanjima, odgledanim filmovima i pročitanim knjigama nastoje prodati korisniku nove filmove i knjige. Osim za komercijalne svrhe, strojno učenje ima veliki utjecaj i na istraživanja u znanosti. Npr. ML metode koriste se za otkrivanje novih čestica u fizici, analizu molekula koje bi potencijalno mogle postati lijekovi za određene bolesti te pronalazak novih planeta. Primjena strojnog učenja na asteroide je tema ovog rada.

Strojno učenje se u svijetu podatkovnih znanosti (engl. *Data Science*)[4] predstavlja kao jednostavan mehanizam pomoću kojega je moguće približiti se rješenju problema. No kako bi se pravilno pristupilo problemu, potrebno je znati odabrati pravu metodu strojnog učenja, njezine jače i slabije strane te razumjeti što od nje očekujemo nakon što je upotrijebimo. Za odabir prave metode strojnog učenja, potrebno je dobro razumijevanje podataka s kojima raspolažemo kako bismo mogli izgraditi dobar model. Na strojno učenje možemo gledati kroz dvije sfere - kao na računalnu pozadinu analize podataka (algoritmi, metode, izračuni i sl.) te kao na statistički okvir čijim povezivanjem imamo mogućnost analizirati podatke. Radi se o interdisciplinarnom području istraživanja koje uključuje računalne znanosti, statistiku, umjetnu inteligenciju te mnoga druga područja.

Dijelimo strojno učenje na dva pristupa - nadzirano strojno učenje (engl. *Supervised machine learning*) i nenadzirano strojno učenje (engl. *Unsupervised machine learning*) [10]. Kod nadziranog strojnog učenja gradi se model podataka koji

[4] interdisciplinarno područje u čijem središtu se nalazi izvlačenje informacija iz podataka

predviđa oznake (engl. *Labels*) za nove podatke na temelju ulaznih, početnih podataka. Kod nenadziranog grade se modeli podataka bez poznavanja oznaka početnih podataka. Odnosno, kod nadziranog učenja podaci već imaju određene karakteristike koje želimo predvidjeti, dok kod nenadziranog otkrivamo strukturu unutar podataka bez definiranja cilja algoritma. Zamislimo da imamo bazu fotografija životinja. Prilikom nadziranog učenja konkretizirali bismo koja vrsta životinje se nalazi na svakoj slici te bismo onda, prilikom unosa novih slika u bazu, naučili model da prepoznaje koje životinje se nalaze na novim slikama. Na istom primjeru, nenadziranim učenjem ne bismo konkretizirali o kojim životinja se radi na slikama unutar baze, već bi algoritam sam prema sličnostima/razlikama morao grupirati slike životinja u iste ili različite skupine. Sukladno navedenome, kod nadziranog učenja postoji svojevrsni učitelj koji pruža nadgledanje rada algoritama i provjeru rezultata u usporedbi s već poznatim, dok kod nenadziranog učenja podaci govore sami za sebe i predviđaju moguća rješenja. Postoji mogućnost korištenja tzv. polunadziranog strojnog učenja (engl. *Semi-supervised learning*) koji je kombinacija dva osnovna pristupa strojnom učenju. U početnim podacima postoje i označeni i neoznačeni primjeri unutar podataka. Neoznačenim se primjerima nakon grupiranja dodjeljuju oznake susjednih već označenih primjera, tj. dobiju oznaku u ovisnosti o tome u kojoj su grupi [11]. Ovaj rad se bavi metodama nenadziranog strojnog učenja.

3.1 Algoritmi grupiranja

U slučajevima kada nemamo skup označenih uzoraka, tj. kada je potrebno odrediti zakonitosti koje se nalaze u podacima koristimo nenadzirano učenje. Jedan od pristupa nenadziranom učenju naziva se grupiranje (engl. *Clustering*). U takvim metodama podatke dijelimo u grupe koje nazivamo *klasterima*. Uzorci koji su slični (u ovisnosti o njihovim atributima) svrstavaju se u isti klaster. Algoritmi grupiranja nastoje naučiti optimalnu raspodjelu podataka. Uspješne implementacije algoritama grupiranja možemo pronaći u različitim područjima: klasifikacija biljaka i životinja u biologiji, pronalazak seizmološki opasnih područja s obzirom na prijašnje potrese, identificiranje grupa korisnika koji imaju slične obrasce ponašanja na Internetu u marketinške svrhe, ili izvlačenje najpopularnijih tema koje se spominju na društvenim mrežama za analizu trendova.

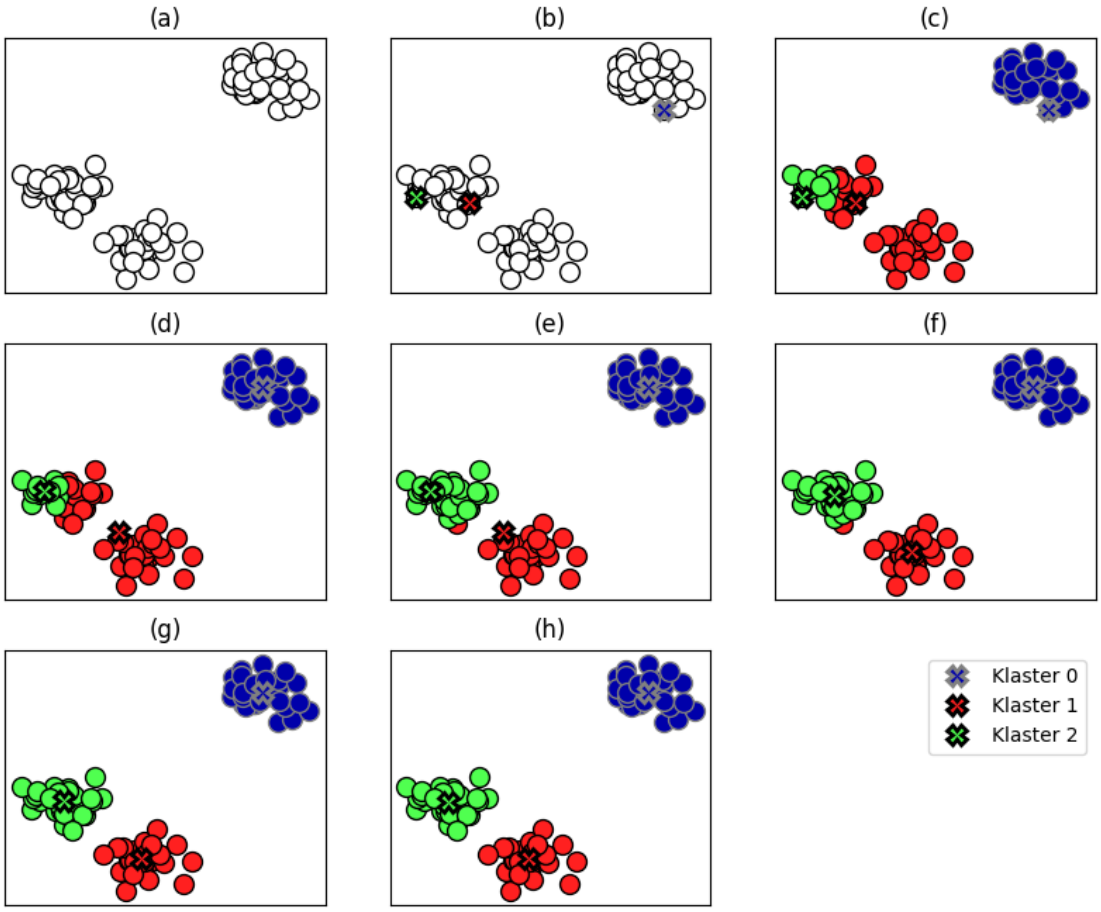
Kako koristimo algoritme grupiranja nad podacima koji nemaju oznake, prilikom evaluacije rezultata postoji poteškoća s određivanjem kvalitete izvršavanja algoritma i dobivenog modela. Najčešće nemamo mogućnost usporedbe s drugim podacima i sukladnu provjeru dobivenog. Zbog toga se algoritmi grupiranja smatraju najkorisnijim metodama kada se koriste u fazi proučavanja korelacija između podataka. U ovom radu koristimo algoritme grupiranja za klasifikaciju asteroida koji su potencijalno opasni za Zemlju. Podatke usporedjujemo s klasifikacijama asteroida koje definira NASA.

3.2 Algoritam k-srednjih vrijednosti (KMEANS)

Za NEO podatke smo prvo koristili algoritam *k*-srednjih vrijednosti (engl. k*-means algorithm*, KMEANS) koji je najpoznatiji algoritam grupiranja i jednostavno se implementira. Grupiranje algoritmom *k*-srednjih vrijednosti se koristi kada analiziramo neoznačene podatke, odnosno podatke koji nemaju definirane kategorije (grupe). Cilj algoritma je pronaći grupe unutar podataka, gdje je broj grupa reprezentiran varijablom *k* koja je unaprijed određena. Možemo reći da ovaj algoritam pronalazi centre grupa koji su predstavnici određenog dijela podataka. Algoritam iteracijom, u ovisnosti o karakteristikama podataka, pridružuje svaku točku unutar skupa podataka nekoj od *k* grupa. Metoda radi s *k* centara grupa gdje su centri određeni kao one lokacije koje minimiziraju udaljenost svake točke od centra grupe u kojoj se nalaze. Svaka točka (podatak, objekt) je bliže centru vlastite grupe nego centru susjedne.

S obzirom na to da koristi iterativan proces prilikom kojeg ne procjenjuje sve moguće redoslijede slaganja grupa nakon svake iteracije, radi se o relativno brzom algoritmu. Prvo se pretpostavlja *k* broj centara nakon čega ponavlja dva koraka dok rješenje ne konvergira. Prvi korak dodjeljuje točke najbližem centru - mijenjaju se očekivanja kojoj grupi pripada koja točka. Drugi korak postavlja centre grupa kao aritmetičke sredine točaka koje su u toj iteraciji dodijeljene toj grupi. Uzimajući navedeno u obzir, svako ponavljanje daje bolje rezultate od prijašnje iteracije s kojima dobivamo bolje određene karakteristike grupa no to nužno ne znači kako ćemo moći odrediti globalno najbolje rješenje. Izvršavanje algoritma završava kada se dodjeljivanje točaka grupama više ne mijenja.

Na Slici 3.1 (a) vidimo podatke nad kojim će se izvesti algoritam *k*-srednjih vri-



Slika 3.1: Rad algoritma k-srednjih vrijednosti, k=3 [12]

jednosti. Ti podaci su reprezentirani točkama tako da vidimo tri moguće grupe i odabiremo k=3 za inicijalizaciju KMEANS algoritma. Na Slici 3.1 (b) su nasumično odabrana tri centra grupa koji su reprezentirani zelenom, crvenom i plavom oznakom X. Prilikom svake iteracije ovi centri će se prilagođavati u odnosu na točke unutar njihovih klastera. Na Slici 3.1 (d) već možemo primijetiti da su se točke podijelile u tri grupe te je potrebno ponovno odrediti centre klastera zbog nove raspodjele točaka po klasterima. Slike 3.1 (e)-(g) prikazuju dodjeljivanje novih točaka određenim klasterima te sukladne prilagodbe centara klastera u ovisnosti o svim točkama određenog klastera. Slika 3.1 (h) prikazuje kraj izvršavanja algoritma gdje vidimo tri dobro određena klastera s odgovarajućim centrima. Za Sliku 3.1 korišten je KMEANS algoritam iz biblioteke *mglearn* koja sadrži metode pomoću kojih se može demonstrirati rad različitih algoritama [12].

Parametar koji određuje broj klastera potrebno je unijeti unaprijed. Postavlja se pitanje njegovog određivanja zbog toga što težimo definiranju broja grupa koje će

uključivati optimalnu složenost dobivenog modela. Metoda kojom bismo odredili točnu vrijednost k ne postoji no, ovisno o primjeni, moguća je procjena. Broj grupa se može odrediti npr. vizualizacijom podataka u 2D prostoru (po potrebi, treba reducirati podatke). Drugi način je grafički prikaz ovisnosti kriterijske funkcije o varijabli k. Kriterijska funkcija u slučaju KMEANS algoritma je prosječna udaljenost točaka unutar klastera od njegovog centra. Kako povećanje broja grupa (tj. povećanje k) smanjuje udaljenost između točaka, kod odgovarajuće vrijednosti parametra k promjena kriterijske funkcije vise neće biti značajna te se iz grafa može odrediti k. Kod nekih primjena varijabla k može biti unaprijed poznata.

Potencijalne mane KMEANS algoritma javljaju se prilikom grupa s kompliciranom raspodjelom granica (nelinearnim granicama), no tada je moguće koristiti druge algoritme strojnog učenja. Također, prilikom povećanja broja uzoraka može se usporiti rad algoritma. Poteškoću s grupiranjem podataka spomenutog tipa možemo vidjeti na Slici 3.2 koja prikazuje kako KMEANS (k=2) nije bio dobar odabir algoritma za prikazane podatke. Umjesto detekcije dvaju zasebnih oblih oblika, algoritam je linearno raspodijelio granicu podataka. Bolje rezultate bismo mogli dobiti odabirom veće vrijednosti k koja bi, prema karakteristikama podataka, razdijelila podatke na više grupa čijom bismo kasnijom evaluacijom i procjenom rješenja mogli rekonstruirati željena svojstva koja vidimo na Slici 3.2 desno.

KMEANS algoritam grupiranja kojeg koristimo u radu je implementiran kroz Pythonovu biblioteku *scikit-learn*. Ova biblioteka je detaljnije opisana u Poglavlju 3.4.

3.3 DBSCAN algoritam grupiranja

DBSCAN (engl. *Density-Based Spatial Clustering of Application with Noise*) algoritam grupiranja pronalazi točke koje se nalaze u područjima prostora (koji je povezan s karakteristikama uzoraka) u kojima su gusto raspoređene odnosno blizu jedna drugoj. Klasteri su gusta područja podataka između kojih se nalaze relativno prazna područja koja potencijalno sadržavaju šum (engl. *Noise*).

Potrebno je definirati dva parametra za rad DBSCAN algoritma [13]:

- *eps*, maksimalan radijus
- *min_samples*, minimalan broj uzoraka koji čine klaster.

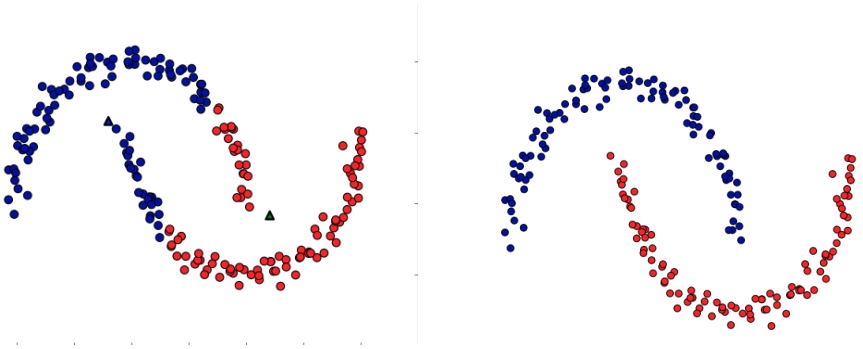
Kako bi točka bila definirana kao točka jezgre[5] (engl. *Core Sample*) mora postojati najmanje *min_samples* točaka unutar udaljenosti *eps* od te točke. Sve točke jezgre koje su međusobno bliže od udaljenosti *eps* pripadaju istom klasteru. Varijabla *eps* označava što u određenom slučaju znači blizina točaka, tj. kada su točke u klasteru blizu jedna drugoj - ako odaberemo preveliki broj, sve točke će biti dijelom jednog klastera, a ako odaberemo premali broj, može se dogoditi da u rezultatima nemamo točke jezgre.

Kod DBSCAN algoritma uočavamo kako nije potrebno unaprijed odrediti broj klastera (kao kod KMEANS algoritma), no potrebno je pažljivo odabrati varijable *eps* i *min_samples* koje implicitno određuju broj klastera koji ćemo imati nakon izvršavanja algoritma. U usporedbi s KMEANS algoritmom, DBSCAN je prikladniji za korištenje nad podacima s nelinearnom granicom i u sličajevima kad klasteri nisu proporcionalnih veličina. DBSCAN algoritam je sporiji od KMEANS algoritma, jer s većom količinom podataka zahtjeva više računalne memorije i procesorskih kapaciteta.

Ukratko, algoritam na početku izvršavanja odabere nasumičnu točku A unutar podataka te odredi sve točke koje su od točke A na udaljenosti *eps* ili manje. Ako je broj točaka koje je odredio manji od definirane vrijednosti varijable *min_samples*, točka A se označava kao točka šuma. Ako je broj točaka koje je algoritam odredio da se nalaze unutar *eps* od točke A veći od *min_samples*, točka A postaje točka jezgre i dobija oznaku klastera. Posjećuju se sve susjedne točke (unutar *eps*) te ako nekoj nije dodijeljen klaster, dobiva oznaku tog novodefiniranog klastera. Kada prilikom posjećivanja susjednih točaka algoritam dođe do druge točke jezgre, posjećuju se i njezine susjedne točke. Tako se veličina klastera povećava do situacije u kojoj više ne postoji susjednih točaka jezgre unutar *eps* od točke A. Nakon što se spomenuto dogodi, algoritam bira drugu nasumičnu točku koja nije bila posjećena unutar dotadašnjeg izvršavanja algoritma te proces kreće ispočetka.

Ovakav algoritam, za razliku od KMEANS, omogućava određivanje klastera kompliciranijih oblika te, s obzirom na koncept točaka šuma, određuje točke koje nemaju vlastiti klaster. Na Slici 3.2 vidimo da DBSCAN algoritam dobro određuje dva klastera iz podataka za koje KMEANS algoritam s $k=2$ nije pružao zadovoljavajuće rješenje [12]. Potrebno je napomenuti kako rezultati ovise o redoslijedu posjećivanja podataka jer granične točke (one na granici unutar *eps*) mogu pripadati (biti su-

[5] točka unutar guste regije podataka

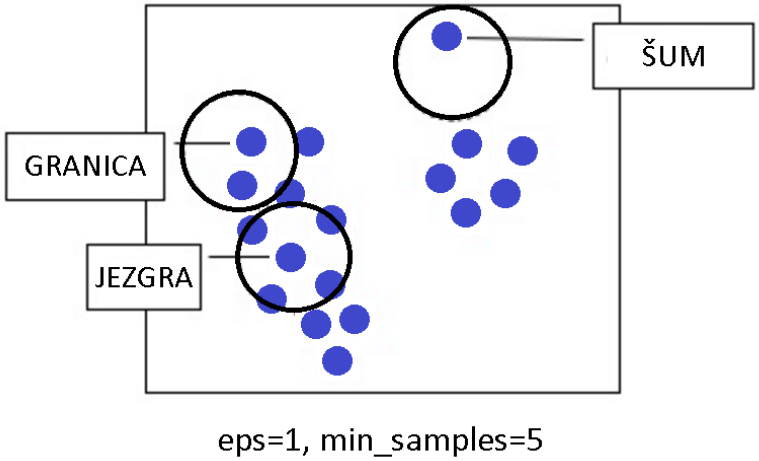


Slika 3.2: Usporedba rada KMEANS (lijevo) i DBSCAN (desno) algoritma [13]

sjedne) većem broju točaka jezgara. Na kraju izvršavanja DBSCAN algoritma imamo 3 tipa točaka:

- točke jezgre
- točke granice (engl. *Boundary point*)
- točke šuma.

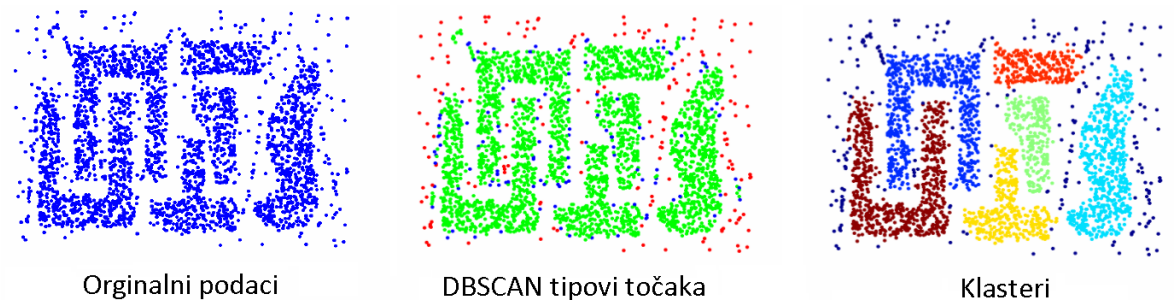
Na Slici 3.3 vidimo proces određivanja različitih tipova točaka pomoću DBSCAN algoritma. Slika 3.4 prikazuje originalne podatke, određivanje tipova točaka te krajnji rezultat izvršavanja algoritma za šest klastera s dobro određenim šumom. U središnjem dijelu slike (DBSCAN tipovi točaka), točke zelene boje su točke jezgre, plava boja predstavlja granične točke dok su crvenom bojom označene točke šuma.



Slika 3.3: Rezultantne točke DBSCAN algoritma [14]

Pravilan odabir parametara je od ključne važnosti. Povećanjem varijable *eps* povećavamo i broj točaka koje će biti uključene u određeni klaster. Ako koristimo

prevelik *eps*, može se dogoditi da podaci koji bi trebali biti u dva klastera, postanu dijelom samo jednog klastera. Kada povećavamo vrijednost varijable *min_samples*, manje točaka će biti točke jezgre, a više će ih biti točke šuma. Preporučljivo je podatke skalirati prije korištenja algoritma ovog tipa kako bi svi podaci imali sličan interval vrijednosti.



Slika 3.4: Rad DBSCAN algoritma [15]

3.4 Osnovne biblioteke i metode

U ovom dijelu teksta ukratko opisujemo informatičke pojmove koje smo koristili u radu: programski jezik *Python*, biblioteke *urllib*, *numpy*, *pandas*, *scikit-learn*, *matplotlib*, sustav *GitHub*, metode smanjenja dimenzija podataka, te *JSON* i *CSV* datoteke.

3.4.1 Python

Python [16] je interpreterski, interaktivni, objektno orijentirani programski jezik koji je naziv dobio po poznatoj britanskoj televizijskoj seriji *Monty Python's Flying Circus*. Naglasak je na jednostavnoj sintaksi te preglednosti koda i ujedinjavanju korisnih značajki drugih programskih jezika. Radi se o besplatnom programskom jeziku visoke razine, s velikom zajednicom korisnika, odličnom dokumentacijom i potporom.

Python ima mogućnost korištenja u različitim okruženjima - npr. kao skripta koja se pokreće kroz terminal, ili kao programski kod koji se zapisuje i izvršava interaktivno korak po korak u nekom od editora. Mogućnosti primjene su vrlo široke. Pyton se može koristiti kao programski jezik koji služi kao uvod u osnove programiranja te sve do razvoja Web aplikacija i programiranja grafičkih korisničkih sučelja (engl. *Graphical User Interface*, GUI). U ovom radu korištena je verzija Python 3.6.

3.4.2 Urllib

Pythonova urllib biblioteka sadrži definicije klasa i funkcija kojima možemo pristupati Internet stranicama kroz programski kod. Njezinim korištenjem možemo preuzimati podatke sa stranica, ili ih slati na njih, te raditi željene izmjene prilikom stvaranja zahtjeva prema stranici [18]. Korištenjem urllib biblioteke spajamo se na poslužitelje na kojima se nalaze sadržaji stranice. Većina servera mora imati zaštitu od prevelikog broja zahtjeva prema njoj kako bi mogla pravilno i neometano funkcionirati. Za otvaranje URL stranica korišten je modul *urllib.request.urlopen*. U Poglavlju 4 je detaljnije opisano preuzimanje podataka sa servera.

3.4.3 NumPy i Pandas biblioteke

Jedan od osnovnih paketa unutar Pythona koji sadrži metode linearne algebre i Fourierove transformacije je NumPy [21]. Pored ostalog, sadrži mogućnosti za upotrebu *n*-dimenzionalnih polja. S obzirom na to da smo koristili *SciKit-Learn* biblioteku i njezine implementacije algoritama strojnog učenja, bilo je potrebno spremiti podatke u 2D polje koje algoritmi mogu koristiti za daljnju analizu. Podaci koje algoritam koristi nalaze se u 2D polju oblika *broj_uzoraka, broj_karakteristika*.

Pandas smo koristili kako bismo mogli urediti i analizirati podatke pomoću osnovne strukture ove biblioteke koja se zove *DataFrame*. Ova struktura je slična datoteci Microsoft Excela i drugih tabličnih programa. Pandas biblioteka omogućava predstavljanje podataka kao tablica gdje redovi označavaju broj uzoraka (engl. *Samples*), a stupci podatke o uzorcima tj. njihove atribute [22]. Moguće je, pomoću različitih metoda iz Pandas biblioteke, vršiti željene operacije nad tablicom. Stupci unutar dataframe-a mogu sadržavati različite tipove podataka što nije moguće kod NumPy polja te je upravo zato i korišten u ovom radu zbog toga što radimo s različitim atributima asteroida. Na primjer atribut iz NEO baze *is_potentially_hazardous_asteroid* je logički (engl. *Boolean*) tip vrijednosti, dok je *absolute_magnitude_h* broj.

3.4.4 SciKit-Learn

Scikit-Learn je Pythonov paket otvorenog koda koji sadrži velik izbor algoritama strojnog učenja (klasifikacija, regresija, grupiranje itd.) s opsežnom dokumentacijom i podrškom prilikom korištenja [20]. Prvo smo primijenili KMEANS algoritam

pomoću instanciranja klase *sklearn.cluster.KMeans* i određivanja broja klastera. Za vrijeme izvršavanja algoritma svaka točka u podacima dodijeli se jednom od klastera i dobije njegovu oznaku. Ako npr. imamo 4 klastera, njihove brojčane oznake su brojevi od 0-3. Oznake je moguće pronaći u *labels_* atributu. Oznake koje dobivamo ovom metodom nemaju konkretno značenje u smislu da (ako se sjetimo primjera sa životinjama na početku ovog poglavlja) možemo znati o kojoj životinji se radi na određenoj slici, već nam samo govore da neke slike imaju slična svojstva i na temelju toga su grupirane u isti klaster.

Koristimo metodu *predict( )* ako imamo nove točke koje želimo dodijeliti nekim klasterima. Na taj način svaku od novih točaka dodijelimo onom klasteru čijem centru je ta točka najbliža. Ovim postupkom ne mijenjamo postojeći model. Centre klastera možemo pronaći u atributu *cluster_centers_*.

Osim KMEANS korišten je i DBSCAN algoritam grupiranja kojeg smo instancirali pomoću klase *sklearn.cluster.DBSCAN*. Algoritam daje dobre rezultate za podatke čiji su klasteri sličnih gustoća. Nakon pozivanja *fit( )* funkcije koja izvršava grupiranje, unutar atributa *labels_* mogu se pronaći oznake klastera. Točke šuma imaju vrijednost -1 unutar ovog atributa, dok točke unutar klastera poprimaju vrijednosti od 0 do (broj_klastera-1). Implementacija algoritama KMEANS i DBSCAN je opisana u Poglavljima 4.3 te 4.4.

3.4.5 Matplotlib

Za grafički prikaz podataka korištena je biblioteka *Matplotlib* [25]. Ova biblioteka omogućava vizualizaciju podataka i puno se koristi u Pythonu. Vizualizacija podataka je važan dio strojnog učenja. Pravilno određivanje oblika grafova i osnovnih osi te mijenjanje pristupa evaluacije podataka prilikom grafičkog prikaza rezultata može pružiti različite uvide u dobivena rješenja i potencijalno omogućiti bolju analizu.

3.4.6 GitHub

GitHub je online platforma na kojoj je moguće udomljavati projekte koji se temelje na Gitu. *Git* je vrsta kontrole verzije sistema (engl. *Version control system*, VCS) otvorenog koda koja omogućava spremanje te praćenje promjena koje se događaju nad datotekama i mapama koje su pod njezinim nadzorom [17]. Naglasak je na sigurnosti

projekata i računalnog koda. U slučaju gubitaka nekih podataka, grešaka prilikom programiranja i slično, Git omogućava povratak projekta u prijašnje stanje. Osim toga, kada postoji potreba za suradnju između više programera na istom projektu, ovakve platforme omogućavaju transparentne izmjene koda s vidljivim autorima.

Git posjeduje određene specifičnosti zbog kojih se često koristi. Za razliku od većine drugih VCS-ova, Git ne sprema datoteke nakon svake izmjene već pamti samo promjene koje su se dogodile u odnosu na prijašnju verziju spremljene datoteke te se poziva na nju. Većinu postupaka moguće je raditi lokalno na računalu tako da je funkcionalan i kada pristup vanjskoj mreži nije moguć. Pojedinosti podešavanja Git-a neće biti spomenute u ovom radu, ali napominjemo kako postoji mnogo uputa koje se mogu pronaći na Internetu. Git se može jednostavno podesiti i po potrebi povezati s GitHub-om.

3.4.7 Smanjenje dimenzija podataka

Osim algoritama grupiranja koristili smo još jednu metodu strojnog učenja u svrhu lakše vizualizacije rezultata - smanjenje broja dimenzija (engl. *Dimension reduction*). Kako bi podaci bili lakši za interpretaciju od strane drugih algoritama i čovjeka koristimo metode smanjenja dimenzija. Ove metode pronalaze nove načine prikazivanja višedimenzionalnih podataka koji sadrže veliki broj različitih atributa. Prilikom procesa smanjivanja broja dimenzija početnih podataka nastoji se zadržati važna svojstva koja su u njima sadržana, ali koristeći manji broj atributa.

Jedan od algoritama ovog tipa je *PCA* (engl. *Principal component analysis*) kojeg smo koristili kako bismo prije korištenja algoritama grupiranja reducirali dimenzije početnih podataka na dvije dimenzije. Time smo omogućili brže i efikasnije izvršavanje modela strojnog učenja na bazi NEA asteroida. PCA transformira podatke i uklanja komponente s niskom varijancom. U 2D skupovima podataka pronalazi prvu glavnu komponentu (engl. Principal component) koja sadrži maksimalnu moguću varijancu te drugu glavnu komponentu koja je ortogonalna na smjer prve. Svaka glavna komponenta predstavlja linearnu kombinaciju originalnih podataka. Pravac glavne komponente je takav da zadržava najviše dostupnih informacija. Upotreba ovog algoritma je preporučena prilikom korištenja višedimenzionalnih podataka jer, osim što olakšava kasniju vizualizaciju rezultata, poboljšava razumijevanje odnosa između podataka.

Općenito, pri reduciranju dimenzija želimo sačuvati što veću količinu informacija unutar podataka [23]. Osim PCA, kao metodu smanjenja dimenzija podataka koristili smo i t-SNE (engl. t-distributed Stochastic Neighbor Embedding [24]) algoritam. Ova metoda omogućava bolju vizualizaciju podataka zbog toga što ne radi transformaciju podataka (kao PCA) već gradi model novog oblika prikaza podataka. Započinje nasumičnim prikazom svih točaka koje u daljnjem procesu u slučaju bliskih točaka dodatno približava, dok one koje su udaljenije dodatno širi unutar prostora prikaza podataka. Za razliku od PCA, radi se o računalno zahtjevnom algoritmu čije izvršavanje može trajati značajno dulje.

3.4.8 JSON

JSON (engl. *JavaScript Object Notation*) je notacija koja omogućava komunikaciju između preglednika i poslužitelja. Kako podaci koji se šalju prilikom takve komunikacije mogu biti isključivo tekstualnog oblika, JSON omogućava odgovarajuću sintaksu kojom se podaci u JavaScript objektima mogu izmjenjivati bez komplikacija. JavaScript objekti se mogu pretvarati u JSON oblik, a moguća je i transformacija u suprotnom smjeru. U nastavku slijedi primjer JSON sintakse preuzete s NeoWs API-a.

```
{
  "near_earth_object_count": 16540,
  "close_approach_count": 450062,
  "last_updated": "2017-06-15",
  "source": "All the NEO data is from NASA JPL NEO team.",
  "nasa_jpl_url": "http://neo.jpl.nasa.gov/"
}
```

Kao što možemo vidjeti, JSON objekti zapisani su u obliku ključa i vrijednosti povezane s tim ključem te su okruženi vitičastim zagradama. Na dan 15. 6. 2017. godine u NEO bazi nalazilo se 16 540 objekata. Za usporedbu, prilikom početka pisanja ovog rada u bazi se nalazilo 15 965 objekata što ukazuje na svakodnevno otkrivanje većeg broja novih objekata u blizini Zemlje i njihovog praćenja.

3.4.9 CSV

U *CSV* (engl. *Comma-Separated Values*) dokumentu spremaju se tablični podaci u tekstualnom obliku pri čemu su podaci (kao što i naziv kaže) najčešće razdvojeni zarezima. Moguće je definirati i druge vrste razdvajanja podataka. Podatake preuzete iz NASA-ine baze smo spremali u CSV datoteku. Atributi korištene CSV datoteke su opisani u Poglavlju 4.1.

4 Rezultati i analiza

Kao temelj za ovaj rad korišten je projekt *Avenge the dinosaurs* (ATD) [26] napisan za natjecanje (hackathon) Space Apps Challenge koje je 2016. godine organizirala NASA. Autor izvornog ATD projekta je Andrei Paleyes zaposlen kao programer u Amazonu. Ideja je bila preuzeti kod uz suglasnost autora (koja zbog otvorenog pristupa GitHuba nije bila neophodna), te ga izmijeniti i nadopuniti novim metodama koristeći ažurirane podatke o asteroidima koji su u međuvremenu sakupljenii u NeoWs (engl. *Near Earth Object Web Service*) bazi. U vrijeme hackathona u NeoWS bazi nalazilo se oko 14 tisuća objekata. U ovom radu promatrali smo skoro 17 tisuća objekata. Osim većeg broja asteroida, povećala se i točnost nekih njihovih karakteristika koje su već bile prisutne u bazi. Bolji opis asteroida je rezultat novih promatranja, mjerenja i izračuna varijabli.

Kako se radi o projektu napravljenom za hackathon koji najčešće zahtijeva od autora realizaciju zadatka u vrlo kratkom vremenu, nema mnogo dokumentacije za ATD. Analizom koda projekta s ciljem njegove izmjene i nadogradnje stvorili smo odgovarajuću pozadinu za analizu asteroida. Željeli smo provjeriti mogu li algoritmi grupiranja pokazati postojanje krivo opisanih asteroida kategoriziranih kao ne-PHA ili PHA. Odnosno, provjeravamo postoje li unutar NASA-ine baze asteroidi koji su kategorizirani kao PHA, no ustvari bi trebali biti ne-PHA. Pored toga želimo provjeriti postoje li asteroidi koji su kategorizirani kao ne-PHA a mogli bi potencijalno biti opasni za Zemlju. Pitamo se možemo li iz podataka NeoWs baze, nakon korištenja algoritama grupiranja, iz rezultata izvući značajne poveznice.

Osim KMEANS algoritma grupiranja i PCA algoritma za smanjenje dimenzija podataka koji su upotrijebljeni u ATD projektu, koristili smo DBSCAN algoritam gdje smo podatke reducirali i pomoću t-SNE algoritma. Kod smo preuzeli putem GitHub-a korištenjem opcije fork. Fork radi kopiju repozitorija kojeg preuzimamo te ga povezuje s našim korisničkim računom. Tada je moguće raditi izmjene na projektu bez utjecaja na originalni projekt. Najčešće se koristi u svrhu predlaganja izmjena koda autoru originalnog projekta te zbog razloga zbog kojeg smo ga mi koristili, kao polazišnu točku novog projekta ili razvoja originalnog koda. Preuzimanjem ATD projekta stvorili smo verziju koda lokalno na računalu s Ubuntu OS-om korištenjem Git-a. Kopiju repozitorija stvaramo na računalu pomoću naredbe *git clone urlrepozito-*

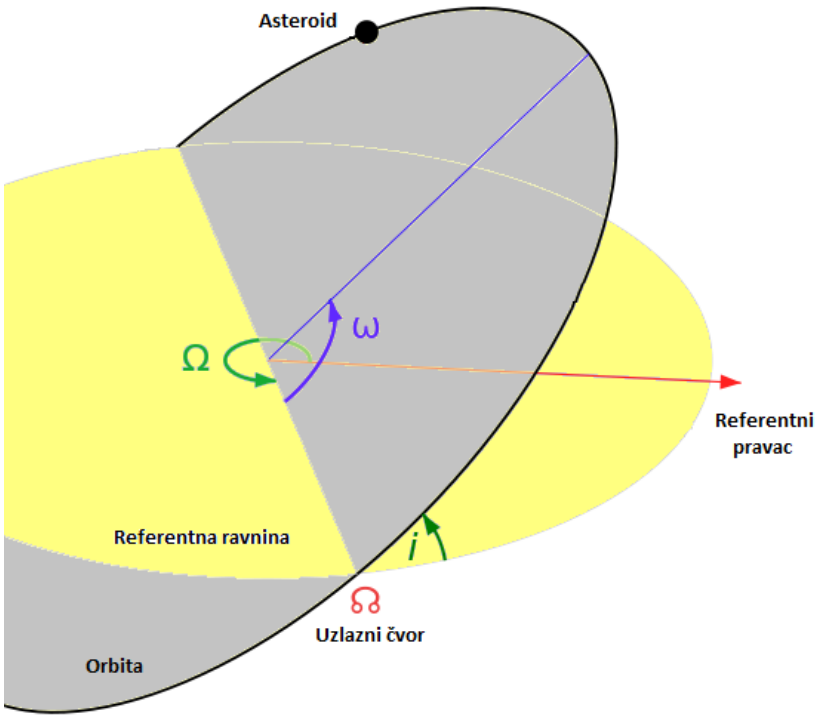
rija u terminalu. Programiranje vezano uz ovaj rad je uglavnom bilo rađeno lokalno na računalu, bez čestih sinkronizacija s GitHub-om no konačne verzije koda smo postavili na GitHub [27].

4.1 Preuzimanje podataka o asteroidima

Web API (engl. *Application Program Interfaces*) je aplikacijsko programsko sučelje koje služi za preuzimanje podataka sa stranica poslužitelja. Njegova upotreba je prikladnija od preuzimanja ili spajanja na samu statičku bazu jer omogućava definiranje zahtjeva (razmjenu poruka) prema i od servera. Postoje javno definirane krajnje točke (engl. *Endpoints*) kojima klijenti mogu pristupiti. Web API korišten u ovom radu naziva se Near Earth Object Web Service (NeoWs). NeoWs je Internet servis pomoću kojeg je moguće pretraživati asteroide koji su upisani u bazu asteroida u blizini Zemlje prema njihovim karakteristikama. Osim podataka o asteroidima tamo možemo pronaći i podatke o kometima. Zbog njihove sličnosti te činjenice da NeoWs ne definira jasnu razliku između kometa i asteroida ne očekujemo veliki utjecaj na rezultate grupiranja. Koristimo sve dostupne podatke u bazi, ali ćemo po potrebi izdvojiti komete.

Korišteni podaci su javno dostupni te se za većinu osnovnih zahtjeva može, kao kod većine API-a, koristiti neka vrsta osnovne autorizacije zahtjeva. U ovom slučaju radi se o univerzalnoj autentifikaciji s api ključem (DEMOKEY). No, s obzirom kako smo preuzimali sve podatake o svim objektima unutar baze takva razina autentifikacije ključa nije bila dovoljna zbog ograničenja o 30 zahtjeva prema API-u za određenu IP adresu unutar sat vremena, odnosno 50 zahtjeva unutar jednog dana. Nakon spomenutog broja zahtjeva API blokira sve naredne zahtjeve s IP adrese tog korisnika na određeni period vremena (sat ili dan). Zbog toga ne bismo mogli uspješno preuzeti podatke o asteroidima. Kako bismo sve potrebne podatke koje smo kasnije koristili dohvatili bez ograničenja danih DEMOKEY autentifikacijom, bilo je potrebno zatražiti API ključ pomoću kojeg je moguće predati dovoljan broj zahtjeva za podacima u jednom satu. Ubrzo nakon registracije za *NASA developer key*, dostavljen je ključ kojim više nismo bili ograničeni prilikom pristupa podacima. U svrhu zaštite podataka, u kodu unutar rada umjesto pravog ključa koristimo DEMOKEY sa stranice. Karakteristike odnosno atribute podatka koje smo preuzimali o svakom asteroidu možemo

vidjeti u Tablici 4.1 zajedno s njihovim kratkim opisom.



Slika 4.1: Vizualni prikaz kuteva Ω i ω opisanih u Tablici 4.1 [28]

Originalni kod je pisan u Pythonu 2 te je prvi zadatak bio uspješno prevođenje u Python 3. Korištenu biblioteku urllib2 u originalnom projektu više nismo bili u mogućnosti koristiti zbog činjenice da je razdijeljena u nekoliko modula u Pythonu 3. Kao zamjenu, koristili smo *urllib.request* koji sadrži definirane klase i funkcije kojima se može pristupati URL-u stranicama pomoću funkcije urlopen. Promjenom metode otvaranja stranica API-a pojavila se poteškoća s prepoznavanjem enkodiranja znakova podataka (engl. *character encoding*) prilikom povlačenja podataka u json obliku. Kako bismo to riješili konkretizirali smo standard za prikaz znakova korištenjem Unikoda (engl. *Unicode*) kojim dekodiramo odgovor kojeg dobivamo od strane API-a kako bismo podatke mogli koristiti.

Kako se radi o github kodu kojeg autor može mijenjati s vremena na vrijeme u ovisnosti o parametrima koje testira, te kako NeoWs API podliježe stalnim izmjenama, prvo pokretanje preuzetog koda je vodilo do problema. Koristila su se polja koja nisu definirana, priložena CSV datoteka na kojoj su se izvršavali algoritmi grupiranja za vrijeme hackathona nije imala karakteristike one datoteke koja bi trebala nastati iz priloženog koda, dobiveni grafovi na natjecanju su se razlikovali od našeg pokretanja istog koda na istoj CSV datoteci. Sukladno navedenome, bilo je

Varijabla	Opis parametra
name	Oznaka asteroida koja označava godinu i vrijeme kada je asteroid otkriven. Mali broj asteroida ima i dodatan naziv.
neo_reference_id	Službeni broj asteroida koji označava redni broj njegovog otkrića
is_potentially_hazardous_asteroid	*True* iil *False* vrijednost koja govori da li je asteroid klasificiran kao potencijalno opasan.
absolute_magnitude_h	H, apsolutna magnituda objekta. Više u Poglavlju 2.
estimated_diameter	Govori o minimalnim i maksimalnim vrijednostima procijenjenog promjera asteroida. Iskoristili smo ih kako bismo izračunali prosječan promjer svakog asteroida, *average_diameter*.
orbit_id	Oznake orbite asteroida
minimum_orbit_intersection	*MOID*, predstavlja minimalnu udaljenost između orbita dva objekta
jupiter_tisserand_invariant	T_J, Tisserandov parametar u odnosu na Jupiter. Više u Poglavlju 2.
epoch_osculation	Vrijeme promatranja
eccentricity	e, ekscentricitet orbite. Više u Poglavlju 2.
semi_major_axis	a, velika poluos orbite. Više u Poglavlju 2.
inclination	i, inklinacija orbite. Više u Poglavlju 2.
ascending_node_longitude	Ω, orbitalni parametar koji predstavlja kut između referentnog pravca (normala na referentnu ravninu) i pravca uzlaznog čvora. Prikaz se nalazi na Slici 4.1
orbital_period	T, period orbite, vrijeme potrebno tijelu da prođe punu orbitu oko drugog tijela.
perihelion_distance	q, udaljenost perihela. Više u Poglavlju 2.
perihelion_argument	ω, kut između pravca ulaznog čvora prema perihelu asteroida. Reprezentacija kuta nalazi se na Slici 4.1
aphelion_distance	Q, udaljenost afela. Više u Poglavlju 2.
perihelion_time	t_p, vrijeme prolaska tijela kroz perihel
mean_anomaly	M, kut kojeg bi tijelo s istim periodom orbite zatvarao s perihelom/afelom kada bi se gibalo kružnom orbitom sa stalnom brzinom
mean_motion	n, kutna brzina potrebna tijelu kako bi prošlo punu orbitu

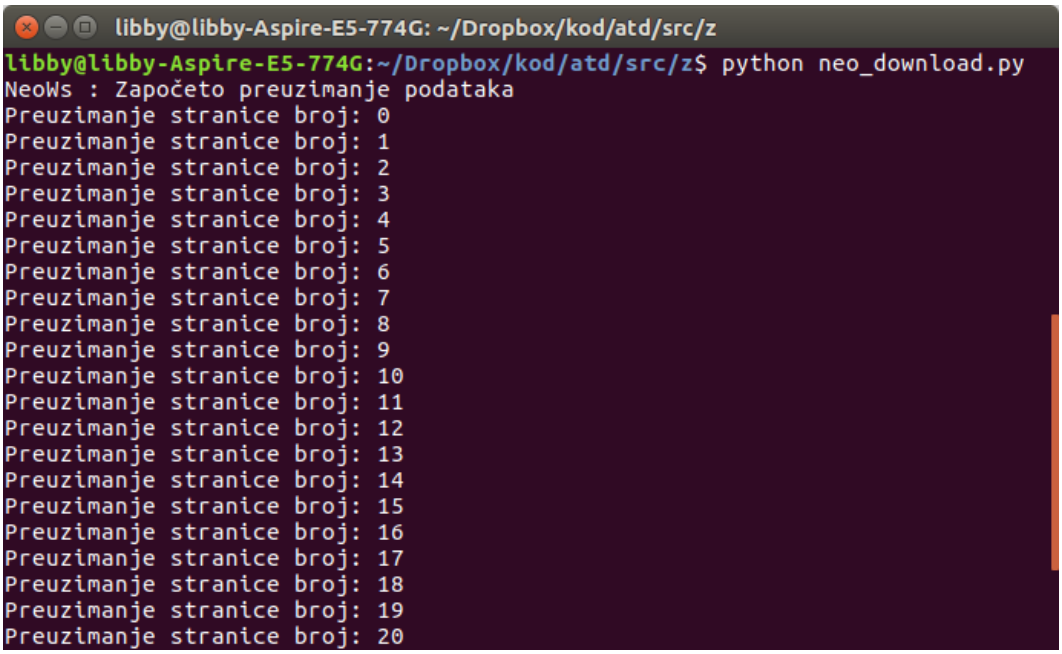
Tablica 4.1: Prikaz parametara korištenih prilikom grupiranja podataka

potrebno podesiti princip preuzimanja podataka novim uvjetima kako bismo dobili odgovarajuću CSV datoteku s ispravno unesenim podacima. Za preuzimanje podataka napisana je funkcija *load_page_data* koja koristi definirani format NeoWs API-a. Podaci su podijeljeni prema stranicama na kojima se nalaze. Funkcija kao argument uzima samo broj stranice. Dodatna otežavajuća okolnost je bila činjenica da su u NeoWs bazu u međuvremenu upisali nekoliko asteroida koji nisu imali sve atribute. Uz pažljivo proučavanje mjesta gdje je dolazilo do prekida preuzimanja podataka prilikom pokretanja koda, zaključili smo kako se radi o dva asteroida koja u json obliku nisu imali ključeve *absolute_magnitude_h* te *estimated_diameter_min* i *estimated_diameter_max*. Konkretno, u pitanju su sljedeći asteroidi [29]:

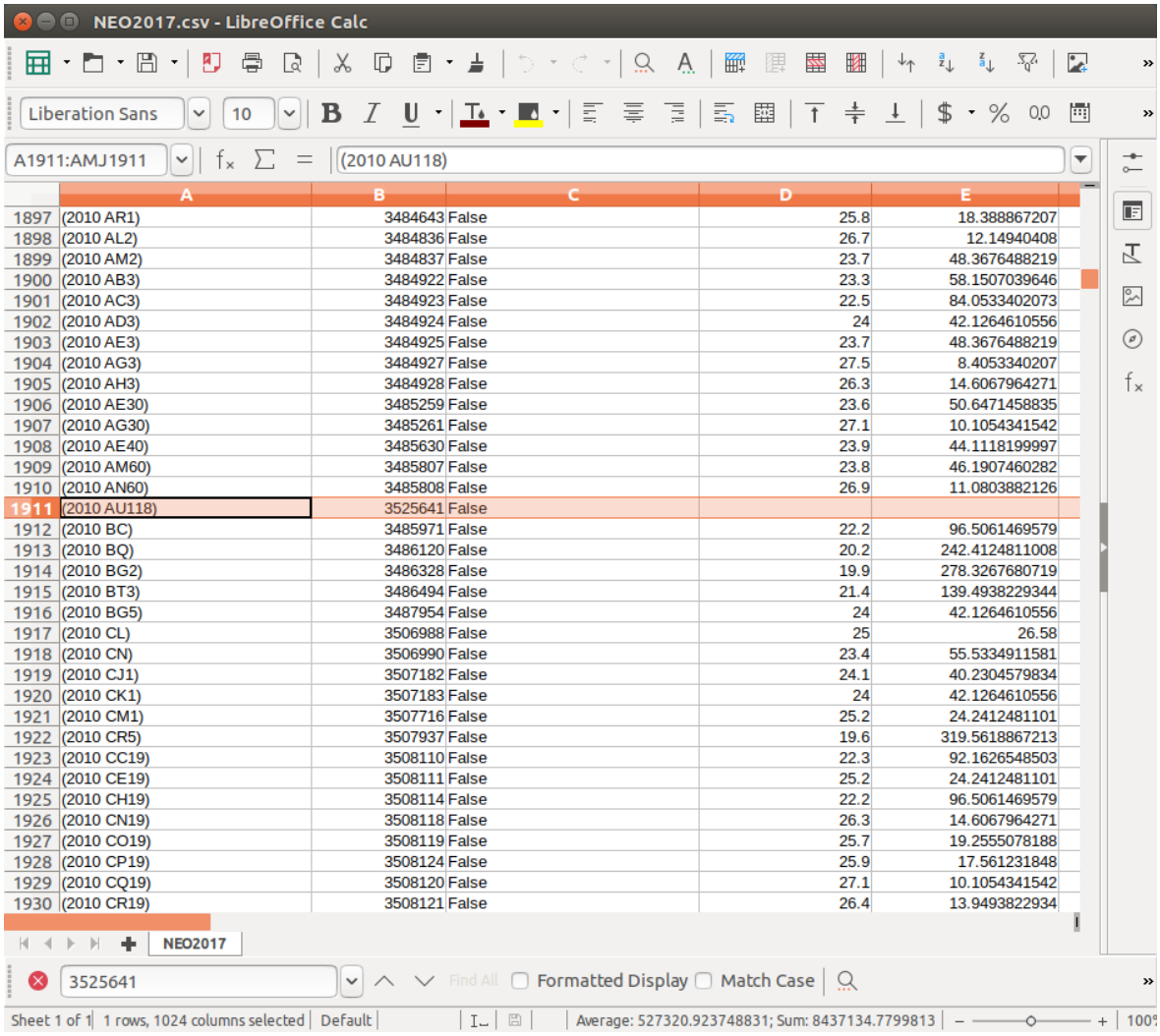
- 2010 AU11, neo_reference_id=3525641
- 2010 GZ60, neo_reference_id=3593312

Zbog spomenutih praznih ključeva prilikom preuzimanja podataka bilo je potrebno redefinirati način pristupa rječniku (engl. *dictionary*, dict). Dotadašnje korištenje dict[ključ] sintakse kako bismo pristupili argumentima asteroida je zbog nepostojećih ključeva vraćala *KeyError* pogrešku. Python podiže KeyError u slučajevima kada se koristi dict objekt koji u sebi ne sadrži ključ. Korištenjem *get()* funkcije nad rječnikom dobivamo mogućnost definiranja uobičajene vrijednosti (koja je bez konkretizacije *None*) koju rječnik može uspješno vratiti bez dizanja pogreške kao u slučaju kada smo zasebno pristupali dijelovima rječnika. Na ovaj način, asteroidi bez svih napomenutih atributa koje nastojimo preuzeti i dalje dobivaju stupac u CSV datoteci s tim atributima no s praznim vrijednostima ćelija. Osim toga, i dalje je bilo moguće pristupanje ugniježđenim json podacima uzastopnim korištenjem *get( )* funkcije.

U poglavlju 4.2 opisujemo rješenje čišćenja podataka s praznim vrijednostima atributa kako bismo nad podacima mogli izvršiti željene algoritme. Na Slici 4.2 možemo vidjeti izvršavanje konačnog koda za preuzimanje podataka s NeoWs servisa. Nakon završetka izvršavanja imamo CSV datoteku koja sadrži željene atribute unutar stupaca o svakom preuzetom objektu. Objekti su navedeni u redovima. Ekransku sliku malog dijela datoteke možemo vidjeti na Slici 4.3 na kojoj su vidljive i prazne ćelije.



Slika 4.2: Preuzimanje podataka s NeoWs



	A	B	C	D	E
1897	(2010 AR1)	3484643	False	25.8	18.388867207
1898	(2010 AL2)	3484836	False	26.7	12.14940408
1899	(2010 AM2)	3484837	False	23.7	48.3676488219
1900	(2010 AB3)	3484922	False	23.3	58.1507039646
1901	(2010 AC3)	3484923	False	22.5	84.0533402073
1902	(2010 AD3)	3484924	False	24	42.1264610556
1903	(2010 AE3)	3484925	False	23.7	48.3676488219
1904	(2010 AG3)	3484927	False	27.5	8.4053340207
1905	(2010 AH3)	3484928	False	26.3	14.6067964271
1906	(2010 AE30)	3485259	False	23.6	50.6471458835
1907	(2010 AG30)	3485261	False	27.1	10.1054341542
1908	(2010 AE40)	3485630	False	23.9	44.1118199997
1909	(2010 AM60)	3485807	False	23.8	46.1907460282
1910	(2010 AN60)	3485808	False	26.9	11.0803882126
1911	(2010 AU118)	3525641	False		
1912	(2010 BC)	3485971	False	22.2	96.5061469579
1913	(2010 BQ)	3486120	False	20.2	242.4124811008
1914	(2010 BG2)	3486328	False	19.9	278.3267680719
1915	(2010 BT3)	3486494	False	21.4	139.4938229344
1916	(2010 BG5)	3487954	False	24	42.1264610556
1917	(2010 CL)	3506988	False	25	26.58
1918	(2010 CN)	3506990	False	23.4	55.5334911581
1919	(2010 CJ1)	3507182	False	24.1	40.2304579834
1920	(2010 CK1)	3507183	False	24	42.1264610556
1921	(2010 CM1)	3507716	False	25.2	24.2412481101
1922	(2010 CR5)	3507937	False	19.6	319.5618867213
1923	(2010 CC19)	3508110	False	22.3	92.1626548503
1924	(2010 CE19)	3508111	False	25.2	24.2412481101
1925	(2010 CH19)	3508114	False	22.2	96.5061469579
1926	(2010 CN19)	3508118	False	26.3	14.6067964271
1927	(2010 CO19)	3508119	False	25.7	19.2555078188
1928	(2010 CP19)	3508124	False	25.9	17.561231848
1929	(2010 CQ19)	3508120	False	27.1	10.1054341542
1930	(2010 CR19)	3508121	False	26.4	13.9493822934

Slika 4.3: Prikaz sadržaja CSV datoteke

4.2 Priprema podataka

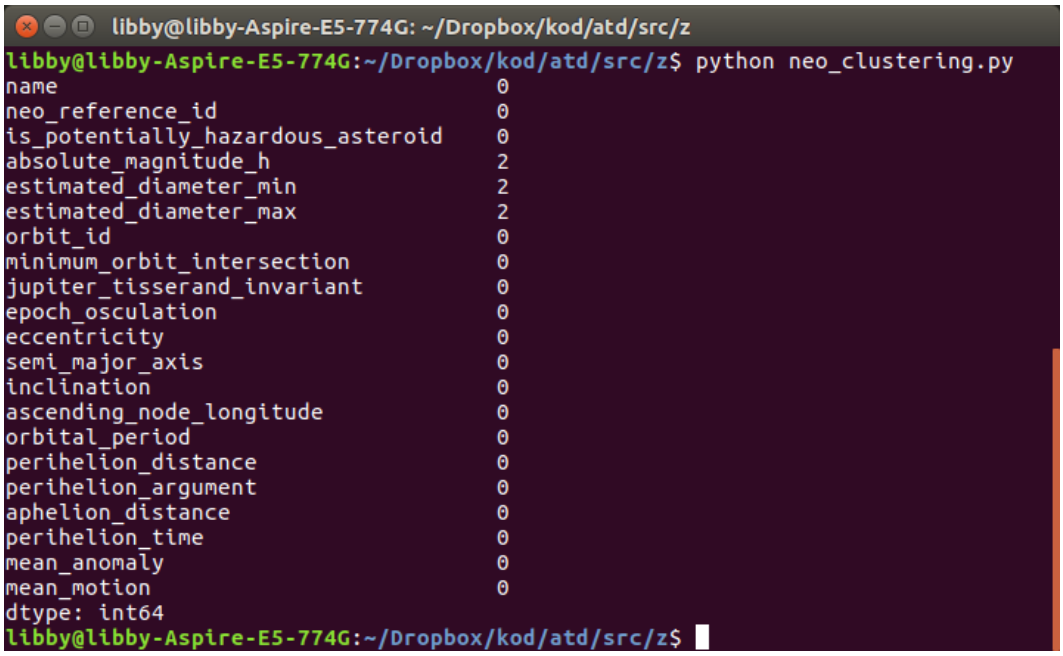
Podatke koje ćemo obrađivati je prije upotrebe metoda strojnog učenja potrebno detaljno analizirati. Tako se dolazi do odgovarajućeg načina pripreme podatka, odnosno do čišćenja nepotrebnih informacija te prilagodbe njihovog oblika kako bi bili prikladniji za primjenu algoritama. Ako se nad podacima ne izvrši dovoljno dobro čišćenje, krajnji rezultat može voditi do neispravnih zaključaka. Kvalitetna priprema podataka npr. uključuje odbacivanje atributa koji nedostaju, ili njihovo punjenje s adekvatnim zamjenama, te isključivanje vrijednosti koje su za zadani problem nemoguće.

CSV datoteka s podacima sadrži prazna polja kod dva spomenuta asteroida no kako želimo upotrijebiti sve dostupne informacije o svakom asteroidu s ciljem izbjegavanja odstupanja unutar rezultata, uklonili smo ih iz tablice. U program smo učitali CSV datoteku dobivenu preuzimanjem podataka, tj. učitali smo je u *pandas.DataFrame* koji je od podataka formirao numpy polje u dvije dimenzije. To polje se sastoji od redaka i stupaca s oznakama.

Funkcija *pandas.DataFrame.dropna* briše oznaku uz koju u ovom slučaju postoje neka prazna mjesta. Spomenuta mjesta su u dataframe-u reprezentirana *NaN* vrijednostima (Slika 4.4). *NaN* (engl. *Not A Number*) reprezentira nedefiniranu vrijednost koja se ne može prikazati. Prolaskom kroz cijeli dataframe, funkcija je pronašla redove koji sadrže *NaN* vrijednosti koje je zatim obrisala u cijelosti. S obzirom kako svaki redak ima svoj broj, na ovaj način nastala je praznina u dataframe-u koja kasnije može stvoriti poteškoće prilikom povezivanja podataka. Na primjer, ako je izbačeni redak čiji je indeks 2, samo indeksiranje će nakon *dropna* umjesto 0,1,2,3,... davati 0,1,3,... itd.

Željeli smo formirati konačnu CSV datoteku koja sadrži samo podatke koje koristimo. Zbog toga smo dobiveni dataframe (bez određenih redaka) ispisali u novu CSV datoteku koju smo kasnije ponovno učitali u program te u novi dataframe kojeg koristimo kao početni uvjet za grupiranje podataka. Još jedno moguće rješenje problema s indeksima je ponovno namještanje indeksiranja dataframe-a pomoću funkcije *pandas.DataFrame.reset_index*.

Podaci su općenito jako osjetljivi na skaliranje, no prije korištenja algoritama korisno je podesiti vrijednosti podataka unutar nekog intervala. Tako transformirani podaci zadržavaju oblik i karakteristike originalnih podataka no u drugačijem, priklad-



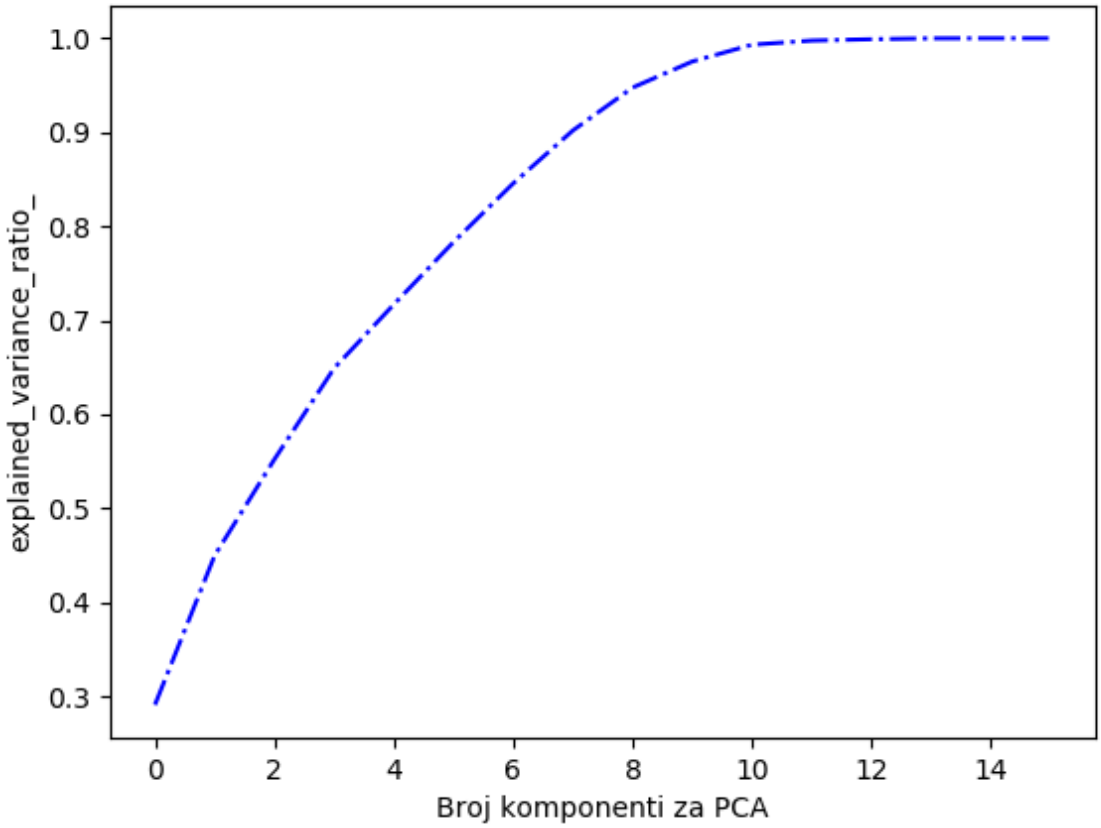
Slika 4.4: Prikaz broja *NaN* vrijednosti po atributu asteroida

nijem mjerilu za obradu i vizualizaciju. Korištena je funkcija *sklearn.preprocessing.scale* koja normira podatke uz određenu os, odnosno odabrali smo normiranje svakog atributa asteroida zasebno tako da prosječna vrijednost podataka bude oko nule. Kako bi mogli koristiti algoritme grupiranja važno je standardizirati podatke. Isto tako, kako ne bi dolazilo do potencijalno krivih transformacija podataka, različiti atributi moraju međusobno imati slične skale vrijednosti.

Osim čišćenja i skaliranja podataka, koristili smo i različite pristupe prilikom smanjivanja dimenzija podataka. Složenost algoritama ovisi o broju dimenzija podataka (broju atributa) i broju uzoraka nad kojima se izvršava. Kako bismo smanjili vrijeme izračunavanja i potrošnju memorije, smanjili smo broj dimenzija ulaznih podataka. Dobivanje informacija iz podataka poboljšano je kada su oni objašnjeni s manje atributa. Stvara se okruženje za razumijevanje do tad nepoznatih poveznica unutar podataka. Ako se redukcija podataka ispravno oblikuje, ne dolazi do velikih gubitaka informacija o pojedinim uzorcima. Osim navedenog, grafička reprezentacija podataka postaje olakšana čime strukture, poveznice i vanjske točke unutar podataka postaju naglašenije. Redukciju podataka smo prvo izvršili pomoću PCA metode pozivanjem *sklearn.decomposition.PCA* klase [30].

```
1 pca_data=PCA(n_components=2).fit_transform(data)
```

U gore navedenoj liniji koda smo primijenili PCA metodu na podatke te ih transformirali u oblik s dvije dimenzije. Pomoću varijable *explained_variance_ratio_* možemo



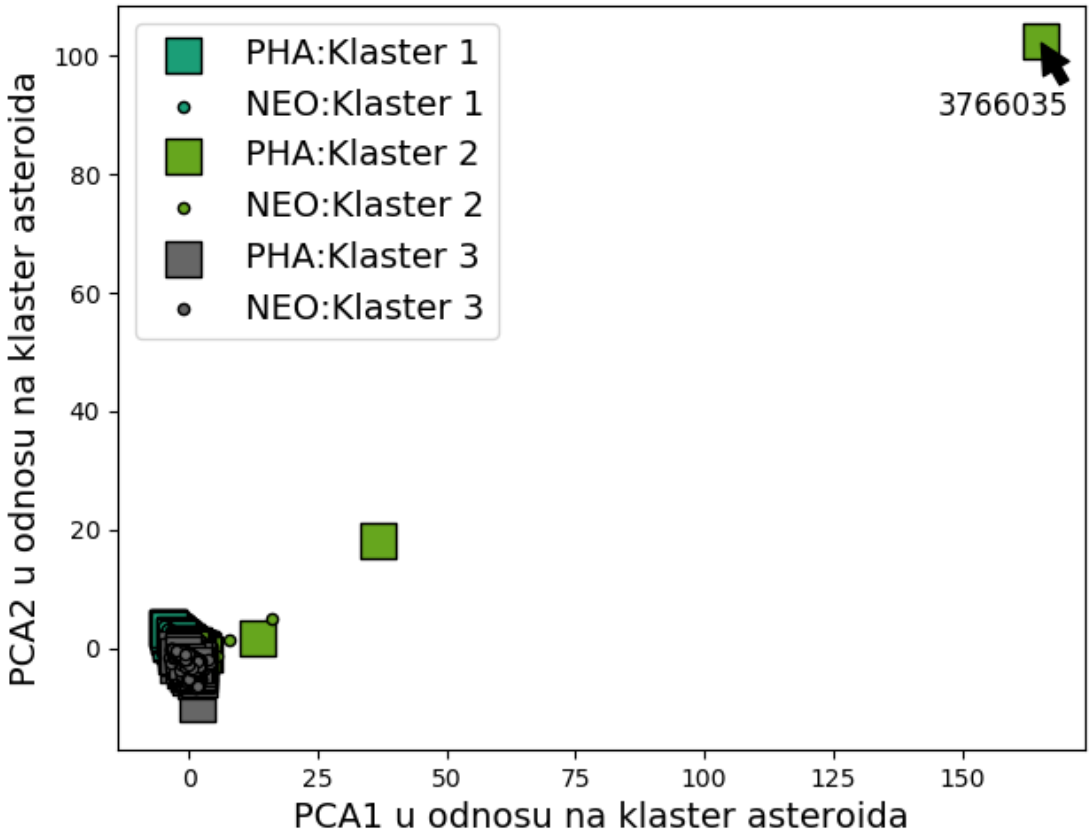
Slika 4.5: Očuvanje informacija o asteroidima u ovisnosti o broju dimenzija

provjeriti koliko varijance zadržava svaka od dobivenih komponenti PCA. Dobili smo kako za dvije dimenzije imamo veliki gubitak podataka. Prva dimenzija PCA (PCA1 u daljem tekstu) zadržava tek 29.11% podataka, dok s drugom (PCA2 u daljem tekstu) dolazimo do 44.99%. Idealno bi bilo reducirati dimenzije podataka na broj dimenzija kojim se informacije očuvaju u što većem broju. Na Slici 4.5 možemo vidjeti kako bi s $n = 10$ očuvali većinu podataka. No, kako nam PCA uglavnom služi kao metoda za vizualizaciju podataka, dimenzije podataka smo ipak smanjili s $(16557, 16)$ na $(16557, 2)$.

Kao atribute asteroida koje skaliramo i kojima reduciramo dimenzije koristili smo sve preuzete parametre koji imaju brojčanu vrijednost kako bismo vidjeli kakvo grupiranje podataka ćemo dobiti. Dakle, koristimo sljedeće atribute asteroida:

```
columns = [
     "absolute_magnitude_h",
    "average_diameter",
     "minimum_orbit_intersection",
    "jupiter_tisserand_invariant",
    "epoch_osculation",
    "eccentricity",
    "semi_major_axis",
```

```
    "inclination",
    "ascending_node_longitude",
    "orbital_period",
    "perihelion_distance",
    "perihelion_argument",
    "aphelion_distance",
    "perihelion_time",
    "mean_anomaly",
    "mean_motion"]
```



Slika 4.6: Komet 2016 VZ18 (3766035) čije se karakteristike razlikuju od većine drugih objekata

Prvi rezultati sadržavali su objekt koji je bio na velikoj udaljenosti od preostalih objekata dobivenih algoritmom grupiranja. Nakon provjere njegovog *neo_reference_id* oznake (3766035[6]) u NeoWs bazi ustanovili smo kako je u pitanju komet te smo ga uklonili iz analize. Kako NeoWs baza sadrži i komete, mogli smo očekivati ovaj ishod kod objekata koji nemaju slične karakteristike kao većina drugih. Odstupanje karakteristika navedenog kometa možemo vidjeti na Slici 4.6. Komet smo uklonili iz korištenih podataka kako bismo izbjegli potencijalne negativne utjecaje na rezultate. Isto tako, s obzirom na to kako smo time smanjili intervale vrijednosti kojima

[6] https://ssd.jpl.nasa.gov/sbdb.cgi?sstr=3766035

raspolažemo, olakšali smo i vizualizaciju podataka.

4.3 Rezultati KMEANS grupiranja

Koristimo nenadzirano strojno učenje primjenom algoritama grupiranja. Zbog toga KMEANS algoritmu dajemo podatke o objektima bez oznake radi li se o ne-PHA objektu, ili PHA objektu. Promatramo način na koji će algoritam grupirati podatke koji je NASA označila kao PHA i ne-PHA. Potencijalno opasni asteroidi prikazani su pomoću kvadrata (□), dok su ne-PHA objekti prikazani pomoću krugova (∘). Različite boje na grafovima predstavljaju različite klastere unutar kojih su objekti grupirani.

Cilj algoritma je grupirati asteroide u različite klastere na temelju sličnosti većeg broja karakteristika tako da se svaki uzorak nađe u klasteru s čijim članovima ima najveću sličnost tj. najmanju moguću varijancu. U narednom kodu inicijaliziramo KMeans na 3 centra grupiranja (k=*n_clusters*) s nasumičnim odabirom podataka za početne točke centara klastera.

```
1 clusterator = KMeans(init="random", n_clusters=3)
2 clusterator.fit(pca_data)
3 cluster = clusterator.predict(pca_data)
4 labels = clusterator.labels_
5 centers=clusterator.cluster_centers_
```

Pomoću *fit( )* funkcije izračunavamo centre klastera, a pomoću *predict( )* dodjeljujemo svaku točku najbližem centru. Sve točke dobivaju oznaku nekog od klastera. Ekvivalentna funkcija za navedeno je *fit_predict( )*.

Važan korak u KMEANS grupiranju je dobar odabir varijable k. Koristili smo tzv. analizu silueta (engl. *Silhouette analysis*) [31]. U ovoj metodi pokrećemo KMEANS na podacima za odabrani interval vrijednosti za koje smatramo kako bi mogle biti dobre za varijablu k. Za svaku od vrijednosti k iz intervala, računa se mjera koja pokazuje koliko je svaka točka jednog klastera blizu točkama susjednih klastera. Pomoću ove metode dobivamo ideju koliko se klastera nalazi unutar podataka. Analiza izračunava koeficijent silueta s koji govori koliko je određeni uzorak udaljen od susjednih klastera za unaprijed određen broj k. Taj koeficijent ima vrijednost unutar $[-1, 1]$, gdje vrijednosti blize nuli ukazuju kako je uzorak jako blizu granici susjednog klastera dok vrijednosti koeficijenta bliži jedan označavaju kako se radi o uzorku koji je jako udaljen od susjednih klastera. Negativne vrijednosti koeficijenta (bliže minus

jedan) uglavnom ukazuju na pogrešku u grupiranju, odnosno da je uzorak dodijeljen krivom klasteru. Dakle, viši koeficijent *s* (bliži jedan) govori o vrijednosti *k* koja ima bolje određene klastere odnosno gusto grupiranje uzoraka, dok onaj bliže nuli govori o klasterima koji se preklapaju. Koeficijent za jedan uzorak računa se koristeći sljedeću formulu, gdje a predstavlja srednju udaljenosti između uzorka i svih ostalih uzoraka unutar klastera, a b srednju udaljenost između uzorka i svih drugih točaka najbližeg, susjednog klastera [31]:

$$s = \frac{b - a}{\max(a, b)} \tag{4.1}$$

Za računanje koeficijenata koristili smo *sklearn.metrics.silhouette_score*. Nakon upotrebe analize silueta nad svim uzorcima asteroida dobili smo sljedeće rezultate:

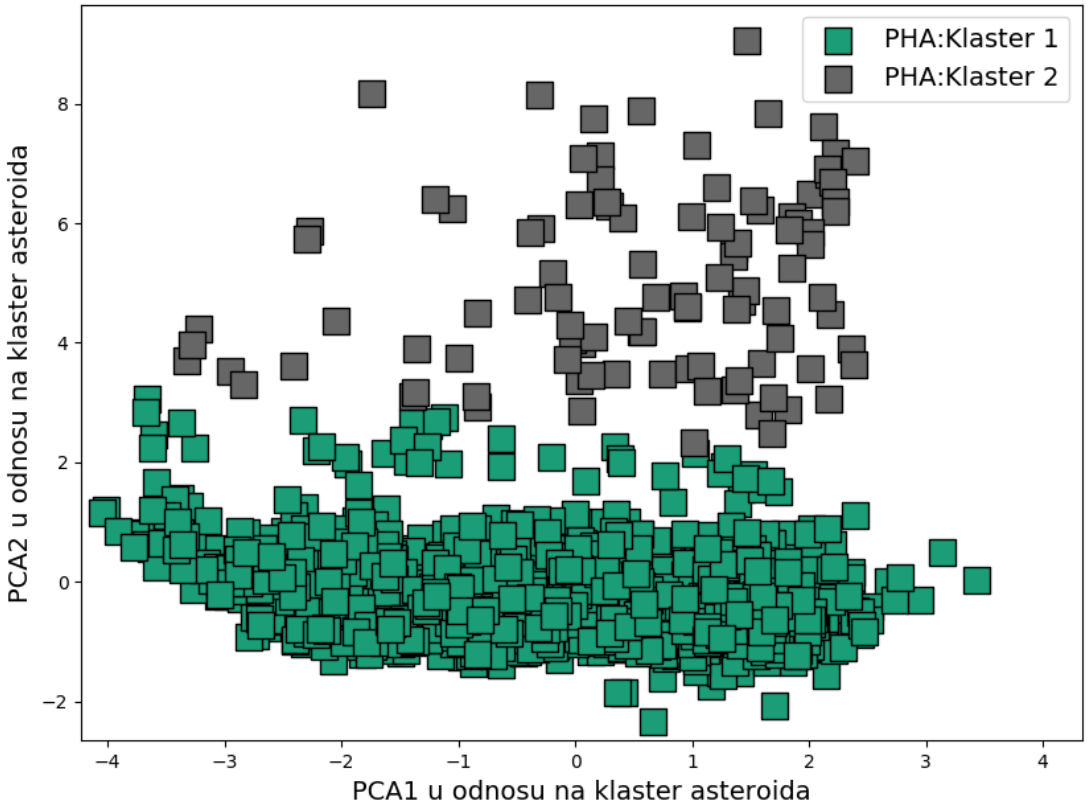
- Za $k = 2$ prosječna vrijednost *s* je : 0.388470374478
- Za $k = 3$ prosječna vrijednost *s* je : 0.399605839262
- Za $k = 4$ prosječna vrijednost *s* je : 0.400320734303
- Za $k = 5$ prosječna vrijednost *s* je : 0.352645108943
- Za $k = 6$ prosječna vrijednost *s* je : 0.330640990948
- Za $k = 7$ prosječna vrijednost *s* je : 0.345837509235

Uvidom u rezultate zaključili smo kako su dobivene vrijednosti koeficijenta *s* relativno niske. Najbolji rezultati su dani za k=4, no kako bi se dobili dobro definirani klasteri potrebni su rezultati od 0.7 naviše. Moguće je kako nam neki od atributa asteroida dodaju šumove u rezultatima, ili samo korištenje algoritama grupiranja ne odgovara ovom tipu podataka.

Kada iz obrade uklonimo atribute *absolute_magnitude_h* i *minimum_orbit_intersection*, dobivamo nešto bolje prosječne vrijednosti koeficijenata gdje je opet preporuka koristiti k=4:

- Za $k = 2$ prosječna vrijednost *s* je : 0.437631402366
- Za $k = 3$ prosječna vrijednost *s* je : 0.500249593664
- Za $k = 4$ prosječna vrijednost *s* je : 0.501237357732
- Za $k = 5$ prosječna vrijednost *s* je : 0.430924745888

- Za $k = 6$ prosječna vrijednost s je : 0.436789981657
- Za $k = 7$ prosječna vrijednost s je : 0.385953292026

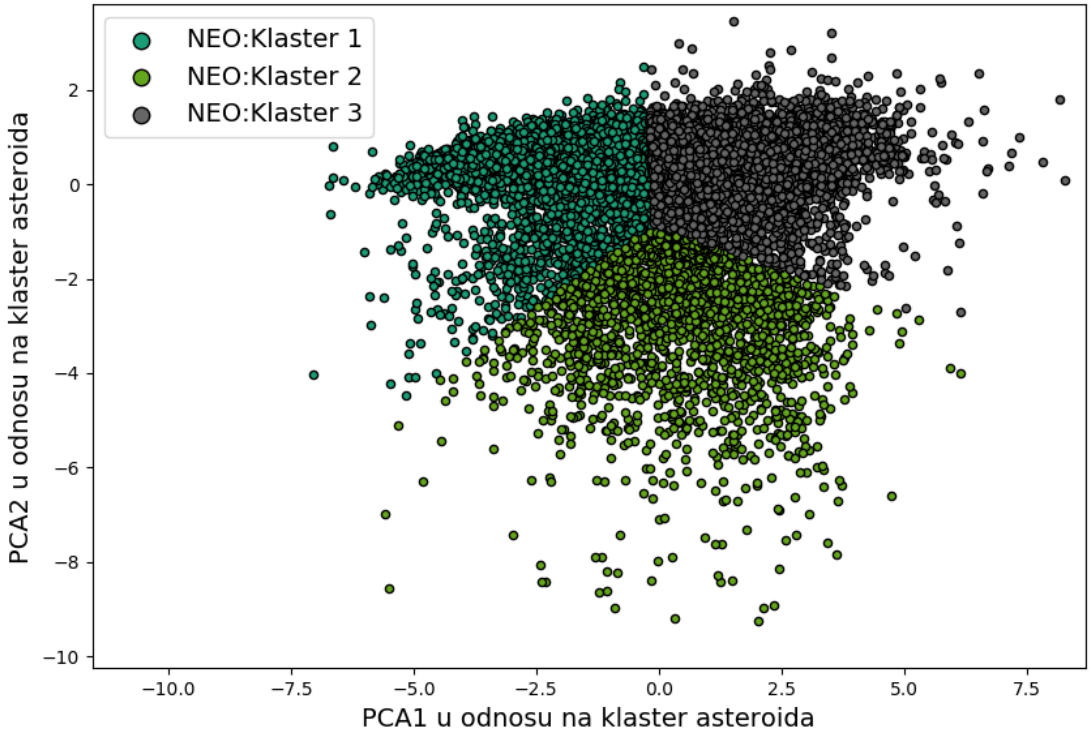


Slika 4.7: Klasteri nad PHA objektima, k=2

S obzirom kako smo bolje rezultate dobivali kada bi izbacili dva parametra o kojima ovisi sama definicija da li je asteroid PHA ili nije, odlučili smo provjeriti koeficijente koje dobivamo samo za dio dataframe-a u kojemu se nalaze PHA asteroidi, odnosno ne-PHA asteroidi, bez korištenja tih parametara. Kada smo upotrijebili sve parametre u ovim slučajevima, rezultati nisu bili značajno različiti od onih dobivenih za sve parametre na cijelom dataframe-u.

Kada upotrijebimo sve dostupne informacije samo o asteroidima koji su definirani kao PHA dobivamo puno bolje rezultate za mogućnost grupiranja podataka s k=2 što smo grafički prikazali na Slici 4.7.

- Za $k = 2$ prosječna vrijednost s je : 0.683139958986
- Za $k = 3$ prosječna vrijednost s je : 0.659528367563
- Za $k = 4$ prosječna vrijednost s je : 0.501852905327



Slika 4.8: Klasteri nad ne-PHA objektima, k=3

- Za $k = 5$ prosječna vrijednost s je : 0.419010926502
- Za $k = 6$ prosječna vrijednost s je : 0.384664752602
- Za $k = 7$ prosječna vrijednost s je : 0.38743119512

Na spomenutoj slici radi bolje vizualizacije grupiranja nismo prikazali točke koje predstavljaju vanjske asteroide ovog grupiranja, tzv. netipične vrijednosti (engl. *outliers*). Netipične vrijednosti su unutar rezultata značajno udaljene od prikazanih klastera iako njima pripadaju. Spomenut ćemo kako su u pitanju objekti čiji su *neo_reference_id* identifikatori 3757574, 3024715 i 3012393.

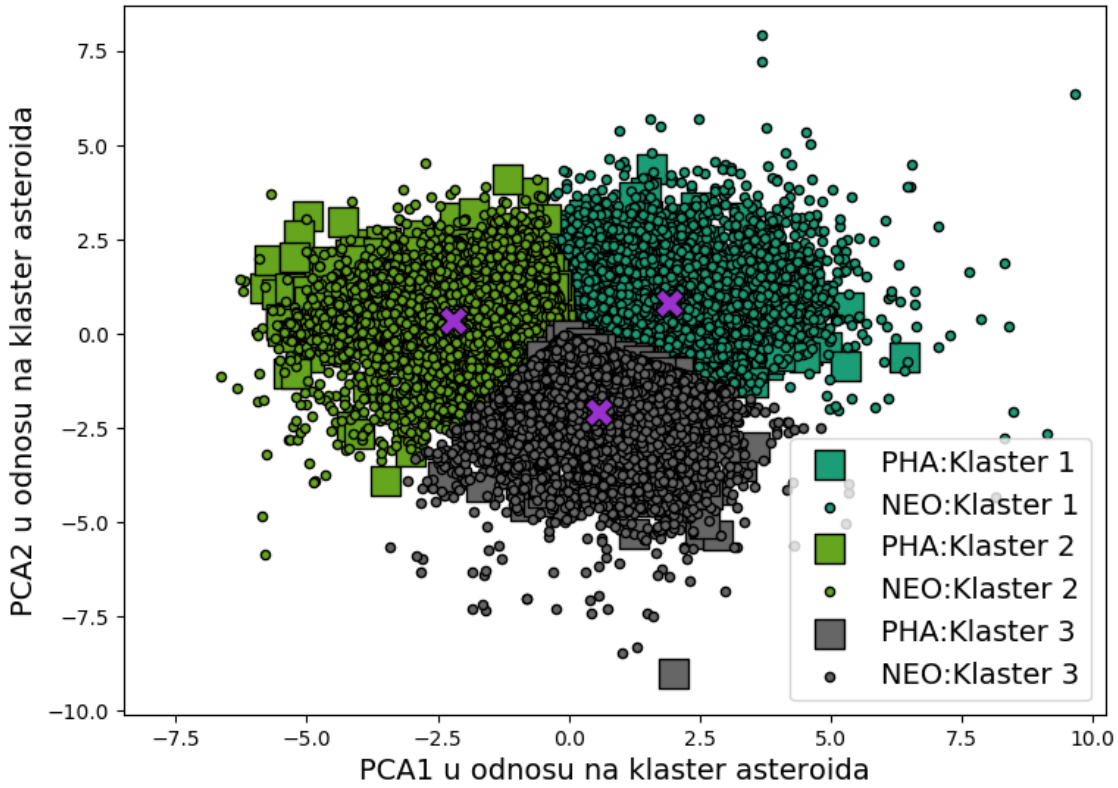
Za ne-PHA asteroide dobivamo lošiju mogućnost određivanja klastera gdje je k=3 najbolji odabir.

- Za $k = 2$ prosječna vrijednost s je : 0.453301614344
- Za $k = 3$ prosječna vrijednost s je : 0.495044545591
- Za $k = 4$ prosječna vrijednost s je : 0.430451718564
- Za $k = 5$ prosječna vrijednost s je : 0.440393129522
- Za $k = 6$ prosječna vrijednost s je : 0.395136394289

- Za $k = 7$ prosječna vrijednost s je : 0.394842330851

Grupiranje smo prikazali na Slici 4.8. I u ovoj vizualizaciji smo uklonili netipične vrijednosti čije identifikatore ćemo nabrojati: 3683246, 2001036, 3555018, 2000433.

S obzirom na male razlike u koeficijentima s prilikom izvršavanja KMEANS nad svim podacima, odlučili smo upotrijebiti sve dostupne atribute asteroida, uključujući *MOID* i apsolutnu magnitudu za k=3. Slika 4.9 prikazuje uvećani prikaz dobivenih klastera nad podacima u kojima oznaka X predstavlja njihove centre.

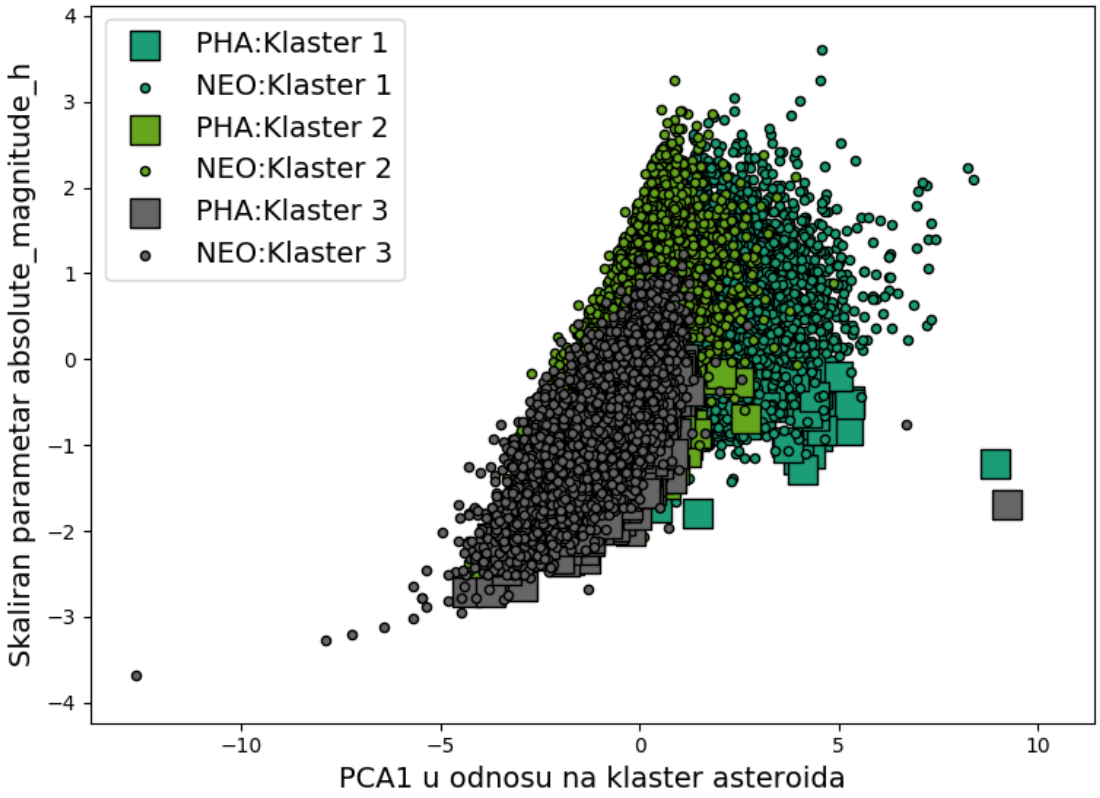


Slika 4.9: KMEANS (PCA) klasteri (svi objekti, k=3)

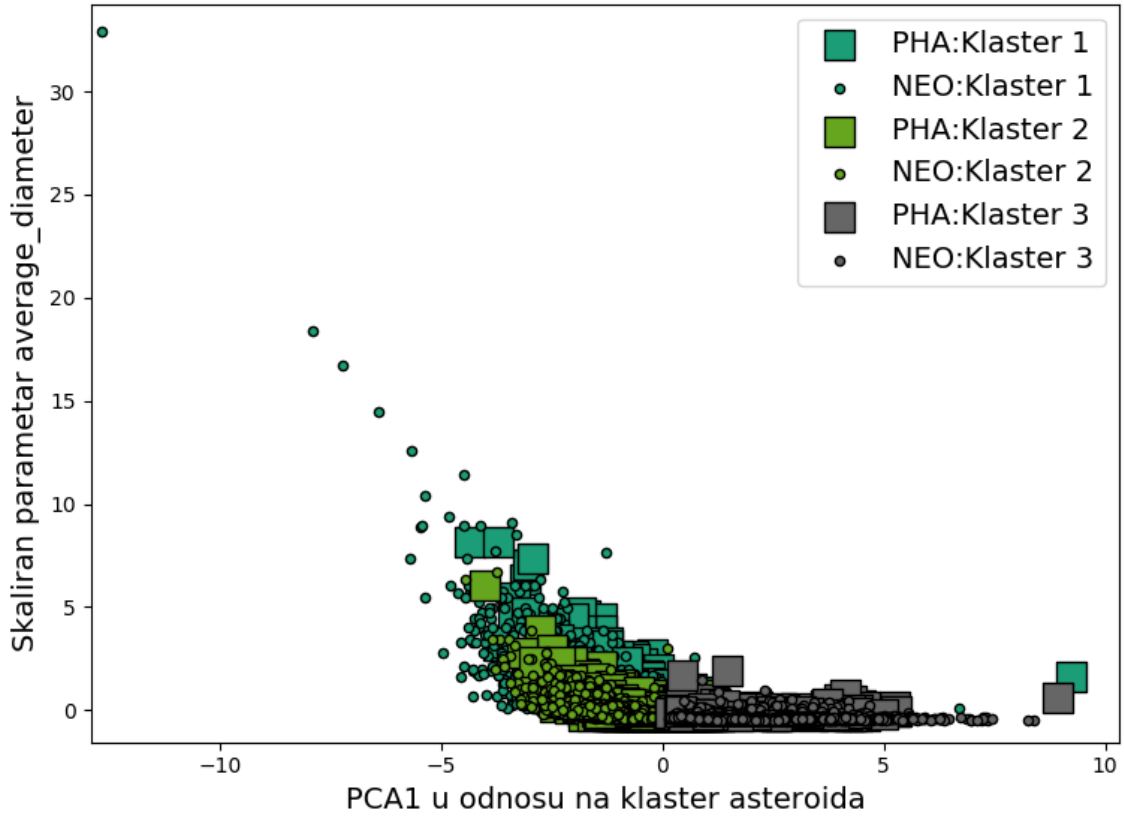
Zbog prilagođenije vizualizacije klastera na Slikama 4.10-4.14 nisu prikazane dvije netipične vrijednosti (PHA 3757574, ne-PHA 2001036). Spomenuti 2001036 je najveći NEA naziva 1036 Ganymed. Asteroid 3757574 ima značajno drugačije orbitalne parametre od ostalih NEO: najveću veliku poluos, najveći orbitalni period i najveći ekscentricitet. Pretpostavljamo kako smo dva spomenuta asteroida mogli i ukloniti iz korištenih podataka s obzirom na to kako se radi o asteroidima s iznimno različitim karakteristikama od ostalih. Navedeno ne isključuje mogućnost kako se potencijalno radi o pogrešnoj klasifikaciji kod tih asteroida. Na 3D vizualizaciji u odnosu na *MOID* (Slika 4.15) iz podataka smo prije grupiranja uklonili navedene

asteroide.

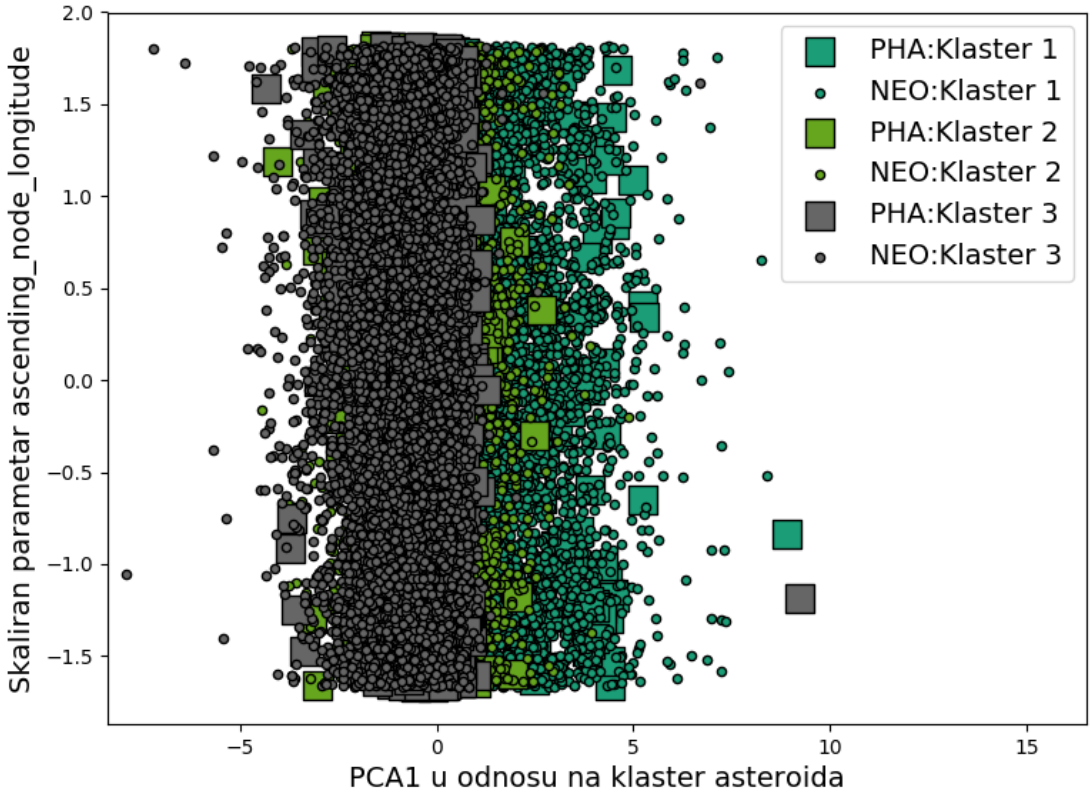
Na rezultatima prilikom usporedbe s *MOID* (Slika 4.14) i *H* (Slika 4.10) dobivamo određene pravilnosti, no kako je grupiranje nenadzirano strojno učenje, nismo u mogućnosti tim klasterima dati odgovarajuće oznake. Pravilnosti uočavamo upravo radi ovisnosti kategorizacije asteroida o tim varijablama, no klasteriranje ne ukazuje na mogućnost izricanja željenog zaključka. Postoji mogućnost da su netipične vrijednosti u odnosu na klastere naši rezultati. Moguće je i kako su svi ne-PHA asteroidi koji se na slikama nalaze u klasteru s PHA asteroidima upravo ti koji su pogrešno klasificirani. Ipak pretpostavljamo da se radi o nemogućnosti metode da uspješno klasificira PHA i ne-PHA objekte koristeći trenutne podatke iz baze NeoWs. Slika 4.9 prikazuje kako su ne-PHA i PHA asteroidi raspršeni kroz sve klastere bez vidljivog značajnijeg grupiranja na temu potencijalne opasnosti za Zemlju.



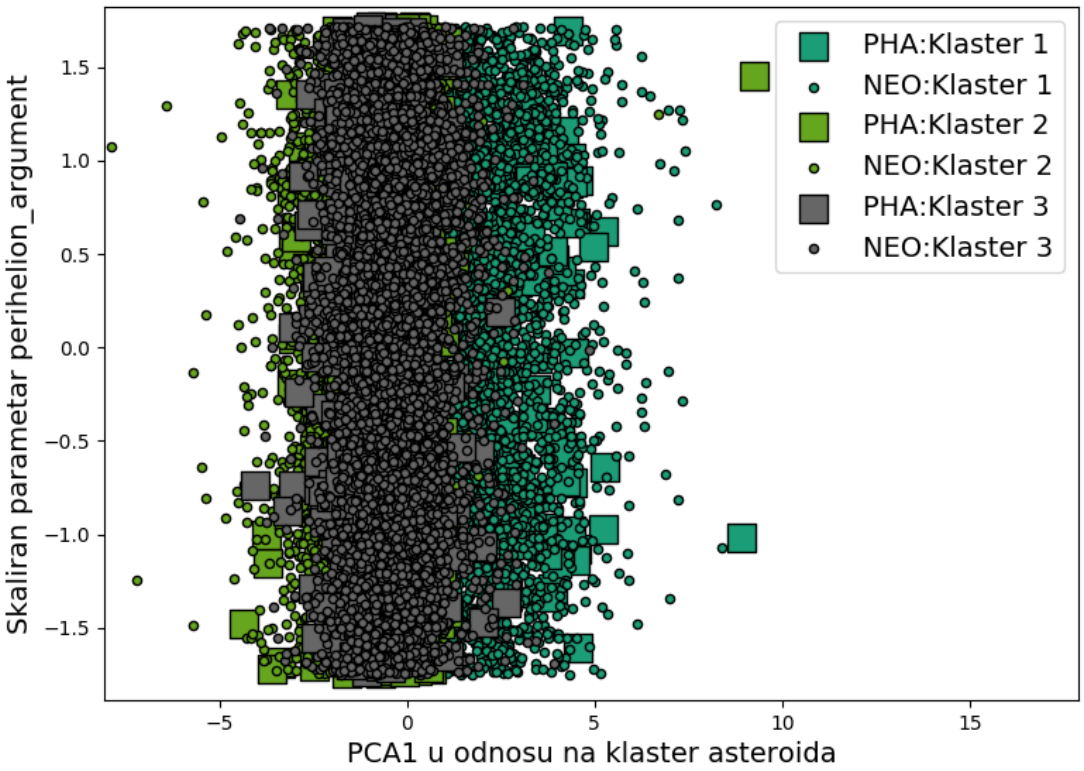
Slika 4.10: KMEANS (*absolute_magnitude_h*) klasteri (svi objekti, k=3)



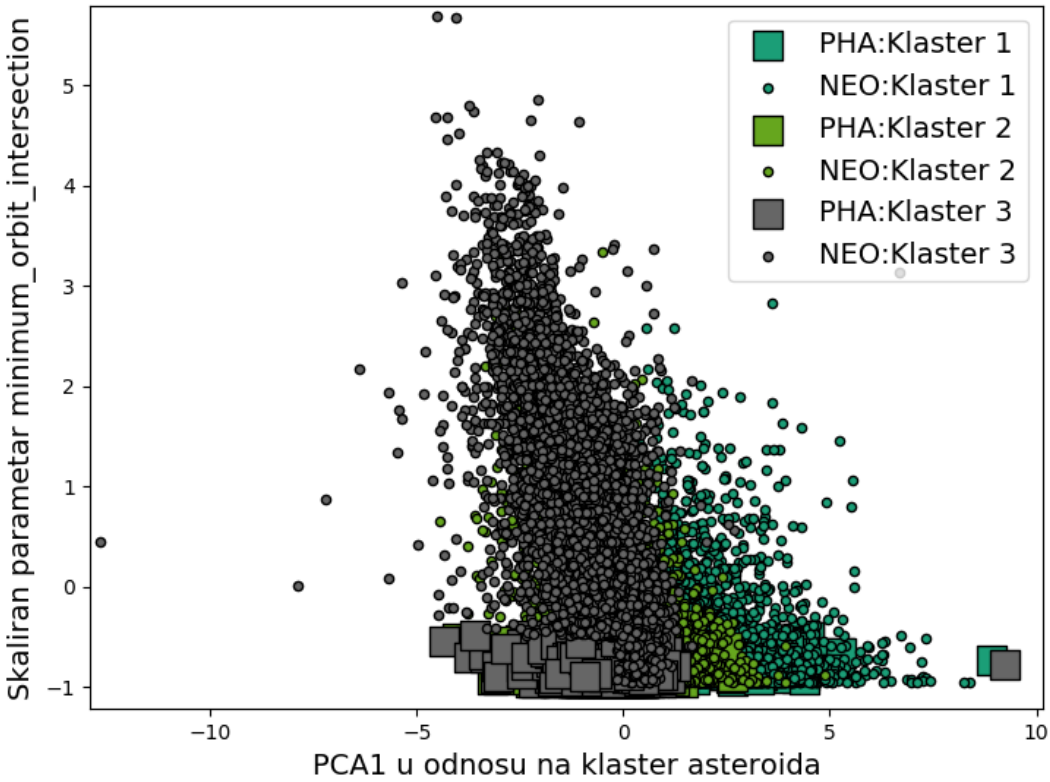
Slika 4.11: KMEANS (*average_diameter*) klasteri (svi objekti, k=3)



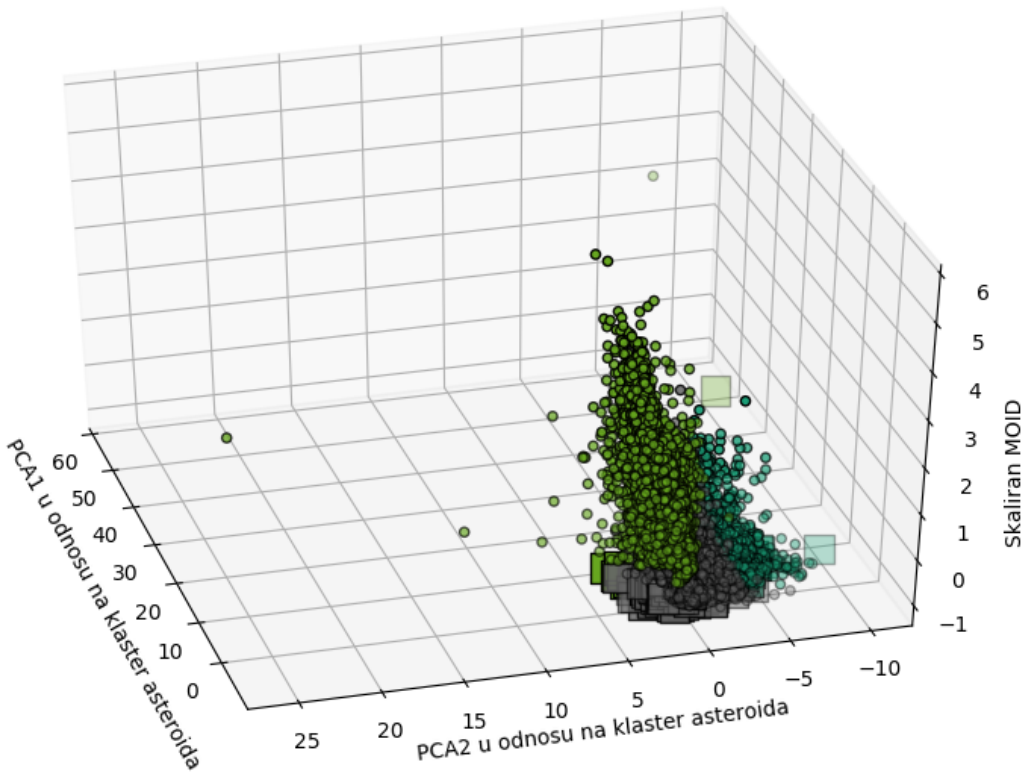
Slika 4.12: KMEANS (*ascending_node_longitude*) klasteri (svi objekti, k=3)



Slika 4.13: KMEANS (*perihelion_argument*) klasteri (svi objekti, k=3)



Slika 4.14: KMEANS (*minimum_orbit_intersection*) klasteri (svi objekti, k=3)



Slika 4.15: KMEANS (*minimum_orbit_intersection*) klasteri (svi objekti, k=3)

4.4 Rezultati DBSCAN grupiranja

Pored KMEANS algoritma, koji je bio realiziran i u projektu ATD, implementirali smo algoritam DBSCAN. U vizualizaciji podataka zadržavamo oznake opisane kod KMEANS algoritma. DBSCAN je odabran zbog toga što, osim što pripada u kategoriju algoritama grupiranja, prikazuje anomalije unutar podataka kroz pojam "točaka šuma". S obzirom na dani problem, u kojemu iz podataka grupiranjem i usporedbom grupiranja s otprije poznatim oznakama danima od strane NASA-e (ne-PHA ili PHA), moguće je kako su upravo neke od točaka šuma asteroidi koji su potencijalno krivo klasificirani.

Prilikom odabira parametara *eps* i *min_samples*, postalo je očito kako algoritam teži samo jednom klasteru s određenim brojem točaka šuma. Postoji nekoliko mogućih objašnjenja dobivenih točaka šuma. Prvo je da unutar podataka postoje pogreške ili nepreciznosti koje ih uzrokuju. Isto tako, moguće je kako su potrebni neki dodatni atributi unutar podataka koji bi objasnili pojavu tih točaka. Moguće je i da korištenje algoritama grupiranja nad ovim skupom podataka nije prikladno.

Jedno od mogućih objašnjenja točaka šuma na kojemu temeljimo analizu dobivenog je da postoje pogreške u oznakama podataka. DBSCAN grupiranje smo realizirali pomoću *sklearn.cluster.DBSCAN* klase čije smo instanciranje prikazali u narednom primjeru koda. Pomoću *fit_predict( )* funkcije smo izvršili grupiranje te su svi podaci dobili oznaku klastera ili točke šuma.

```
1 clusterator = DBSCAN(eps,min_samples)
2 cluster = clusterator.fit_predict(pca_data)
3 labels = clusterator.labels_
```

Odlučili smo provjeriti dobiveno na istim atributima asteroida kao i za KMEANS slučaj (navedenim u Poglavlju 4.2), ali koristeći sljedeće kombinacije metoda nakon skaliranja podataka:

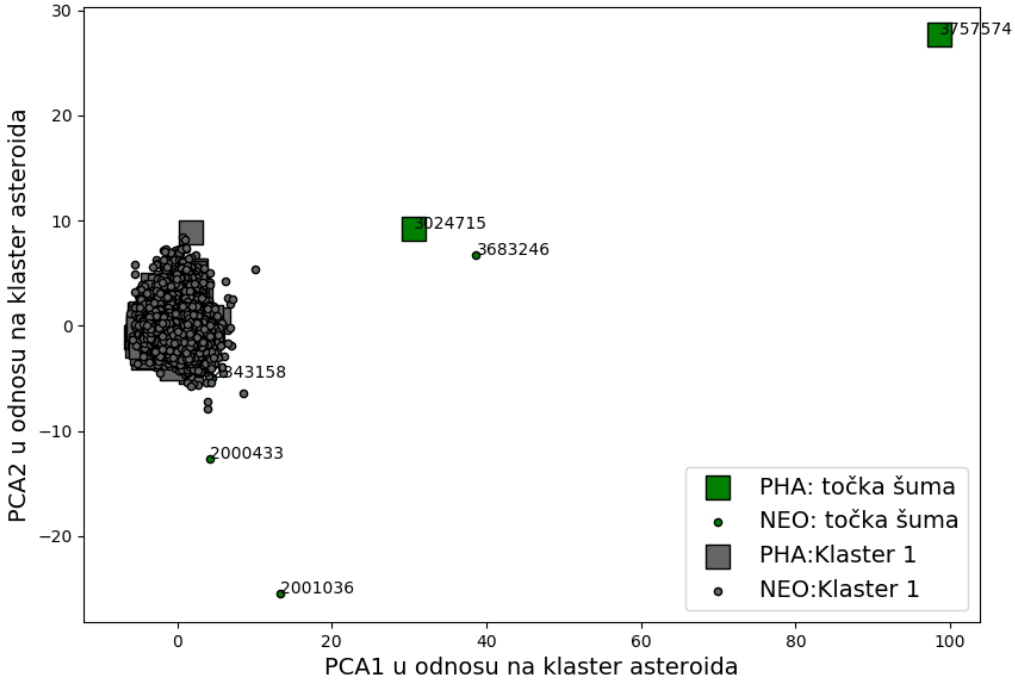
- nad podacima izvršiti redukciju pomoću PCA metode, no grupiranje izvršiti nad originalnim skaliranim podacima (Slika 4.16)
- nad podacima izvršiti redukciju pomoću t-SNE metode, grupiranje izvršiti nad originalnim skaliranim podacima (Slika 4.17)
- nad podacima izvršiti redukciju pomoću PCA metode, grupiranje izvršiti nad podacima dobivenim PCA metodom (Slika 4.18)

Vizualizacija	Točke šuma
Slika 4.16	3024715 3683246 2000433 2001036 3757574 2343158
Slika 4.17	3024715 3683246 2000433 2001036 3757574 2343158
Slika 4.18	3024715 3683246 2001036 3757574
Slika 4.19	-

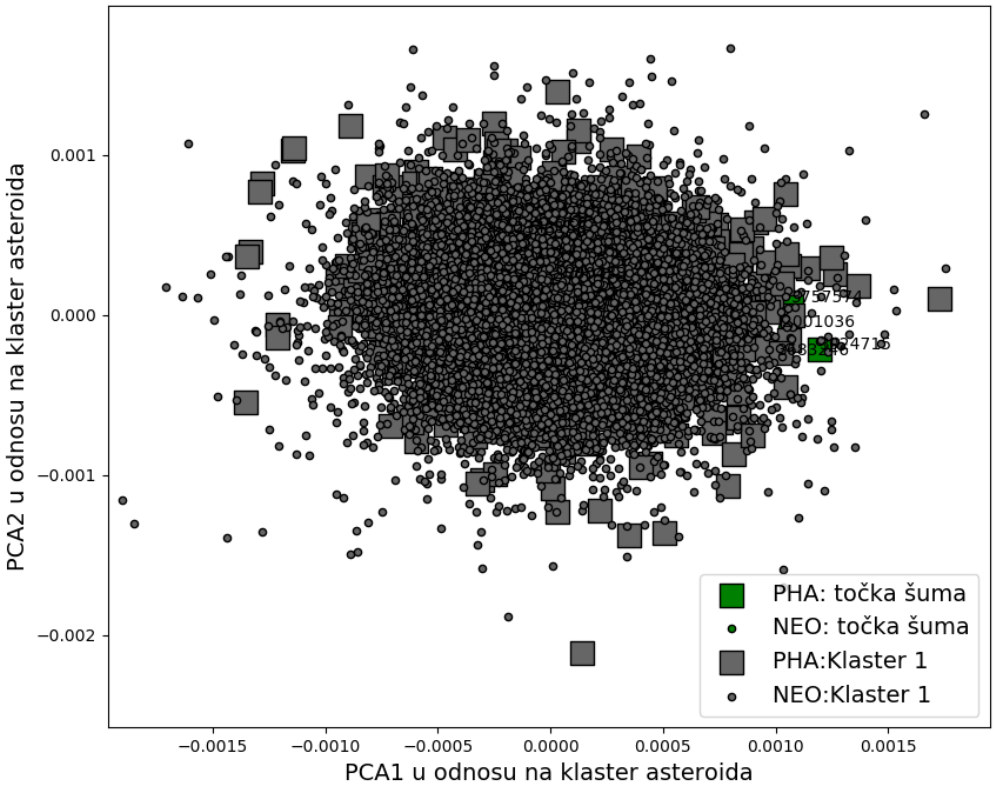
Tablica 4.2: Rezultati DBSCAN algoritma grupiranja

- nad podacima izvršiti redukciju pomoću t-SNE metode, grupiranje izvršiti nad podacima dobivenim t-SNE metodom (Slika 4.19)

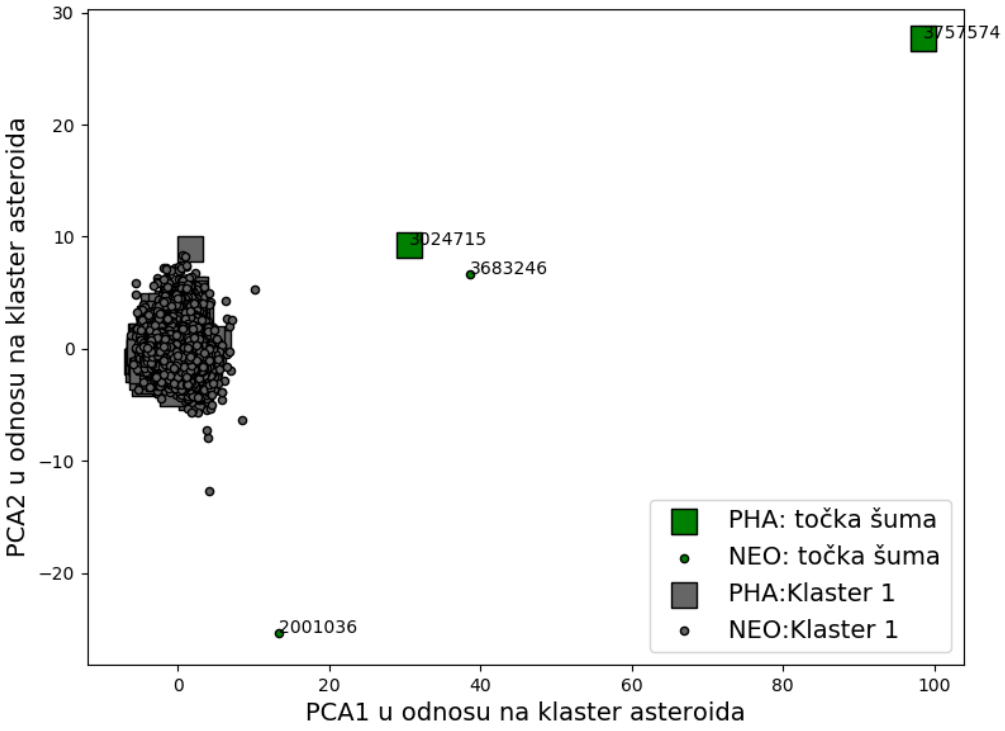
U Tablici 4.2 možemo vidjeti dobivene točke šuma za navedene slučajeve. Usporedbom Slika 4.16, 4.17 i tablice primjećujemo da dobivamo jednake točke šuma, ali s obzirom da klasteriranje vršimo nad originalnim podacima razlika niti ne može postojati. No, kako metode redukcije dimenzija podataka koristimo prilikom vizualizacije rezultata vidljivo je poboljšanje prikaza prilikom korištenja t-SNE algoritma gdje ne primjećujemo rasipanje točaka kao u slučaju korištenja PCA algoritma. Na Slikama 4.18 i 4.19 vidimo još značajniju razliku upotrebe dvaju metoda. Kada koristimo reducirane podatke za grupiranje asteroida, manje točaka pripada točkama šuma odnosno, algoritam može bolje prepoznati točke klastera. Kada koristimo DBSCAN algoritam nad podacima koji su samo skalirani dobivamo više točaka šuma nego u slučaju skaliranih i reduciranih podataka. Zanimljivo je da upotreba DBSCAN algoritma nad skaliranim i reduciranim podacima pomoću t-SNE algoritma (Slika 4.19) ne producira točke šuma što bi se moglo iskoristiti prilikom upotrebe drugih metoda strojnog učenja nad podacima u daljnjem pristupu problemu reklasifikacije asteroida u blizini Zemlje.



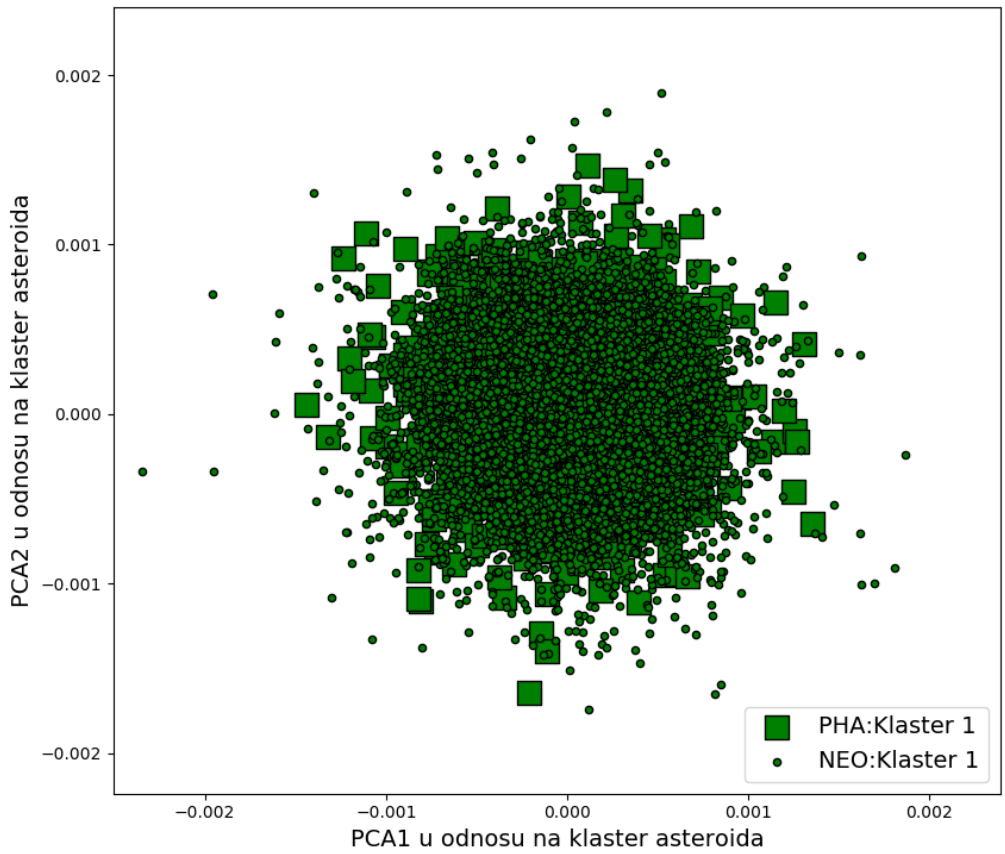
Slika 4.16: DBSCAN (PCA) nad skaliranim podacima



Slika 4.17: DBSCAN (t-SNE) nad skaliranim podacima



Slika 4.18: DBSCAN (PCA) nad skaliranim i reduciranim podacima



Slika 4.19: DBSCAN (t-SNE) nad skaliranim i reduciranim podacima

5 Zaključak

NASA je 2016. godine održala Space Apps Challenge kojim su od sudionika natjecanja tražili razvoj metoda strojnog učenja za analizu podataka o astronomskim objektima koji bi mogli predstavljati opasnost za život na Zemlji. Jedno od predloženih rješenja bio je projekt Avenge the dinosaurs koji je napisao Andrei Paleyes (programer iz tvrtke Amazon). Koristeći početnu ideju navedenog projekta željeli smo provjeriti mogu li se algoritmi grupiranja primijeniti na problem klasifikacije asteroida u blizini Zemlje. Pokušaj podjele asteroida u blizini Zemlje na potencijalno opasne (PHA) i one koji to nisu (ne-PHA) rađen je pomoću nenadziranih algoritama strojnog učenja. Prije primjene tih algoritama asteroidi nisu označeni kao PHA ili ne-PHA. Kasnije smo koristili odgovarajuće oznake iz NeoWs baze kako bismo procijenili ispravnost dobivenih rezultata. Kod nenadziranih metoda strojnog učenja problem je procjena rezultata, tj. određivanje jesmo li uspješno dobili nove informacije iz podataka.

Nakon primjene KMEANS i DBSCAN algoritama grupiranja nad podacima ne možemo tvrditi kako smo uspješno klasificirali asteroide kao PHA i ne-PHA pomoću ove metode strojnog učenja. Kao vrlo uspješnu primjenu algoritama grupiranja mogli bi definirati situaciju u kojoj imamo PHA asteroide u jednom, a ne-PHA u drugom klasteru. Takav rezultat izvršavanjem algoritama grupiranja nismo primijetili. Ne postoji apsolutni kriterij procjene rezultata grupiranja podataka. On ovisi o prikladnosti podataka, njihovom poznavanju i krajnjem cilju kojeg smo željeli ostvariti. Primjena algoritama KMEANS i DBSCAN nad podacima iz NeoWs baze je bila korisna za bolje razumijevanje odnosa između atributa asteroida. Time smo potvrdili kako je korisnost upotrebe algoritama grupiranja najveća upravo prilikom stvaranja slike o skrivenim odnosima između podataka.

Vjerujemo kako bi se s većom količinom adekvatnih podataka potencijalno mogla uspješno napraviti PHA/ne-PHA klasifikacija. Na primjer, spajanjem podataka iz NeoWs te Minor Planet Center[7] baza i korištenjem više podataka o gibanju asteroida te prikladnijom procjenom odgovarajućih atributa bi se potencijalno moglo doći do odgovarajućih informacija. Preporučili bismo korištenje i nadziranog strojnog učenja, npr. binarne klasifikacije, metoda poput SVM, neuronskih mreža i stabla odlučivanja.

[7] http://www.minorplanetcenter.net/data

Metode proučavanja svemira se razvijaju ubrzano. *Large Synoptic Survey Telescope* (LSST) projekt (nakon dovršetka konstrukcije početkom 2022. godine) će provesti desetogodišnje istraživanje svemira kojim se planira doći do 200 petabajta podataka. Baza koju smo koristili u ovom radu sadrži manje od 17 000 asteroida. Pretpostavke su da će u podacima LSST-a biti oko 5 milijuna asteroida. Zbog toga postaje sve relevantnije razvijati metode strojnog učenja za analizu već postojećih manjih skupova podataka o asteroidima.

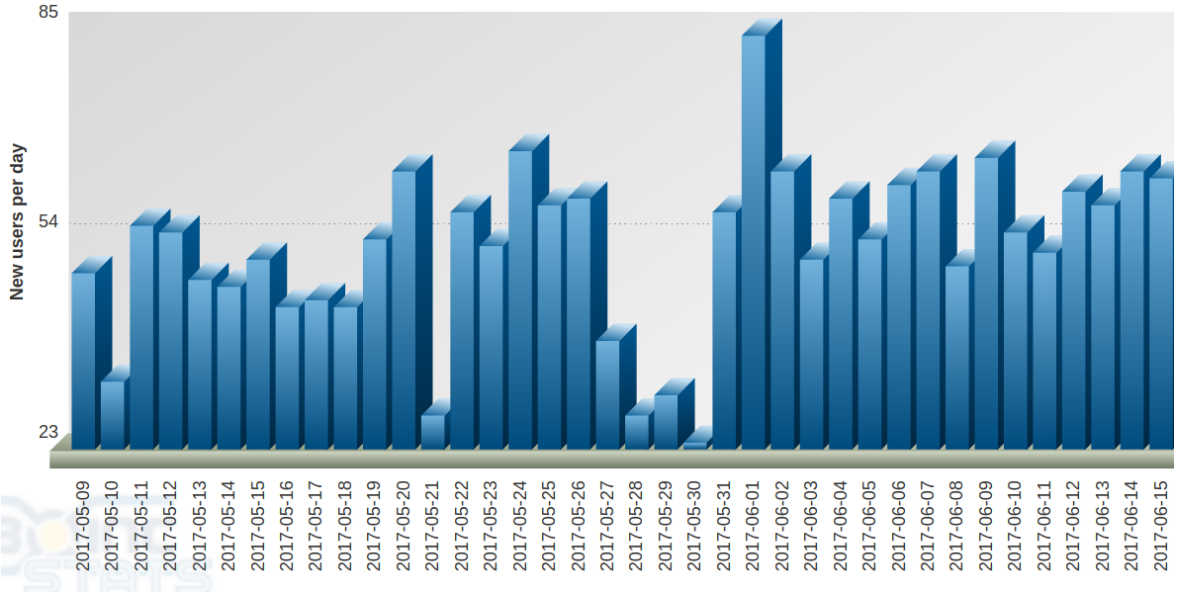
Dodatak

Dodatak A Asteroids@home

Projekt *Asteroids@home* je razvijen od strane Astronomskog Instituta u Pragu kako bi se koristili vanjski računalni resursi [32]. Korisnici poklanjaju projektu procesorsko vrijeme svojih osobnih računala te preuzimaju pripremljeni program u svrhu analize fotometrijskih podataka o asterodima. Analiziraju se podaci dobiveni iz različitih izvora proučavanja asteroida, od velikih znanstvenh istraživanja do promatranja astronoma amatera. U ovakvim projektima vidimo mogućnost približavanja astronomije i informatike velikom skupu gradjana. Korisnici vide da je moguće pomagati u znanstvenim istraživanjima na jednostavan način.

Osnovni program koji omogućava donaciju procesorskog vremena kada je računalo u mirovanju (engl. *Idle time*) naziva se BOINC (engl. *Berkeley Open Infrastructure for Network Computing*). Pomoću njega mogu se koristiti resursi bilo kojeg računala (spojenog na Internet) za analizu fizičkih karakteristika asteroida. U BOINC-u postoje i drugi projekti poput Rosetta@home (analizira oblike proteina), ili SETI@home (traži inteligentni život u svemiru). Mogućnost sudjelovanja u ovakvim projektima imaju svi građani - od učenika i studenata, preko korisnika čija zanimanja nemaju velikih poveznica s projektom kojeg su odabrali, do umirovljenika. Povezuju ih samo činjenice da svojim računalima omogućavaju napredovanje željenog projekta te posjedovanje znanstvene znatiželje i želje za učenjem. Svatko tko posjeduje računalo može iskusiti dijelove istraživačkog procesa što potiče sudjelovanje javnosti u znanosti te razvoj interesa za astrofiziku i informatiku kod mlađih korisnika. Do kraja 2015. godine oko 60 tisuća korisnika (odnosno 100 tisuća računala) sudjelovalo je u Asteroids@home projektu [34]. Od onda imamo konstantan rast broja korisnika do današnjih 100 tisuća. Na Slici A.1 vidimo broj novih korisnika po danu u određenom periodu svibnja i lipnja 2017. godine.

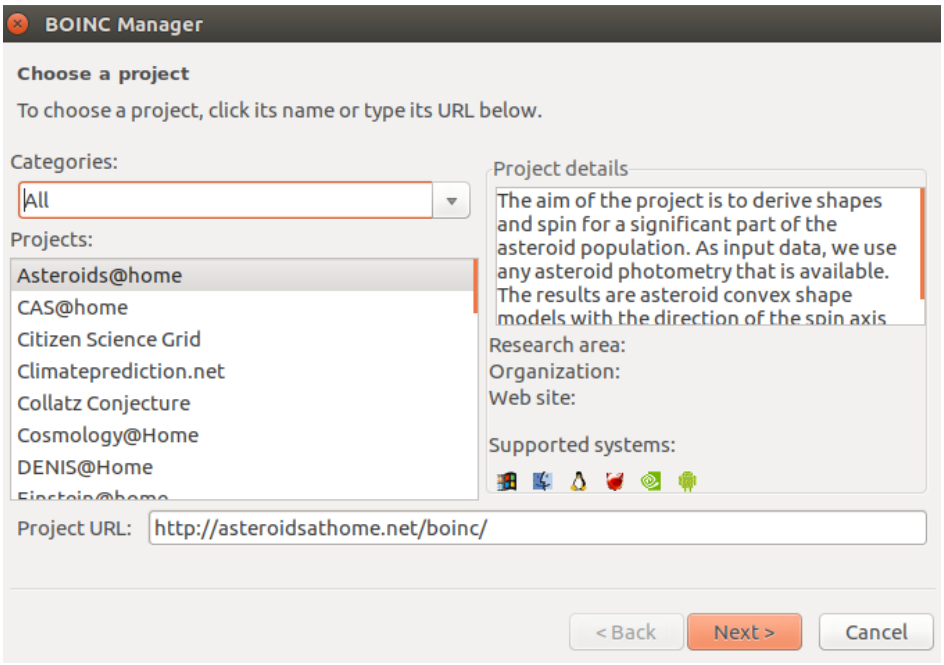
Iako je do danas mnogo asteroida otkriveno i stalno se otkrivaju novi, ne postoji mnogo saznanja o fizičkim svojstvima pojedinačnih asteroida. Karakteristike poput oblika, rotacijskog perioda i smjera rotacijske osi su poznati samo za mali broj asteroida. Projekt Asteroids@home korištenjem inverzne metode svjetlosne krivulje stvara fizičke modele asteroida. Kako asteroidi rotiraju u vremenu te su nepravilnog



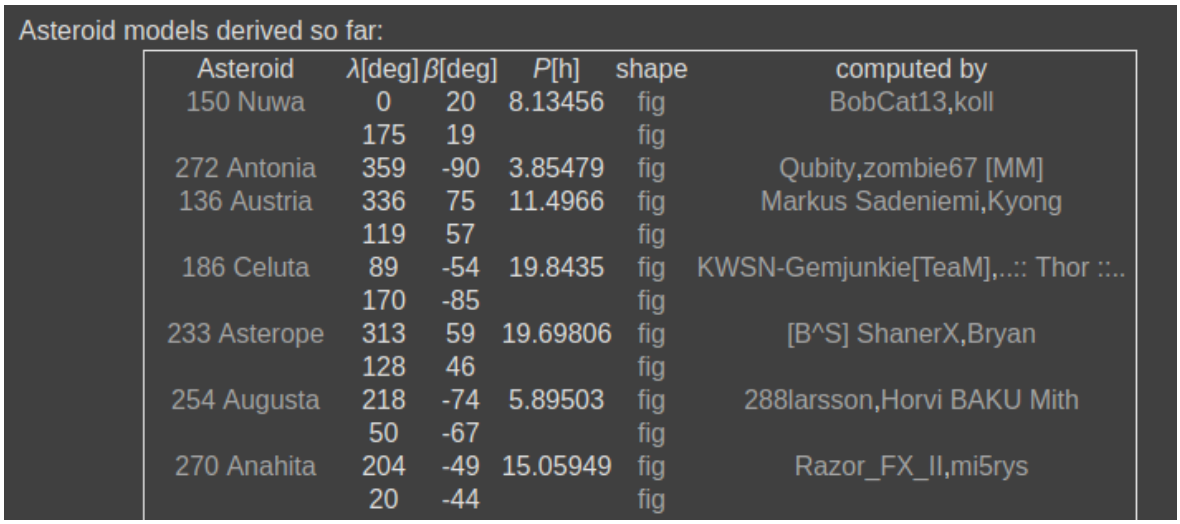
Slika A.1: Statistika korištenja Asteroids@home: broj novih korisnika po danu [33]

oblika, količina svjetlosti koju reflektiraju ovisi o vremenu. Tu promjenu sjaja u vremenu nazivamo svjetlosnom krivuljom. Stvorila se potreba za projektom upravo zbog računalne zahtjevnosti spomenute metode. Veliki broj asteroida i promjena sjaja kroz vrijeme stvaraju veliki skup podataka te je za procjenu fizičkih parametara potrebno koristiti sva dostupna računala. Određeni problem izračuna se u ovom projektu dijeli na manje zadatke. Različita računala preuzimaju i rade na tim manjim zadacima. Rezultati se na kraju ujedine u konačna rješenja i tako se dobivaju parametri asteroida. U projektu Asteroids@home je do sada istraženo više od 100 tisuća asteroida [34].

Najjednostavnija instalacija je ona s grafičkim sučeljem. Instalacija programa je brza i jednostavna za sve operacijske sustave. Zbog toga Asteroids@home mogu uspješno instalirati i osobe čije poznavanje korištenja računala nije veliko. Instalacija programa na Linuxu sastoji se od jednostavnog korištenja terminala gdje instaliramo Bionic Manager (BM) i Core Client (CC). Npr. za Ubuntu se koristi: *sudo aptitude install boinc-client boinc-manager*. Nakon instalacije (Slika A.2) vidimo mogućnost dodavanja željenih projekata kojima ćemo dopustiti korištenje CPU-a računala. Računalu se nakon odabira projekta na temelju njegovih osobina (količina RAM-a i procesor) dodjeljuje niz zadataka od strane projektnog poslužitelja zaduženog za raspored zadataka. Računalo tada preuzima podatke s poslužitelja zajedno s programima koji izvršavaju proračune. Znatiželjni korisnik se može uputiti u novi svijet istraživanja asteroida. Nakon završetka izračuna, stvaraju se datoteke s rješenjima koje se prosljeđuju natrag poslužitelju te se dobivaju novi zadaci.



Slika A.2: Dodavanje projekta Asteroids@home u BIONIC Manager

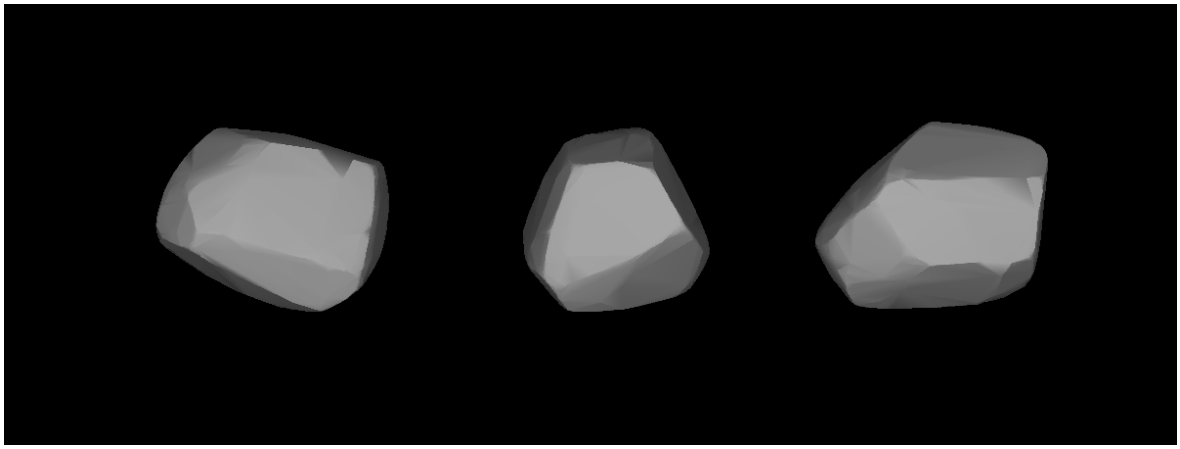


Asteroid	λ[deg]	β[deg]	P[h]	shape	computed by
150 Nuwa	0	20	8.13456	fig	BobCat13,koll
	175	19		fig	
272 Antonia	359	-90	3.85479	fig	Qubity,zombie67 [MM]
136 Austria	336	75	11.4966	fig	Markus Sadeniemi,Kyong
	119	57		fig	
186 Celuta	89	-54	19.8435	fig	KWSN-Gemjunkie[TeaM],..:: Thor ::..
	170	-85		fig	
233 Asterope	313	59	19.69806	fig	[B^S] ShanerX,Bryan
	128	46		fig	
254 Augusta	218	-74	5.89503	fig	288larsson,Horvi BAKU Mith
	50	-67		fig	
270 Anahita	204	-49	15.05949	fig	Razor_FX_II,mi5rys
	20	-44		fig	

Slika A.3: Nekoliko rezultata sa Asteroids@home stranice [35]

Ovakvi projekti imaju značajan utjecaj na popularizaciju znanosti. Ako se korisniku predstavi kako je jedino što je potrebno da sudjeluje u otkrivanju stvarnog izgleda asteroida njegovo računalo, u ljudskoj prirodi je da žele znati više o predmetu istraživanja. Zapitat će se sto su asteroidi, kako ih otkrivamo, zašto je ovaj projekt važan i slično. Zanimljivo je da korisnici mogu direktno vidjeti rezultate koji su dobiveni proračunima na njihovim računalima. Tako se potiče daljnja suradnja i razvoj interesa za astronomiju i znanost općenito. Na primjeru stranice sa Slike A.3 se vidi da korisnici mogu po imenima vidjeti rezultate njihovih računala te točne nazive i 3D slike asteroida do kojih su pomogli doći. Postoji sustav pohvala korisnika kroz

mehanizme korisnika dana (engl. *User of the day*) i mogućnost sakupljanja bodova u ovisnosti koliko računalnog vremena je dano projektu.



Slika A.4: 3D model asteroida dobiven pomoću Asteroids@home [35]

Kada korisnik odabere opciju *fig* sa stranice, može vidjeti slike dobivenih asteroida. Npr. sudjelovanjem dvaju korisnika prilikom izračuna za asteroid naziva 829 Academia dobiven je objekt sa Slike A.4. Osim slika mogu pristupiti i informacijama o pojedinačnim asteroidima koja ih vodi na NASA-inu stranicu. Tu su napisane zanimljivosti o asteroidima i odgovori na pitanja. Nova pitanja i nerazjašnjeni problemi te informatička podloga projekta su načini privlačenja mladih prema znanosti.

Literatura

[1] Zanimljivosti o asteroidima (NASA), https://tinyurl.com/yd9y9xr4, 5. 5. 2017.

[2] O imenovanju asteroida, http://www.minorplanetcenter.net/iau/info/HowNamed.html, 10. 7. 2017.

[3] Galerija slika povezanih uz asteroide (NASA), https://solarsystem.nasa.gov/planets/asteroids/galleries, 2. 7. 2017.

[4] Originalna slika opisa parametara eliptične putanje asteroida, https://www.math.ksu.edu/math240/book/capstone/ellipse.php, 2. 7. 2017.

[5] Ridpath, I. A Dictionary of Astronomy. 2nd rev. ed. Oxford University Press, 2016., http://www.oxfordreference.com/view/10.1093/oi/authority.20110803104727545, 2. 7. 2017.

[6] Morbidelli, A.; Bottke Jr., W.F.; Michel, P. Origin and Evolution of Near-Earth Objects, 2002., https://tinyurl.com/y97v2et3, 25. 5. 2017.

[7] Prikaz NEO orbita u odnosu na Zemljinu orbitu, https://cneos.jpl.nasa.gov/about/neo_groups.html, 25. 4. 2017.

[8] NASA-in članak o asteroidima, Lipanj 2017., https://solarsystem.nasa.gov/news/2017/06/26/10-things-june-26, 10. 7. 2017.

[9] Statistika broja otkrivenih NEA u odnosu na njihovu veličinu, https://cneos.jpl.nasa.gov/stats/size.html, 2. 7. 2017.

[10] VanderPlas, J. Python Data Science Handbook. O'Reilly Media. 2016.

[11] Polunadzirano strojno učenje, http://scikit-learn.org/stable/modules/label_propagation.html, 25. 6. 2017.

[12] K-means i DBSCAN vizualizacija, temeljni kod, https://tinyurl.com/y8sqhocc, 5 .6. 2017.

[13] Guido, S.; Müller, A. Introduction to Machine Learning with Python. O'Reilly Media. 2016.

[14] Slika rezultantnih točaka DBSCAN algoritma, https://tinyurl.com/y8ybksut, 5. 6. 2017.

[15] Prikaz DBSCAN algoritma https://tinyurl.com/y845hdgx, 5. 6. 2017.

[16] O Pythonu, https://www.python.org/about/, 25. 6. 2017.

[17] GitHub, https://github.com/, 25. 6. 2017.

[18] Biblioteka urllib, https://docs.python.org/3/library/urllib.html#module-urllib, 25. 6. 2017.

[19] JSON, http://www.json.org/, 25. 6. 2017.

[20] Scikit learn projekt, http://scikit-learn.org/stable/, 25. 6. 2017.

[21] NumPy biblioteka, http://www.numpy.org/, 25. 6. 2017.

[22] Pandas projekt, http://pandas.pydata.org/, 25. 6. 2017.

[23] Géron, A. Hands-On Machine Learning with Scikit-Learn and TensorFlow. O'Reilly Media. 2017.

[24] Algoritam t-SNE, http://scikit-learn.org/stable/modules/generated/sklearn.manifold.TSNE.html, 10. 7. 2017

[25] Matplotlib biblioteka, https://matplotlib.org/, 25. 6. 2017.

[26] Paleyes, A. Avenge the dinosaurs projekt, https://github.com/apaleyes/avenge-the-dinosaurs, 2017.

[27] Finalni kod diplomskog rada, https://github.com/lib686/avenge-the-dinosaurs, 2017.

[28] Opis orbitalnih parametara (originalna slika) Ω i ω, http://www.columbia.edu/~my2317/asteroidproject.html, 10. 7. 2017.

[29] NASA NeoWs Open API, pretrazivanje pojedinačnih NEO objekata, https://api.nasa.gov/api.html#neows-browse, 2017.

[30] O PCA metodi, http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html, 26. 6. 2017.

[31] Analiza silueta, ”2.3.9.5. Silhouette Coefficient”, http://scikit-learn.org/stable/modules/clustering.html, 26. 6. 2017.

[32] Asteroids@Home stranica, http://asteroidsathome.net, 30. 5. 2017.

[33] Asteroids@Home statistika, https://boincstats.com/en/stats/134/project/detail, 5. 7. 2017.

[34] Durech, J.; Hanus, J.; Vanco, R. Asteroids@home - A BOINC distributed computing project for asteroid shape reconstruction. // Astronomy and Computing 13 (2015), str. 80-84 , https://arxiv.org/abs/1511.08640

[35] Asteroids@home stranica s rezultatima, http://asteroidsathome.net/scientific_results.html, 5. 7. 2017.