VILNIAUS UNIVERSITETO
KAUNO FAKULTETAS

SOCIALINIŲ MOKSLŲ IR TAIKOMOSIOS INFORMATIKOS INSTITUTAS

Finansų technologijų studijų programa
Kodas 6211BX022

LUKAS PALIONIS

MAGISTRO BAIGIAMASIS DARBAS

AUTOMATIZUOTŲ SANDORIŲ APTIKIMO METODAS SOLANOS TINKLE

Kaunas 2025

VILNIAUS UNIVERSITETO
KAUNO FAKULTETAS

SOCIALINIŲ MOKSLŲ IR TAIKOMOSIOS INFORMATIKOS INSTITUTAS

LUKAS PALIONIS

MAGISTRO BAIGIAMASIS DARBAS

AUTOMATIZUOTŲ SANDORIŲ APTIKIMO METODAS SOLANOS TINKLE

Magistrantas _____
(parašas)
Darbo vadovas _____
(parašas)
_____
(darbo vadovo mokslo laipsnis, mokslo pedagoginis vardas, vardas ir pavardė)

Darbo įteikimo duomenys
Registracijos Nr.

Kaunas 2025

TURINYS

LENTELIŲ SĄRAŠAS 5

PAVEIKSLŲ SĄRAŠAS 6

ĮVADAS 8

1. AUTOMATIZUOTŲ SANDORIŲ APTIKIMO METODO SOLANOS TINKLE, LITERATŪROS APŽVALGA 11

1.1. Blokų grandinių technologija 11

1.2 Solanos transakcijų duomenys 16

1.3 Botų apibrėžimas 19

1.4 Esamos botų aptikimo sistemos 23

1.5 Mašininio mokymosi taikymas botų aptikimui 26

1.6 Vertinimo metrikos 29

2. AUTOMATIZUOTŲ SANDORIŲ APTIKIMAS SOLANOS TINKLE METODOLOGIJA 32

2.1. Automatizuotų sandorių aptikimo metodas Solanos tinkle tyrimo pobūdis 32

2.2. Automatizuotų sandorių aptikimo metodas Solanos tinkle tyrimo metodas 32

2.3. Automatizuotų sandorių aptikimas Solanos tinkle tyrimo metodo statistika 33

2.4. Empirinio taikomojo Automatizuotų sandorių aptikimo metodo Solanos tinkle tyrimo atlikimas 34

3. AUTOMATIZUOTŲ SANDORIŲ APTIKIMO SOLANOS TINKLE EKSPERMENTINĖ DALIS 39

3.1. Neapdorotų duomenų rinkimas ir vertinimas 39

3.2. Duomenų apdorojimas ir požymių inžinerija 42

3.3 Duomenų aprašomoji statistika 46

3.4 Klasterizacijos ir anomalijų aptikimas 55

3.4.1 K-means klasterizacijos taikymas elgsenos modelių identifikavimui 55

3.4.2. DBSCAN klasterizacijos metodo taikymas 63

3.4.3.Anomalijų aptikimas taikant Isolation Forest metodą 67

3.5 Hierarchinis klasterizacijos ir anomalijų aptikimo metodas 74

3.5.1. K means klasterių hierarchinė analizė 75
3.5.2.DBSCAN klasterių hierarchinė analizė 80
3.5.3. DBSCAN, K-means ir Isolation Forest metodų hierarchinės anomalijų analizės palyginimas 83
3.6 Ekspermento vertinimas 85
IŠVADOS 93
LITERATŪRA 95
PRIEDAI 98
1 PRIEDAS 98
2 PRIEDAS 99
3 PRIEDAS 100
4 PRIEDAS 101
5 PRIEDAS 102

LENTELIŲ SĄRAŠAS

Lentelė 1. Solana ir kitų blokų grandinių našumo palyginimas....14
Lentelė 2. Botų tipai, funkcionalumas ir susijusios rizikos kriptovaliutų rinkose....19
Lentelė 3. Pagrindiniai iššūkiai botų aptikime....24
Lentelė 4. Aprašomosios statistikos struktūra....33
Lentelė 5. Duomenų tvarkymo procesas....34
Lentelė 6. Hierarchinės analizės metodų taikymo planas....36
Lentelė 7. Neapdorotų duomenų struktūra....39
Lentelė 8. Duomenų kokybės vertinimo lentelė....41
Lentelė 9. Json tipo duomenų tvarkymas....42
Lentelė 10. Požymių grupavimas ir reikšmingumas....43
Lentelė 11. Centrinės tendencijos....45
Lentelė 12. Duomenų variacijos rodikliai....46
Lentelė 13. Pasiskirstymo formos rodikliai....48
Lentelė 14. Klasterių analizės apibendrinti duomenys....57
Lentelė 15. Klasterizacijos vertinimo metrikos pagal K reikšmes....58
Lentelė 16. DBSCAN klasterizacijos metrikų rezultatai....64
Lentelė 17. Anomalijų ir normalaus profilio požymių palyginimas....67
Lentelė 18. Statistinė požymių analizė ir reikšmingumo lygiai....68
Lentelė 19. K means klasterių savybių palyginimas....76
Lentelė 20. Anomalijų piniginės pagal K mean klasterius....77
Lentelė 21. DBSCAN klasterių savybių palyginimas....80
Lentelė 22. Anomalijų piniginės pagal DBSCAN klasterius....81
Lentelė 23. DBSCAN, K-mean ir Isolation Forest metodų anomalijų identifikavimo rezultatai.82
Lentelė 24. Anomalijų aptikimo vertinimo metrikų rezultatai pagal metodus....84
Lentelė 25. Kombinuotų metodų anomalijų aptikimo vertinimo metrikų rezultatai....85
Lentelė 26. Metodų stiprybės ir trūkumai....87
Lentelė 27. Botų aptikimo metodų rezultatų palyginimas su ankstesniais tyrimais....88
Lentelė 28. Ekspermento metodų pritaikymo galimybės Solana ekosistemoje....89

PAVEIKSLŲ SĄRAŠAS

1pav. Blokų grandinės struktūra 11
2 pav. Proof-of-History grandinė su laiko žymėmis 14
3 pav. Rizikos valdymo sistema Mango Markets platformoje 22
4 pav. Botų Aptikimo Technikos 25
5 pav. Automatizuotų sandorių aptikimo metodas Solanos tinkle parinkimo etapai 38
6 pav. Piniginės adreso įvedimo laukas 39
7 pav. Piniginės adreso įvedimo vartotojo sąsaja 51
8 pav. Transakcijų dažnio pasiskirstymas 52
9 pav. Vidutinės transakcijos sumos pasiskirstymas 53
10 pav. Požymių pasiskirstymo Box Plot analizė 54
11 pav. Optimalaus klasterių skaičiaus (K) paieška naudojant Elbow metodą 55
12pav. Silhouette Score priklausomybė nuo Cluster K skaičiaus 56
13 pav. Calinski-Harabasz indekso priklausomybė nuo klasterių skaičiaus (K) 57
14 pav. Galutinė K-means klasterizacija pagrindinių komponentų plokštumoje 57
15pav. Klasterių ypatybių pasiskirstymo analizė pagal elgesio metrikas 58
16 pav. Kryžminės validacijos metrikos pagal klasterių skaičių (K) 61
17 pav. DBSCAN klsaterizavimo PCA vizuolizacija 63
18pav. Savybių svarbos vizualizacija DBSCAN klasterizacijoje 64
19pav. Klasterių dydžio pasiskirstymas pagal DBSCAN metodą 65
20 pav. Anomalijų ir požymių koreliacija 68
21. pav. Anomalijų aptikimo požymių svarba 71
22pav. Izoliacijos miško metodo veikimo įvertinimas pagal ROC kreivę 72
23 pav. Anomalijų vizualizacija naudojant PCA 72
24 pav. Anomalijų vizualizacija naudojant t-SNE 73

SANTRAUKA (užsienio kalba)

Palionis Lukas (2025). Automated transaction detection in the Solana network. MBA Graduation Paper. Kaunas: Vilnius University, Kaunas Faculty, Institute of Social Sciences and Applied Informatics. 70 p.

SUMMARY

The aim of this thesis is to develop an automated transaction detection method for identifying bot activity in the Solana blockchain network. The main objectives include summarizing the types of automated transactions (bots) and their impact on blockchain systems. Creating a methodology for wallet behavior analysis and anomaly detection in the Solana network using clustering and anomaly detection algorithms. Applying K-means, DBSCAN, and Isolation Forest algorithms to identify typical and atypical wallet behavior patterns; and analyzing the results and providing recommendations for improving the security and transparency of the Solana network. The study employs a quantitative approach, utilizing real transaction data collected from the Solana blockchain via the JSON-RPC API. The data is preprocessed, normalized, and stored in an SQLite database. The methodology involves feature engineering; clustering (K-means and DBSCAN); and anomaly detection (Isolation Forest), implemented using Python libraries such as Pandas, NumPy, and Scikit-learn. The effectiveness of the methods is evaluated using metrics such as the Silhouette coefficient, Calinski–Harabasz index, Precision, Recall, and F1-score.

The experimental results, based on the analysis of 100 Solana network wallets, demonstrate the ability of the proposed method to identify automated transaction patterns (including MEV, “sandwich,” and frequency trading bots) and detect anomalies. In particular, the combination of DBSCAN and Isolation Forest proves to be the most effective, achieving a favorable balance between Precision, Recall, and F1-score.

The main conclusions highlight the importance of adapting bot detection methods to the specific characteristics of the Solana network and the need for regular model updates to keep pace with the evolving nature of bot activity. The thesis also underlines the potential of the proposed methodology for real-time integration into network security systems and DeFi platforms, strengthening both security and market transparency. The work consists of 86 pages of text (excluding tables, figures, list of abbreviations, references, and appendices), 24 figures, and 28 tables.

ĮVADAS

Blockchain technologija, ypatingai kriptovaliutų srityje, pastarąjį dešimtmetį sulaukė didelio susidomėjimo tiek akademinėje bendruomenėje, tiek visuomenėje (Nakamoto, 2008). Viena iš naujausių ir sparčiausiai augančių blockchain platformų yra Solana, pasižyminti itin dideliu pralaidumu ir maža transakcijų kaina (Klarman et al., 2019). Solanos tinklas pritraukia vis daugiau vartotojų ir kūrėjų, tačiau kartu kyla ir naujų iššūkių, susijusių su tinklo saugumu ir stabilumu (Yakovenko, 2020).

Viena iš potencialių grėsmių Solanos tinklo veikimui yra automatizuoti sandoriai, generuojami išmaniųjų sutarčių ar kitų autonominių subjektų. Tokie sandoriai gali sudaryti reikšmingą dalį bendro transakcijų srauto ir potencialiai paveikti tinklo našumą bei patikimumą (Kim et al., 2018). Norint užtikrinti sklandų Solanos veikimą ir laiku pastebėti galimas anomalijas, būtina turėti efektyvius metodus automatizuotiems sandoriams aptikti ir stebėti.

Nepaisant augančio poreikio, šiuo metu trūksta specializuotų įrankių ir metodų, skirtų būtent Solanos tinklo analizei ir monitoringui (Buterin, 2014). Dauguma esamų sprendimų yra pritaikyti kitiems blockchain tinklams, tokiems kaip Ethereum ar Bitcoin, ir ne visuomet gali būti tiesiogiai perkelti į Solanos kontekstą dėl skirtingų techninių charakteristikų ir veikimo principų (G. Wood, 2014).

Temos aktuolumas:

Kriptovaliutų ir blokų grandinės technologijos užima vis svarbesnę vietą šiuolaikinėje FinTech ekosistemoje, ypač decentralizuotų finansų (angl. DeFi) sektoriuje, kuris leidžia vykdyti finansines operacijas be tarpininkų. Solana tinklas, pasižymintis dideliu operacijų greičiu ir žemomis sąnaudomis, yra viena iš sparčiausiai augančių DeFi platformų. Tačiau šio tinklo dinamiką reikšmingai veikia automatizuoti sandoriai, kuriuos vykdo robotai, iškraipantys rinkos aktyvumą ir mažinantys duomenų patikimumą. Tokie automatizuoti procesai sukuria dirbtinį likvidumą ir iškreipia kainas, apsunkindami tiek rinkos analizę, tiek tinklo skaidrumo užtikrinimą (Niedermayer, 2024).

Nepaisant plačių tyrimų Ethereum ir Bitcoin tinkluose, Solana tinklas vis dar lieka nepakankamai išanalizuotas. Automatizuotų sandorių ir jų poveikio Solanos tinklo rinkos stabilumui bei patikimumui tyrimas yra ypač svarbus, nes šios anomalijos gali daryti įtaką ne tik kainų dinamikai, bet ir investuotojų pasitikėjimui DeFi ekosistema. Be to, tokių technologijų kaip

K-means, DBSCAN ir Isolation Forest taikymas Solanos ekosistemos analizėje suteikia galimybę giliau suprasti tinklo elgseną ir rinkos manipuliacijų modelius.

Didėjant DeFi ir kitų decentralizuotų rinkų svarbai, šio darbo aktualumas taip pat kyla iš būtinybės sukurti pažangius metodus, padedančius užtikrinti tinklo skaidrumą ir stabilumą. Šio tyrimo rezultatai ne tik prisidės prie Solanos tinklo analizės, bet ir pateiks praktinius sprendimus, pritaikomus kitose DeFi platformose, stiprinant kriptovaliutų rinkos tvarumą ir patikimumą (Tianfu et al., 2024)..

Darbo problema: Kaip efektyviai aptikti automatizuotus sandorius ir botų veiklą Solana ekosistemoje, siekiant užtikrinti tinklo skaidrumą ir sumažinti rinkos manipuliacijų riziką?

Darbo hipotezė: Tinkamai parinkus ir sujungus klasterizacijos bei anomalijų aptikimo metodus, galima patikimiau ir efektyviau identifikuoti automatizuotas perlaidas Solana blokų grandinėje, palyginti su atskirų metodų taikymu.

Darbo tikslas: Sukurti ir patikrinti automatizuotą Solanos tinklo automatizuotų sandorių nustatymo modelį, pasitelkiant klasterizacijos ir anomalijų aptikimo metodus.

Darbo uždaviniai:

1. Apibendrinti automatizuotų sandorių (botų) veiklos tipus bei jų poveikį blokų grandinės sistemoms.
2. Sukurti metodologiją, skirtą piniginių elgsenos analizei ir anomalijų aptikimui Solanos tinkle, naudojant klasterizacijos ir anomalijų aptikimo algoritmus.
3. Pritaikyti K-means, DBSCAN ir Isolation Forest algoritmus, siekiant identifikuoti tipinius ir netipinius piniginių elgsenos modelius.
4. Atlikti gautų rezultatų analizę ir pateikti rekomendacijas Solanos tinklo saugumo ir rinkos skaidrumo didinimui.

Darbo objektas: Piniginės elgsenos analizė Solanos ekosistemoje, ypatingą dėmesį skiriant automatizuotų sandorių ir anomalijų aptikimui.

Darbo strūktūra: susideda iš literatūros analizės, kurioje nagrinėjami automatizuotų sandorių ir botų veiklos aptikimo metodai, ypatingą dėmesį skiriant klasterizacijos ir anomalijų aptikimo algoritmams, tokiems kaip K-means, DBSCAN ir Isolation Forest. Toliau aprašoma darbo metodologija, kurioje detalizuojama duomenų rinkimo, apdorojimo ir analizės eiga. Darbo

eksperimentinėje dalyje atliekama 100 piniginių elgsenos analizė, vizualiai pavaizduojant duomenis ir interpretuojant gautus rezultatus, siekiant nustatyti anomalijas ir netipinius piniginių veiklos modelius.

Darbo naujumas: Šiame darbe pristatomas novatoriškas požiūris į automatizuotų sandorių ir botų veiklos aptikimą Solana blokų grandinėje, pasitelkiant klasterizacijos ir anomalijų aptikimo metodų derinį. Darbas išsiskiria tuo, kad siūlomas metodas yra specializuotas būtent Solanos tinklui, atsižvelgiant į jo unikalias charakteristikas ir veikimo principus. Siūloma metodologija apima ne tik žinomus algoritmus, bet ir jų optimizavimą bei pritaikymą specifiniam kontekstui.

Naudoti literatūros šaltiniai: Darbe remiamasi moksliniais straipsniais, nagrinėjančiais blockchain technologiją Etherium ir Solanos tinkluose, botų veiklos ypatumus ir aptikimo metodus. Taip pat naudojami šaltiniai, aptariantys klasterizacijos ir anomalijų aptikimo algoritmų veikimo principus ir taikymo sritis. Analizuojami naujausi tyrimai, susiję su automatizuotų sandorių problemomis decentralizuotų finansų kontekste.

Darbo ir tyrimo metodai: Darbe taikomi mašininio mokymosi metodai, tokie kaip K-means, DBSCAN klasterizavimas ir Isolation Forest anomalijų aptikimas. Duomenų rinkimui naudojama Solanos JSON RPC API, o duomenų apdorojimui ir analizei pasitelkiamos Python bibliotekos, tokios kaip Pandas, NumPy ir Scikit-learn. Metodų efektyvumo vertinimui naudojamos metrikų sistemos, įskaitant Silhouette koeficientą, Calinski-Harabasz indeksą ir kitos.

Darbo praktinėje dalyje: Atliekamas eksperimentas su 100 Solanos tinklo piniginių duomenimis, siekiant identifikuoti automatizuotų sandorių ir botų veiklos požymius. Duomenys vizualizuojami ir interpretuojami, pateikiant išsamią klasterių analizę ir anomalijų aptikimo rezultatus. Palyginami skirtingų metodų derinių rezultatai, siekiant nustatyti optimaliausią automatizuotų sandorių aptikimo strategiją.

Tyrimo ribotumai: Darbe pripažįstama, kad naudojami duomenys atspindi tik dalį Solanos tinklo piniginių, todėl rezultatų apibendrinimas visam tinklui turi būti atliekamas atsargiai. Taip pat pabrėžiama, jog botų veiklos pobūdis nuolat kinta, todėl sukurti metodai turi būti reguliariai atnaujinami ir tobulinami.

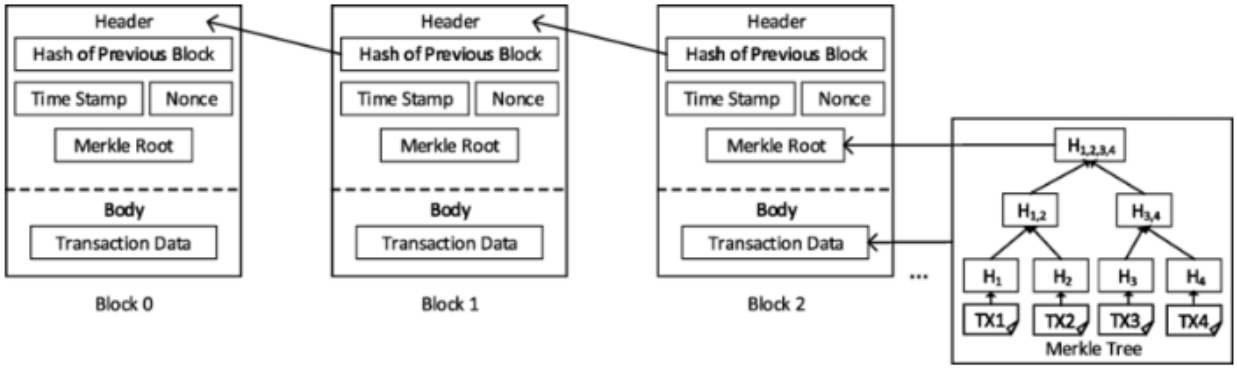
1. AUTOMATIZUOTŲ SANDORIŲ APTIKIMO METODO SOLANOS TINKLE, LITERATŪROS APŽVALGA

Šiame skyriuje pateikiama literatūros apžvalga, kurioje nagrinėjama automatizuotų sandorių, dar vadinamų botų veikla, reikšmė ir poveikis decentralizuotoms finansų sistemoms. Apžvelgiami pagrindiniai sukčiavimo tipai, susiję su botų naudojimu kriptovaliutų rinkose, bei analizuojami metodai, kurie pasitelkiami tokioms anomalijoms aptikti. Šiame skyriuje taip pat pateikiama mokslinių darbų, kuriuose tyrinėjami mašininio mokymosi algoritmai, botu aptikimui.

1.1. Blokų grandinių technologija

Blokų grandinė (angl. *blockchain*) yra paskirstytų duomenų saugojimo technologija, leidžianti saugiai ir skaidriai saugoti, valdyti bei dalintis informacija tarp įvairių šalių nenaudojant centralizuoto valdymo. Ši technologija remiasi kriptografiniais metodais bei konsensuso algoritmu, užtikrinančiu sistemos saugumą ir patikimumą (Zheng. Z et al. 2018).

Blokų grandinės esmę sudaro duomenų blokai, kurie sujungti į grandinę, naudojant kriptografinius metodus (žr. 1 paveikslą). Kiekvienas naujas blokas yra prijungiamas prie grandinės, įtraukiant ne tik naujus duomenis, bet ir nuorodą (angl. *hash*) į prieš tai buvusį bloką (Nakamoto, 2008). Norint pakeisti ar suklastoti bet kokią bloko informaciją, reikėtų iš naujo pergeneruoti visus po jo sekančius grandinės blokus, o tai yra praktiškai neįmanoma dėl milžiniškų skaičiavimo resursų poreikio.



Šaltinis: Nakamoto, S. (2008). *Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Review*.

1pav. Blokų grandinės struktūra

Blokų grandinės struktūra, kur kiekvienas blokas susideda iš antraštės ir kūno. Antraštėje yra ankstesnio bloko hash, laiko žyma, nonce ir Merkle šaknis, kuri reprezentuoja visų transakcijų hash. Kūne saugomi transakcijų duomenys. Merkle medis leidžia efektyviai patikrinti, ar tam tikra transakcija yra bloke, naudojant hierarchinę hash struktūrą, kurioje atskirų transakcijų hash sujungiami į galutinę Merkle šaknį.

Priklausomai nuo taikymo srities, blokų grandinės gali būti viešos (angl. public), kai prie jų gali prisijungti bet kas, arba privačios (angl. private), skirtos naudoti uždarame vartotojų rate, pavyzdžiui, įmonės viduje (Wüst ir Gervais, 2018). Taip pat jos gali būti be leidimų (angl. permissionless), kai visi dalyviai turi vienodas teises, arba su leidimais (angl. permissioned), kai yra kontroliuojama, kas ir kokį vaidmenį atlieka sistemoje.

Viena pagrindinių blokų grandinės savybių yra decentralizuotas valdymas. Skirtingai nei tradicinėse centralizuotose sistemose, blokų grandinėje nėra vieno valdančio autoriteto ar tarpininko. Bach, Mihaljevic ir Zagar (2018), teigia, kad visi tinklo mazgai (angl. nodes) kompiuteriai, palaikantys blokų grandinės veikimą, kartu dalyvauja konsensuso procese ir patvirtina kiekvieną naują bloką bei jame esančius duomenis. Populiariausi konsensuso algoritmai yra proof-of-work (PoW), kurį naudoja Bitcoin, ir proof-of-stake (PoS), kurį naudoja Ethereum 2.0 ir kitos naujesnės blokų grandinės.

Kita svarbi blokų grandinių savybė - duomenų nepakeičiamumas (angl. immutability). Kadangi blokų grandinė yra paskirstytas registras, kiekvienas tinklo mazgas turi savo grandinės kopiją. Bandymas pakeisti vieno mazgo duomenis būtų atmestas kitų mazgų, kadangi jų kopijos neatitiktų. Tai praktiškai neįmanoma padaryti didelėje decentralizuotoje sistemoje. Duomenų įvestis į blokų grandinę taip pat yra negrįžtama, įrašius duomenis jų nebegalima ištrinti ar koreguoti (Drescher, 2017).

Blokų grandinės technologija turi daug potencialių pritaikymo sričių, ne tik kriptovaliutoms. Casino, Dasaklis ir Patsakis (2019) akcentuoja, kad blokų grandinės technologija tinka visur, kur reikalingas skaidrus, saugus ir decentralizuotas duomenų valdymas tarp nepasitikėjimo šalių. Blokų grandinės gali būti taikomos ir išmaniosiose sutartyse (angl. smart contracts) - kompiuteriniuose protokoluose, automatiškai vykdančiuose sutarčių sąlygas, kai įvykdomos iš anksto apibrėžtos sąlygos. Tai leidžia sutarčių sudarymą ir vykdymą perkelti į decentralizuotą ir saugią aplinką.

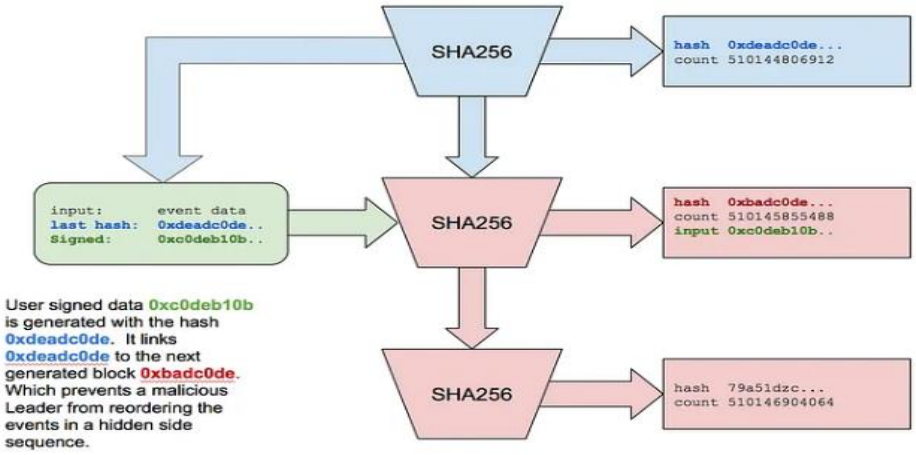
Nepaisant blokų grandinių pranašumų, šiai technologijai būdingi ir tam tikri iššūkiai. Vienas didžiausių yra mastelio keitimo (angl. *scalability*)- gebėjimas apdoroti didelius duomenų kiekius ir transakcijas sparčiai augant vartotojų skaičiui (Yli-Huumo. et al 2016). Bitcoin ir Ethereum blokų grandinės gali įvykdyti atitinkamai tik apie 7 ir 30 transakcijų per sekundę, kas yra pastebimai lėčiau, nei centralizuotos mokėjimo sistemos. Sprendimai šiai problemai apima blokų grandinių fragmentavimą (angl. *sharding*), šalutinės grandinės (angl *sidechains*) ir kitus optimizavimo metodus (Ferdous. et al 2020).

Kiti iššūkiai yra privatumo užtikrinimas, neteisėtos veiklos prevencija bei reguliavimo ir valdymo klausimai. Nors duomenys blokų grandinėse yra pseudonimiški, analitiniais metodais įmanoma susieti transakcijas su realiu pasauliu (Reid ir Harrigan, 2013). Neteisėta veikla, tokia kaip pinigų plovimas, taip pat išnaudoja blokų grandinių decentralizaciją. To prevencijai gali pasitarnauti įvairūs kontrolės mechanizmai. Pagaliau, šiuo metu blokų grandinės dar nėra pakankamai reguliuojamos teisinėmis priemonėmis, o jų valdymas kelia iššūkių dėl decentralizuotos prigimties (Zheng et al., 2018).

Viena iš naujausių blokų grandinių platformų yra Solana, sukurta siekiant išspręsti mastelio keitimo problema. Solana išsiskiria savo unikaliu Proof-of-History (PoH) konsensuso algoritmu, kuris leidžia blokų grandinei turėti įgimtą "laikrodį" ar laiko nuorodą (Möser et al., 2017). Tradicinėse blokų grandinėse, tokiose kaip Bitcoin ar Ethereum, laikas nėra įgimtas pačiame protokole, o blokų kūrimo greitis priklauso nuo konsensuso mechanizmo, tokio kaip Proof-of-Work (PoW). Tai gali lemti nepastovų blokų kūrimo greitį ir sunkumus nustatant įvykių seką.

PoH algoritmas sprendžia šią problemą, naudodamas kriptografinę laiko žymėjimo funkciją. Ši funkcija generuoja unikalią išvestį, vadinamą "laikrodžio žyme", kuri gali būti naudojama įrodyti, kad tam tikras įvykis įvyko po arba prieš kitą įvykį, nepriklausomai nuo to, kiek laiko praėjo tarp šių įvykių (Yakovenko, A. 2018). Tai leidžia Solana blokų grandinei turėti įgimtą laiko sampratą ir efektyviai apdoroti transakcijas be papildomų vėlinimų.

PoH veikia kaip kriptografinė laiko žyma, leidžianti mazgams susitarti dėl įvykių sekos nenaudojant daug ryšio resursų. Tai pasiekiama naudojant rekursinę vienkryptę funkciją, pvz., SHA-256 maiša, kuri nuolat generuoja naujus blokus su laiko žymėmis. Kiekviena laiko žyma priklauso nuo ankstesnės žymos, sukuriant grandinę, kuri gali būti naudojama kaip laiko nuoroda (2 pav.).



Šaltinis: Yakovenko, A. (2018). Solana: A new architecture for a high performance blockchain

2 pav. Proof-of-History grandinė su laiko žymėmis

PoH grandinė yra generuojama centralizuotai, tačiau gali būti patikrinta decentralizuotai. Tai reiškia, kad kiekvienas Solana tinklo mazgas gali savarankiškai patikrinti PoH grandinės tikslumą ir naudoti ją kaip patikimą laiko šaltinį, nepriklausomai nuo tinklo vėlinimų ar atskirų mazgų gedimų (Solana Documentation, 2023).

Naudojant PoH, Solana gali pasiekti ypač didelį transakcijų pralaidumą, nes transakcijos gali būti apdorojamos lygiagrečiai, o ne nuosekliai. Tai leidžia Solana tinklui apdoroti tūkstančius transakcijų per sekundę, kas yra žymiai daugiau nei tradicinės blokų grandinės (Yakovenko, A. 2018).

Taigi, Proof-of-History yra kertinė Solana inovacija, leidžianti blokų grandinei turėti įgimtą laiko sampratą ir pasiekti ypač aukštą našumą. Tai išskiria Solana iš kitų blokų grandinių ir suteikia jai unikalų pranašumą skalabilumo ir efektyvumo srityse.

Solana taip pat išsiskiria savo unikalia blokų struktūra. Vietoj įprastų blokų, Solana naudoja duomenų struktūrą, pavadintą Gulf Stream, kuri leidžia mazgams iš anksto patvirtinti būsimas transakcijas ir taip sumažinti vėlavimą. Tai leidžia Solana pasiekti itin aukštą našumą - iki 65 000 transakcijų per sekundę, kai tuo tarpu Ethereum pajėgumas siekia tik apie 30 transakcijų per sekundę (Yakovenko, A. 2018) (žr. 1 lentelę).

1 lentelė

Solana ir kitų blokų grandinių našumo palyginimas

Blokų grandinė	Konsensuso mechanizmas	Transakcijų greitis (TPS)	Vėlavymas	Mastelis	Pagrindinė architektūra
Solana	Proof of History (PoH) + PoS	Iki 65 000	Subsekundinė	Aukštas	Optimizuota su PoH, kad būtų užtikrintas didelis pralaidumas ir mažas vėlavimas
Ethereum	Proof of Stake (PoS)	Iki 30	Aukštas	Vidutinis	Palaiko išmaniąsias sutartis, naudoja „sharding“ Ethereum 2.0
Bitcoin	Proof of Work (PoW)	Iki 7	Aukštas	Žemas	Linijinė blokų grandinė, pritaikyta saugumui, o ne greičiui
Avalanche	Snowball (Hibridinė)	iki 4500	Žemas	Aukštas	DAG pagrindu su blokų grandinės integracija
IOTA	Proof of Authority (PoA) + Koordinatorius	iki 1000	Žemas	Aukštas	DAG struktūra (Tangle), tinkama IoT

Šaltinis: Kahmann, F., Honecker, F., Dreyer, J., Fischer, M., & Tönjes, R. (2023). *Performance Comparison of Directed Acyclic Graph-Based Distributed Ledgers and Blockchain Platforms*, 12(12), 257

Solanos blokų grandinė palaiko išmaniąsias sutartis, parašytas Rust, C++ ar C programavimo kalbomis, kas suteikia lankstumą kūrėjams (Solana Documentation, 2023). Tai leidžia kurti decentralizuotas programas (angl. *dApps*) įvairiose srityse, įskaitant decentralizuotus finansus (angl. *DeFi*), žaidimus, socialinę mediją ir kt.

Botų aptikimas ir prevencija yra viena svarbiausių sričių užtikrinant Solana tinklo saugumą ir stabilumą. Tam gali būti naudojami įvairūs metodai, pradedant nuo paprastų taisyklėmis grįstų algoritmų iki sudėtingų mašininio mokymosi modelių (Yakovenko., 2018). Tačiau botų aptikimas Solana tinkle vis dar yra mažai ištirta sritis, reikalaujanti tolesnių tyrimų ir naujų sprendimų.

Blokų grandinių technologija siūlo decentralizuotą, saugų ir skaidrų būdą duomenų saugojimui ir dalijimusi, tačiau susiduria su skalabilumo, privatumo ir reguliavimo iššūkiais. Solana blokų grandinė siekia spręsti mastelio keitimo problemas naudodama inovatyvius sprendimus, tokius kaip Proof-of-History konsensuso algoritmas ir Gulf Stream duomenų struktūra, taip pasiekdama ženkliai didesnį našumą nei kitos blokų grandinės.Vis dėlto, Solana susiduria su botų veiklos problema, kuri tampa ypač aktuali dėl šios blokų grandinės aukšto našumo ir mažų transakcijų mokesčių.

Apibendrinant, blokų grandinių technologija, įskaitant Solana, turi didelį potencialą, tačiau susiduria su reikšmingais iššūkiais, iš kurių botų veikla yra viena svarbiausių. Tolimesni tyrimai ir inovatyvūs sprendimai šioje srityje bus būtini siekiant užtikrinti blokų grandinių patikimumą ir tvarumą ateityje.

1.2 Solanos transakcijų duomenys

Solanos tinkle transakcijos generuoja įvairius duomenis, kurie yra esminiai analizuojant automatizuotų botų veiklą decentralizuotose finansų (angl. *DeFi*) programose. Šios programos pritraukia botus, kurie naudojasi rinkos neefektyvumu, vykdydami didelės spartos operacijas, tokias kaip arbitražas ar likvidumo manipuliavimas (Ye Wang et al., 2023). Kad būtų galima nustatyti tokią veiklą, būtina išsamiai analizuoti Solanos transakcijų struktūrą, vidinių instrukcijų mechanizmus ir sąskaitų valdymą.

Transakcijų Struktūra

Kiekviena Solanos transakcija turi specifinę struktūrą, kurią sudaro įvairūs komponentai, būtini operacijoms atlikti ir transakcijos kilmei patikrinti, Yakovenko (2018) teigia:

- Signatures yra base58 koduoti skaitmeniniai parašai, susieti su transakcijos accountKeys masyve esančiais viešaisiais raktais. Pirmasis parašas veikia kaip pagrindinis transakcijos identifikatorius, leidžiantis patvirtinti jos kilmę ir autentiškumą. Transakcijų analizėje šis parašas yra esminis autentiškumo įrodymas, galintis padėti identifikuoti botus, kurie gali generuoti kelias transakcijas iš to paties rakto.

Message yra pagrindinė transakcijos informacija, kurioje nurodomos sąskaitos, parašai ir instrukcijos:

- Account Keys yra viešųjų raktų sąrašas, kuriame nurodytos visos transakcijoje naudojamos sąskaitos, įskaitant inicijatorių, gavėją ir programų sąskaitas. Botai, dažnai atliekantys pasikartojančias transakcijas, paprastai sąveikauja su tomis pačiomis

sąskaitomis arba programomis, todėl šių raktų sąrašas gali būti naudojamas botų elgsenai aptikti.

- Header nustato, kiek parašų reikia ir kurios sąskaitos gali būti tik skaitomos ar naudojamos skaitymo-rašo režimu. Ši informacija yra svarbi siekiant atpažinti botus, kurie vykdo pakartotinius veiksmus naudodami tas pačias sąskaitas.
- Recent Blockhash yra hashas iš neseniai patvirtinto bloko, užtikrinantis transakcijos aktualumą ir apsaugantis nuo pakartotinių atakų. Kadangi botai dažnai veikia labai aukštu dažniu, šis hashas padeda identifikuoti anomalias pasikartojančias transakcijas, kurios gali rodyti botų veiklą.
- Instructions apibrėžia, kokios operacijos turi būti atliktos per transakciją, tokios kaip žetonų pervedimas ar programų kvietimai.
- Program ID Index nurodo programos sąskaitą, kuri vykdo instrukciją.
- Accounts yra sąskaitų indeksai, perduodami programai, pagal kuriuos vykdomos transakcijos.
- Data yra programos specifiniai duomenys, užkoduoti base58, kurie detalizuoja operacijos parametrus.

Solanos transakcijų struktūra suteikia analitikams galimybę aptikti botų elgesį, atsižvelgiant į jų sąveiką su tam tikromis programomis, dažnų sąskaitų naudojimą ir operacijų pobūdį (Zwang, Somin, Pentland, ir Altshuler, 2018).

Vidinės Instrukcijos

Vidinės instrukcijos, dar vadinamos (angl. *cross-program invocations*), leidžia programoms sąveikauti su kitomis Solanos blokų grandinės programomis, užtikrindamos sudėtingų operacijų vykdymą vienoje transakcijoje. Šis mechanizmas yra ypač svarbus analizuojant sudėtingus botų veiksmus DeFi erdvėje Kitzler (2023) teigia, kad botai dažnai atlieka kelias operacijas, pavyzdžiui, žetonų apsikeitimus, likvidumo koregavimą ar arbitražą. Pagrindiniai kiekvienos vidinės instrukcijos komponentai apima:

- Index nurodo transakcijos instrukcijos poziciją, iš kurios buvo inicijuota vidinė instrukcija.
- Instructions sąrašas, vykdomas vidinės transakcijos metu, įskaitant:

- Program ID Index: Program ID Index accountKeys masyve nurodo programos sąskaitą, kuri vykdo instrukciją.
- Accounts nurodo sąskaitų indeksų sąrašą, kurie perduodami programai.
- Data yra instrukcijai būdingi duomenys, užkoduoti base58, pateikiantys įvesties parametrus.

Vidinių instrukcijų analizė suteikia galimybę detaliau suprasti transakcijų logiką, ypač kai botai naudoja kelias programas vienos transakcijos metu. Analizuojant šias sekas galima aptikti specifinius elgsenos modelius, būdingus botams (Piet, Fairoze, ir Weaver, 2022).

Sąskaitos

Solanos sąskaitos yra pagrindiniai duomenų saugojimo ir sąveikos vienetai pagal Yakovenko (2018), esminiai transakcijų analizei. Kiekviena sąskaita turi šiuos specifinius atributus:

- Data Storage sąskaitos turi baitų masyvą (angl. *data*), kuriame saugoma sąskaitos būsena arba vykdomasis kodas. Botai gali naudotis šiuo saugojimo mechanizmu, siekdami kaupti informaciją, susijusią su rinkos manipuliavimu arba dažnomis operacijomis.
- Ownership reiškia, kad kiekviena sąskaita turi priskirtą savininką - programą, kuri gali modifikuoti sąskaitą. Tik savininko programa gali atlikti pakeitimus, todėl šis elementas užtikrina sąskaitos duomenų saugumą ir kontrolę.
- Executable Flag, ar sąskaita turi vykdomąjį kodą. Sąskaitos su vykdomumo teisėmis dažniausiai atstovauja programas, o nevykdomosios sąskaitos saugo vartotojų arba operacijų duomenis.
- Lamports kiekviena sąskaita turi balansą lamportais, kurie yra mažiausia SOL dalis (1 SOL = 1 milijardas lamportų), ir šis balansas yra stebimas lamports lauke. Botų identifikacijai svarbu sekti lamportų balansus, nes dažnos manipuliacijos ar arbitražas gali rodyti anomalias balanso fluktuacijas.

Transakcijų Metaduomenys

Solanos tinkle, priešingai nei Ethereum, nėra dujų sistemos. Vietoj jos naudojamas Instruction Memory Limit – kiekviena transakcija turi specifinius atminties ribojimus, kurie turi įtakos operacijos sudėtingumui, kurį galima atlikti per vieną transakciją. Šie metaduomenys yra

ypač svarbūs identifikuojant didelių resursų poreikio veiksmus, kurie gali rodyti botų veiklą, nes botai dažnai vykdo operacijas, reikalingas aukšto atminties kiekio (Qin et al., 2021).

Solanos transakcijų struktūra, vidinės instrukcijos ir sąskaitų atributai suteikia tvirtą pagrindą botų veiklos analizei. Botai, veikiantys DeFi programose, dažnai naudojasi programų iškvietimų kombinacijomis, dažnais sąskaitų veiksmais ir pasikartojančiomis vidinėmis instrukcijomis, kurios ilgainiui atskleidžia jų veiklos modelius. Fokusuojantis į šiuos komponentus ir jų metaduomenis, galima atskirti botų veiklą nuo įprastų vartotojų sąveikų, prisidedant prie skaidresnės ir sąžiningesnės DeFi ekosistemos .

1.3 Botų apibrėžimas

Botai yra automatizuotos programos, sukurtos atlikti iš anksto apibrėžtas užduotis įvairiose srityse, įskaitant decentralizuotų finansų (angl. *DeFi*) platformas. Botai gali būti suprogramuoti vykdyti įvairias funkcijas, nuo paprastų ir pasikartojančių užduočių automatizavimo iki sudėtingų algoritmų, pagrįstų dirbtinio intelekto ir mašininio mokymosi principais, įgyvendinimo (Zheng et al. 2018).

Botų naudojimas sparčiai išaugo per pastaruosius kelerius metus, apimant tokias sritis kaip klientų aptarnavimas, e-prekyba, socialinė žiniasklaida ir finansinės paslaugos. Klientų aptarnavimo srityje pokalbių botai naudojami atsakyti į dažnai užduodamus klausimus, teikti rekomendacijas ir padėti klientams naršyti svetaines. E-prekyboje botai gali automatizuoti atsargų valdymą, kainų optimizavimą ir produktų rekomendacijas. Socialinės žiniasklaidos platformose botai naudojami turiniui kurti, sąveikai su vartotojais palaikyti ir netgi dezinformacijai skleisti (Xie et al., 2019).

Finansų sektoriuje botai sukėlė revoliuciją prekyboje, leisdami automatizuoti sudėtingas strategijas ir vykdyti sandorius milisekundžių greičiu (Wüst ir Gervais, 2018). Algoritminėje prekyboje botai analizuoja rinkos duomenis, identifikuoja modelius ir priima prekybos sprendimus pagal iš anksto apibrėžtus algoritmus. Aukšto dažnio prekybos (angl. *HFT*) botai pasinaudoja minimaliais kainų skirtumais ir vykdo didelius sandorių kiekius, siekdami gauti pelno iš nedidelių rinkos neefektyvumo (Bach, Mihaljevic ir Zagar, 2018).

Atsiradus blokų grandinės technologijai ir decentralizuotiems finansams, botams atsirado nauja pritaikymo sritis. DeFi ekosistemose, pagrįstose išmaniosiomis sutartimis ir

decentralizuotomis aplikacijomis (dApps), botai gali automatizuoti įvairias finansines operacijas, tokias kaip prekybą, skolinimą ir likvidumo tiekimą (Mingxiao et al., 2017). Botai tapo neatsiejama DeFi ekosistemos dalimi, pasinaudodami unikaliais blokų grandinės technologijos privalumais, tokiais kaip decentralizacija, nepatikimumas ir programuojamumas (Drescher, 2017).

Tradicinėse finansų rinkose botai plačiai naudojami automatizuoti prekybai ir vykdyti sudėtingas algoritminės prekybos strategijas. Tačiau decentralizuotų finansų eroje botai įgavo dar svarbesnį vaidmenį, nes jie gali sąveikauti su išmaniosiomis sutartimis ir dApps be žmogaus įsikišimo. Botai gali stebėti rinkos sąlygas, analizuoti on-chain duomenis ir priimti prekybos sprendimus pagal sudėtingus algoritmus (Niedermayer, Saggese ir Haslhofer. 2024).

Nors botai suteikia daug privalumų, kaip didesnis efektyvumas, mažesnės sąnaudos ir 24/7 veikimo galimybės, jie taip pat kelia įvairių iššūkių ir rizikų. Yli-Huumo et al. (2016), teigimu vienas iš pagrindinių susirūpinimų yra botų galimybė trikdyti rinkas ir sukelti finansinį nestabilumą. Botai taip pat, gali būti naudojami neteisėtoms veikloms, tokioms kaip rinkos manipuliacijos, pinigų plovimas ir neteisėtas prekių pardavimas (Ferdous et al., 2020). Įvairūs botų tipai, jų pagrindinės funkcijos ir su jais susijusios rizikos kriptovaliutų rinkoje (žr. 2 lentelę).

2 lentelė

Botų tipai, funkcionalumas ir susijusios rizikos kriptovaliutų rinkose

Botų tipai	Funkcionalumas	Rizikos ir iššūkiai
Aukšto dažnio prekybos (HFT) botai	Įvykdo daug sandorių per milisekundes, siekiant pasinaudoti mažais kainų neefektyvumais.	Rinkos nestabilumas, likvidumo sumažėjimas, galimos sisteminės rizikos.
Arbitražo botai	Nustato ir išnaudoja kainų skirtumus tarp skirtingų DEX arba tokenų porų pelno siekimui.	Kainų svyravimai, galimi front-running išpuoliai ir rinkos manipuliavimas.
Rinkos formavimo botai (MEV)	Užtikrina likvidumą DEX ir gauna pajamas iš spredų.	Dirbtinai padidintas likvidumas, galimi konfliktai su organiniais rinkos dalyviais.
Likvidumo ūkininkavimo botai	Automatizuoja staking arba likvidumo užtikrinimą, siekiant maksimaliai padidinti pajamas DeFi platformose.	Paskatų išnaudojimas, staking taisyklių pažeidimas ir atlygių koncentracija.
Sniping botai	Atpažįsta ir vykdo sandorius palankiais momentais, pavyzdžiui, naujų tokenų paleidimo metu.	Front-running, nesąžininga praktika ir kitų dalyvių nepalankumas.
"Pump-and-Dump" botai	Koordinuoja ir vykdo rinkos manipuliavimo schemas siekiant pelno.	Apgaulė, nuostoliai mažmeniniams prekybininkams ir rinkos pasitikėjimo sumažėjimas.
Sandwich botai	Išnaudoja pavedimų sekas, įterpdami savo sandorius tarp pirkimo ir pardavimo užsakymų (sandwich attack).	Pavedimų manipuliavimas, nesąžininga praktika ir didesni sandorių kaštai kitiems dalyviams.
Front-Running botai	Nustato dar neįvykdytus pavedimus mempool'e ir pateikia savo sandorį su pranašumo prioritetu.	Nesąžininga praktika, didesni sandorių kaštai aukoms ir pasitikėjimo rinkos sistema mažėjimas.

Šaltinis: sudaryta autoriaus.

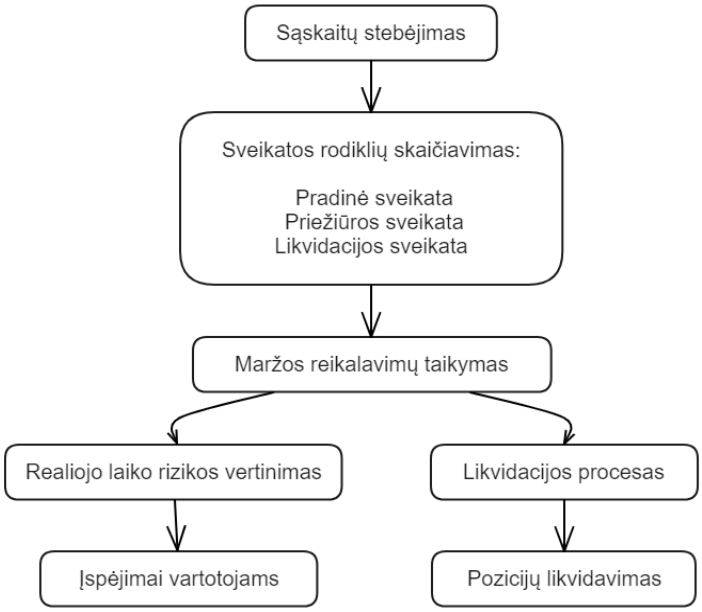
Nepaisant šių iššūkių, botai ir toliau atlieka svarbų vaidmenį besivystančioje DeFi ekosistemoje. Ethereum tinkle botai plačiai naudojami sandoriams decentralizuotose biržose (DEX), tokiose kaip Uniswap ir Sushiswap, automatizuoti (Reid ir Harrigan, 2013). Šie botai stebi rinkos sąlygas ir vykdo prekybos strategijas, pasinaudodami kainų skirtumais tarp skirtingų decentralizuotų keityklų ar žetonų porų. Tačiau botų veikla Ethereum DeFi ekosistemoje sukėlė susirūpinimą dėl rinkos manipuliavimo, nesąžiningos prekybos praktikos ir sisteminės rizikos (Möser et al., 2017).

Solana blockchain išsiskiria savo didelės spartos infrastruktūra, maža operacijų kaina ir gebėjimu apdoroti didelį kiekį operacijų per sekundę (Casino,Dasaklis, ir Patsakis, 2019). Šios savybės padarė Solanos blokų grandinės tinką patrauklią botams, siekiantiems pasinaudoti sparčiai besivystančia DeFi ekosistema. Solana tinkle veikia kelios populiarios DEX, tokios kaip Raydium, Jupiter ir Serum, kuriose botai aktyviai dalyvauja.

Dėl savo greičio ir efektyvumo Solana tapo pagrindiniu blockchain tinklu aukšto dažnio prekybai ir sudėtingoms algoritminės prekybos strategijoms (Ferdous, Chowdhury, Hoque ir Colman., 2020). Botai Solana ekosistemoje gali vykdyti įvairias funkcijas, įskaitant arbitražą tarp skirtingų decentralizuotų keityklų (angl. *DEX*), rinkos formavimą, teikiant likvidumą, ir dalyvavimą likvidumo ūkininkavimo programose, tokiose kaip Raydium's Fusion Pools (Reid ir Harrigan, 2013).

Nors specifinių mokslinių tyrimų apie botų veiklą Solana ekosistemoje yra nedaug, galime pritaikyti bendresnes įžvalgas iš botų elgsenos kitose DeFi platformose. Tyrėjai nustatė, kad botai gali sudaryti didelę dalį prekybos apimties DEX, tokiose kaip Uniswap, ir turėti reikšmingą įtaką kainų dinamikai (Möser et al., 2017). Botai taip pat buvo siejami su rinkos manipuliacijomis, tokiomis kaip " pump-and-dump " schemomis ir pasinaudojimu išmaniųjų sutarčių pažeidžiamumais (Yakovenko., 2018).

Solanos projektai, tokie kaip Mango Markets, įgyvendino priemones, skirtas apsisaugoti nuo botų manipuliacijų, pavyzdžiui, didelio dažnio prekybos) maržos reikalavimus. Taip pat, buvo sukurtos botų stebėjimo priemonės, siekiant geriau suprasti jų elgesį Solana DeFi protokoluose (Mango Markets, 2021).



Šaltinis: Sudaryta autoriaus pagal Mango Markets. (2021). High Frequency Trading and the Mango Margin Account. Mango Markets Docs.

3 pav. Rizikos valdymo sistema Mango Markets platformoje

Mango Markets rizikos valdymo sistema nuolat stebi vartotojų paskyras, vertina jų turtą ir įsipareigojimus, naudodama svorinius skaičiavimus, kad nustatytų pagrindinius sveikatos rodiklius: Pradinė sveikata, Palaikymo sveikata ir Likvidavimo sveikata. Šie rodikliai lemia, ar paskyra gali atidaryti naujas pozicijas, ar reikia koreguoti maržą, ar būtina likvidacija. Sistema taiko griežtus maržos reikalavimus, kad būtų išvengta per didelio sverto, kuris dažnai išnaudojamas aukšto dažnio prekybos botų. Ji vykdo realaus laiko rizikos vertinimus, siekdama identifikuoti rizikingas paskyras, ir prireikus aktyvuoja įspėjimus arba likvidavimo procesus. Likvidacija stabilizuoja paskyras, parduodant turtą ir koreguojant įsipareigojimus, kol sveikatos rodiklis pasiekia priimtiną lygį, taip užtikrinant rinkos stabilumą ir sumažinant manipuliuojančių botų poveikį.

Nors botai gali padidinti rinkos efektyvumą ir likvidumą, jie taip pat gali sukelti rinkos sutrikimus ir kelti sisteminę riziką. Siekiant užtikrinti tvarią ir sąžiningą DeFi ekosistemos plėtrą, būtina toliau tirti botų elgesį, kurti pažangius aptikimo įrankius ir įgyvendinti rizikos mažinimo strategijas. Tai apima on-chain duomenų analizę, mašininio mokymosi metodų taikymą ir rizikos

mažinimo priemonių, tokių kaip sandorių mokesčiai ir išmaniųjų sutarčių auditas, įgyvendinimą (Kahmann et al., 2023).

Apibendrinant galima teigti, kad botai atlieka svarbų, nors ir prieštaringą vaidmenį Solana DeFi ekosistemoje. Nors specifinių tyrimų apie botus Solana tinkle gali būti mažai, galime pasimokyti iš botų elgsenos kitose DeFi platformose ir pritaikyti šias žinias Solana kontekstui. Kadangi Solana ekosistema ir toliau sparčiai vystosi, svarbu atidžiai stebėti botų veiklą ir kurti atitinkamas rizikos valdymo strategijas.

1.4 Esamos botų aptikimo sistemos

Botų aptikimas tapo svarbiu uždaviniu decentralizuotų finansų (DeFi) ekosistemose, siekiant užtikrinti rinkos vientisumą ir apsaugoti vartotojus nuo kenkėjiškos veiklos. Įvairūs tyrėjai ir praktikai pasiūlė botų aptikimo sistemas, skirtas stebėti ir identifikuoti botų elgesį blockchain tinkluose, tokiuose kaip Ethereum, Bitcoin ir Solana.

Šiuo metu naudojamos botų aptikimo sistemos dažniausiai taiko griežtai apibrėžtas euristikas ir orientuojasi į itin pelningus sandorius, tokius kaip maksimalios išgaunamos vertės (angl. *MEV*) sandorius. Dvi labiausiai paplitusios sistemos, MEV-inspect pagal Zheng (2018) ir (Eigenphi, 2022) pagal, daugiausia dėmesio skiria dažniausiai pasitaikantiems MEV tipams, būtent arbitražui, sandwitch atakoms (taip pat vadinamoms sumuštiniais) ir likvidavimui.

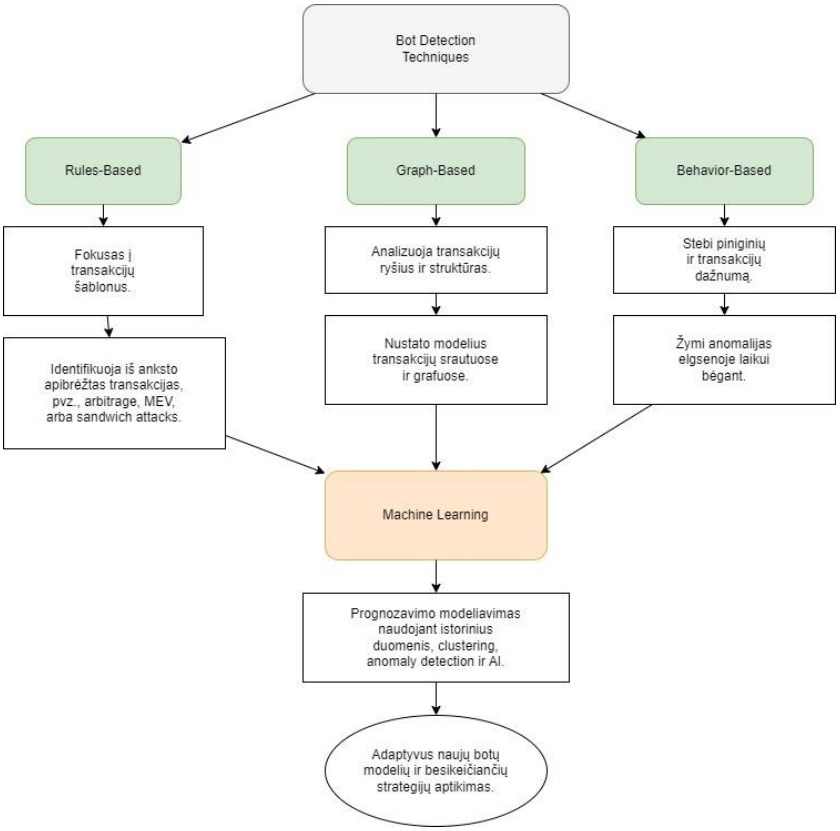
Arbitražas išnaudoja kainų skirtumus decentralizuotose biržose, siekiant gauti pelno. Sandwitch atakos siunčia sandorį, kad padidintų kainą, leidžia kitam pirkėjui pirkti už aukštą kainą, o tada siunčia dar vieną sandorį, kad parduotų tai, kas buvo nupirkta anksčiau, ir gautų pelno. Likvidavimas Ethereum tinkle apima įkeistų aktyvų pardavimą skolai padengti. Šis pardavimas paprastai vyksta pirkėjui palankiu kursu, todėl tai yra patrauklus sandoris.

MEV-inspect yra Flashbots komandos sukurtas įrankis, skirtas aptikti ir kategorizuoti MEV sandorius Ethereum blokų grandinėje. Tai atliekama analizuojant sandorių pėdsakus kiekviename Ethereum bloke, kuriame sandoriai identifikuojami kaip susiję su MEV arba ne. Procesas apima išsamių duomenų išgavimą iš kiekvieno bloko, su atskirais sandoriais susijusių pėdsakų vertinimą naudojant taikomųjų programų dvejetainę sąsają (angl. *ABI*), siekiant identifikuoti su apsikeitimais ir likvidavimu susijusias funkcijas, ir vėliau identifikuoti įvairias MEV formas, įskaitant tris pagrindines: arbitražus, likvidavimo įvykius arba sumuštinių atakas (Eigenphi, 2022). MEV-

inspect yra pagrįstas taisyklėmis ir leidžia pridėti naujas specifikacijas bei būdus klasifikuoti sandorius kaip tam tikro tipo MEV, tačiau norint pridėti naujus tipus reikia žinoti apie konkrečius protokolus, su kuriais sąveikaujama.

EigenPhi (2022)vertina MEV botų paplitimą Ethereum ir Binance Smart Chain tinkluose. Nors Eigenphi sistema yra uždaro kodo, jie atskleidė savo metodą savo svetainėje (Wüst ir Gervais, 2018). Atsižvelgiant į sudėtingus ir potencialiai beribius sandorių struktūrų variantus, vien tik iš anksto nustatytų sandorių šablonų atitikimo nepakanka naujiems MEV šablonams atskleisti. Norėdama efektyviai identifikuoti arbitražą sandoriuose, EigenPhi žengia žingsnį toliau nei MEV-inspect ir nustato universalią sandorių abstrakciją, kad galėtų aptikti bendresnius arbitražus.

Chen et al. (2018) pasiūlė metodą, pagrįstą mašininiu mokymusi, siekiant aptikti prekeivio botus, kurie manipuliuoja rinka Ethereum pagrindu veikiančiose decentralizuotose biržose. Jų metodas naudoja įvairias sandorių ir piniginių savybes, tokias kaip sandorių dažnumas, sandorio dydis ir sąveikos šablonai, kad identifikuotų įtartinus botų valdomus piniginius (žr. 4 paveikslą). Šios technikos sudaro tvirtą sistemą, leidžiančią aptikti botų veiklą decentralizuotų finansų ekosistemose.



Šaltinis: Sudaryta autoriaus pagal Chen, W., Zheng, Z., Cui, J., Ngai, E., Zheng, P., ir Zhou, Y. (2018). Detecting ponzi schemes on ethereum: Towards healthier blockchain technology. In Proceedings of the 2018 World Wide Web Conference.

4 pav. Botų Aptikimo Technikos

Kiti tyrėjai nagrinėjo botų veiklą ne tik Ethereum tinkle, bet ir kituose blockchain tinkluose. Autoriai Hu et al. (2019) analizavo botų elgesį Bitcoin tinkle, siekdami aptikti pinigų plovimo veiklą. Jie naudojo klasterizavimo ir anomalijų aptikimo technikas, kad identifikuotų įtartinus piniginių šablonus ir sandorių grafus, būdingus pinigų plovimo botams.

Daian et al. (2020) analizavo decentralizuotų biržų pažeidžiamumą, susijusį su "front-running" atakomis, kai didelės dažnio prekybos (angl. *HFT*) botai išnaudoja blockchain transakcijų viešumą ir tinklo vėlavimą, siekdami gauti pelną. Jų tyrimas atskleidė, kad HFT botai gali manipuliuoti rinkos dinamika ir kelti riziką DEX naudotojams bei konsensuso stabilumui, taip pat išskyrė pagrindinius iššukius botų aptikime (žr. 3 lentelę).

3 lentelė

Pagrindiniai iššūkiai botų aptikime

Iššūkiai	Poveikis	Siūlomi sprendimai
Besikeičiantis botų elgesys	Sunkiau aptikti naujas botų strategijas	Adaptyvūs machine learning modeliai
Didelės transakcijų apimtys	Didėjantys ištekliai skaičiavimo resursams	Efektyvūs graph-based algoritmai
Ribotas duomenų skaidrumas	Sunkumai identifikuojant HFT botus tarp platformų	Bendradarbiavimas tarp blockchain bendruomenių

Šaltinis: Daian, P., Goldfeder, S., Kell, T., Zhao, X., Bentov, I., Breidenbach, L., ir Juels, A. (2020). Flash boys 2.0: Frontrunning in decentralized exchanges, miner extractable value, and consensus instability. In 2020 IEEE Symposium on Security and Privacy (SP) (pp. 910-927). IEEE.

Apibendrinant galima teigti, kad esamos botų aptikimo sistemos daugiausia dėmesio skiria dominuojantiems botų tipams, tokiems kaip arbitražo botams, sandwitch atakos botams ir likvidavimo botas Ethereum tinkle. Tačiau botų kraštovaizdis sparčiai vystosi, apimant vis įvairesnius botų tipus ir naujas DeFi ekosistemas. Siekiant išlaikyti veiksmingą botų aptikimą, svarbu toliau plėtoti adaptyvias sistemas, kurios gali prisitaikyti prie kintančių botų elgsenos modelių ir aptikti naujus botų tipus įvairiuose blockchain tinkluose, įskaitant Solana. Tai apima pažangių mašininio mokymosi ir dirbtinio intelekto metodų taikymą bei bendradarbiavimą tarp skirtingų blokų grandinių bendruomenių.

1.5 Mašininio mokymosi taikymas botų aptikimui

Mašininis mokymasis tapo galinga priemone botų aptikimui įvairiose srityse, įskaitant socialinių tinklų svetaines, internetines platformas ir blokų grandinių ekosistemas. Anot Cresci (2020) mašininio mokymosi metodai gali automatiškai išmokti sudėtingus šablonus iš didelių duomenų rinkinių, leidžiančių efektyviai identifikuoti botų elgesį.

Klasterizavimo algoritmai, tokie kaip K-means ir DBSCAN (angl. *Density-Based Spatial Clustering of Applications with Noise*), Niedermayer, Saggese ir Haslhofer (2024), teigimu gali būti sėkmingai pritaikomi botų aptikimui blokų grandinių tinkluose. K-means algoritmas siekia suskirstyti duomenų taškus į k klasterių, minimizuojant atstumų kvadratų sumą tarp duomenų taškų ir jų priskirtų klasterių centrų. Tikslo funkcija apibrėžiama kaip:

$$J = (n = 1toN)(k = 1toK)r_nk||x_n -_k ||^2$$

Čia r_nk yra dvejetainė priklausomybės kintamasis, nurodantis, ar duomenų taškas x_n priklauso klasteriui k, o μ_k yra klasterio k centras.

DBSCAN algoritmas identifikuoja tankius taškų regionus, atsižvelgdamas į du parametrus: eps (maksimalų atstumą tarp dviejų taškų, kad jie būtų laikomi kaimynais) ir minPts (mažiausią taškų skaičių, reikalingą tankiam regionui sudaryti). Taškai, turintys mažiau nei minPts kaimynų eps atstumu, laikomi triukšmu arba anomalijomis (Nidhi ir Patel, 2016).
DBSCAN: Duomenų taškas p laikomas šerdies tašku, jei aplink jį, tam tikro ϵ spindulio srityje, yra bent minPts kitų taškų.:

$$N_\epsilon(p) = \{q \in D \mid \mathrm{dist}(p,q) \leq \epsilon\}$$

D yra visas duomenų rinkinys, dist(p,q) reiškia atstumą tarp taškų p q, o minPts yra mažiausias reikalingas taškų skaičius, kad būtų galima suformuoti klusterį.

Chen et al. (2018) naudojo K-means algoritmą, kad sugrupuotų Ethereum pinigines pagal jų sandorių elgesį ir identifikuotų Ponzi schemas. Tuo tarpu Wu et al. (2021) analizavo kriptovaliutų sandorius tinklo požiūriu ir parodė, kaip klasterizavimo metodai gali padėti aptikti anomalijas ir botų veiklą.

Izoliavimo miškas (Isolation Forest) yra dar vienas mašininio mokymosi metodas, naudojamas anomalijų ir botų aptikimui. Autoriai Miftahuddin ir Al-Ghifary (2024) remiasi prielaida, kad anomalijas galima lengviau izoliuoti nuo normalių duomenų taškų. Izoliavimo medžiai konstruojami atsitiktinai parenkant požymį ir atsitiktinai parenkant skilimo tašką tarp maksimalios ir minimalios to požymio reikšmės. Anomalijos turi trumpesnius kelius izoliavimo medžiuose, nes joms reikia mažiau skaidymų, kad būtų izoliuotos. Taško x anomalijos balas apibrėžiamas kaip:

$$s(x, n) = 2^(- E(h(x))/c(n))$$

Čia E(h(x)) yra vidutinis kelio ilgis taškui x visuose izoliavimo medžiuose, o c(n) yra normalizavimo konstanta, priklausanti nuo duomenų taškų skaičiaus n.

Kaip teigia Shwartz-Ziv ir Armon (2022) ansamblio mokymosi metodai (angl. *ensemble learning*), tokie kaip atsitiktinis miškas (angl. *random forest*), Adaboost ir gradientinis stiprinimas (angl. *gradient boosting*), yra ypač veiksmingi sprendžiant klasifikavimo užduotis su lentelių duomenimis. Atsitiktinis miškas konstruoja daugybę sprendimų medžių apmokymų metu ir išveda klasę, kuri yra vidutinė atskirų medžių prognozė. Kiekvienas medis konstruojamas naudojant atsitiktinę duomenų poaibį ir atsitiktinį požymių poaibį kiekviename mazge.

Adaboost algoritmas iteratyviai apmoko silpnus klasifikatorius, suteikdamas didesnius svorius neteisingai klasifikuotiems duomenų taškams kiekviename etape. Galutinis klasifikatorius yra svertinė silpnųjų klasifikatorių suma:

$$H(x) = sign((t = 1toT)_t * h_t(x))$$

h_t(x) yra silpnas klasifikatorius t, o α_t yra svoris, priskirtas klasifikatoriui remiantis jo tikslumo rezultatu.

Gradientinis stiprinimas nuomone taip pat iteratyviai apmoko sprendimų medžius, tačiau kiekvienas medis apmokomas prognozuoti ankstesnių medžių liekanų gradientą (Chengsheng, Huacheng ir Bing, 2017). Galutinis modelis yra visų medžių suma:

$$F_M(x) = F_0(x) + (m = 1toM)_m * h_m(x)$$

F_0(x) yra pradinis spėjimas, h_m(x) yra sprendimų medis m, o γ_m yra medžio m mokymosi greitis.

Šie metodai gali būti pritaikyti botų aptikimui blokų grandinių sandorių duomenyse. Li, Guan ir Lee (2023) naudojo XGBoost algoritmą, gradientinio stiprinimo variaciją, arbitražo botų identifikavimui Ethereum decentralizuotose biržose (DEX) ir parodė, kad šis metodas gali pasiekti aukštą tikslumą aptinkant botų veiklą.

Solana blokų grandinė tapo patrauklia platforma botams dėl savo didelio našumo infrastruktūros ir mažų sandorių mokesčių (Yakovenko, 2018). Norint veiksmingai aptikti botus Solana ekosistemoje, būtina sukurti aktualias požymių aibes, kurios perteiktų botų elgesio ypatumus. Siddamsetti (2024) analizavo Bitcoin blokų grandinę anomalijų aptikimo požiūriu ir pasiūlė požymių inžinerijos strategijas, skirtas botų veiklai identifikuoti.

Nepaisant šių pasiekimų, mašininio mokymosi taikymas botų aptikimui blokų grandinėse vis dar susiduria su tam tikrais iššūkiais. Vienas pagrindinių iššūkių yra patikimų anotacijų trūkumas (Bonifazi et al. 2022). Dauguma esamų botų aptikimo sistemų remiasi rankinio požymių kūrimo ir taisyklių nustatymo metodais, kurie reikalauja išsamių žinių apie konkrečias botų veiklos rūšis. Tai apriboja šių sistemų gebėjimą prisitaikyti prie naujų botų elgesio šablonų. Mašininio mokymosi metodai gali pagerinti botų aptikimo sistemų lankstumą ir gebėjimą aptikti nežinomus botų tipus. Tačiau norint efektyviai apmokyti ir įvertinti mašininio mokymosi modelius, reikalingi kokybiški anotacijų duomenys.

Zwang et al. (2018) teigia, kad ženklinimo iššūkį galima spręsti derinant rankines anotacijas su kitais metodais, tokiais kaip elgesio šablonų analizė ar euristinės taisyklės. Autoriai naudojo laiko intervalų analizę, kad nustatytų neįprastus sandorių šablonus ir automatiškai sugeneruotų botų elgesio anotacijas. Bonifazi et al. (2022) analizavo įvairius metrikas, tokias kaip sandorių vertė ir tinklo struktūra, norėdami kategorizuoti Ethereum pinigines ir identifikuoti

galimą botų veiklą. Derinant šiuos metodus su tiksliomis rankinėmis anotacijomis, galima sukurti kokybiškesnius anotacijų duomenis mašininio mokymosi modelių apmokymui.

Kitas iššūkis yra požymių inžinerija - informatyvių požymių, atspindinčių botų elgesio ypatumus, kūrimas. Dauguma esamų darbų remiasi paprastomis metrikomis, tokiomis kaip sandorių dažnis ar vertė (Xing, Shu, Zhao, Li ir Guo, 2022). Norint tiksliai modeliuoti sudėtingą botų elgesį, gali prireikti išvestinių požymių, kurie perteikia laiko bei erdvės šablonus ir ryšius tarp piniginių. Wu et al (2020), teigimu grafo teorijos metrikos, tokios kaip centrališkumas ar klasterizacijos koeficientas, gali būti naudojamos norint išryškinti svarbias piniginių tinklo savybes. Giliojo mokymosi (angl. *deep learning*) metodai, tokie kaip grafo konvoliuciniai tinklai (angl. *GNN*), taip pat gali būti naudojami automatizuoti požymių išmokimui iš neapdorotų grafo duomenų.

Apibendrinant, mašininio mokymosi metodai siūlo daug žadančių galimybių pagerinti botų aptikimo galimybes blokų grandinių ekosistemose. Neprižiūrimi metodai, tokie kaip klasterizavimas ir anomalijų aptikimas, gali padėti identifikuoti įtartinus elgesio šablonus be išankstinių žinių apie konkrečius botų tipus. Tuo tarpu prižiūrimi ansamblio mokymosi metodai, tokie kaip atsitiktiniai miškai ir gradientinis stiprinimas, gali pasiekti aukštą klasifikavimo tikslumą, kai yra pakankamai anotacijų duomenų. Derinant šiuos metodus su išmaniomis duomenų ženklinimo strategijomis, pažangia požymių inžinerija ir gilaus mokymosi technikomis, galima sukurti lanksčias ir galingas botų aptikimo sistemas, pritaikytas dinamiškiems ir sudėtingiems blokų grandinių duomenims. Ateities tyrimai turėtų toliau tirti šias kryptis, siekiant pagerinti botų aptikimo efektyvumą ir pritaikomumą įvairiose blokų grandinių platformose bei taikymo srityse.

1.6 Vertinimo metrikos

Vertinimo metrikos yra esminės, siekiant įvertinti mašininio mokymosi modelių, skirtų botų aptikimui Solana ekosistemoje, našumą ir efektyvumą. Šiame skyriuje aptarsime įvairias metrikas, naudojamas klasifikavimo modelių veiklos vertinimui, įskaitant tikslumą, preciziškumą, atšaukimą (jautrumą), F1 įvertį, specifiškumą, plotą po ROC kreive (AUROC) ir plotą po

tikslumo-atšaukimo kreive (AUPRC). Taip pat aptarsime kryžminio validavimo svarbą, siekiant užtikrinti botų aptikimo modelių patikimumą ir generalizavimą.

Tikslumas yra teisingų prognozių (tiek teisingų teigiamų (TP), tiek teisingų neigiamų (TN)) proporcija tarp visų tirtų atvejų (Sokolova ir Lapalme, 2009). Jis apskaičiuojamas pagal formulę:

Tikslumas = (TP + TN) / (TP + TN + FP + FN)

čia FP yra klaidingų teigiamų atvejų skaičius, o FN - klaidingų neigiamų atvejų skaičius.

Tikslumas yra bendras modelio veiklos matas, tačiau jis gali būti klaidinantis, kai susiduriama su nesubalansuotais duomenų rinkiniais. Preciziškumas yra teisingų teigiamų prognozių dalis tarp visų modelio atliktų teigiamų prognozių (Davis ir Goadrich 2016). Jis apskaičiuojamas taip:

Preciziškumas = TP / (TP + FP)

Preciziškumas parodo, kokia dalis modelio identifikuotų botų iš tikrųjų yra botai. Atšaukimas (jautrumas) yra teisingų teigiamų prognozių dalis tarp visų faktinių teigiamų atvejų duomenų rinkinyje (Xie at el, 2019). Jis apskaičiuojamas pagal formulę:

Atšaukimas = TP / (TP + FN)

Atšaukimas nurodo, kokią dalį faktinių botų modelis gali aptikti.

F1 įvertis yra harmoninis preciziškumo ir atšaukimo vidurkis, pateikiantis subalansuotą modelio veiklos matą (Wüst ir Gervais, 2018).Jis ypač naudingas, kai susiduriama su nesubalansuotais duomenų rinkiniais, kai teigiamų ir neigiamų atvejų skaičius yra nevienodas. F1 įvertis apskaičiuojamas taip:

F1 = 2 * (Preciziškumas * Atšaukimas) / (Preciziškumas + Atšaukimas)

Specifiškumas yra teisingų neigiamų prognozių dalis tarp visų faktinių neigiamų atvejų duomenų rinkinyje (Bach, Mihaljevic ir Zagar, 2018). Jis apskaičiuojamas pagal formulę:

Specifiškumas = TN / (TN + FP)

Specifiškumas parodo, kokia dalis ne botų yra teisingai klasifikuojama kaip ne botai.

AUROC yra grafinis modelio gebėjimo atskirti teigiamus ir neigiamus atvejus vaizdavimas, atvaizduojant tikrąjį teigiamą dažnį (TPR) prieš klaidingą teigiamą dažnį (FPR) esant įvairioms ribinėms reikšmėms. TPR ir FPR apskaičiuojami taip:

TPR = TP / (TP + FN) FPR = FP / (FP + TN)

AUROC reikšmė 1 rodo tobulą klasifikatorių, o 0,5 rodo atsitiktinį spėjimą. Kuo didesnis AUROC, tuo geresnis modelio gebėjimas atskirti botus nuo ne botų.

AUPRC yra grafinis modelio veiklos vaizdavimas preciziškumo ir atšaukimo atžvilgiu esant skirtingoms ribinėms reikšmėms, ypač naudingas turint reikalų su nesubalansuotais duomenų rinkiniais (Drescher, D., 2017). Kuo didesnis AUPRC, tuo geresnė modelio veikla.

Kryžminio validavimo metodai, tokie kaip k-dalių kryžminis validavimas arba stratifikuotas k-dalių kryžminis validavimas, yra svarbūs užtikrinant botų aptikimo modelių patikimumą ir generalizavimą (Casino, Dasaklis ir Patsakis, 2019). Šie metodai duomenis suskirsto į k nesikertančių dalių ir apmoko bei testuoja modelį su skirtingų dalių kombinacijomis. Proceso kartojimas leidžia įvertinti modelio našumo vidurkį ir standartinį nuokrypį, taip suteikiant patikimesnį modelio veiklos įvertinimą.

Greta standartinių klasifikavimo metrikų, tyrime gali būti naudojamos ir specifinės srities metrikos, susijusios su botų aptikimu Solana ekosistemoje. Pavyzdžiui, Chen et al. (2018) analizuoja konkretų botų tipą - rinkos manipuliavimo botus Bitcoin ekosistemoje. Tyrimo metodologiją ir išvadas galima adaptuoti Solana ekosistemoje, siekiant įvertinti panašių manipuliacinių botų aptikimo dažnį ir jų poveikį. Be to, Dey et al. (2016) vertina finansinius nuostolius, patirtus dėl front-running botų veiklos decentralizuotose keityklose Ethereum blokų grandinėje. Nors tai skirtinga ekosistema, panašius metodus galima taikyti vertinant finansinius nuostolius, patirtus dėl botų veiklos Solana ekosistemoje, ypač susijusius su arbitražo ar likvidavimo botais.

Apibendrinant, vertinimo metrikos yra labai svarbios vertinant botų aptikimo modelių, taikomų Solana ekosistemoje, našumą ir efektyvumą. Preciziškumas, atšaukimas, F1 įvertis, AUROC, AUPRC ir kryžminio validavimo metodai suteikia išsamų modelių veiklos vaizdą ir leidžia palyginti skirtingus metodus. Derinant standartines klasifikavimo metrikas su specifinėmis srities metrikomis ir adaptuotomis įžvalgomis iš kitų blokų grandinių, galima gauti visapusišką botų aptikimo modelių vertinimą Solana ekosistemoje. Vizualizacijos, tokios kaip sumaišymo matrica, ROC kreivė ir preciziškumo-atšaukimo kreivė, taip pat gali padėti skaitytojams geriau suprasti šias metrikas ir modelių veiklą. Apskritai, skyrius apie vertinimo metrikas yra svarbus magistro darbo komponentas, parodantis tyrėjo gebėjimą kritiškai įvertinti ir palyginti mašininio mokymosi modelius konkrečiame botų aptikimo Solana ekosistemoje kontekste.

2. AUTOMATIZUOTŲ SANDORIŲ APTIKIMAS SOLANOS TINKLE METODOLOGIJA

Skyriuje pateikiama automatizuotų sandorių aptikimo metodologija, kurioje aprašomas empirinio taikomojo modelio tyrimas, jame naudotų duomenų šaltinis bei pačių duomenų analizė, metodas ir jo programinio kodo pavyzdys, kurio šablonas buvo naudojamas šiame tiriamajame darbe. Taip pat šiame skyriuje pateikiamas modelio gerumo vertinimo metrikų aprašymas ir tyrimo eigos etapai.

2.1. Automatizuotų sandorių aptikimo metodas Solanos tinkle tyrimo pobūdis

Tyrimo tikslas yra sukurti efektyvų metodą automatizuotų sandorių (botų) aptikimui Solanos blockchain tinkle. Pagrindinis tyrimo uždavinys - identifikuoti įtartinus sandorių šablonus ir anomalijas, kurie gali būti susiję su botų veikla. Tyrimo paskirtis - pagerinti Solanos tinklo saugumą ir apsaugoti vartotojus nuo galimų manipuliacijų rinkoje. Tyrimas apims tris blogųjų botų tipus: MEV (Miner/Maximal Extractable Value) botus, „sandwich“ botus ir „frequency trading“ botus. Visi jie pasižymi manipuliacinėmis praktikomis bei veikia siekdami neteisėto pranašumo ar žalos rinkos dalyviams, todėl kelia didesnę grėsmę Solanos tinklo skaidrumui.

Tyrimo tipas yra kiekybinis, paremtas realių Solanos tinklo transakcijų duomenų analize. Naudojant statistinius ir mašininio mokymosi metodus, siekiama nustatyti botų elgesio dėsningumus ir sukurti automatizuotą aptikimo įrankį.

Tyrimo objektas yra Solanos blockchain tinklo transakcijų duomenys. Analizuojami sandorių šablonai, dažnumas, adresų sąsajos galinčioss indikuoti botų veiklą.

2.2. Automatizuotų sandorių aptikimo metodas Solanos tinkle tyrimo metodas

Duomenų šaltinis, naudojami duomenys bus gaunami iš Solanos blockchain tinklo, pasitelkiant Solana JSON RPC API. Atsitiktinai bus atrinkta 1000 transakcijų iš 100 Solanos piniginių (angl. *wallets*).

Duomenų analizei bus taikomi aprašomosios statistikos metodai, siekiant įvertinti pagrindinių požymių pasiskirstymą ir variaciją, taip pat bus pritaikyti mašininio mokymosi algoritmai, tokie kaip K-means klasterizavimas, DBSCAN ir izoliavimo miškai (angl. *Isolation Forest*), siekiant identifikuoti anomalijas ir botų elgesio šablonus

Metodas šiame tyrimas yra apibūdinamas kaip empirinis taikomas, skirtas automatizuotų sandorių aptikimui Solanos tinkle. Tyrime naudojami mašininio mokymosi algoritmai, tokie kaip K-means, DBSCAN klasterizavimai ir Isolation Forest, siekiant identifikuoti elgsenos šablonus ir anomalijas, būdingas botų veiklai. Tyrimas buvo atliktas naudojant Python programavimo kalbą ir bibliotekas, tokias kaip Scikit-learn ir Pandas, kurios leido įgyvendinti klasterizacijos ir anomalijų aptikimo procesus. Duomenų saugojimui naudota SQLite duomenų bazė, o algoritmų parametrų optimizavimas atliktas taikant kryžminio patikrinimo metodiką. Gauti rezultatai bus pateikti vizualiai suprantama forma, tokia kaip diagramos ir grafikai, o jų interpretacija padės suprasti automatizuotų sandorių elgesio modelius.

2.3. Automatizuotų sandorių aptikimas Solanos tinkle tyrimo metodo statistika

Šiame poskyryje pateikiama išsami apžvalga statistinių metodų, naudojamų automatizuotų sandorių (botų) aptikimui Solanos blockchain tinkle. Taikomi metodai apima aprašomąją statistiką, anomalijų aptikimo metrikas, klasterizavimo vertinimo statistiką ir anomalijų aptikimo statistiką.

Aprašomoji statistika padės suprasti bendras duomenų charakteristikas ir identifikuoti potencialius botų elgesio požymius. Inferencinė statistika leis statistiškai patikrinti, ar stebimi sandorių šablonai reikšmingai skiriasi nuo atsitiktinių.

Klasterizavimo vertinimo statistika padės užtikrinti prasmingą piniginių grupavimą pagal panašius elgesio šablonus.

Galiausiai, anomalijų aptikimo statistika, gaunama naudojant Izoliavimo miško algoritmą, leis kiekybiškai įvertinti piniginių tikimybę būti botais ir modelio efektyvumą skiriant anomalias pinigines nuo įprastų. Taikant šiuos statistinius metodus, bus galima išsamiai išanalizuoti Solanos tinklo transakcijų duomenis, identifikuoti įtartinus automatizuotų transakcijų elgesio šablonus ir sukurti patikimą automatizuotų sandorių aptikimo metodą.

4 lentelė

Aprašomosios statistikos struktūra

Kategorija	Matas	Aprašymas	Svarba botų aptikimui
Centrinės tendencijos	**Mean** (Vidurkis)	Vidutinė tokių metrikų kaip sandorių dažnis ir sandorio suma reikšmė.	Nustato pinigines, kurios nukrypsta nuo vidurkio, kas gali rodyti botus, veikiančius kraštutinumais.
	Median (Mediana)	Vidurinė reikšmė surūšiuotuose duomenyse, mažiau paveikiama išskirčių.	Išryškina pinigines su nenormaliais sandorių laiko tarpais ar dažniu.
	Mode (Moda)	Dažniausiai pasitaikanti reikšmė, pvz., sandorio dydis ar naudojama programa.	Aptinka pasikartojančius botų šablonus, pvz., nuolat vienodo dydžio sandorius.
Kintamumas (sklaida)	**Standard Deviation**	Matuoja duomenų išsibarstymą aplink vidurkį, pvz., sandorių sumos kintamumą.	Botai dažnai pasižymi mažu kintamumu dėl pasikartojančių šablonų, o testavimo botai – didesniu.
	Range (Intervalas)	Skirtumas tarp didžiausios ir mažiausios reikšmių, pvz., sandorių dažnio intervalo.	Parodo ekstremalias elgsenas (pvz., nuolat aukštą ar žemą aktyvumą).
	Interquartile Range (IQR) (Kvartilinis plotis)	Intervalas tarp 25 ir 75 procentilių, išskyrus išskirtis.	Aptinka pinigines, kurių sandoriai dažnai patenka už įprastų ribų.
Pasiskirstymo forma	**Skewness** (Asimetrija)	Matuoja duomenų pasiskirstymo asimetriškumą, pvz., teigiamą asimetriją sandorių dažnyje.	Parodo pinigines, kurios prisideda prie reikšmingos asimetrijos, dažnai botus.
	Kurtosis (Ekscesas)	Matuoja duomenų pasiskirstymo „uodeguotumą“, pvz., sunkias uodegas sandorių sumoje.	Išryškina pinigines su dažnai pasikartojančiais ar retai dideliais sandoriais.
Tarpusavio ryšiai	**Correlation Coefficients** (Koreliacijos koeficientai)	Matuoja ryšius tarp požymių porų, pvz., dažnis vs. balanso pokyčiai.	Aptinka botams būdingas priklausomybes, pvz., dažną aktyvumą su mažais balanso pokyčiais.
Pasiskirstymo šablonai	**Histograms** (Histogramos)	Vizualizuoja tokių požymių kaip sandorių dažnis ar suma pasiskirstymą.	Aptinka botų šablonus kaip smailes histogramose.
	Box Plots (Stačiakampės diagramos)	Parodo pagrindinių metrikų išsibarstymą ir išskirtis.	Išryškina ekstremalias sandorių elgsenos reikšmes, padedančias aptikti potencialius botus.

Šaltinis: sudaryta autoriaus.

2.4. Empirinio taikomojo Automatizuotų sandorių aptikimo metodo Solanos tinkle tyrimo atlikimas

Šio skyriaus tikslas – aprašyti empirinį automatizuotų sandorių aptikimo metodo tyrimą, atliktą naudojant Solanos tinklo duomenis. Tyrimas apima duomenų rinkimą, apdorojimą, funkcijų inžineriją, klasterizaciją bei anomalijų aptikimą, siekiant identifikuoti automatizuotų (botų) sandorių elgsenos modelius.

1 etapas. Duomenų rinkimas ir apdorojimas Duomenys surinkti iš viešai prieinamos Solanos blokų grandinės, naudojant RPC API endpointą: https://rpc.shyft.to.

Tyrimo duomenų rinkinį sudaro:

- 100 piniginių adresų, atrinktų pagal įvairias aktyvumo charakteristika (dažnumą, sąveikas su DeFi platformomis).
- Iki 1000 sandorių kiekvienai piniginei, įtraukiant tokius metaduomenis kaip balansas, programų sąveikos ir laiko žymės.

2 etapas. Duomenys buvo apdoroti naudojant .NET pagrindu sukurtą backend programine struktūra, kuri užtikrino:

- Filtravimą: Pašalinti neišsamūs arba pasikartojantys sandoriai.
- Normalizavimą: Sandorių sumų, dažnių ir kitų metrikų standartizavimas.
- Funkcijų išskyrimą: Sugeneruoti pagrindiniai požymiai, tokie kaip sandorių dažnis, aktyvumo nuoseklumas ir mokesčių santykiai.

5 lentelė

Duomenų tvarkymo procesas

Kategorija	Laukai	Numatytoji reikšmė arba sprendimas
Skaitmeniniai laukai	Transakcijos sumos	0
	Skaičiavimo reikšmės (pvz., dažnis, koeficientai)	0
Tekstiniai laukai	Tuščios reikšmės	Tuščia eilutė "" arba žyma "Random"
Masyvai	Tušti masyvai	Tuščias masyvas []
Kontekstinis null reikšmių sprendimas	Timestamp reikšmės	Keičiama blokų laiku
	Amount reikšmės	Skaičiuojama iš Pre Balance ir Post Balance pokyčių
	Program reikšmės	Keičiama pagal Program ID
Kokybės užtikrinimas	Null reikšmių analizė	Kiekvieno stulpelio null reikšmių dalis stebima
	Duomenys su dideliu null reikšmių skaičiumi	Pažymimi kaip mažiau patikimi arb apanaikinami

Šaltinis: sudaryta autoriaus.

3 etapas. Visi apdoroti duomenys saugomi SQLite duomenų bazėje, siekiant užtikrinti sklandų ryšį su Python klasterizacijos bei anomalijų aptikimo paslaugomis.

Požymių inžinerija - duomenų rinkinys papildytas reikšmingais požymiais, skirtomis klasterizacijai ir anomalijų aptikimui, požymiai buvo parinkti remiantis literatūroje aptartais

būdais, siekiant užtikrinti metodų pritaikomumą ir rezultatų palyginamumą su kitų blokų grandinių tyrimais:

- Sandorių dažnis: Bendras sandorių skaičius per pasirinktą laikotarpį.
- Vidutinė sandorio suma: Tipinis kiekvieno sandorio dydis.
- Sandorio sumos standartinis nuokrypis: Sandorio sumų variacijos laipsnis.
- Mokesčio ir sumos santykis: Metrika, atskleidžianti neefektyvius arba botų vykdomus sandorius.
- Unikalių gavėjų skaičius: Gavėjų adresų įvairovė.
- Sąskaitos vaidmenų įvairovė: Įvairių sąveikos tipų skaičius sandoriuose.
- Instrukcijų sudėtingumas: Programų instrukcijų skaičius ir tipai sandoryje.
- Sėkmės rodiklis: Sėkmingų sandorių dalis, palyginti su visais.
- Slotų anomalijų rodiklis: Nukrypimai nuo įprasto slotų panaudojimo modelio.
- DEX sandorių dažnis: Sandoriai, susiję su decentralizuotomis biržomis.
- Programų pasiskirstymo balas: Programų įvairovė ir dažnumas.
- Aktyvios valandos per dieną: Vidutinis aktyvumo valandų skaičius per dieną.
- Aktyvumo nuoseklumas: Sandorių aktyvumo pasiskirstymo vienodumas.

4 etapas. Klasterizacija ir anomalijų aptikimas Funkcijos buvo analizuojamos naudojant Python, taikant šiuos metodus:

K-means klasterizacija:

- Grupuoja pinigines pagal jų elgsenos panašumus.
- Optimalus klasterių skaičius nustatytas naudojant Elbow Method ir Silhouette Score.

DBSCAN klasterizacija:

- Aptinka tankius sandorių grupių modelius, kurie gali rodyti botų veiklą.
- Parametrai:
    - eps – kaimynystės dydis.
    - minPts – minimalus taškų skaičius klasteryje.

Isolation Forest anomalijų aptikimas:

- Nustato reikšmingai nuo normos nukrypusius sandorių modelius.
- Fokusuojasi į sandorius, kurių elgesys išskirtinis.

5 etapas. Šiame etape bus atlikta hierarchinė analizė, skirta sukurti anomalijų izoliacijos miško medžius, siekiant įvertinti, kaip kiekvienas metodas DBSCAN, K-means ir Isolation Forest metodų sugebėjimą atskirti botų ir žmonių pinigines. Kiekvienam metodui bus sukurtos anomalijų izoliacijos miško medžių struktūros, kurios leis atskirti botų ir žmonių pinigines pagal jų elgsenos ypatybes. Rezultatai bus susieti su klasteriais, siekiant įvertinti kiekvieno metodo gebėjimą identifikuoti specifines anomalijas. Bei kokios metodų kombinacijos pasižymi geriausiais rezultatais Pagrindinis dėmesys bus skirtas analizuoti, kaip šie metodai identifikuoja anomalijas, atsižvelgiant į duomenų tankumo, retumo ypatybes.

6 lentelė

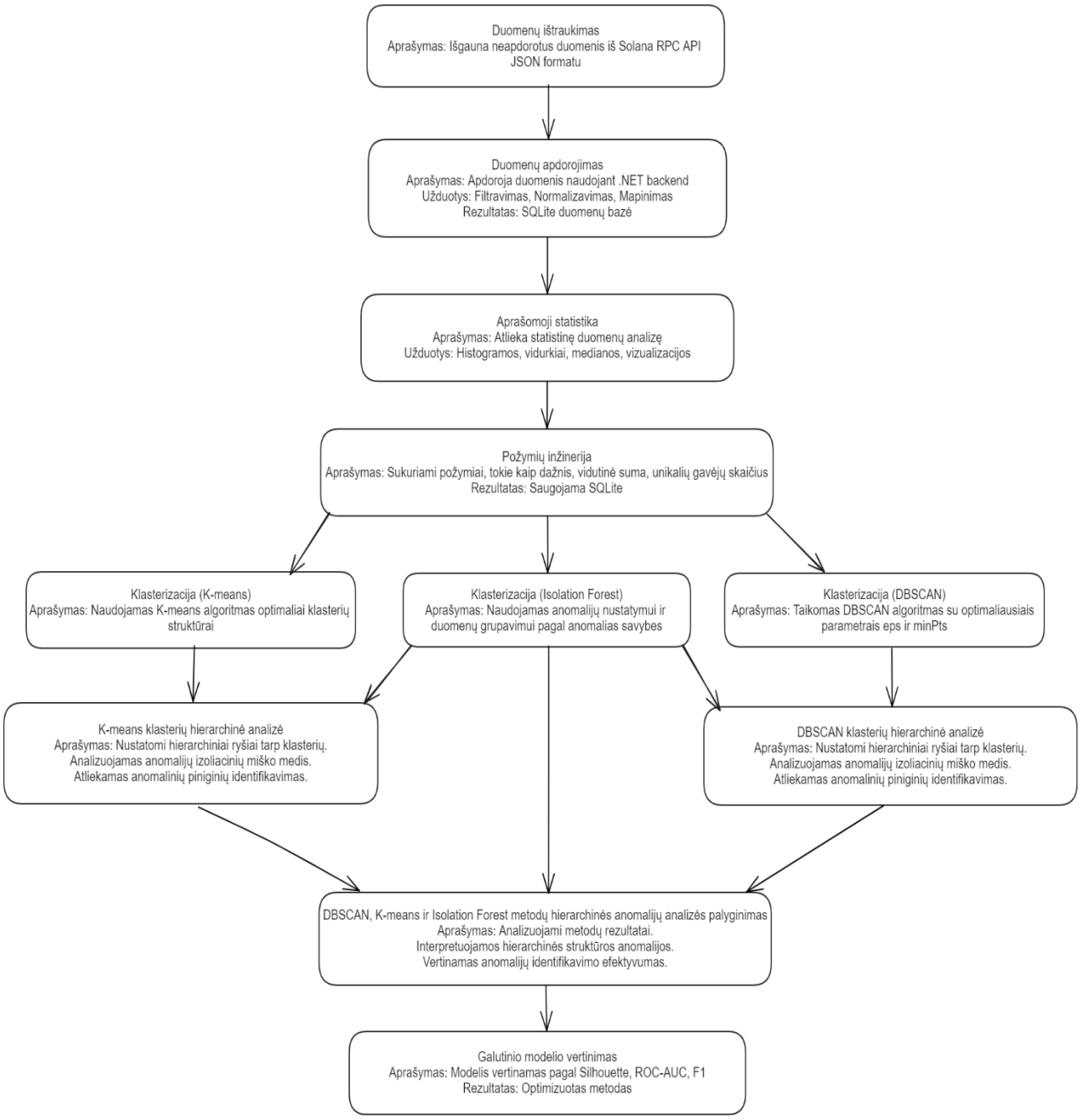
Hierarchinės analizės metodų taikymo planas

Aspektas	Aprašymas
Metodai	DBSCAN, K-means, Isolation Forest
Tikslas	Sukurti anomalijų izoliacijos miško medžius, skirtus botų ir žmonių piniginėms atskirti pagal elgesio ypatybes.
Pagrindinis dėmesys	Anomalijų klasifikavimas, analizuojant duomenų tankumą, retumą, transakcijų elgesio modelius.
Rezultatų vizualizacija	Anomalijų miško schemos, pateikiančios piniginių elgesio modelius.
Galutinis rezultatas	Įvertinimas kiekvieno metodo gebėjimas identifikuoti specifines anomalijas bei jų taikymo ribotumai.

Šaltinis: sudaryta autoriaus

6 etapas. Rezultatų analizė Gauti klasterizacijos ir anomalijų aptikimo rezultatai buvo įvertinti pagal šiuos rodiklius:

- Silhouette Score: Klasterizacijos modelių grupių kohesijos ir separacijos vertinimas.
- ROC-AUC: Anomalijų aptikimo tikslumo įvertinimas, taikant Isolation Forest.
- Precision, Recall ir F1-score: Vertinama, kai yra žinoma kurios piniginės yra žmonių, o kurios botų, leidžiant palyginti rezultatų tikslumą.



Šaltinis: sudaryta autoriaus remiantis aprašyto tyrimo atlikimo etapais.

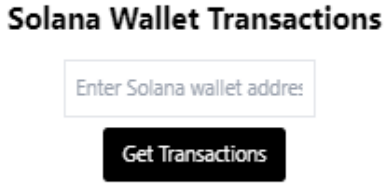
5 pav. Automatizuotų sandorių aptikimo metodas Solanos tinkle parinkimo etapai

3. AUTOMATIZUOTŲ SANDORIŲ APTIKIMO SOLANOS TINKLE EKSPERMENTINĖ DALIS

Šiame skyriuje pateikiami automatizuotų sandorių aptikimo rezultatai, susiję su botų elgesio nustatymu Solanos tinkle. Aprašomas duomenų analizės procesas, eksperimentai su mašininio mokymosi algoritmais bei modelio optimizavimas, siekiant sukurti efektyvų automatizuotų sandorių aptikimo modelį. Pateikiama analizė, kaip buvo pasirinktas galutinis modelis, ir aprašoma eksperimentų bei galutinio modelio statistika.

3.1. Neapdorotų duomenų rinkimas ir vertinimas

Sukurtas Angular pagrindu veikiantis vartotojo sąsajos komponentas „Solana Wallet Transactions", leidžiantis įvesti bet kurio Solanos tinklo piniginės adresą ir gauti susijusius transakcijų duomenis. Vartotojo sąsaja buvo sukurta taip, kad būtų intuityvi ir lengvai naudojama (žr. 5 paveikslą).



Šaltinis: sudaryta autoriaus.

6 pav. Piniginės adreso įvedimo laukas

Kai vartotojas įveda Solanos piniginės adresą ir paspaudžia mygtuką „Get Transactions“, sistema automatiškai inicijuoja užklausą į Solanos tinklo RPC API. Užklausa atliekama pasitelkiant programoje aprašytą SolanaService paslaugą, kuri yra integruota į Wallet Transactions komponentą. Ši paslauga yra esminė duomenų surinkimo dalis, nes ji ne tik siunčia API užklausą,

bet ir atsako už gautų duomenų tvarkymą, filtravimą ir validavimą, užtikrindama, kad duomenys būtų kokybiški ir tinkami tolesnei analizei.

SolanaService paslauga naudoja API metodus getSignaturesForAddress ir getTransaction, kad surinktų išsamią informaciją apie piniginės transakcijas. Metodas getSignaturesForAddress grąžina transakcijų parašų sąrašą ir jų slot numerius, nurodančius transakcijų patvirtinimo kontekstą. Tuo tarpu getTransaction pateikia detales apie kiekvieną transakciją, įskaitant balanso pokyčius, transakcijos ir vykdytas instrukcijas. Šie metodai leidžia gauti ne tik bazinius finansinius duomenis, bet ir sudėtingesnius elgesio šablonus, kurie yra būtini analizuojant ir identifikuojant automatizuotą veiklą Solanos tinkle. Ši paslauga taip pat apdoroja duomenis JSON formatu, pritaikydama juos patogiai vartotojo sąsajos vizualizacijai. Paslauga įdiegta taip, kad galėtų apdoroti klaidas, pavyzdžiui, kai pateikiamas netinkamas ar neegzistuojantis piniginės adresas. Jei API grąžina klaidos atsaką, SolanaService tinkamai valdo šiuos atvejus, pateikdama vartotojui informatyvų pranešimą ir apsaugodama sistemą nuo tolimesnių klaidų. Paslauga taip pat yra atsakinga už užklausų efektyvumą – optimizuota taip, kad dideli duomenų rinkiniai būtų perduodami nuosekliai ir neviršytų RPC API užklausų limitų. Viso šio proceso metu Solana Service užtikrina, kad visi gauti duomenys būtų paruošti tolesniems analitiniams žingsniams, tokiems kaip požymių inžinerija, klasterizacija ir anomalijų aptikimas. Toks integruotas sprendimas ne tik palengvina duomenų surinkimą, bet ir užtikrina patikimą, efektyvų bei vartotojui patogų procesą, kuris yra būtinas moksliniam tyrimui apie automatizuotų sandorių aptikimą Solanos tinkle.

Surinkti neapdoroti duomenys, iš API, apima išsamią informaciją apie kiekvieną transakciją (žr. 6 lentelę).

7 lentelė

Neapdorotų duomenų struktūra

Stulpelis	Aprašymas
Parašas (Signature)	Unikalus transakcijos identifikatorius.
Laiko žyma (Timestamp)	Laiko žyma, nurodanti, kada įvyko transakcija.
Slot	Šlot numeris Solanos tinkle, kuriame buvo užregistruota transakcija.
Pradinė balanso vertė (Pre Balance, SOL)	Piniginės balansas prieš transakciją SOL valiuta.
Galutinė balanso vertė (Post Balance, SOL)	Piniginės balansas po transakcijos SOL valiuta.
Suma (Amount, SOL)	Transakcijos suma SOL valiuta.
Mokestis (Fee, SOL)	Transakcijos mokestis SOL valiuta.

Gavėjo adresas (Receiver Address)	Piniginės, gavusios transakciją, adresas.
Būsena (Status)	Transakcijos būsena („Įvykdyta“, „Nepavyko“).
Reikalingi parašai (Required Signatures)	Parašų skaičius, reikalingas patvirtinti transakciją.
Tik skaitomi pasirašyti sąskaitos (Readonly Signed Accounts)	Sąskaitų, kurios tik skaitomos ir yra pasirašytos transakcijai, sąrašas.
Tik skaitomi nepasirašyti sąskaitos (Readonly Unsigned Accounts)	Sąskaitų, kurios tik skaitomos ir nėra pasirašytos transakcijai, sąrašas.
Naujausias blokhašas (Recent Blockhash)	Naujausia su transakcija susijusi blokhašo vertė.
Transakcijos pranešimas (Transaction Message)	Išsamus transakcijos pranešimas.
Neapdorotos instrukcijos (Raw Instructions)	Žaliavos instrukcijos, vykdytos per transakciją.
Žaliavos vidinės instrukcijos (Raw Inner Instructions)	Vidinės instrukcijos, susijusios su transakcija.
Žurnalo pranešimai (Log Messages)	Žurnalai, sugeneruoti vykdant transakciją.

Šaltinis: sudaryta autoriaus.

Surinkti duomenys buvo saugomi dideliame JSON faile, kuriame kiekviena transakcija buvo atvaizduota kaip atskiras objektas su aukščiau minėtais atributais. Duomenų apdorojimas ir paruošimas Siekiant paruošti duomenis tolesnei analizei, buvo atlikti keli apdorojimo ir valymo žingsniai:

1. JSON failo nuskaitymas ir transakcijų išgavimas.
2. Trūkstamų ar netinkamų reikšmių („null", „undefined") pašalinimas ar pakeitimas.
3. Duomenų tipų konvertavimas (eilučių pavertimas skaičiais).
4. Naujų požymių generavimas (laiko žymų konvertavimas į datos formatą, mokesčių ir sumos santykio apskaičiavimas).
5. Duomenų filtravimas, pašalinant nereikalingas ar besidubliuojančias transakcijas.

Apdoroti duomenys buvo saugomi SQLite duomenų bazėje, kuri yra kompaktiška, greita ir patikima reliacinė duomenų bazė. SQLite duomenų bazė buvo pasirinkta dėl jos paprastumo, našumo ir galimybės lengvai integruoti su kitomis technologijomis, tokiomis kaip Python ir .NET. Duomenų saugojimas SQLite duomenų bazėje užtikrino efektyvų duomenų valdymą ir lengvą prieigą tolesnei analizei. Duomenų kokybė buvo vertinama pagal šiuos kriterijus (žr. 7 lentelę). Atlikus duomenų vertinimą, nustatyta, kad surinkti duomenys yra pakankamai išsamūs, unikalūs, nuoseklūs ir tikslūs, kad būtų galima atlikti automatizuotų sandorių aptikimo analizę. Tam tikri

stulpeliai, tokie kaip „Slot", „Recent Blockhash" ar „Log Messages", buvo nuspręsta neįtraukti į tolesnę analizę, nes jie nebuvo laikomi aktualiais tyrimui.

8 lentelė

Duomenų kokybės vertinimo lentelė

Kriterijus	Klausimas	Vertinimo tikslas
Išsamumas	Ar yra trūkstamų reikšmių? Ar visi stulpeliai užpildyti?	Užtikrinti, kad surinkti duomenys yra pilni ir tinkami analizei, be praleistų reikšmių.
Unikalumas	Ar yra besidubliuojančių transakcijų?	Išvengti pasikartojančių transakcijų, kurios gali iškreipti analizės rezultatus.
Nuoseklumas	Ar duomenų tipai ir formatai yra nuoseklūs?	Patikrinti, kad visi stulpeliai laikosi vienodų duomenų tipų ir formatų.
Tikslumas	Ar reikšmės yra tikslios ir atitinka tikrovę?	Užtikrinti, kad duomenys būtų patikimi ir neprieštarautų Solanos tinklo realiems duomenims.
Aktualumas	Ar surinkti duomenys yra aktualūs tyrimo tikslui?	Įsitikinti, kad duomenys yra tinkami naudojimui botų aptikimo algoritmuose.

Šaltinis: sudaryta autoriaus.

Apibendrinant, sukurta duomenų rinkimo sąsaja, leidžianti greitai ir patogiai gauti Solanos tinklo transakcijų neapdorotus duomenis. Surinkti duomenys buvo išvalyti ir įvertinti, siekiant užtikrinti jų tinkamumą tolesnei kintamųjų inžinerijai .

3.2. Duomenų apdorojimas ir požymių inžinerija

Šiame skyriuje aprašomas procesas, kaip surinkti duomenys apdorojami .NET aplinkoje, naudojant backend modelį, siekiant paruošti juos tolesnei analizei. Darbo metu buvo taikomas domenų valdomos architektūros (angl. *Domain-Driven Design*) ir švarios architektūros (angl. *Clean Architecture*) principų derinys. Šie principai užtikrino aiškią atsakomybių atskirtį, padalijant projektą į atskirus sluoksnius: API, Application, Domain ir Infrastructure. Toks struktūrizavimas leido garantuoti eksperimentų metodų testuojamumą bei pagrindinių metodų vienos atsakomybės (angl. *Single Responsibility Principle*) laikymąsi.

Pateikiama informacija apie null reikšmių tvarkymą, požymių inžineriją, bei JSON laukų, tokių kaip „Raw Instructions", „Raw Inner Instructions" ir „Log Messages", žemėlapio (angl. *mapper*) kūrimą ir apdorojimą. Duomenų apdorojimas užtikrina, kad visi duomenys būtų tinkamai struktūrizuoti ir paruošti naudoti analitiniais tikslais. Tuščių (angl. *Null*) reikšmių tvarkymu siekiama užtikrinti duomenų vientisumą ir nuoseklumą.

Numatytojų (angl. *Default*) reikšmių suteikimas:

Skaitmeniniams laukams:

- Transakcijos sumos: 0.
- Skaičiavimo reikšmės (pvz., dažnis, koeficientai): 0.0.

Tekstiniams laukams:

- Tuščios eilutės "" arba žyma „Random".

Masyvams:

- Tuščias masyvas [].

Kontekstinis null reikšmių sprendimas:

- Trūkstamos „Timestamp" reikšmės keičiamos blokų laiku.
- Trūkstamos „Amount" reikšmės skaičiuojamos iš „Pre Balance" ir „Post Balance" pokyčių.
- Trūkstamos „Program" reikšmės keičiamos pagal „Program ID".

Kokybės užtikrinimas:

- Kiekvieno stulpelio null reikšmių dalis analizuojama ir stebima.
- Duomenys su dideliu null reikšmių skaičiumi pažymimi kaip mažiau patikimi.

JSON tipo duomenų laukų žemėlapio (angl. *mapper*) kūrimas JSON struktūrose, tokiose kaip „Raw Instructions", „Raw Inner Instructions" ir „Log Messages", pateikiami sudėtingi duomenys, kuriuos būtina tinkamai interpretuoti (žr. 8 lentelę).

9 lentelė

Json tipo duomenų tvarkymas

Laukas	Aprašymas ir apdorojimas
Raw Instructions	- Paruošiamos vykdytos instrukcijos.
	- Ištraukiama: programos ID, instrukcijų tipas (pvz., „transfer", „swap"), dalyviai, parametrai.
Raw Inner Instructions	- Apdorojami įdėti masyvai.
	- Kiekviename lygyje identifikuojamas programos tipas ir operacijų detalės.
Log Messages	- Išskirstomi logai pagal prefiksus: „Program .invoke", „Program success".
	- Analizuojamos klaidų žinutės ir jų įtaka perlaidai.

Šaltinis: sudaryta autoriaus.

Požymių inžinerija yra vienas svarbiausių etapų kuriant automatizuotų sandorių aptikimo modelį. Šio proceso tikslas – iš pradinių duomenų išskirti specifinius požymius, kurie atspindi pagrindines elgsenos charakteristikas ir padeda atskirti automatizuotas (angl. *botų*) veiklas nuo žmogaus inicijuotų transakcijų Solanos tinkle. Toliau pateikiama (žr. 8 lentelę). išsami analizė, kaip požymiai buvo pasirinkti, apskaičiuojami ir kodėl jie yra svarbūs.

10 lentelė

Požymių grupavimas ir reikšmingumas

Požymis	**Apskaičiavimas**	**Svarba**
Transaction Frequency	Transakcijų skaičius per valandą pagal piniginės adresą.	Aukštas transakcijų dažnis yra vienas pagrindinių botų veiklos rodiklių.
Average Transaction Amount	Vidutinė transakcijos suma SOL valiuta, apskaičiuojama iš „Amount“ lauko.	Leidžia nustatyti standartinę botų transakcijų struktūrą (mažas, bet dažnas sumas).
StdDev Transaction Amount	Standartinis nuokrypis nuo transakcijų sumų vidurkio.	Parodo transakcijų sumų kintamumą; botai dažnai atlieka pasikartojančias transakcijas su mažais nuokrypiais.
Fee-to-Amount Ratio	Mokesčio ir sumos santykis: „Fee“ / „Amount“.	Botai dažnai optimizuoja mokesčius, išlaikydami specifinius santykius.
Unique Receivers Count	Skaičiuojamas unikalių gavėjų skaičius, pašalinant savęs pervedimus.	Botai paprastai vykdo transakcijas į ribotą adresų skaičių.
Account Role Diversity	Skaičiuojami unikalūs paskyros vaidmenys, pvz., „Signer“, „Writable“.	Leidžia įvertinti sąveikų sudėtingumą; botų paskyros yra labiau apribotos.
Instruction Complexity	Analizuojamas instrukcijų sudėtingumas pagal masyvų ilgį ir įdėtą struktūrą.	Botai dažnai naudoja sudėtingesnes instrukcijas (pvz., „swap“ ar „liquidity pool“ operacijas).
Success Ratio	Skaičiuojama, kokia transakcijų dalis buvo sėkminga (statusas „Completed“).	Botai paprastai turi aukštą sėkmės lygį dėl optimizuotų veiksmų.
DEX Transaction Frequency	Transakcijų, susijusių su decentralizuotomis biržomis, dažnis.	Botai dažnai veikia decentralizuotose biržose („Raydium“, „Serum“).
Program Distribution Score	Programų naudojimo įvairovės ir koncentracijos įvertis.	Leidžia nustatyti, ar veikla orientuota į kelias specifines programas (tipiška botams).
Active Hours Per Day	Vidutinis aktyvių valandų skaičius per dieną.	Botai veikia nuolat, o žmogaus aktyvumas būna ribotas pagal dienos laiką.
Activity Consistency	Intervalų tarp transakcijų variacija.	Botų veikla pasižymi pastoviais intervalais tarp transakcijų.

Šaltinis: sudaryta autoriaus.

Požymių inžinerijos kūrimas ir validacija prasidėjo nuo pradinės duomenų analizės ir domeno žinių kaupimo. Pradiniai duomenys buvo analizuojami siekiant suprasti jų struktūrą ir

galimybes. Blockchain ekspertai ir decentralizuotų finansų (DeFi) operatoriai dalijosi žiniomis apie žinomus botų modelius, kurie leido identifikuoti svarbiausias savybes.

Toliau buvo atliekama žinomų botų elgesio analizė. Buvo tiriami arbitrage botai, MEV botai ir prekybos botai, siekiant suprasti jų laiko, programų naudojimo ir transakcijų modelius. Remiantis šiomis įžvalgomis, buvo kuriamos hipotezės apie potencialius požymius, kurie galėtų padėti atskirti botų elgesį nuo žmonių.

Iš pradžių buvo sukurti baziniai požymiai, kurie vėliau buvo ištestuoti statistiniais metodais, tokiais kaip koreliacijos ir reikšmingumo testai, bei mašininio mokymosi algoritmų pagalba, pavyzdžiui, K-means, DBSCAN ir Isolation Forest. Ypač svarbūs pasirodė techniniai rodikliai, tokie kaip instrukcijų sudėtingumas ir vaidmenų įvairovė, nes jie atskleidė botų operacijų pastovumą ir specifiką. Galiausiai, sukurti požymiai buvo ištestuoti su žinomų botų ir žmonių transakcijomis realiojo pasaulio sąlygomis. Šis žingsnis padėjo patvirtinti požymių diskriminacinį pajėgumą ir užtikrinti, kad jie būtų naudingi praktiniuose scenarijuose.

Kodėl šie požymiai buvo pasirinkti?

Šie požymiai buvo pasirinkti dėl jų gebėjimo tiksliai atskleisti automatizuotą veiklą ir jų efektyvumo, naudojant klasterizacijos algoritmus. Vienas iš svarbiausių kriterijų buvo diskriminacinė galia – požymiai pasižymi aukštu botų ir žmonių elgsenos atskyrimo tikslumu, užtikrina nuoseklius rezultatus ir žemą klaidingai teigiamų atvejų skaičių. Taip pat buvo atsižvelgta į skaičiavimo efektyvumą. Pasirinkti požymiai lengvai apskaičiuojami naudojant turimą Solanos tinklo informaciją, reikalauja minimalių resursų ir gali būti greitai apdorojami.

Pasirinkti požymiai papildo vienas kitą ir apima skirtingus elgsenos aspektus. Jie dirba kartu, kad pateiktų išsamų automatizuotos veiklos vaizdą. Požymiai taip pat yra atsparūs bandymams juos apeiti ar paslėpti botų veiklą, nes remiasi fundamentaliais elgsenos principais, kuriuos sunku imituoti. Galiausiai, požymiai išlieka informatyvūs ir patikimi, nepaisant kintančių tinklo sąlygų. Jie gali būti naudojami ilgalaikiam botų aptikimui ir stebėjimui, užtikrinant nuoseklų ir patikimą veikimą. Taip pat pasižymi praktiniais privalumais, tokiais kaip duomenų prieinamumas, skaičiavimo paprastumas ir galimybė juos atnaujinti bei prižiūrėti. Jie puikiai sąveikauja su pasirinktais mašininio mokymosi algoritmais: K-means algoritmas identifikuoja elgsenos grupes, DBSCAN atranda netipines tankio zonas, o Isolation Forest aptinka anomalijas remiantis šiais požymiais.

Požymių kūrimas ir pasirinkimas buvo iteratyvus ir duomenimis grįstas procesas, kurio metu didelis dėmesys buvo skiriamas praktiniam efektyvumui, įgyvendinimo galimybėms, aptikimo tikslumui, skaičiavimo našumui ir ilgalaikiam patikimumui. Gauti požymiai užtikrina aukštą botų aptikimo tikslumą ir efektyvumą, naudojant pasirinktus mašininio mokymosi algoritmus Solanos tinkle.

Apibendrinant, duomenų apdorojimas ir kintamųjų inžinerija yra esminiai žingsniai, siekiant paruošti Solanos tinklo duomenis botų aptikimo analizei. Požymių atranka buvo vykdoma, atsižvelgiant į jų įtaką, skaičiavimo efektyvumą ir suderinamumą su mašininio mokymosi algoritmais.

3.3 Duomenų aprašomoji statistika

Šiame skyriuje aprašomos duomenų aprašomosios statistikos taikymo metodikos, skirtos analizuoti botų veiklos požymius Solanos tinklo sandorių duomenyse. Naudojamos aprašomosios statistikos priemonės padeda geriau suprasti pagrindines duomenų charakteristikas, nustatyti anomalijas ir įvertinti botų veiklos modelius.

11 lentelė

Centrinės tendencijos

Metrika	Transakcijų dažnis	Vidutinė transakcijos suma	Aktyvios valandos per dieną
Vidurkis	0.117117	0.323475	0.391671
Mediana	0.026898	0.294437	0.230909
Moda	0.206384	0.038064	0

Šaltinis: sudaryta autoriaus.

Transakcijų dažnis buvo pasirinktas, nes tai vienas pagrindinių rodiklių, rodantis, kaip dažnai piniginė sąveikauja su blokų grandine. Ši metrika leidžia atskirti robotus, kurie paprastai vykdo dažnas ir nuolatines operacijas, nuo žmonių, kurie transakcijas vykdo rečiau ir netolygiais intervalais. Transakcijų dažnio metrika (žr. 10 lentelę) rodo dešinįjį pasiskirstymo šališkumą. Aukštesnis vidurkis, palyginti su mediana, reiškia, kad nors dauguma piniginių veikia mažo dažnio režimu, kelios itin aktyvios piniginės ženkliai padidina vidurkį. Tai dar labiau patvirtina moda, atspindinti dažniausiai pasitaikantį transakcijų dažnį duomenų rinkinyje. Toks pasiskirstymo šališkumas leidžia išskirti dvi skirtingas elgsenos grupes: mažo aktyvumo pinigines, greičiausiai

atspindinčias žmonių elgesį, ir didelio aktyvumo pinigines, kurios gali būti susijusios su robotais. Tolimesnė piniginių, kurių dažnis yra aukštesnis už vidurkį, analizė galėtų padėti efektyviau identifikuoti robotų elgesio modelius.

Vidutinė transakcijos suma parodo, kokia yra tipiška piniginės operacijų vertė. Robotai dažnai vykdo mažas ir pastovias transakcijas, tuo tarpu žmonių transakcijos yra labiau įvairios. Vidutinės transakcijos sumos metrika (žr. 10 lentelę) rodo simetriškesnį pasiskirstymą, palyginti su transakcijų dažniu. Tačiau moda, kuri yra reikšmingai žemesnė, atspindi dažniausiai pasitaikančią transakcijos sumą. Ši žema moda gali reikšti mikrotransakcijas, kurios dažnai būdingos robotų veiklai. Tuo tarpu vidurkis ir mediana rodo šiek tiek platesnį transakcijų sumų diapazoną. Ši stebėsena leidžia manyti apie dviejų elgsenos grupių buvimą: robotus, veikiančius su mažomis, nuosekliomis transakcijomis, ir žmones, vykdančius transakcijas su labiau įvairiomis sumomis. Vidurkio ir medianos artumas leidžia manyti, kad abi grupės daro panašią įtaką duomenų rinkiniui, tačiau išsamesnė mažų transakcijos sumų analizė galėtų atskleisti aiškesnius robotų elgesio modelius.

Aktyvios valandos per dieną parodo, kiek laiko piniginė vidutiniškai aktyvi per dieną. Robotai paprastai būna aktyvūs visą parą, o žmonės – tik tam tikru paros metu, pavyzdžiui, darbo valandomis arba nebūna aktyvus kas diena. Aktyvių valandų per dieną metrika (žr. 10 lentelę) rodo, kad piniginės vidutiniškai yra aktyvios mažiau nei pusvalandį per dieną. Mediana, mažesnė nei vidurkis, atspindi dešinįjį šališkumą: mažuma piniginių yra aktyvios ilgą laiką, tikėtina, dėl automatizuoto robotų veikimo. Moda, kuri rodo neveiklumą kaip dažniausią būseną, gali reikšti arba neveiklias pinigines, arba žmonių vartotojus, kurie įsitraukia tik sporadiškai. Vidurkio ir modos kontrastas atskleidžia esminį skirtumą tarp automatizuoto ir rankinio elgesio, kur robotai pasižymi ilgesniu ir nuoseklesniu aktyvumu, o žmonių vartotojai – labiau epizodišku ir ribotu įsitraukimu.

. Variacijos rodikliai, tokie kaip standartinis nuokrypis, diapazonas ir interkvartilinis diapazonas, suteikia papildomų įžvalgų apie tai, kaip plačiai piniginių aktyvumo rodikliai skiriasi nuo vidurkių. Tai leidžia nustatyti, ar elgsenos modeliai yra vienodi visoje populiacijoje, ar egzistuoja ryškūs skirtumai tarp žmonių ir robotų veiklos (žr. 11 lentelę).

12 lentelė

Duomenų variacijos rodikliai

Metrika	Transakcijų dažnis	Vidutinė transakcijos suma	Transakcijos sumų standartinis nuokrypis	Aktyvios valandos per dieną

Standartinis nuokrypis	0.195167	0.295399	0.269087	0.345586
Interkvartilinis diapazonas (IQR)	0.200668	0.54632	0.517681	0.717165

Šaltinis: sudaryta autoriaus.

Standartinio nuokrypio vertės parodo bendrą kiekvienos metrikos duomenų variaciją. Mažiausią standartinį nuokrypį turi Transakcijų dažnis, kas rodo, jog daugumos piniginių aktyvumo lygiai yra gana panašūs. Tuo tarpu Vidutinė transakcijos suma ir Transakcijos sumų standartinis nuokrypis demonstruoja didesnę variaciją, kas atspindi skirtingas piniginių transakcijų vertes. Aktyvios valandos per dieną turi didžiausią standartinį nuokrypį, kas rodo reikšmingus skirtumus tarp piniginių aktyvumo laikotarpių. Tai leidžia atskirti robotus, kurie būna aktyvūs ilgesnį laiką, nuo žmonių, kurie veikia epizodiškai.

Interkvartilinis diapazonas (angl. *IQR*) suteikia papildomų įžvalgų apie duomenų pasiskirstymo vidurinius 50 %. Transakcijų dažnis pasižymi mažiausiu IQR, kas patvirtina, jog dauguma piniginių veikia siaurame aktyvumo diapazone. Tuo tarpu Vidutinė transakcijos suma ir Transakcijos sumų standartinis nuokrypis rodo platesnius IQR, kas atspindi įvairesnę transakcijų elgseną. Aktyvios valandos per dieną turi plačiausią IQR, kas reiškia reikšmingą dienos aktyvumo laikų įvairovę tarp piniginių. Šis rodiklis pabrėžia šios metrikos svarbą skiriant žmonių ir automatizuotą veiklą.

Diapazonas (angl. *Range*) buvo vienodas (lygus 1) visoms metrikoms, nes duomenys buvo normalizuoti į [0, 1] intervalą. Nors diapazono įtraukimas gali būti naudingas pradiniame duomenų analizės etape ar norint patvirtinti, kad normalizacija buvo atlikta teisingai, šiuo atveju jis neatskleidžia papildomos reikšmingos informacijos apie duomenų variaciją. Vienoda reikšmė visose metrikose atspindi tik duomenų apdorojimo etapą, o ne realius duomenų pasiskirstymo skirtumus. Todėl Diapazono stulpeli pašalinau iš lentelės, siekiant aiškesnės ir labiau į veiksmingus rodiklius orientuotos informacijos.

Apibendrinant, Standartinio nuokrypio ir IQR rodikliai leidžia pastebėti kelis svarbius modelius. Transakcijų dažnio mažas standartinis nuokrypis ir IQR rodo, kad dauguma piniginių vykdo panašų kiekį operacijų, išskyrus kelias išsiskiriančias, greičiausiai robotų valdomas pinigines. Tuo tarpu Vidutinė transakcijos suma ir jos variacija atskleidžia platesnį pasiskirstymą, kas gali atspindėti robotų mažas ir pastovias transakcijas bei žmonių didesnį operacijų įvairumą. Aktyvios valandos per dieną, turinčios didžiausią dispersiją, išryškina skirtumus tarp robotų nuolatinio aktyvumo ir žmonių epizodiško naudojimosi sistema.

Šališkumas ir kurtosis rodiklai leidžia giliau suprasti, kaip duomenys yra pasiskirstę, ar jie atspindi natūralius elgsenos modelius, ar rodo nenatūralius, automatizuotos veiklos požymius. Šie rodikliai suteikia papildomų įžvalgų apie galimą robotų ir žmonių veiklos atskyrimą pagal duomenų struktūrą (žr. 12 lentelę).

13 lentelė

Pasiskirstymo formos rodikliai

Metrika	Transakcijų dažnis	Vidutinė transakcijos suma	Aktyvios valandos per dieną	Aktyvumo pastovumas
Šališkumas (Skewness)	2.69	0.44	0.76	0.24
Kurtosis (Kurtosis)	7.77	-1.02	-1.19	-1.43

Šaltinis: sudaryta autoriaus.

Transakcijų dažnio metrika demonstruoja stiprų šališkumą ir labai aukštą smailumą, kas yra klasikinis robotų veiklos požymis. Toks pasiskirstymas rodo automatizuotus, pasikartojančius prekybos modelius, kurie labai skiriasi nuo natūralios žmonių veiklos. Žmonių elgsenoje paprastai pasitaiko didesnė transakcijų dažnio įvairovė, todėl toks tolygus aktyvumas leidžia įtarti sistemingai valdomas pinigines.

Vidutinės transakcijos sumos metrika pasižymi neigiamu kurtosiu, kas parodo „pernelyg tobulą“ atsitiktinumą. Tai reiškia, kad elgsena atrodo dirbtinai subalansuota, kas gali reikšti programuotą bandymą imituoti natūralų atsitiktinumą. Toks plokštesnis pasiskirstymas nebūdinga žmonėms, kurie paprastai vykdo tiek mažas, tiek didesnes transakcijas, o robotai, priešingai, demonstruoja nuosekliai pasikartojančius modelius.

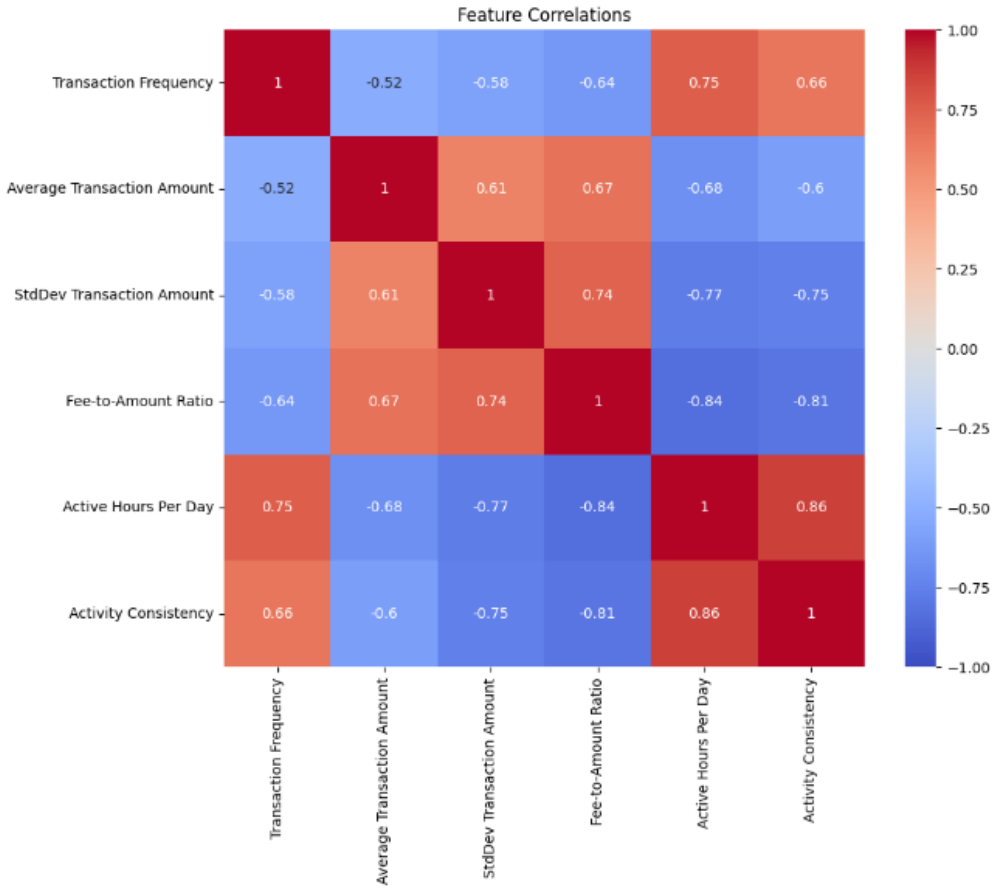
Aktyvių valandų per dieną metrika rodo vienodą aktyvumo pasiskirstymą be aiškių poilsio laikotarpių, kas nėra būdinga natūraliai žmogaus veiklai. Tai pabrėžia nenatūralų nuoseklumą, kai veikla vykdoma tolygiai per visus laikotarpius. Žmonių elgsena dažnai priklauso nuo dienos ritmo ir kitų aplinkybių, kas šioje analizėje nėra pastebima.

Aktyvumo pastovumo metrika taip pat demonstruoja pernelyg vienodą elgesį, kas rodo programuotą veiklą. Tokie rodikliai, kaip vienodas aktyvumo lygis ir dirbtinis atsitiktinumas, yra pagrindiniai robotų veiklos indikatoriai. Natūrali žmonių veikla pasižymi didesniu variacijų diapazonu, kuris šiuo atveju nėra pastebimas.

Apibendrinant, šie rodikliai kartu parodo sistemingą elgesį, kuris smarkiai skiriasi nuo natūralių žmogaus veiklos modelių. Aukštas kurtosis rodiklis transakcijų dažnyje ir neigiamas

kurtosis kitose metrikose atspindi programuotą atsitiktinumą ir dirbtinai sukurtą tolygumą. Tai rodo, kad nagrinėjamuose duomenyse greičiausiai yra robotų veiklos požymių, o žmonių elgesys tokiose metrikose nebūtų toks nuoseklus ir vienodas. Toks sistemingas elgesys yra aiškus automatizuotos veiklos, būdingos prekybos robotams, požymis.

Koreliacijos tarp skirtingų funkcijų padeda suprasti, kaip šios metrikos tarpusavyje susijusios, ir leidžia atpažinti elgesio modelius, būdingus tiek žmonių, tiek automatizuotų sistemų veiklai. Vertinant funkcijų sąveiką, galima giliau įsigilinti į duomenų struktūrą ir atskleisti



esminius robotų bei žmonių elgsenos skirtumus.

Šaltinis: sudaryta autoriaus.

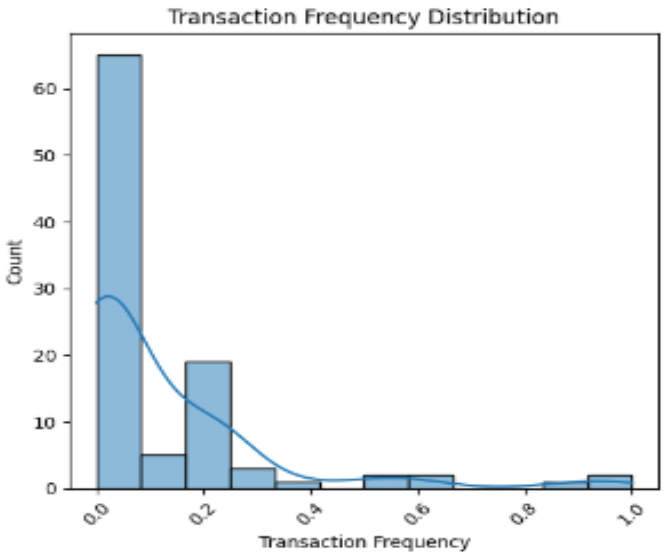
7 pav. Piniginės adreso įvedimo vartotojo sąsaja

Stiprios teigiamos ir neigiamos koreliacijos tarp tam tikrų metrikų rodo sistemingą ir automatinį elgesį, kuris skiriasi nuo natūralios žmonių veiklos, dažnai pasižyminčios atsitiktinumu ir nenuoseklumu (žr. 6 paveikslą). Viena iš stipriausių teigiamų koreliacijų nustatyta tarp Aktyvių valandų per dieną ir Aktyvumo pastovumo. Šis ryšys rodo, kad kuo ilgiau piniginė yra aktyvi, tuo nuoseklesnis jos elgesys, kas būdinga automatizuotiems robotų modeliams. Taip pat reikšmingas teigiamas ryšys matomas tarp Transakcijų dažnio ir Aktyvių valandų per dieną, kas leidžia manyti, kad daugiau valandų aktyvumo tiesiogiai lemia dažnesnes operacijas – tai tipiška automatizuotos prekybos strategija.

Stiprios neigiamos koreliacijos taip pat išskiria ryškų robotų elgesio modelį. Pavyzdžiui, stipri neigiama koreliacija tarp Standartinio transakcijų sumų nuokrypio ir Aktyvių valandų per dieną rodo, kad kuo ilgiau piniginė veikia, tuo mažesnė yra operacijų vertės variacija. Tai rodo nuoseklias, suplanuotas transakcijas, kurios nėra būdingos žmonių veiklai. Panašiai labai stipri neigiama koreliacija tarp Mokesčių ir sumos santykio bei Aktyvių valandų per dieną atskleidžia sistemingą mokesčių valdymą – tai dar vienas būdingas robotų požymis.

Apibendrinant, stiprios koreliacijos tarp šių metrikų rodo sistemingą ir struktūrizuotą elgseną, būdingą robotams. Šie rezultatai pabrėžia, kad natūrali prekyba dažnai pasižymi atsitiktinumu, o tokie ryškūs ir logiškai paaiškinami ryšiai tarp funkcijų dažniausiai signalizuoja

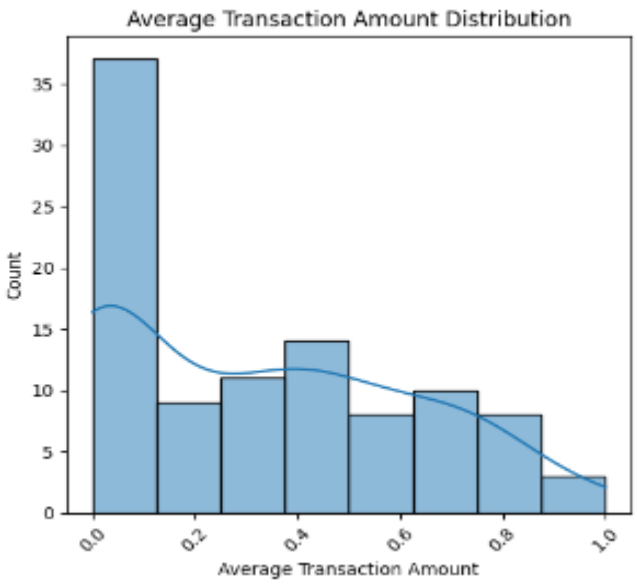
apie automatizuotą sprendimų priėmimą ir optimizaciją. Toks sistemingumas rodo, kad pasirinktos funkcijos yra itin tinkamos robotų veiklos analizei ir identifikavimui.



Šaltinis: sudaryta autoriaus.

8 pav. Transakcijų dažnio pasiskirstymas

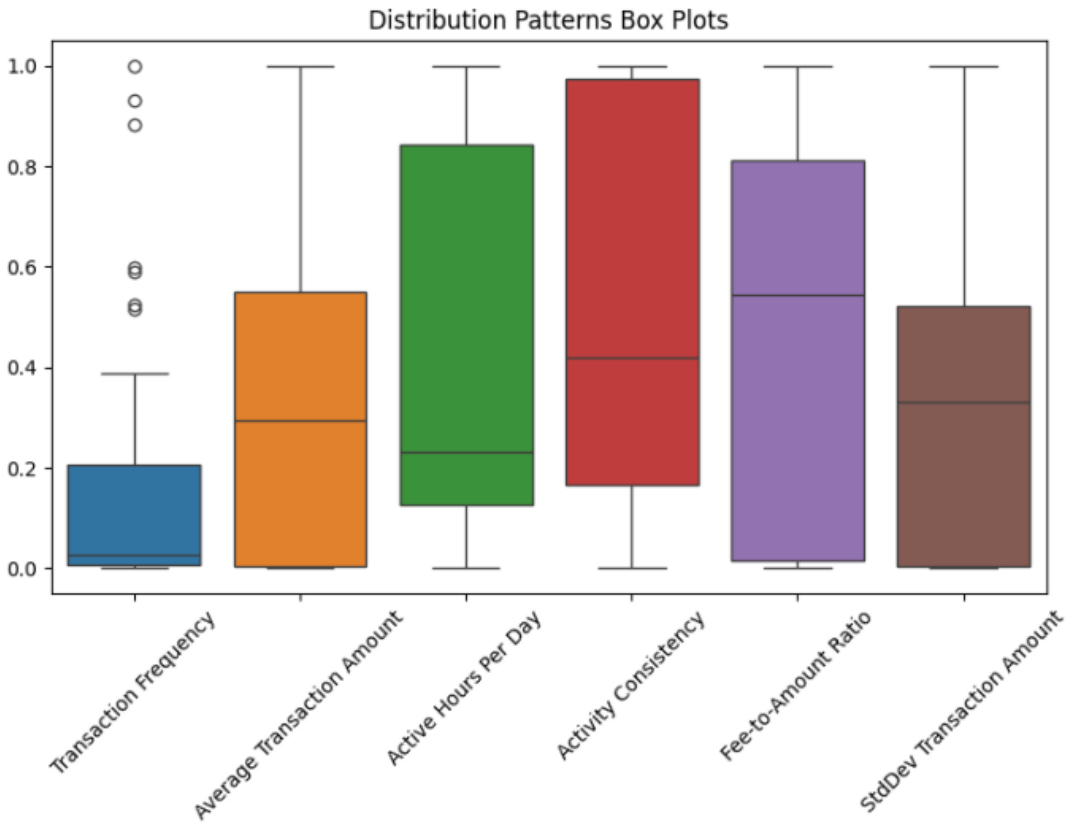
Transakcijų dažnio pasiskirstymo analizė parodo staigų dažnių koncentracijos padidėjimą (žr. 7 paveikslą), pradinėje histogramoje, kuris greitai mažėja, sudarydamas stipriai dešinį šališkumą. Didžioji transakcijų dalis yra labai žemo dažnio diapazone, o aukštesniuose dažniuose pastebimas ilgas uodegos (angl. *Long tail*) modelis su mažesniu, bet pastoviu aktyvumu. Toks modelis atspindi sistemingai suplanuotus prekybos intervalus, kurie būdingi robotams. Histogramoje nėra sklandžių perėjimų tarp skirtingų piko taškų, o tai rodo ne natūralų, o algoritminį aktyvumą, pagrįstą automatizuota prekybos logika.



Šaltinis: sudaryta autoriaus.

9 pav. Vidutinės transakcijos sumos pasiskirstymas

Vidutinės transakcijos sumos pasiskirstymo analizė atskleidžia aiškų struktūrizuotą grupavimą pagal specifines sumas, kas rodo automatizuotą sprendimų priėmimą. Histogramoje matomi ryškūs žingsnių modeliai (angl. Steps), kur transakcijų sumos pasiskirsto pagal griežtai apibrėžtus intervalus. Toks modelis rodo, kad operacijos yra iš anksto suplanuotos ir vykdomos pagal algoritmus, o ne natūraliai priimamus sprendimus, būdingus žmonių elgsenai. Grupavimas pagal specifinius intervalus leidžia robotams vykdyti efektyvias operacijas su mažomis pasikartojančiomis sumomis, dažnai siekiant sumažinti mokesčius ar optimizuoti prekybos strategijas. Staigūs perėjimai tarp verčių patvirtina nenatūralų atsitiktinumą, būdingą programinei prekybai. Analizė pabrėžia robotų prekybos logikos prioritetą tikslumui ir nuoseklumui, išskirdama ją iš žmonių elgsenos.



Šaltinis: sudaryta autoriaus.

10 pav. Požymių pasiskirstymo Box Plot analizė

Box plot vizualizacijos atskleidžia aiškius robotų veiklos požymius, kurie atsispindi sistemingose metrikų variacijose ir suplanuotuose elgesio modeliuose (žr. 8 paveikslą). Transakcijų dažnis pasižymi maža mediana ir siauru interkvartiliniu diapazonu (IQR), kas rodo nuoseklų elgesį su retkarčiais pasitaikančiais staigiais aktyvumo pliūpsniais (angl. *outliers*). Vidutinės transakcijos sumos platus IQR ir simetriškas pasiskirstymas rodo įvairias, bet suplanuotas transakcijas. Tuo tarpu Aktyvumo pastovumo ir Aktyvių valandų per dieną metrikos demonstruoja didelę variaciją ir išnaudotą pilną diapazoną, kas patvirtina suplanuotą ir nuoseklų robotų aktyvumą. Mokesčių ir sumos santykis rodo vidutinišką sklaidą ir nuoseklią mokesčių kontrolę, kas atspindi automatizuotą strategijų optimizaciją

Apibendrinant, Duomenų aprašomosios statistikos skyriuje atlikta analizė parodė reikšmingus skirtumus tarp automatizuotos ir natūralios elgsenos modelių. Pasiskirstymo formos rodikliai atskleidė sistemingą šališkumą ir kurtosis reikšmes, būdingas robotų veiklai, o funkcijų koreliacijos išryškino stiprius ryšius tarp metrikų, pabrėžiančius nuoseklų ir suplanuotą aktyvumą. Pasiskirstymo modeliai, nagrinėti per histogramas ir dėžinių diagramų analizę, papildomai parodė

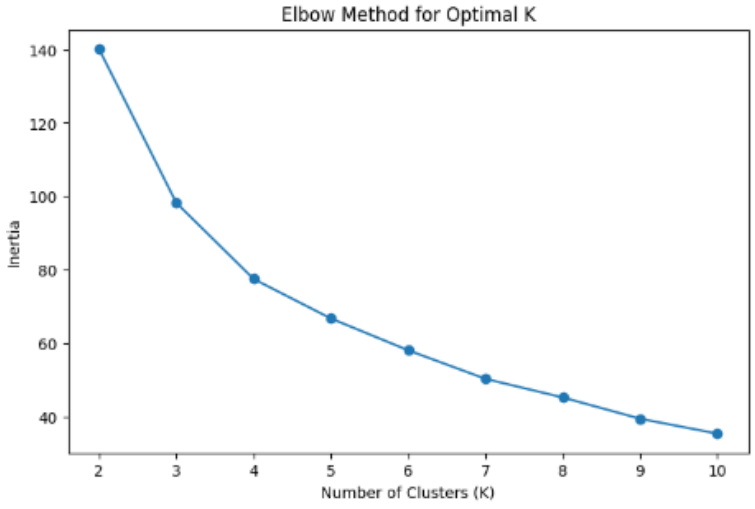
automatizuotų prekybos sprendimų struktūrizuotumą ir strateginį tikslumą, leidžiantį efektyviai atskirti robotų veiklą nuo žmonių.

3.4 Klasterizacijos ir anomalijų aptikimas

Šiame skyriuje bus nagrinėjami klasterizacijos ir anomalijų aptikimo metodai, siekiant analizuoti piniginių elgsenos modelius Solana ekosistemoje. Klasterizacijos procese bus taikomi K-means ir DBSCAN algoritmai, kurie leis grupuoti pinigines pagal jų elgesio panašumus, atskleidžiant pagrindines elgsenos struktūras. K-means algoritmas leis identifikuoti aiškiai apibrėžtus klasterius, kai tuo tarpu DBSCAN metodas padės aptikti tankiai išsidėsčiusius klasterius bei atpažinti triukšmą ar išskirtines reikšmes. Anomalijų aptikimui bus naudojamas Isolation Forest algoritmas, kuris identifikuos elgsenos anomalijas.

3.4.1 K-means klasterizacijos taikymas elgsenos modelių identifikavimui

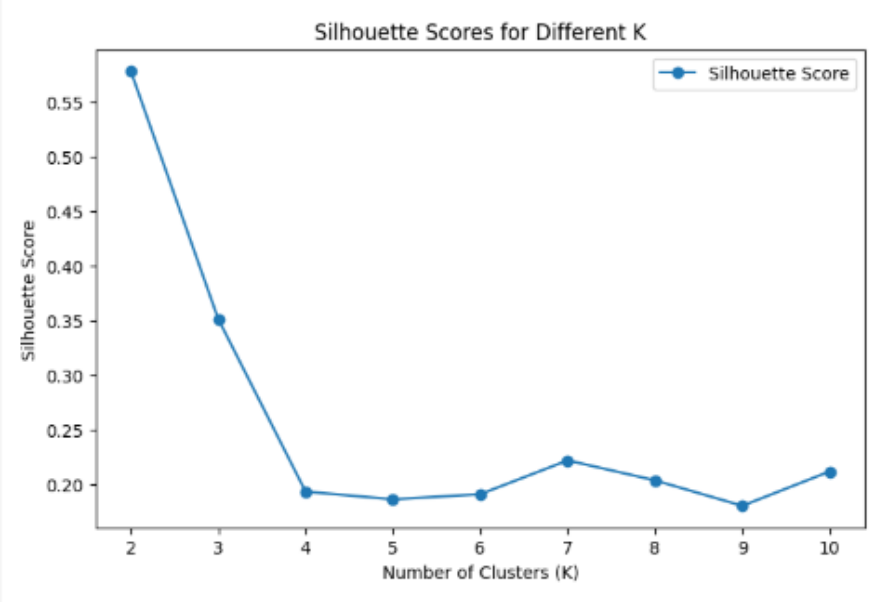
Atlikus K means metodu klasterizacija buvo nustatytas optimalus klasterių skaičius. Remiantis Elbow metodo ir Silhouette balų analize, K=3 užtikrina geriausią klasterizacijos kokybės ir modelio sudėtingumo pusiausvyrą, leidžiančią tiksliai grupuoti duomenis ir išvengti per didelio klasterių skaičiaus.



Šaltinis: sudaryta autoriaus.

11 pav. Optimalaus klasterių skaičiaus (K) paieška naudojant Elbow metodą

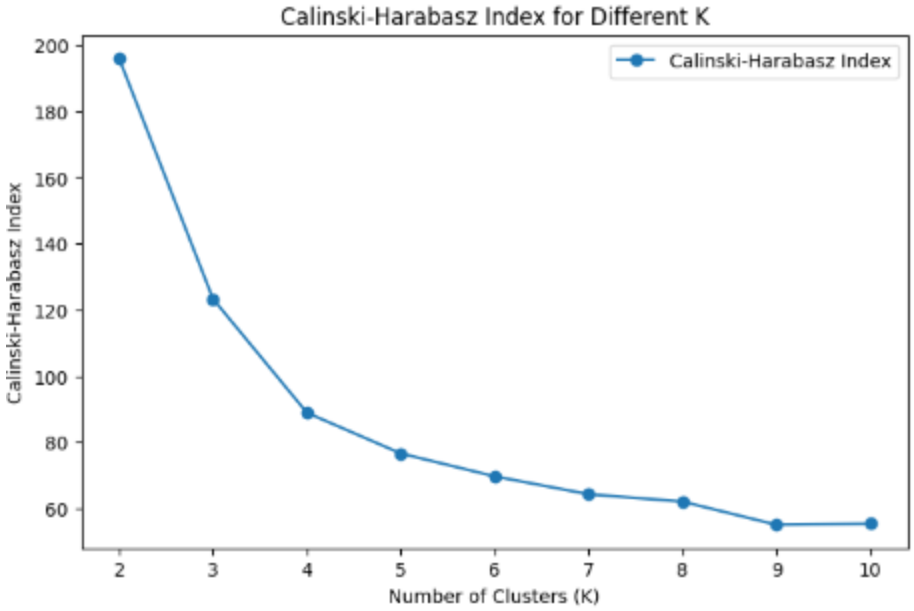
Elbow metodą (žr. 9 paveikslą), iliustruoja klasterių skaičiaus (K) ir inercijos reikšmių sąryšį. Pastebima, kad inercijos reikšmės sparčiai mažėja nuo K=2 iki K=3, tačiau vėliau mažėjimo tempas ženkliai sulėtėja. Šis „alkūnės“ taškas ties K=3 nurodo optimalų klasterių skaičių, užtikrinantį pusiausvyrą tarp klasterizacijos kokybės ir sudėtingumo. Šio taško pasirinkimas remiasi siekiu išvengti per didelio klasterių skaičiaus, kuris galėtų apsunkinti interpretaciją ir neefektyviai padidinti skaičiavimų sudėtingumą. Užtikrina, kad klasteriai bus aiškiai atskirti, o rezultatai – lengviau interpretuojami.



Šaltinis: sudaryta autoriaus.

12pav. Silhouette Score priklausomybė nuo Cluster K skaičiaus

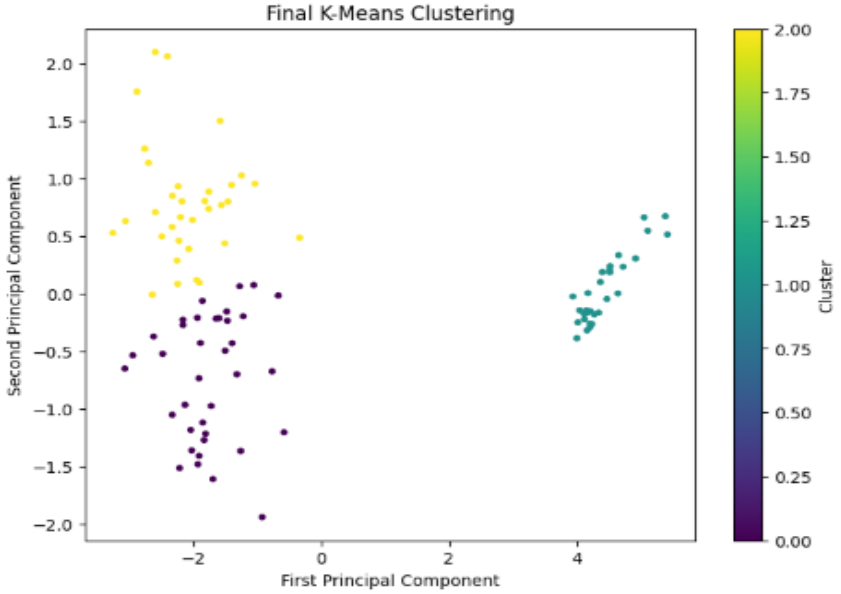
Grafikas iliustruoja Silhouette balų kaitą, priklausomai nuo klasterių skaičiaus (K). Didžiausias Silhouette balas pastebimas ties K=2, tačiau K=3 taip pat išlaiko pakankamai aukštą reikšmę, rodydamas gerą klasterių atskyrimo ir vidinės konsistencijos balansą. Mažėjant balui nuo K=4 ir toliau, pastebima, kad klasterių kokybė prastėja dėl didesnių vidinių skirtumų tarp klasterių arba nepakankamo jų atskyrimo. Analizė papildo Elbow metodo įžvalgas, pateikdama statistinį pagrindimą pasirinktam klasterių skaičiui.



Šaltinis: sudaryta autoriaus.

13 pav. Calinski-Harabasz indekso priklausomybė nuo klasterių skaičiaus (K)

Calinski-Harabasz indekso reikšmių pokytį didėjant klasterių skaičiui (K). Aukščiausia indekso reikšmė pasiekiama ties K=2, o ties K=3 reikšmė siekia **123.2795**, nurodanti gerą klasterių vidinės konsistencijos ir jų tarpusavio atskyrimo balansą. Toliau didinant klasterių skaičių, indekso reikšmė ženkliai mažėja, kas rodo silpnesnį klasterizacijos efektyvumą. Aukštos reikšmės ties K=3 patvirtina, kad klasteriai yra pakankamai tankūs viduje ir gerai atskirti vienas nuo kito.

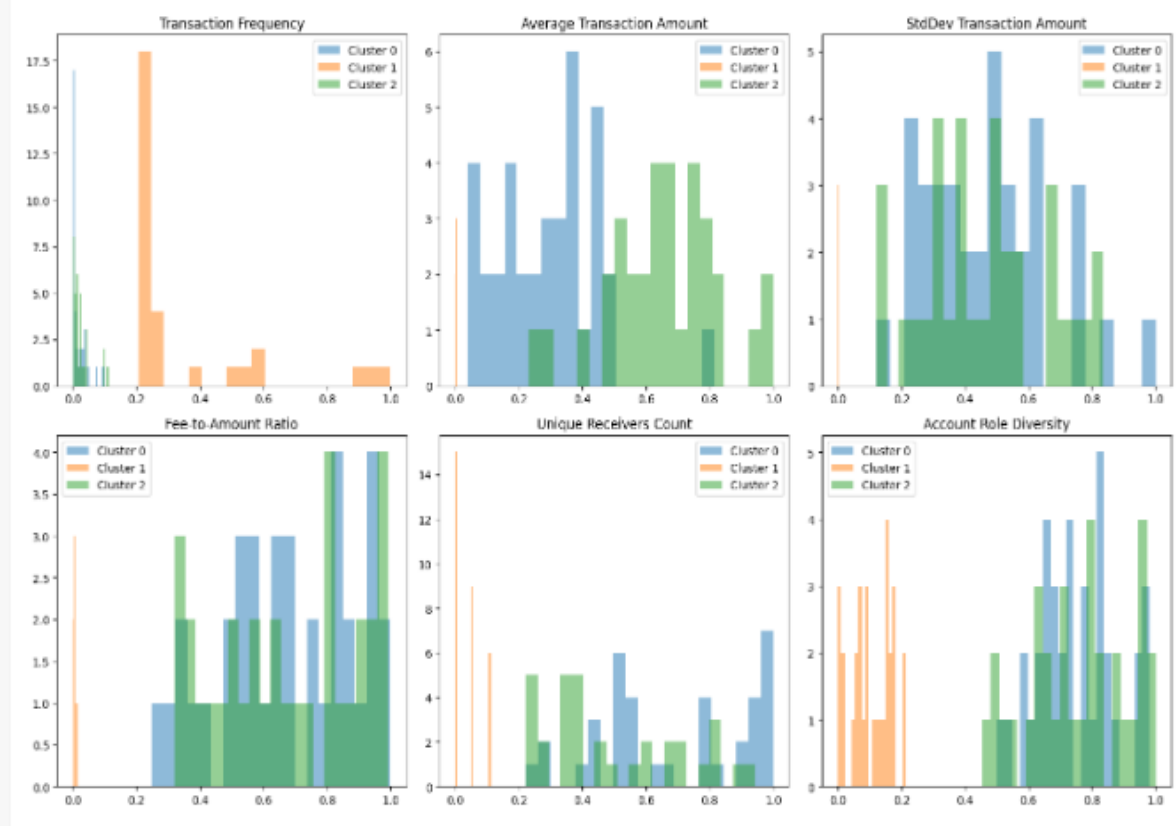


Šaltinis: sudaryta autoriaus.

14 pav. Galutinė K-means klasterizacija pagrindinių komponentų plokštumoje

Galutinis K-means klasterizacijos rezultatas su optimaliu klasterių skaičiumi K=3. Grafikas atvaizduoja duomenų klasterius pagrindinių komponentų erdvėje, siekiant sumažinti duomenų dimensijų skaičių ir vizualiai pabrėžti klasterių struktūrą. Pagrindinių komponentų analizė (PCA) leidžia efektyviai suspausti informaciją ir aiškiai atskirti klasterius.

Paveiksle matyti trys klasteriai, pažymėti skirtingomis spalvomis. Kairiajame grafiko krašte esantis geltonasis ir violetinis klasteriai turi tam tikrą sąryšį, tačiau jų atskyrimas yra aiškus, o tai rodo, kad kiekvienas klasteris turi savitų ypatybių. Dešiniajame krašte esantis žalsvas klasteris yra ryškiai atskirtas nuo kitų dviejų klasterių, kas pabrėžia jo unikalias charakteristikas. Toks aiškus atskyrimas rodo gerą K-means modelio veikimą ir tinkamą pasirinkto K=3 parametrą.



Šaltinis: sudaryta autoriaus.

15pav. Klasterių ypatybių pasiskirstymo analizė pagal elgesio metrikas

Elgesio metrikų pasiskirstymą atskiruose klasteriuose, Transakcijų dažnis, Vidutinė transakcijos suma, Transakcijos sumų standartinis nuokrypis, Mokestinio efektyvumo rodiklis (Fee-to-Amount Ratio), Unikalių gavėjų skaičius ir Sąskaitų vaidmenų įvairovė (angl. *Account*

Role Diversity). Vizualizacijos leidžia įvertinti kiekvienos metrikos variaciją ir elgesio skirtumus tarp trijų klasterių. Matosi aiškūs skirtumai tarp klasterių. Pavyzdžiui, Klasteris 0 išsiskiria mažu transakcijų dažniu ir siauru pasiskirstymu, kas rodo mažą aktyvumą, galimai būdingą žmonių elgesiui. Klasteris 1 turi aukštesnį transakcijų dažnį ir didesnę unikalių gavėjų įvairovę, kas gali būti rodiklis sudėtingesnės prekybos elgsenos arba robotų veiklos. Tuo tarpu Klasteris 2 pasižymi aukštesne vidutine transakcijos suma ir nuosekliu mokestiniu efektyvumu, kas gali atspindėti strategines arba automatizuotas prekybos operacijas. Tai parodo skirtingus prekybos elgesio modelius kiekviename klasteryje. Pabrėžia esminius rodiklius, kurie padeda atskirti žmonių ir robotų veiklą Solana ekosistemoje.

14 lentelė

Klasterių analizės apibendrinti duomenys

Metrika	Reikšmė
Atstumas tarp klasterių 0 ir 1	6.22
Atstumas tarp klasterių 0 ir 2	1.55
Atstumas tarp klasterių 1 ir 2	6.56
Klasterio 0 variacija	1.07
Klasterio 1 variacija	0.46
Klasterio 2 variacija	1
Klasterio 0 dydis	37 (37.0%)
Klasterio 1 dydis	30 (30.0%)
Klasterio 2 dydis	33 (33.0%)
Vidutinė transakcijos suma	0.33 vienetai nuo vidurkio
Sėkmės santykis	0.25 vienetai nuo vidurkio
Programų pasiskirstymo rodiklis	0.23 vienetai nuo vidurkio

Šaltinis: sudaryta autoriaus.

Klasterių analizės duomenys leidžia daryti kelias svarbias išvadas apie klasterių struktūrą ir jų elgesio modelius (žr. 13 lentelę). Pirma, atstumų tarp klasterių analizė rodo, kad kai kurie klasteriai turi aiškų tarpusavio atskyrimą, pavyzdžiui, vienas iš jų pasižymi mažesniu atstumu su kitu klasteriu, kas gali rodyti tam tikrus panašumus elgesio modeliuose. Tuo tarpu didesni atstumai tarp kitų klasterių rodo reikšmingus skirtumus, kas yra naudinga analizuojant skirtingas grupių ypatybes. Klasterių vidinės variacijos skirtumai taip pat yra reikšmingi. Vienas iš klasterių pasižymi itin homogeniškų elgesiu, o tai gali reikšti, kad ši grupė yra gana vienoda pagal savo elgesį ir atspindi tam tikrą specifinę strategiją ar veiklos pobūdį. Kiti klasteriai turi didesnę variaciją, kas rodo, jog šių grupių piniginės vykdo įvairesnes operacijas ir galimai apima tiek

žmogaus, tiek automatizuotos veiklos modelius. Klasterių dydžių pasiskirstymas yra gana subalansuotas. Vieno klasterio, kuris sudaro didesnę duomenų dalį, dydis gali reikšti jo bendrą svarbą sistemoje. Tačiau mažesni klasteriai, ypač turintys mažiau vidinės variacijos, gali būti susiję su konkretesniais ir aiškesniais elgesio modeliais.

Galiausiai, pateikti rodikliai, tokie kaip vidutinė transakcijos suma ir sėkmės santykis, atskleidžia ryškius skirtumus tarp klasterių. Vienas iš jų turi didesnę vidutinę transakcijos sumą, kas rodo efektyvesnę veiklą. Tuo tarpu mažesnis sėkmės santykis kitame klasteryje gali rodyti klaidų ar nesėkmingų operacijų dažnumą, būdingą automatizuotoms sistemoms arba mažiau efektyvioms piniginėms. Tai leidžia daryti prielaidą, kad klasteriai atspindi skirtingus tiek žmonių, tiek automatizuotų piniginių elgsenos modelius.

15 lentelė

Klasterizacijos vertinimo metrikos pagal K reikšmes

K Value	Mean Silhouette Score	Silhouette StdDev	Mean Calinski-Harabasz Score	Calinski-Harabasz StdDev
3	0.4511	±0.1433	86.0836	±42.8743
4	0.3397	±0.1620	61.9929	±30.8450
5	0.3077	±0.1629	54.8625	±30.0459
6	0.3301	±0.1528	58.6931	±33.6858
7	0.3168	±0.1688	53.7738	±29.1997
8	0.3365	±0.2012	61.8108	±35.7105
9	0.2485	±0.1113	59.4222	±32.9799
10	0.1797	±0.1285	57.6202	±32.7104

Šaltinis: sudaryta autoriaus.

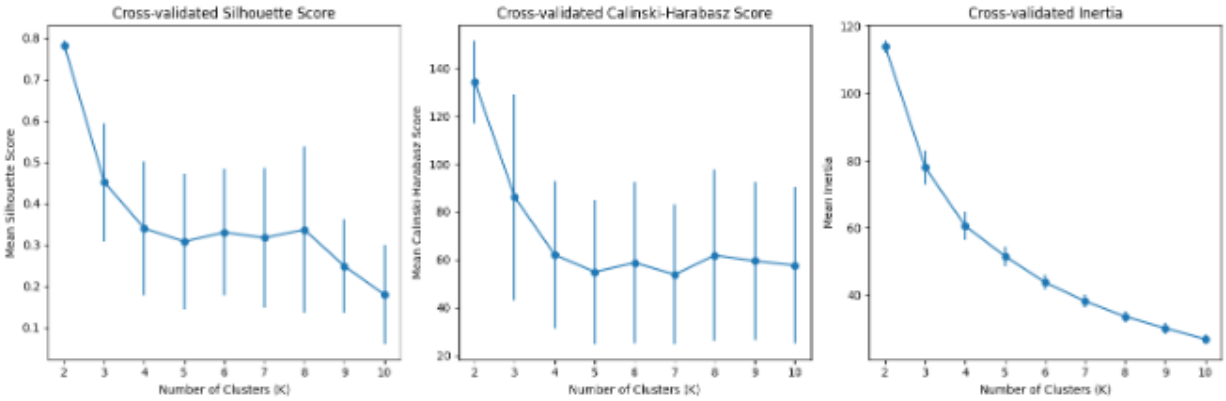
Matome K-means metodo rezultatus, gautus klasterizuojant duomenis su skirtingomis K reikšmėmis (nuo 2 iki 10). Kiekvienai K reikšmei pateikiami du rodikliai: "Mean Silhouette Score" (vidutinis silueto įvertis) ir jo standartinis nuokrypis ("Silhouette StdDev"), taip pat "Mean Calinski-Harabasz Score" (vidutinis Calinski-Harabasz indeksas) ir jo standartinis nuokrypis ("Calinski-Harabasz StdDev").

Analizuojant "Mean Silhouette Score" stulpelį, matome, kad didžiausias vidutinis silueto įvertis (0.7816) pasiekiamas, kai K = 2. Tai rodo, kad duomenys geriausiai išsiskiria į dvi grupes. Didėjant K reikšmei, vidutinis silueto įvertis mažėja, o tai reiškia, kad duomenų išskyrimas į daugiau klasterių tampa mažiau optimalus.

"Silhouette StdDev" stulpelis parodo silueto įverčio standartinį nuokrypį. Mažesnis standartinis nuokrypis rodo didesnį klasterių stabilumą ir homogeniškumą. Šiuo atveju, standartinis nuokrypis didėja, didėjant K reikšmei, o tai rodo, kad didesnis klasterių skaičius veda prie didesnės klasterių įverčių dispersijos.

"Mean Calinski-Harabasz Score" ir "Calinski-Harabasz StdDev" stulpeliai pateikia informaciją apie klasterių tankumą ir atskyrimą. Didesnis Calinski-Harabasz indeksas rodo geresnį klasterių atskyrimą. Lentelėje matome, kad didžiausias vidutinis Calinski-Harabasz indeksas (134.1836) pasiekiamas, kai K = 2, o didėjant K reikšmei, šis indeksas mažėja. Tai patvirtina ankstesnę išvadą, kad optimalus klasterių skaičius yra 2.

Apibendrinant, remiantis pateiktais K-means metodo rezultatais, galime daryti išvadą, kad duomenys geriausiai išsiskiria į dvi grupes (K = 2). Tai patvirtina tiek vidutinis silueto įvertis, tiek Calinski-Harabasz indeksas. Didėjant klasterių skaičiui, klasterizacijos kokybė mažėja, o klasterių stabilumas ir homogeniškumas suprastėja



Šaltinis: sudaryta autoriaus.

16 pav. Kryžminės validacijos metrikos pagal klasterių skaičių (K)

Kryžminės validacijos (cross-validation) rezultatus, gautus naudojant skirtingas K reikšmes (nuo 2 iki 10) K-means klasterizacijos metodui. Kryžminė validacija yra svarbi technika, leidžianti įvertinti modelio efektyvumą ir stabilumą, naudojant skirtingas duomenų padalijimo į mokymosi ir testavimo rinkinius strategijas.

Pirmasis grafikas rodo vidutinį silueto įvertį (Mean Silhouette Score) kiekvienai K reikšmei. Silueto įvertis matuoja, kaip gerai objektas priklauso savo klasteriui, palyginus su kitais klasteriais. Aukštesnis įvertis rodo geresnį klasterizacijos rezultatą. Matome, kad didžiausias

vidutinis silueto įvertis pasiekiamas, kai K = 2, o didėjant K reikšmei, įvertis mažėja. Tai rodo, kad optimalus klasterių skaičius yra 2.

Antrasis grafikas vaizduoja vidutinį Calinski-Harabasz indeksą (Mean Calinski-Harabasz Score) kiekvienai K reikšmei. Šis indeksas vertina klasterių tankumą ir atskyrimą. Didesnis indeksas rodo geresnį klasterių atskyrimą. Grafikas patvirtina, kad didžiausias vidutinis Calinski-Harabasz indeksas pasiekiamas, kai K = 2, o didėjant K reikšmei, indeksas mažėja.

Trečias grafikas atspindinti kryžminės validacijos inerciją (Cross-validated Inertia), suteikia reikšmingų įžvalgų apie klasterių skaičiaus įtaką klasterizacijos rezultatams. Inercija rodo bendrą atstumų tarp klasterių centrų ir juose esančių duomenų taškų sumą. Ji matuoja, kaip gerai duomenų taškai priglunda prie klasterio centro – kuo mažesnė inercijos reikšmė, tuo labiau susitelkę yra duomenys klasteriuose.

Kryžminės validacijos naudojimas yra svarbus, nes jis leidžia įvertinti modelio efektyvumą ir stabilumą, naudojant skirtingus duomenų padalijimus. Tai padeda išvengti persimokymo (angl. *overfitting*) ir leidžia geriau suprasti, kaip modelis veiks su naujais, nematytais duomenimis. Kryžminės validacijos rezultatai, pateikti grafikuose, rodo, kad K-means metodas yra stabilus ir duoda panašius rezultatus, nepriklausomai nuo duomenų padalijimo.

Klasterių analizė atskleidė reikšmingus skirtumus tarp piniginių elgsenos modelių, leidžiančius išskirti grupes, kurios galimai atspindi robotų ir žmonių valdomas pinigines. Pavyzdžiui, klasteriai su mažesniu transakcijų dažniu ir didesne aktyvumo pastovumo reikšme greičiausiai atspindi žmonių elgseną, tuo tarpu klasteriai, kuriuose transakcijų dažnis aukštas, aktyvumo valandų skaičius ilgas, o transakcijų sumos nuosekliai mažos, rodo robotų veikimą. Ši diferenciacija suteikia vertingų įžvalgų apie automatizuotą veiklą ir jos atskyrimą nuo natūralių vartotojų elgsenos.

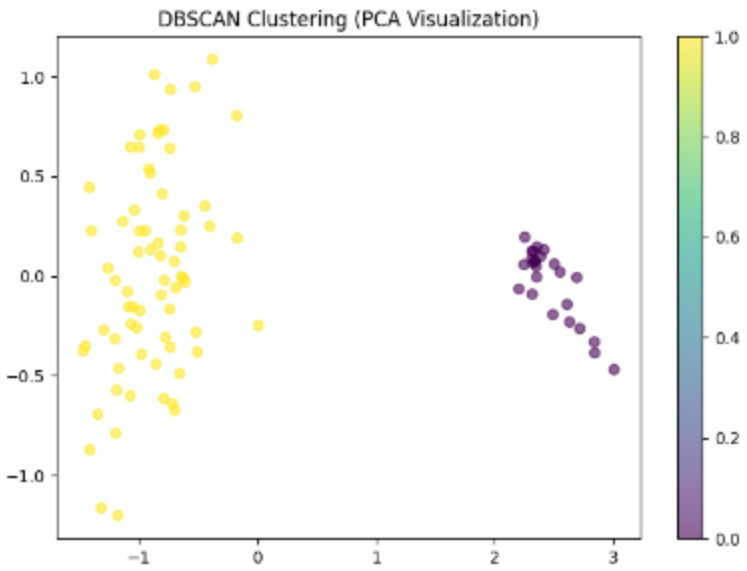
Nepaisant to, pastebėta, kad kai kurie klasteriai rodo dalinį persidengimą, ypač tam tikrų metrikų, kaip transakcijų dažnio ir aktyvumo valandų, atžvilgiu. Tai kelia iššūkių tiksliai atskiriant robotų ir žmonių elgseną, ypač jei egzistuoja mišrios elgsenos piniginės. Tokiais atvejais būtina naudoti papildomas analitines priemones ar kitus algoritmus, siekiant užtikrinti tikslesnę diferenciaciją.

Apibendrinant, klasterizacijos procesas pateikė aiškų struktūrą analizuojant piniginių veiklos modelius, o optimalus klasterių skaičius (K=3), nustatytas pagal Elbow, Silhouette ir Calinski-Harabasz metodus, užtikrino duomenų grupavimo efektyvumą. Šie rezultatai ne tik

leidžia įvertinti pagrindinius elgsenos skirtumus, bet ir parodo būtinybę giliau analizuoti mišrius atvejus, siekiant tiksliau identifikuoti ir atskirti robotų bei žmonių veikimą.

3.4.2. DBSCAN klasterizacijos metodo taikymas

Šiame poskyryje DBSCAN metodas bus taikomas analizuojant Solana ekosistemos piniginių elgseną, siekiant atskleisti sudėtingesnes struktūras ir identifikuoti anomalijas. Algoritmas yra itin efektyvus, kai duomenų rinkinys apima netolygius pasiskirstymus ar tankius grupavimo modelius.



Šaltinis: sudaryta autoriaus.

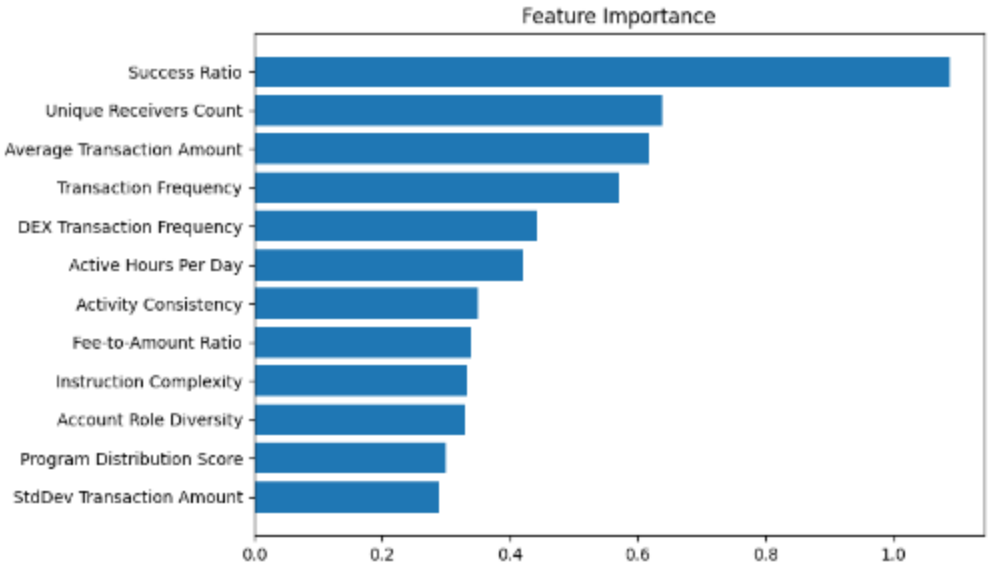
17 pav. DBSCAN klsaterizavimo PCA vizuolizacija

DBSCAN klasterizacijos vizualizacija (žr. 14 paveikslą), kurioje duomenys atvaizduoti naudojant pagrindinių komponentų analizę (PCA). Išskiriami du pagrindiniai klasteriai, pažymėti skirtingomis spalvomis: geltona ir violetine. Kiekvienas taškas atitinka vieną piniginę, o jų pozicijos erdvėje rodo pagrindinius elgesio modelių skirtumus tarp klasterių.

Vizualizacijoje pastebimi aiškiai atskirti klasteriai leidžia daryti keletą svarbių prielaidų. Violetiniai taškai rodo klasterį, kuris yra tankiai sutelktas mažoje erdvėje, o tai gali reikšti homogenišką elgesio modelį, būdingą automatizuotoms arba robotų valdomoms piniginėms. Tuo

tarpu geltoni taškai yra išsklaidyti didesnėje erdvėje, kas leidžia manyti apie didesnę žmonių elgsenos įvairovę.

Akivaizdus atskyrimas tarp šių klasterių rodo, kad DBSCAN algoritmas efektyviai identifikavo duomenų struktūrą ir išskyrė skirtingas elgesio grupes. Šis atskyrimas taip pat pabrėžia DBSCAN metodo privalumus analizuojant blockchain duomenis, nes jis geba ne tik aptikti klasterius, bet ir ignoruoti triukšmą, kuris šiame paveiksle nepateikiamas, nes algoritmas neaptiko triukšmo taškų.



Šaltinis: sudaryta autoriaus.

18pav. Savybių svarbos vizualizacija DBSCAN klasterizacijoje

Svybių svarbos analizė, atlikta naudojant DBSCAN klasterizacijos rezultatus. Grafikas rodo, kurios savybės turėjo didžiausią įtaką piniginių grupavimui į skirtingus klasterius.

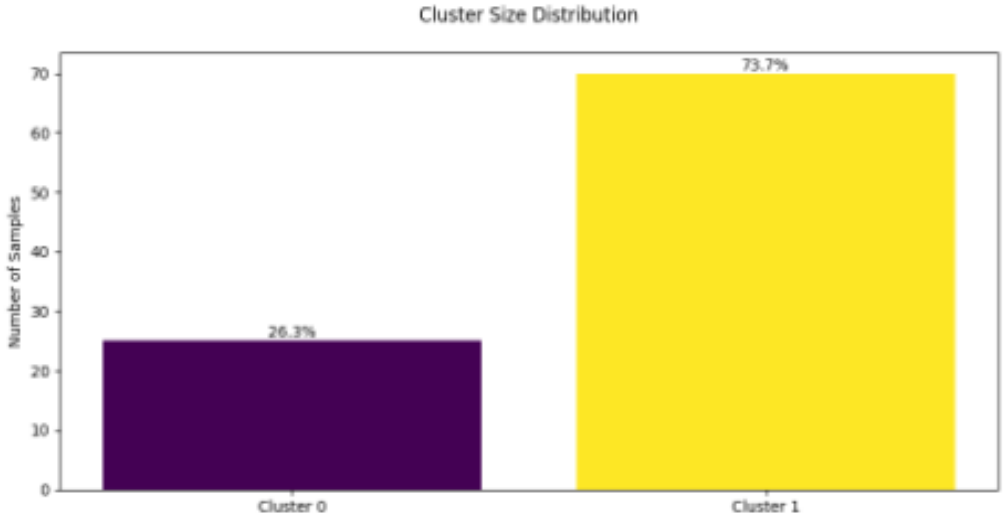
Pirmiausia, sėkmės rodiklis (Success Ratio) išsiskiria kaip svarbiausia savybė. Tai rodo, kad piniginių veiklos rezultatyvumas vaidina esminį vaidmenį atskiriant elgesio modelius. Šis rodiklis gali būti susijęs su tuo, kaip dažnai piniginė pasiekia savo tikslą (pavyzdžiui, sėkmingai užbaigia transakcijas).

Antroje vietoje pagal svarbą yra unikalių gavėjų skaičius (Unique Receivers Count), kuris parodo piniginės sąveikos su kitomis piniginėmis įvairovę. Tai gali būti svarbus indikatorius, atskleidžiantis, ar piniginė veikia kaip žmogaus valdomas subjektas, linkęs turėti daugiau įvairių ryšių, ar kaip robotas, orientuotas į siauresnį veiklos spektrą.

Vidutinė transakcijos suma (Average Transaction Amount) ir transakcijų dažnis (Transaction Frequency) taip pat pateikiami tarp reikšmingiausių savybių. Šie rodikliai padeda identifikuoti tiek aktyvumo modelius, tiek piniginės valdymo strategijas – pavyzdžiui, ar tai žmogaus veikla, apimanti įvairias sumas ir retesnį aktyvumą, ar robotų veikla, pasižyminti dažnesnėmis ir vienodesnėmis transakcijomis.

Mažesnės svarbos savybės, tokios kaip programų paskirstymo balas (Program Distribution Score) ir transakcijų sumų standartinis nuokrypis (StdDev Transaction Amount), taip pat prisideda prie klasterių formavimo, tačiau jų įtaka yra mažesnė. Tai gali reikšti, kad šie rodikliai yra labiau papildomi nei esminiai grupavimo veiksniai.

Šios vizualizacijos pagrindu galime daryti prielaidą, kad klasterizacija remiasi ne tik viena savybe, bet įvairių savybių deriniu, kuris leidžia atskirti skirtingus elgesio modelius. Tolimesnė analizė galėtų apimti detalesnį pagrindinių savybių įtakos klasterių viduje ir tarp jų tyrimą.



Šaltinis: sudaryta autoriaus.

19pav. Klasterių dydžio pasiskirstymas pagal DBSCAN metodą

Grafikas atskleidžia dviejų klasterių dydžių pasiskirstymą mūsų duomenų rinkinyje, kur Klasteris 0 sudaro mažesnę duomenų dalį (26,3%), o Klasteris 1 užima didžiąją dalį (73,7%). Šie rezultatai rodo ryškų skirtumą tarp piniginių elgesio modelių, kas yra esminė įžvalga botų aptikimo kontekste Solana blokų grandinėje.

Klasteris 0: Šis mažesnis klasteris greičiausiai atspindi botų elgseną. Botai dažnai pasižymi aukštu aktyvumu per trumpą laikotarpį, vykdo labai pasikartojančias ir sistemingas operacijas bei

išlaiko pastovų veiklos modelį. Klasterio 0 piniginės turi aukštą aktyvumo lygį, mažas transakcijų sumas ir aukštą sėkmės rodiklį, kas yra tipiniai botų veiklos požymiai. Toks elgesys gali rodyti, kad šios piniginės yra naudojamos tam tikrai specifinei užduočiai, pavyzdžiui, likvidumo tiekimui ar arbitražiniam prekybos algoritmui. Taip pat pastebimas mažas sąveikos su kitais tinklo dalyviais lygis, nes jos turi nedaug unikalių gavėjų ir žemą sąskaitų vaidmenų įvairovę.

Klasteris 1: Didesnis klasteris atspindi žmogaus elgsenos modelius. Šios piniginės vykdo įvairesnes operacijas su platesnėmis transakcijų sumomis, tačiau jų veikla nėra tokia nuosekli kaip botų. Klasterio 1 piniginės turi daugiau unikalių gavėjų, didesnę sąskaitų vaidmenų įvairovę ir yra mažiau aktyvios per parą, kas būdinga žmonėms. Ši grupė rodo natūralesnį ir mažiau struktūruotą operacijų modelį, kuris gali būti susijęs su asmeniniu ar investiciniu naudojimu.

Šių klasterių analizė leidžia daryti išvadą, kad DBSCAN algoritmas sėkmingai atskyrė dvi pagrindines piniginių grupes – botus ir žmones. Mažesnis klasteris (Klasteris 0) rodo sistemingą ir automatizuotą elgesį, kuris greičiausiai yra susijęs su botų veikla, o didesnis klasteris (Klasteris 1) atspindi žmogaus naudotojus su įvairesniu elgesiu.

16 lentelė

DBSCAN klasterizacijos metrikų rezultatai

Klasterizacijos metrika	Reikšmė
Klasterių skaičius	2
Triukšmo taškų skaičius	0
Geriausias Silueto balas	0.6308
Davies-Bouldin indeksas	0.4875
Calinski-Harabasz indeksas	209.5178
Geriausi parametrai	eps=1.4, min_samples=6

Šaltinis: sudaryta autoriaus.

DBSCAN klasterizacijos optimizacijos rezultatai (žr. 15 lentelę) rodo itin aukštą klasterizacijos kokybę pagal kelis pagrindinius metrikos rodiklius**. Silhouette Score, kuris** siekia 0.6308, parodo, kad identifikuoti klasteriai yra gerai apibrėžti ir turi aiškią struktūrą. Tai rodo, jog klasterizacija su pasirinktais parametrais (eps=1.4, min_samples=6) buvo itin sėkminga. Metrika vertina, kaip gerai objektai priskiriami klasteriams, lyginant su kitais klasteriais.

Davies-Bouldin Index, kurio reikšmė 0.4875, pabrėžia klasterių atskirtumo kokybę. Mažesnė indekso reikšmė rodo mažesnį klasterių persidengimą, kas yra svarbus rodiklis, ypač

analizuojant piniginių elgseną blokų grandinės kontekste, kur vienareikšmiška segmentacija yra būtina botų identifikavimui.

Calinski-Harabasz Index, siekiantis 209.5178, taip pat patvirtina aukštą klasterių kokybę. Aukštesnė šio rodiklio reikšmė rodo stipresnį vidinį klasterio tankį ir aiškesnį klasterių atskyrimą.

Klasterių skaičius ir triukšmo taškai*:* buvo identifikuoti du klasteriai, ir nėra triukšmo taškų. Tai reiškia, kad visi duomenų taškai buvo tinkamai priskirti klasteriams, kas liudija gerai parinktus hiperparametrus.

Parametrų reikšmės (eps ir min_samples): Parametrų optimizacija parodė, kad reikšmės eps=1.4 ir min_samples=6 yra tinkamiausios šiam duomenų rinkiniui. Šis pasirinkimas lėmė ne tik aukštą klasterizacijos kokybę, bet ir stabilumą.

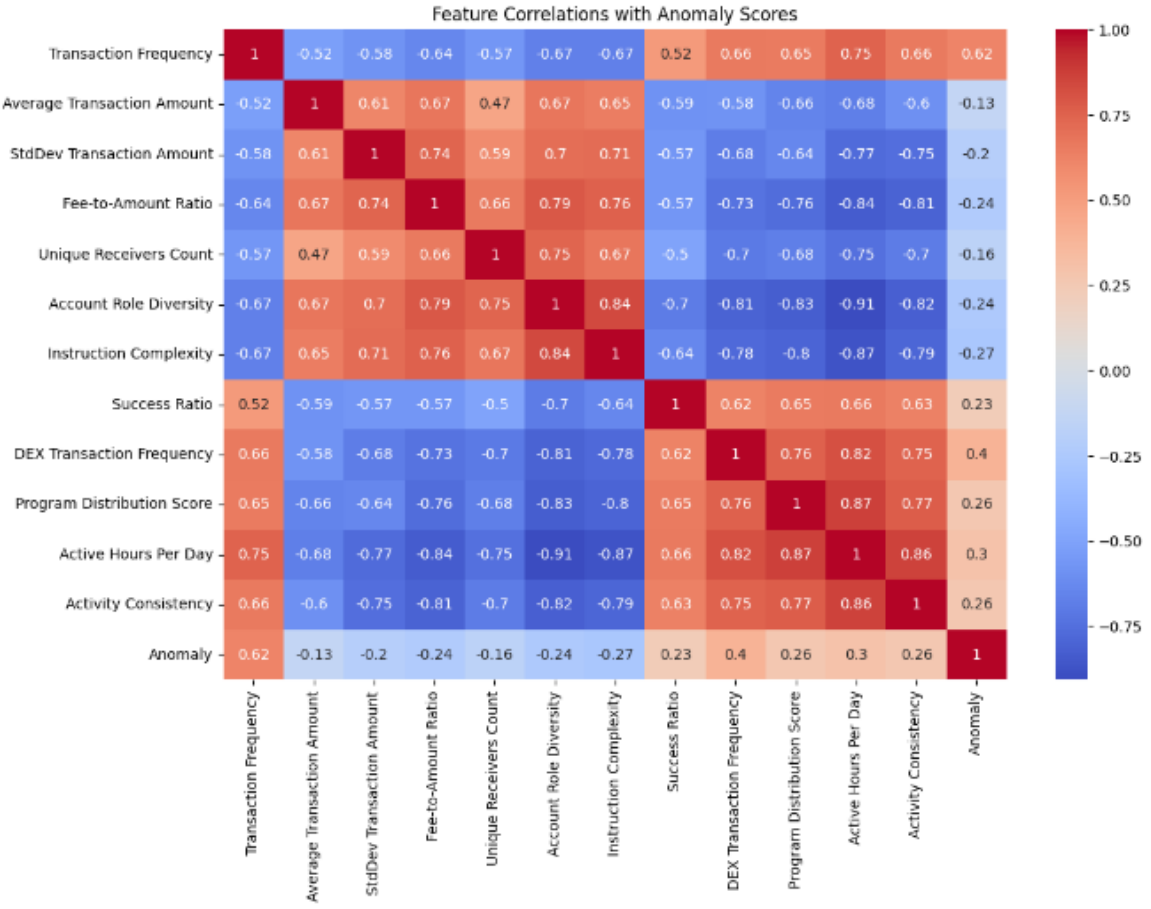
Metrikų tarpusavio sąryšis: Silhouette Score, tiek Calinski-Harabasz Index demonstruoja klasterių vidinį vientisumą ir atskirtį. Šie rodikliai kartu su Davies-Bouldin Index suteikia holistinį vaizdą apie klasterizacijos efektyvumą ir jos patikimumą.

Konteksto pritaikomumas: Kadangi duomenų rinkinys analizuojamas „Solana“ piniginių elgsenos kontekste, šie rezultatai gali būti pritaikomi siekiant atskirti žmonių elgseną nuo robotų veiklos. Gerai apibrėžti klasteriai rodo, kad DBSCAN yra tinkamas algoritmas.

Apibendrinant DBSCAN klasterizacijos rezultatus, galime teigti, kad algoritmas efektyviai išskyrė du skirtingus klasterius, be jokių triukšmo taškų. Tai parodo aukšta pasirinktos klasterizacijos metodologijos kokybė ir tinkamumas analizuojant blokų grandinės piniginių elgseną. Silhouette Score reikšmė atskleidžia, jog klasteriai yra gerai atskirti, o tai yra esminis požymis siekiant nustatyti robotų veiklos pėdsakus. Calinski-Harabasz Index reikšmė rodo, kad klasteriai turi stiprų vidinį tankį ir aiškias ribas, o mažas Davies-Bouldin Index patvirtina, kad klasterių persidengimas yra minimalus. Remiantis rezultatais, galima daryti prielaidą, kad vienas klasteris (didesnės apimties) gali atspindėti natūralų žmonių elgesį, tuo tarpu kitas (mažesnės apimties) yra susijęs su labiau struktūrizuota ir automatizuota veikla, būdinga botų operacijoms. Parametrų eps ir min_samples optimizavimas padėjo užtikrinti, kad visi duomenų taškai būtų tinkamai priskirti klasteriams, kas ypač svarbu analizuojant sudėtingus duomenų rinkinius.

3.4.3.Anomalijų aptikimas taikant Isolation Forest metodą

Izoliacijos miško (angl. Isolation Forest) algoritmas yra specialiai sukurtas efektyviam anomalijų aptikimui duomenų rinkiniuose. Jo veikimo principas remiasi anomalijų atskyrimu nuo įprastų duomenų pagal jų savybių retumą ir unikalumą. Šiame poskyryje Izoliacijos miško algoritmas bus taikomas analizuojant Solana ekosistemos piniginių elgseną, siekiant identifikuoti anomalijas ir išskirti įtartinas veiklos tendencijas.



Šaltinis: sudaryta autoriaus.

20 pav. Anomalijų ir požymių koreliacija

Požymių koreliacija su anomalijų rezultatais, leidžianti įvertinti, kurios metrikos labiausiai susijusios su netipine piniginių elgsena Solana ekosistemoje. Matome, kad aukščiausias koreliacijos koeficientas su anomalijų rezultatais turi"Transaction Frequency" (0.62), kas rodo, jog transakcijų dažnis yra stipriai susijęs su netipišku elgesiu. Tai gali reikšti, kad itin aukštas

transakcijų dažnis yra viena pagrindinių savybių, leidžiančių identifikuoti potencialius robotus ar kenkėjišką veiklą. Kita vertus, „Average Transaction Amount“ ir „StdDev Transaction Amount“ turi žemesnę, neigiamą koreliaciją su anomalijomis. Tai gali reikšti, jog šie požymiai yra mažiau svarbūs identifikuojant netipinius piniginių modelius. Požymiai, tokie kaip „Active Hours Per Day“ ir „Activity Consistency“, taip pat turi teigiamą, bet vidutinę koreliaciją su anomalijomis, rodančią, jog nuoseklus ir ilgas aktyvumas taip pat gali būti indikacija anomalijų. Šilumos žemėlapis (angl. heat map) parodo, jog skirtingi požymiai skirtingai prisideda prie anomalijų aptikimo. Požymių, turinčių reikšmingą koreliaciją su anomalijomis, įtraukimas į modelį sustiprina anomalijų aptikimo proceso efektyvumą.

17 lentelė

Anomalijų ir normalaus profilio požymių palyginimas

Požymis	Anomalijų profilis	Normalus profilis	Skirtumas
Transakcijų dažnis	1.664502	11.365324	-9.700822
Vidutinė transakcijos suma	1.46611	11.566758	-10.100648
Transakcijos sumų dispersija	1.448655	11.564306	-10.105973
Mokesčio ir sumos santykis	1.488862	11.640211	-10.172359
Unikalių gavėjų skaičius	1.590627	11.694917	-10.104289
Vaidmenų įvairovė	1.559287	11.758548	-10.19926
Instrukcijų sudėtingumas	1.47819	11.654833	-10.176643
Sėkmės santykis	1.888826	11.740223	-9.851397
DEX transakcijų dažnis	1.684776	11.473295	-9.788519
Programų pasiskirstymo balas	1.834333	11.639658	-9.805325
Aktyvios valandos per dieną	1.889198	11.588063	-9.784845
Aktyvumo nuoseklumas	1.855903	11.657876	-9.801973

Šaltinis: sudaryta autoriaus.

Anomalijų profilis sistemingai demonstruoja mažesnes reikšmes beveik visuose požymiuose, lyginant su normaliu profiliu. Šie skirtumai gali būti interpretuojami kaip galimi požymiai, rodantys netipinį arba automatizuotą elgesį, kuris nebūdinga įprastiems piniginių naudotojams Solana ekosistemoje.

Pavyzdžiui, transakcijų dažnio ir vidutinės transakcijos sumos reikšmingi skirtumai leidžia daryti prielaidą, kad anomalijų profilio piniginės vykdo daug mažesnį transakcijų skaičių ir

operacijas su mažesnėmis sumomis. Ši tendencija dažnai būdinga botams, kurie gali būti suprogramuoti vykdyti specifines užduotis su ribotais ištekliais.

Kiti rodikliai, tokie kaip mokesčio ir sumos santykis, sėkmės santykis ir DEX transakcijų dažnis, taip pat rodo mažesnes reikšmes anomalijų profilyje. Tai rodo mažesnį šių piniginių įsitraukimą į sudėtingesnes finansines operacijas, kurios yra dažnesnės normaliame profilyje. Vaidmenų įvairovės ir aktyvumo nuoseklumo skirtumai dar labiau sustiprina šią hipotezę, nes anomalijos dažnai rodo mažiau įvairiapusišką ir nuoseklų elgesį. Bendra tendencija rodo, kad anomalijų profilio piniginės yra paprastesnio, mažiau įvairaus elgesio, kas leidžia jas identifikuoti kaip potencialiai automatizuotas arba netipines.

18 lentelė

Statistinė požymių analizė ir reikšmingumo lygiai

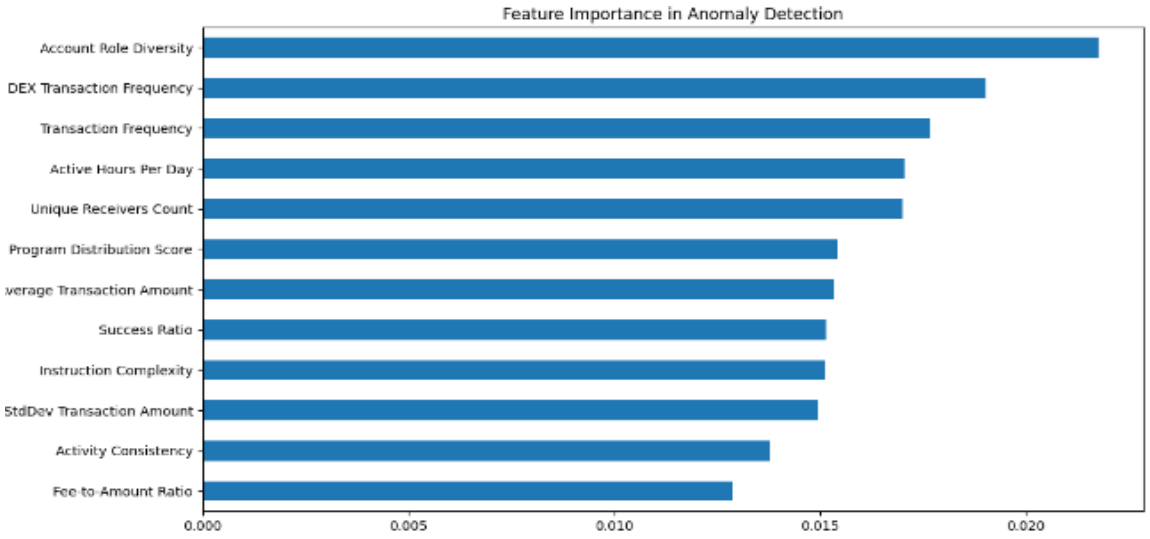
Požymis	Statistikos reikšmė	p reikšmė
Transakcijų dažnis	89.5	0.000833
Vidutinė transakcijos suma	620	0.051467
Transakcijos sumos standartinis nuokrypis	644	0.026197
Mokesčio ir sumos santykis	577	0.1461
Unikalių gavėjų skaičius	595	0.09583
Paskyrų vaidmenų įvairovė	614	0.06034
Instrukcijų sudėtingumas	658	0.01712
Sėkmės santykis	264	0.033059
DEX transakcijų dažnis	159.5	0.000833
Programų pasiskirstymo rodiklis	241	0.016593
Aktyvios valandos per dieną	229	0.011294
Aktyvumo nuoseklumas	276	0.046197

Šaltinis: sudaryta autoriaus.

Statistiniai požymių reikšmingumo rodikliai, leidžia įvertinti, kurie požymiai yra reikšmingi aptinkant anomalijas ir kokią įtaką jie gali turėti tolimesnei analizei. Reikšmingiausi požymiai, turintys mažiausias p reikšmes, yra transakcijų dažnis, DEX transakcijų dažnis, aktyvios valandos per dieną ir instrukcijų sudėtingumas. Tai rodo, kad šie požymiai turi stipriausią poveikį atskiriant įprastus ir anomalinius piniginių elgsenos modelius. Transakcijų dažnis ir DEX transakcijų dažnis atskleidžia, kad automatizuoti veikėjai (robotai) paprastai vykdo didesnio intensyvumo operacijas, kurios pastebimai skiriasi nuo žmonių elgsenos. Aktyvios valandos per

dieną taip pat yra svarbus rodiklis, nes anomalijos dažnai susijusios su piniginėmis, kurios veikia ilgesnį laiką nei įprasti vartotojai.

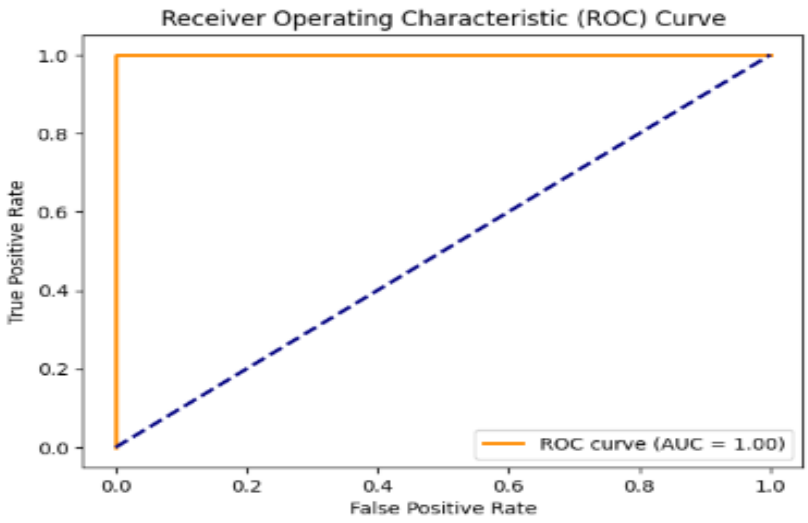
Mažiau reikšmingi požymiai, tokie kaip mokesčio ir sumos santykis ar paskyrų vaidmenų įvairovė, vis dar gali suteikti papildomos informacijos apie elgsenos modelius, tačiau jie nėra tokie stiprūs anomalijų indikatoriai kaip minėti aukščiau. Lentelės duomenys taip pat patvirtina, kad p reikšmės, mažesnės nei 0.05, rodo statistinį reikšmingumą, o tai sustiprina šių požymių įtaką anomalijų detekcijai.



Šaltinis: sudaryta autoriaus.

21. pav. Anomalijų aptikimo požymių svarba

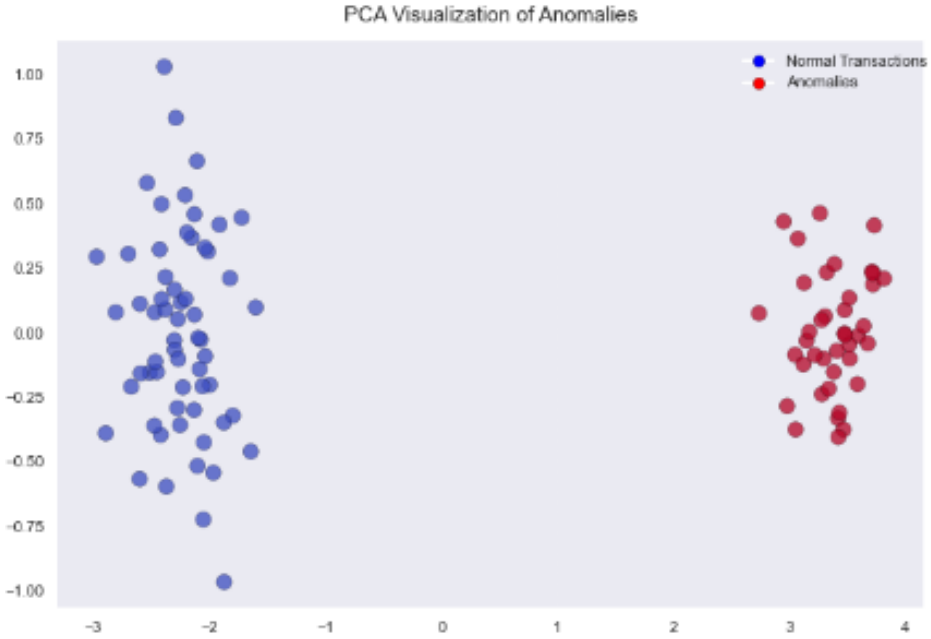
Požymių svarba anomalijų aptikime, leidžia išryškinti metrikas, kurios yra kritiškai svarbios nustatant nenormalius elgsenos modelius Solana ekosistemoje. Požymiai, tokie kaip paskyrų vaidmenų įvairovė, DEX transakcijų dažnis ir transakcijų dažnis, išsiskiria kaip labiausiai įtakojantys anomalijų identifikavimą. Šie rezultatai rodo, kad įvairių paskyrų veiklos pobūdis, intensyvumas ir dažnumas yra pagrindiniai indikatoriai atskiriant įprastus ir neįprastus piniginių elgsenos modelius. Aktyvios valandos per dieną ir unikalių gavėjų skaičius taip pat pasirodo kaip reikšmingi rodikliai, kurie gali nurodyti automatizuotą veiklą ar neįprastus naudojimo modelius. Šie požymiai gali būti naudingi tobulinant anomalijų aptikimo modelius, nes jie atspindi aktyvumo ir transakcijų įvairovę. Mažiau svarbūs požymiai, tokie kaip mokesčio ir sumos santykis ar veiklos pastovumas, gali būti mažiau reikšmingi atskiriant anomalijas, tačiau jų įtraukimas gali suteikti papildomų įžvalgų apie bendrą duomenų struktūrą.



Šaltinis: sudaryta autoriaus.

22pav. Izoliacijos miško metodo veikimo įvertinimas pagal ROC kreivę

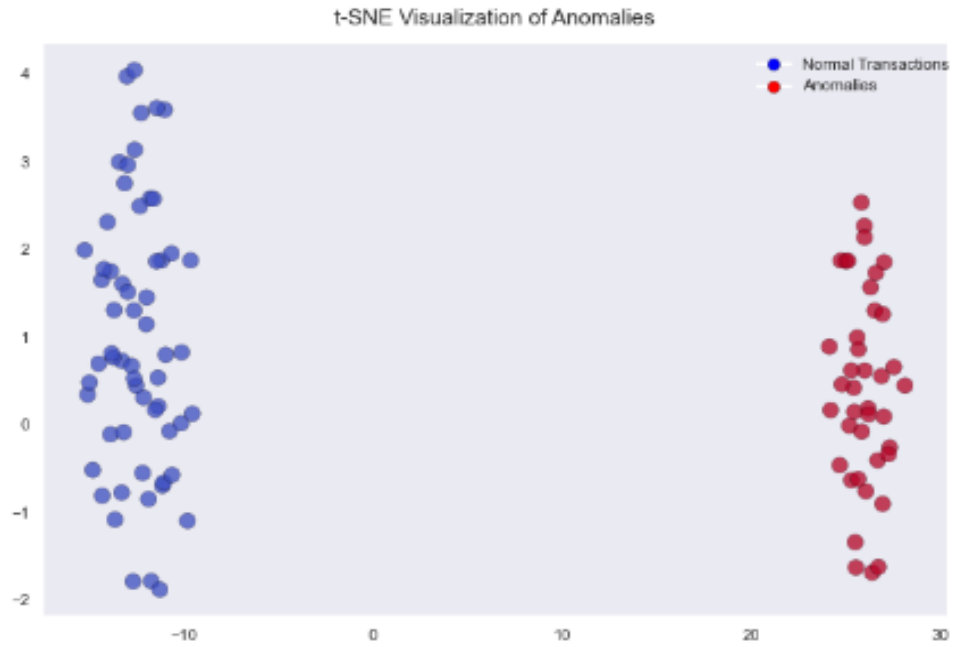
ROC kreivė rodo puikią modelio veikimo kokybę, nes ploto po kreive (AUC) reikšmė yra lygi 1, kas rodo maksimalų modelio jautrumą ir specifiškumą. Šiame grafike akivaizdžiai matomas modelio gebėjimas teisingai klasifikuoti tiek anomalinius, tiek normalius duomenų taškus, be klaidingų teigiamų ar neigiamų rezultatų. Šis rezultatas parodo izoliacijos miško metodo tvirtumą ir tinkamumą anomalijų aptikimui Solana blokų grandinės duomenų analizėje.



Šaltinis: sudaryta autoriaus.

23 pav. Anomalijų vizualizacija naudojant PCA

Anomalijų ir normalių duomenų pasiskirstymą sumažintų dimensijų erdvėje, naudodajant pagrindinių komponentų analizę (PCA). Raudonais taškais pažymėtos anomalijos aiškiai išsiskiria nuo mėlynų taškų, kurie atspindi normalius duomenis. Pastebima, kad anomalijos yra grupuojamos atskirai nuo kitų duomenų taškų, kas rodo išskirtinį izoliacijos miško algoritmo gebėjimą atpažinti netipinę piniginių veiklą Solana blokų grandinėje. Anomalijų koncentracija vienoje erdvės dalyje rodo tam tikrus elgsenos modelius, kurie ženkliai skiriasi nuo normalaus elgesio. Tai leidžia daryti prielaidą, kad šios piniginės gali būti susijusios su automatizuotomis operacijomis arba neįprasta veikla. Vizualizacijos aiškumas patvirtina PCA metodo tinkamumą palengvinti sudėtingų daugiamatių duomenų interpretaciją.



Šaltinis: sudaryta autoriaus.

24 pav. Anomalijų vizualizacija naudojant t-SNE

t-SNE, pateikia vizualizaciją, leidžiančią suprasti duomenų pasiskirstymą mažesnės dimensijos erdvėje. Raudoni taškai žymi anomalijas, o mėlyni – normalius duomenis. Aiškiai matomas anomalijų grupavimas atskirtoje erdvės dalyje, kas rodo jų elgesio reikšmingą skirtingumą nuo normalių duomenų. t-SNE metodas išryškina sudėtingesnius duomenų modelius, kurie gali būti ne tokie akivaizdūs kitomis vizualizacijos priemonėmis. Anomalijos yra susitelkusios aukštos anomalijos reikšmės zonoje, kas leidžia daryti prielaidą, kad šios piniginės yra susijusios su automatizuotais procesais arba neįprasta veikla Solana blokų grandinėje. Šis

grafikas parodo t-SNE metodo pranašumą, kai reikia identifikuoti sudėtingus duomenų ryšius. Tikslesnė anomalijų lokalizacija leidžia geriau suprasti, kaip šios piniginės išsiskiria pagal savo elgseną ir kokių priemonių reikėtų imtis jų tolesniam tyrimui.

Apibendrinant, atlikta izoliacinio miško analizė efektyviai identifikavo anomalijas Solanos tinklo piniginėse, pabrėždama algoritmo tinkamumą aptikti automatizuotų sandorių modelius. Statistiškai reikšmingi požymiai, tokie kaip „Account Role Diversity“ ir „DEX Transaction Frequency“, ženkliai prisidėjo prie aptikimo tikslumo, suteikdami vertingų įžvalgų apie piniginių elgesį. Rezultatai atsklė, kaip svarbu derinti skirtingus analizės metodus ir prioritetizuoti reikšmingus požymius, siekiant kurti efektyvias anomalijų aptikimo sistemas. Praktinis pritaikymas apima ne tik geresnį automatizuotų veiksmų valdymą Solanos tinkle, bet ir galimybę įgyvendinti sprendimus, kurie padėtų kovoti su kenkėjiškais robotais. .

3.5 Hierarchinis klasterizacijos ir anomalijų aptikimo metodas

Hierarchinis klasterizacijos ir anomalijų aptikimo metodas yra esminė analizės dalis, siekiant giliau suprasti duomenų elgsenos modelius ir nustatyti subtilias anomalijas. Šis metodas sujungia K-means klasterizacijos, DBSCAN klasterizacijos ir Isolation Forest algoritmų rezultatų pagrindą, leidžiantį analizuoti duomenis skirtingais pjūviais. Toks integruotas požiūris leidžia ne tik identifikuoti globalius elgsenos modelius, bet ir atskleisti lokalius nukrypimus, kurie gali būti susiję su specifinėmis elgsenos charakteristikomis ar potencialiai įtartina veikla.

Naudojant K-means ir DBSCAN klasterizacijos metodus, duomenys yra grupuojami pagal skirtingus principus: K-means metodas sutelkia homogeniškus duomenų taškus į sferines grupes, o DBSCAN leidžia aptikti netaisyklingos formos klasterius ir atskirus taškus, kurie priskiriami triukšmui. Tačiau šie klasterizacijos metodai nėra tiesiogiai skirti aptikti retas ar neįprastas vertes, kurios gali egzistuoti netgi klasterių viduje. Todėl, siekiant išspręsti šį apribojimą, kiekvieno klasterio viduje (K-means, DBSCAN) buvo taikomas Isolation Forest algoritmas, kuris, remdamasis izoliuotų medžių struktūra, leidžia tiksliai identifikuoti anomalijas.

Šio metodo taikymas yra ypač svarbus, nes jis leidžia išplėsti analizę, naudojant trijų skirtingų metodų privalumus. K-means ir DBSCAN klasterizacija užtikrina stiprią pagrindą elgsenos modelių identifikavimui, o Isolation Forest algoritmas, taikomas klasterių viduje, leidžia

aptikti subtilias anomalijas, kurios galėtų būti praleistos naudojant tik klasterizacijos metodus. Isolation Forest išskirtinumas yra jo gebėjimas efektyviai aptikti nukrypimus, remiantis taškų izoliavimo gyliais, nesiremiant išankstinėmis prielaidomis apie duomenų pasiskirstymą. Hierarchinis požiūris leidžia ne tik aptikti anomalijas, bet ir gauti vertingų įžvalgų apie tai, kaip specifiniai požymiai lemia anomalijų atsiradimą skirtinguose klasteriuose. Ši metodologija sustiprina bendrą analizės tikslumą, nes aptinkami lokaliai svarbūs nukrypimai, kurie galėtų būti praleisti naudojant tik globalius metodus.

Toks sisteminis požiūris užtikrina, kad kiekvienas klasteris būtų analizuojamas detaliai ir atskirai, išryškinant svarbias elgsenos savybes ir anomalijų šaltinius. Galutinis šios metodologijos tikslas yra ne tik identifikuoti anomalijas, bet ir pateikti detalius paaiškinimus bei vizualizacijas, kurios padėtų geriau suprasti sudėtingų duomenų struktūrą ir jų praktinį kontekstą.

3.5.1. K means klasterių hierarchinė analizė

Izoliacijos medis, taikomas K means būdu gautam Klasteriui 0, pateikia išsamią analizę, kaip konkrečios ypatybės lemia anomalijų aptikimą šioje duomenų grupėje. Pagrindinė analizės prielaida grindžiama tuo, kad Isolation Forest algoritmas siekia atskirti duomenų taškus remiantis jų izoliuojamumu, o vizualizacija (1 priedas) leidžia suprasti pagrindinius modelio sprendimus.

Pirmasis skirstymas, atliekamas šaknies mazge, yra pagrįstas ypatybe Account Role Diversity (F5), kurios ribinė reikšmė yra ≤ 0.54. Šis rodiklis reikšmingas formuojant elgsenos modelį, nes apibrėžia, kurios piniginės veikia su mažesne vaidmenų įvairove ir yra dažniau priskiriamos normaliems taškams. Šio skirstymo svarba atsispindi tame, kad jis iškart atskiria dvi pagrindines piniginių grupes pagal jų veiklos elgseną.

Kairėje šakoje, kur piniginės atitinka $F5 \leq 0.54$, izoliuoti taškai toliau skirstomi pagal Fee-to-Amount Ratio (F3) ir Average Transaction Amount (F1). Šios ypatybės aiškiai išskiria anomalijas, kurios gali būti susijusios su neįprastai mažomis operacijomis arba manipuliatyviais veiklos modeliais. Terminaliniuose mazguose, kur stebimos didžiausios anomalijų tikimybės (tamsiai oranžinė spalva), randamos piniginės, kurių F3 reikšmės yra ypatingai žemos, rodant neadekvačius transakcijų santykius.

Dešinėje šakoje, kur piniginės neatitinka $F5 \leq 0.54$ kriterijaus, skirstymas vykdomas remiantis Activity Consistency (F11) ir Program Distribution Score (F9) ypatybėmis. Šios ypatybės rodo, kad anomalijos šioje grupėje dažnai kyla dėl nepastovios veiklos arba labai

intensyvios decentralizuotų biržų operacijų istorijos. Pavyzdžiui, mazguose, kur F11 reikšmės yra žemos, greitai izoliuojami taškai, kuriuos galima laikyti netipinėmis piniginėmis.

Vizualizacijoje matomas spalvų intensyvumas, kuris parodo, kaip algoritmo sprendimai siejami su anomalijos balais. Kuo intensyvesnė oranžinė spalva, tuo didesnė anomalijos tikimybė. Tamsiai oranžiniuose mazguose randamos piniginės, kurios turi reikšmingus nukrypimus nuo klasterio normų ir gali būti siejamos su manipuliatyvia ar automatizuota veikla. Šiame klasteryje stebimas glaudus ryšys tarp anomalijų ir ekonominių rodiklių, tokių kaip Fee-to-Amount Ratio, kurie yra vieni pagrindinių anomalijų aptikimo požymių.

Analizuojant Klasterio 0 medį, akivaizdu, kad šis klasteris pasižymi sisteminėmis anomalijomis, kurios dažniausiai susijusios su transakcijų parametrais. Tokia struktūra leidžia identifikuoti specifinius anomalijų elgsenos modelius, kas yra svarbus žingsnis giliau suprantant Solana blokų grandinės duomenų dinamiką. Analizė rodo, jog Account Role Diversity, kaip pagrindinis skirstymo veiksnys, ne tik atskiria normalius taškus, bet ir padeda nustatyti anomalijas kartu su kitais parametrais, kaip Fee-to-Amount Ratio ir Program Distribution Score.

Klasteriui 1, pateikia gilesnę įžvalgą į šio klasterio duomenų elgseną ir anomalijų aptikimo mechanizmą (2 priedas). Šio klasterio medžio struktūra demonstruoja, kad pagrindinis skirstymo kriterijus šaknies mazge yra Account Role Diversity (F5), kurios ribinė reikšmė yra ≤ -1.43. Šis rodiklis reikšmingai lemia pradinį duomenų grupavimą, atskiriant pinigines su maža vaidmenų įvairove nuo tų, kurių elgsenos įvairovė yra didesnė.

Kairioji šaka, atitinkanti kriterijų F5 ≤ -1.43, yra skirstoma remiantis Program Distribution Score (F9) ir Unique Receivers Count (F4) rodikliais. F9 su ribine reikšme ≤ 1.48 rodo, kad piniginės, kurių veikla yra labiau koncentruota ties keliais programų tipais, yra greitai izoliuojamos. Tolesniame skirstyme, kai pagrindiniu kriterijumi tampa Unique Receivers Count (F4), pastebima, kad anomalijos dažniau siejamos su piniginėmis, kurios vykdo operacijas su ribotu gavėjų skaičiumi.

Dešinėje šakoje, kurioje F5 > -1.43, dominavo tokie požymiai kaip Activity Consistency (F11) ir Success Ratio (F7). F11, kurio ribinė reikšmė yra ≤ -1.39, parodo, kad piniginės su nepastovia veikla dažnai patenka į izoliuojamus mazgus. Šios piniginės gali būti susijusios su nestandartiniais veiklos modeliais arba automatizuotomis operacijomis. F7, kai rodiklis mažesnis už 0.82, identifikuoja pinigines su žemu operacijų sėkmės rodikliu, kas dažnai būna susiję su anomalijomis.

Terminaliniuose mazguose pastebimi ryškūs anomalijų taškai, kuriuos rodo tamsiai oranžinė spalva, pavyzdžiui, mazgas, kurio reikšmė yra 0.95. Šis mazgas apima 3.3 % visų duomenų taškų ir rodo pinigines, kurių veikla reikšmingai nukrypsta nuo klasterio normų. Šio mazgo analizė leidžia daryti prielaidą, kad anomalijos kyla dėl sudėtingų veiklos modelių, kurie susiję su žemu Success Ratio ir žema Activity Consistency. Vizualizacija pabrėžia, kad skirstymo procesas yra orientuotas į požymių kombinaciją, leidžiančią išskirti tiek įprastas, tiek netipines pinigines. Spalvų intensyvumas atspindi anomalijos tikimybę: tamsesnė oranžinė spalva rodo didesnį išskirtinumą, o šviesesni mazgai žymi normalius taškus. Klasterio 1 medis parodo sudėtingą anomalijų modelį, kuris susijęs su veiklos intensyvumu ir transakcijų sėkme. Analizė rodo, kad izoliuojamumo kriterijai, tokie kaip Program Distribution Score, Activity Consistency, ir Success Ratio, yra esminiai aptinkant netipines pinigines, o šių požymių reikšmės gali būti naudingos tolesniam automatizuotos veiklos identifikavimui. Šis medis suteikia galimybę detaliau suprasti piniginių veiklos modelius, kurie susiję su Solana blokų grandinės ekosistema.

Izoliacijos medis, taikomas Klasteriui 2, atskleidžia sudėtingą ir mišrią anomalijų struktūrą šiame klasteryje (3 priedas). Pagrindinis skirstymo kriterijus šaknies mazge yra Account Role Diversity (F5), kurios ribinė reikšmė yra ≤ 0.49. Šis rodiklis, kaip ir ankstesniuose klasteriuose, parodo elgsenos įvairovės svarbą anomalijų aptikimo procese. Piniginės, turinčios mažą vaidmenų įvairovę, yra greičiau izoliuojamos, o tai leidžia išskirti netipines veiklos tendencijas.

Kairioji šaka, kurioje $F5 \leq 0.49$, pasižymi papildomu skirstymu remiantis Fee-to-Amount Ratio (F3) ir Program Distribution Score (F9). Šie požymiai parodo, kad anomalijos dažniausiai kyla dėl netipinių ekonominių rodiklių, tokių kaip neproporcingai mažos transakcijos arba labai ribotas programų naudojimas. Terminaliniuose mazguose, kurių reikšmės yra aukščiausios (pvz., 0.95), izoliuotos piniginės demonstruoja elgseną, kuri gali būti susijusi su sudėtinga ar manipuliatyvia veikla.

Dešinioji šaka, kurioje $F5 > 0.49$, apima pinigines su didesne vaidmenų įvairove. Šios piniginės toliau skirstomos pagal Activity Consistency (F11) ir Success Ratio (F7), kurie atskleidžia anomalijas dėl nepastovaus veiklos intensyvumo arba žemo operacijų sėkmės rodiklio. Pavyzdžiui, mazguose, kur $F11 \leq -0.06$, pastebimos piniginės, kurių veikla yra labai neįprasta ir galimai susijusi su netipinėmis operacijomis.

Vizualizacijoje matomas spalvų intensyvumas pabrėžia anomalijų tikimybę: tamsiai oranžinė spalva atspindi didesnę tikimybę, kad piniginė pasižymi anomalijomis, o šviesesni mazgai nurodo normalius duomenų taškus. Šiame klasteryje tamsiai oranžiniai mazgai, kurių reikšmė yra 0.95, rodo labiausiai išsiskiriančias pinigines, kurios, remiantis rodikliais, galimai vykdo neįprastai sudėtingas ar manipuliatyvias operacijas.

Apibendrinant, Klasterio 2 medis parodo, kad šiame klasteryje anomalijos yra susijusios tiek su ekonominiais rodikliais, tiek su operacijų pastovumo ir sėkmės parametrais. Šio klasterio analizė atskleidžia ne tik tipinių ir netipinių piniginių elgseną, bet ir suteikia įžvalgų apie jų galimą vaidmenį Solana blokų grandinės ekosistemoje. Šie rezultatai pabrėžia, jog net mišriuose klasteriuose galima identifikuoti aiškius anomalijų modelius, remiantis tinkamais požymiais.

Analizuojant visų trijų klasterių izoliacijos medžius, išryškėja svarbūs jų skirtumai ir bendrumai. Kiekviename klasteryje pastebimos skirtingos anomalijų priežastys, kurios atspindi tiek ekonominius rodiklius, tiek veiklos pastovumo ir sudėtingumo parametrus. Visuose klasteriuose pagrindiniai požymiai, lemiantys anomalijų aptikimą, apima Account Role Diversity (F5) ir Activity Consistency (F11). Tai rodo, kad šie parametrai yra universaliai svarbūs anomalijų aptikimo procese ir dažnai susiję su netipinėmis piniginių elgsenomis. Be to, anomalijos visose grupėse išryškėja per tokius rodiklius kaip Fee-to-Amount Ratio (F3) ir Success Ratio (F7), kurie atspindi tiek transakcijų ekonominį pagrįstumą, tiek operacijų efektyvumą.

19 lentelė

K means klasterių savybių palyginimas

Savybės	Klasteris 0	Klasteris 1	Klasteris 2
Dominuojantis tipas	Žmonių piniginės	Botų piniginės	Mišrios žmonių piniginės
Pagrindinės anomalijos	Transakcijų standartinis nuokrypis, Gavėjų skaičius	Veiklos pastovumas, DEX operacijos	Instrukcijų sudėtingumas, Vaidmenų įvairovė
Esminiai rodikliai	Mokesčių ir sumos santykis, Operacijų sėkmės rodiklis	Veiklos pastovumas, Programų paskirstymo balas	Vaidmenų įvairovė, Instrukcijų sudėtingumas
Tipinė anomalija	Didelė transakcijų įvairovė	Intensyvi ir pastovi veikla	Sudėtingos ir specializuotos operacijos

Šaltinis: sudaryta autoriaus.

Anomalijų aptikimas kiekviename klasteryje suteikia galimybę detaliau suprasti specifinių piniginių elgseną, kuri ženkliai nukrypsta nuo klasterio normų. Toliau pateikiama lentelė, kuri

apibendrina kiekviename klasteryje identifikuotas labiausiai anomalijomis pasižyminčias pinigines. Ši lentelė yra sudaryta pagal gautus rezultatus ir apima pagrindines ypatybes, kurios lėmė anomalijų balus.

20 lentelė

Anomalijų piniginės pagal K mean klasterius

Klasteris	Piniginės ID	Anomalijos balas	Pagrindinės ypatybės
0	Žmogus_66	0.5716	Transakcijų standartinis nuokrypis (1.00), Gavėjų skaičius (1.00), Operacijų sėkmės rodiklis (0.8871)
0	Žmogus_45	0.5656	Mokesčių ir sumos santykis (0.9856), Vaidmenų įvairovė (0.9808)
1	Botas_28	0.582	Aktyvios valandos per dieną (0.993), Programų paskirstymo balas (0.9829), Veiklos pastovumas (0.9581)
1	Botas_8	0.5661	Decentralizuotų biržų operacijų dažnis (1.00), Veiklos pastovumas (0.9682)
2	Žmogus_17	0.5667	Vaidmenų įvairovė (0.6672), Instrukcijų sudėtingumas (0.5783)
2	Žmogus_42	0.5502	Gavėjų skaičius (0.8333), Transakcijų standartinis nuokrypis (0.8023)

Šaltinis: sudaryta autoriaus.

Rezultatų interpretacija atskleidžia esminius skirtumus tarp skirtingų klasterių anomalijų. Kiekviename klasteryje anomalijų piniginės pasižymi unikaliomis savybėmis, kurios leidžia giliau suprasti jų elgsenos modelius. Klasteryje 0 aptiktos anomalijos dažniausiai siejamos su žmonių piniginėmis, kurių elgsena išsiskiria sudėtingomis operacijomis. Šio klasterio anomalijos remiasi dideliais transakcijų standartinio nuokrypio rodikliais bei plačiu gavėjų skaičiumi. Tokios ypatybės rodo pinigines, kurios veikia įvairiose srityse arba vykdo daugialypes operacijas, tačiau dėl šios veiklos įvairovės jų elgsena nukrypsta nuo pagrindinių klasterio normų.

Klasteryje 1 dominuoja botų piniginės, kurios pasižymi sistemingomis anomalijomis, aiškiai susijusiomis su intensyvia veikla. Šiame klasteryje anomalijos kyla dėl didelio decentralizuotų biržų operacijų dažnio ir labai pastovaus aktyvumo. Botų piniginės išsiskiria tuo, kad jų veikla yra nuosekliai organizuota, dažnai vykdoma be žymių nukrypimų nuo automatizuotų modelių, tačiau pati šios veiklos intensyvumo struktūra yra netipinė, todėl jos identifikuojamos kaip anomalijos. Šie rezultatai leidžia daryti prielaidą, kad botai šioje grupėje gali būti naudojami prekybos efektyvumui didinti arba manipuliuoti rinkos sąlygomis.

Klasteryje 2 anomalijos yra mišrios ir dažniausiai būdingos žmonių piniginėms, kurios veikia sudėtingose operacijose su didele instrukcijų įvairove. Šio klasterio anomalijos kyla dėl įvairios veiklos paskirties, kurioje įsitraukia įvairių vaidmenų piniginės. Šių piniginių elgsena ne visuomet atitinka pagrindinį klasterio modelį, todėl jos identifikuojamos kaip netipinės. Šie rezultatai rodo, kad klasteris 2 apima tiek įprastas, tiek sudėtingesnes operacijas vykdančias pinigines.

Bendrai, ši analizė pateikia gilias įžvalgas apie anomalijų pobūdį Solana blokų grandinės ekosistemoje. Botų klasterio anomalijos išryškina sistemingumą ir efektyvumo prioritetus, o žmonių piniginės dažniausiai išsiskiria elgsenos įvairove ir sudėtingumu. Šie skirtumai leidžia ne tik klasifikuoti pinigines pagal jų anomalijų modelius, bet ir pateikia vertingų duomenų tolesniam tyrimų ir prevencinių priemonių kūrimui.

3.5.2.DBSCAN klasterių hierarchinė analizė

DBSCAN klasterių analizė, taikant izoliacijos miško metodą, siekiama išplėsti supratimą apie klasterių vidinę struktūrą ir galimus anomalijų modelius. Skirtingai nei K-means metodas, DBSCAN leidžia aptikti tankias duomenų grupes, ignoruojant atskirus taškus, todėl šis metodas yra tinkamas analizuoti sudėtingesnius elgsenos modelius. Izoliacijos miško metodas pritaikomas kiekvienam klasteriui atskirai, siekiant nustatyti ir išanalizuoti retus bei neįprastus duomenų taškus. Tokia integruota metodika leidžia ne tik aptikti anomalijas, bet ir geriau suprasti galimą netipinę elgseną Solanos blokų grandinės ekosistemoje.

Izoliacijos medis, taikomas Klasteriui 0, pateikia giluminę įžvalgą į šio klasterio duomenų elgseną ir anomalijų aptikimo mechanizmą (4 priedas). Klasterio medžio struktūra demonstruoja, kad pagrindinis skirstymo kriterijus šaknies mazge yra Account Role Diversity (F5), kurios ribinė reikšmė yra ≤ 0.103. Rodiklis reikšmingai lemia pradinių duomenų grupavimą, atskiriant pinigines su mažą vaidmenų įvairovę nuo tų, kurių elgsenos įvairovė yra didesnė.

Kairiojoje šakoje, atitinkančioje kriterijų F5 $\leq$ 0.103, yra skirstoma remiantis Program Distribution Score (F9) ir Unique Receivers Count (F4) rodikliais. F9 su ribine reikšme ≤ 0.971 rodo, kad piniginės, kurių veikla labiau koncentruota ties keliais programų tipais, yra greitai izoliuojamos. Tolimesniame skirstyme, kai pagrindiniu kriterijumi tampa Unique Receivers Count (F4), pastebima, kad anomalijos dažniau siejamos su piniginėmis, kurios vykdo operacijas su ribotu gavėjų skaičiumi.

Dešiniojoje šakoje, kurioje F5 > 0.103, dominuoja tokie požymiai kaip Activity Consistency (F11) ir Success Ratio (F7). F11, kurio ribinė reikšmė yra ≤ 0.985, parodo, kad piniginės su pastovia veikla dažnai patenka į izoliuojamus mazgus. Piniginės gali būti susijusios su nestandartiniais veiklos modeliais arba automatizuotomis operacijomis. F7, kai rodiklis mažesnis už 0.862, identifikuoja pinigines su žemu operacijų sėkmės rodikliu, kas dažnai būna susiję su anomalijomis. Terminaliniuose mazguose pastebimi ryškūs anomalijų taškai, kuriuos rodo tamsiai oranžinė spalva, pavyzdžiui, mazgas, kurio reikšmė yra 0.951. Šis mazgas apima 3.3 % visų duomenų taškų ir rodo pinigines, kurių veikla reikšmingai nukrypsta nuo klasterio normų. Vizualizacija pabrėžia, kad skirstymo procesas yra orientuotas į požymių kombinacijas, leidžiančias išskirti tiek įprastas, tiek netipines pinigines. Spalvų intensyvumas atspindi anomalijų tikimybę: tamsesnė oranžinė spalva rodo didesnį išskirtinumą, o šviesesni mazgai apima neutralius taškus.

Klasterio 0 medis atskleidžia sudėtingą anomalijų modelį, kuris susijęs su veiklos intensyvumu ir transakcijų eiga. Analizė parodo, kad izoliuojami mazgai išryškėja, kai tokie požymiai kaip Program Distribution Score, Activity Consistency ir Success Ratio tampa esminiais anomalijų indikatoriais.

Dbcsan Klasterio 1 izoliacinis medis, leidžia giliau suprasti šio klasterio struktūrą ir anomalijų pasiskirstymą (4 priedas). Medžio struktūra rodo, kad pagrindiniu skirstymo kriterijumi šaknies mazge tampa Activity Consistency (F1), kai ribinė reikšmė yra ≤ 0.875. Šis rodiklis svarbus analizuojant piniginių operacijų reguliarumą ir veiklos pastovumą. Tokia skirstymo logika atskleidžia, kad anomalijos dažniau aptinkamos tarp piniginių, kurios pasižymi neįprastu veiklos intensyvumu arba netolygiu veiklos dažnumu.

Kairėje šakoje, kai F1 ≤ 0.875, pastebimi požymiai, susiję su Fee-to-Amount Ratio (F3) ir Program Distribution Score (F9). F3 rodiklis, kai reikšmė mažesnė arba lygi 0.6, padeda identifikuoti pinigines, kurios vykdo operacijas su neįprastai mažu mokesčių ir sumos santykiu, o tai gali reikšti galimą automatizuotų arba neįprastų operacijų modelį. Tuo tarpu F9, kai reikšmė yra ≤ 0.9, rodo pinigines, kurios dalyvauja labai ribotame programų skaičiuje, kas gali reikšti specializuotą arba automatizuotą elgesį.

Dešinėje šakoje, kai F1 > 0.875, pagrindiniai anomalijų indikatoriai yra Success Ratio (F7) ir Average Transaction Amount (F2). F7 rodiklis, kai reikšmė yra ≤ 0.8, dažnai asocijuojamas su sumažėjusiu operacijų sėkmės lygiu, kas gali būti susiję su padidėjusia rizika arba nesėkmingais

automatizuotais bandymais vykdyti operacijas. F2, kai reikšmė yra labai maža (pvz., ≤ 0.001), rodo pinigines, kurios vykdo operacijas su neįprastai mažomis sumomis, galimai siekiant išvengti aptikimo arba testuojant automatizuotus veiksmus. Terminaliniai mazgai vizualiai išryškina ryškias anomalijas, kurios pasižymi itin žemais arba aukštais rodikliais. Tamsiai oranžinės spalvos mazgai (pvz., kai F3 ≤ 0.6 arba F7 ≤ 0.8) atspindi aukšto rizikos lygio pinigines, kurios galimai susijusios su automatizuotais veiklos modeliais ar neteisėta veikla.

Apibendrinant klasterio 1 anomalijos susijusios su veiklos netolygumu ir neįprastomis transakcijų charakteristikomis. Šis klasteris išsiskiria tuo, kad dažnai aptinkami rodikliai, tokie kaip Activity Consistency, Fee-to-Amount Ratio, ir Program Distribution Score, yra svarbūs įvertinant šių piniginių riziką Solana ekosistemoje.

Analizuojant DBSCAN metodu gautus klasterius ir jų izoliuojamų anomalijų modelius, atsiskleidžia svarbūs veiklos ypatumai ir skirtumai tarp botų bei žmonių piniginių. Abiejuose klasteriuose pastebimi svarbūs veiklos anomalijas lemiantys rodikliai. Tiek Mokesčių ir sumų santykis (Fee-to-Amount Ratio), tiek Instrukcijų sudėtingumas (Instruction Complexity) pasirodo esantys kritiniai veiksniai anomalijų aptikimo procese. Rodikliai leidžia identifikuoti specifinius modelius, susijusius su transakcijų eiga ir paskyros elgsenos ypatumais, neatsižvelgiant į tai, ar piniginė priklauso žmogui, ar yra botų valdoma.

21 lentelė

DBSCAN klasterių savybių palyginimas

Savybės	Klasteris 0	Klasteris 1
Dominuojantis tipas	Botų piniginės	Žmonių piniginės
Pagrindinės anomalijos	Transakcijų dažnumo modeliai, standartinis nuokrypis, unikalūs gavėjai	Dideli mokėjimų kiekiai, sudėtingos instrukcijos
Esminiai rodikliai	Mokesčių ir sumų santykis (Fee-to-Amount Ratio), Sėkmės rodiklis (Success Ratio), Programų paskirstymo balas (Program Distribution Score)	Paskyros vaidmenų įvairovė (Account Role Diversity), Instrukcijų sudėtingumas (Instruction Complexity), Aktyvios valandos per dieną (Active Hours Per Day)
Tipinė veikla	Automatizuotos intensyvios operacijos	Įvairialypė žmonių veikla

Šaltinis: sudaryta autoriaus.

Apibendrinant galima teigti, kad abiejų DBSCAN klasterių analizė atskleidė reikšmingus skirtumus tarp botų ir žmonių piniginių elgsenos modelių. Identifikuoti rodikliai, tokie kaip mokesčių ir sumų santykis bei instrukcijų sudėtingumas, leidžia geriau suprasti specifinius veiklos modelius ir anomalijų šaltinius.

22 lentelė

Anomalijų piniginės pagal DBSCAN klasterius

Šaltinis: sudaryta autoriaus.

Lentelėje pateikiami DBSCAN klasterių rezultatai, atskleidžiantys pagrindines anomalijų savybes ir jų pasiskirstymą tarp piniginių. Klasteris 0 daugiausia sudarytas iš botų piniginių, kurių elgsena pasižymi stipriu automatizuotu veiklos pobūdžiu. Šios piniginės turi aukštus anomalijų balus, o pagrindinės jų savybės, tokios kaip Lėšų-Kainos Santykis ir Veiklos Pastovumas, rodo sistemingą ir nuoseklią veiklą. Tokie rodikliai dažnai būdingi piniginėms, naudojamoms algoritminėje prekyboje ar automatizuotose operacijose.

Klasteris	Piniginės ID	Anomalijos balas	Pagrindinės ypatybės
0	Botas_28	1	Lėšų-Kainos Santykis (0.98), Sėkmės Rodiklis (0.87)
0	Botas_8	0.8945	DEX Transakcijų Dažnumas (1.00), Veiklos Pastovumas (0.97)
0	Botas_5	0.7939	Programų Pasiskirstymo Rodiklis (0.89), Aktyvios Valandos per Dieną (0.92)
1	Žmogus_66	1	Transakcijų Standartinis Nuokrypis (1.00), Gavėjų Skaičius (1.00)
1	Žmogus_17	0.8564	DEX Transakcijų Dažnumas (1.00), Transakcijų Dažnis (0.10)
1	Žmogus_45	0.8197	Vaidmenų Įvairovė (0.98), Lėšų-Kainos Santykis (0.99)

Klasteryje 1 dominuoja žmonių piniginės, tačiau jų anomalijų balai taip pat rodo tam tikrus netipinius elgsenos bruožus. Šios piniginės pasižymi dideliu Transakcijų Standartiniu Nuokrypiu ir Gavėjų Skaičiumi, kas gali būti susiję su plačiu operacijų spektru ar netipiniu elgsenos pobūdžiu. Be to, tam tikros piniginės šioje grupėje rodo reikšmingus rodiklius, tokius kaip Vaidmenų Įvairovė, leidžiantys manyti apie sudėtingesnes operacijas ar įvairiapusišką veiklą.

Apibendrinant, lentelė parodo, kaip skirtingos piniginės, priskirtos DBSCAN klasteriams, turi unikalius rodiklius, leidžiančius geriau suprasti jų elgsenos modelius. Ši informacija yra svarbi tolimesniam anomalijų identifikavimui ir gilesniam Solana ekosistemos veiklos supratimui.

3.5.3. DBSCAN, K-means ir Isolation Forest metodų hierarchinės anomalijų analizės palyginimas

Palyginti DBSCAN, K-means ir Isolation Forest metodus, taikomus hierarchinei anomalijų aptikimo analizei Solana blokų grandinės duomenyse. Analizė siekia nustatyti, kuris metodas yra efektyvesnis atsižvelgiant į anomalijų aptikimo rezultatus, duomenų interpretavimo galimybes ir

svarbiausių požymių identifikavimą. Palyginimas leis įvertinti metodų tinkamumą skirtingiems naudojimo atvejams ir pateikti rekomendacijas dėl jų taikymo efektyvumo.

23 lentelė

DBSCAN, K-means ir Isolation Forest metodų anomalijų identifikavimo rezultatai

Metodas	Anomalijų kategorija	Klasteriai	Anomalijų skaičius	Reikšmingiausios ypatybės	Papildomumas su kitais metodais
DBSCAN	Botų piniginės	2	27	Veiklos Pastovumas (72.22%), DEX Transakcijų Dažnumas (27.78%)	Identifikuoja tankiuose regionuose esančias anomalijas su aiškiais elgesio modeliais.
DBSCAN	Žmonių piniginės	2	13	Transakcijų Dažnis (28.45%), Vaidmenų Įvairovė (15.62%), Aktyvios Valandos (15.13%)	Identifikuoja kompleksines žmonių elgesio anomalijas, kurias lemia daugiafaktoriniai bruožai.
K-means	Botų piniginės	3	15	Veiklos Pastovumas (72.22%), Instrukcijų Kompleksiškumas (27.78%)	Nustato globalius anomalijų modelius ir papildo DBSCAN.
K-means	Žmonių piniginės	3	25	Transakcijų Dažnis (28.45%), Gavėjų Skaičius (20.00%), Fee-to-Amount Ratio (15.00%)	Pabrėžia žmonių piniginių veiklos sudėtingumą ir įvairius elgesio modelius.
Isolation Forest	Botų piniginės	-	23	Fee-to-Amount Ratio (25.00%), Veiklos Pastovumas (20.00%), DEX Transakcijų Dažnumas (15.00%)	Patvirtina DBSCAN ir K-means rezultatus, suteikia daugiau pasitikėjimo.
Isolation Forest	Žmonių piniginės	-	17	Gavėjų Skaičius (25.00%), Instrukcijų Kompleksiškumas (20.00%), Aktyvios Valandos (15.00%)	Papildo DBSCAN ir K-means, suteikia papildomų įžvalgų apie sudėtingesnį žmonių elgesį.

Šaltinis: sudaryta autoriaus.

Lentelėje pateikti rezultatai parodo, kaip botų ir žmonių piniginių anomalijos buvo identifikuotos naudojant DBSCAN, K-means ir Isolation Forest metodus. DBSCAN metodas išsiskiria gebėjimu aptikti tankiuose regionuose esančias botų pinigines, kuriose vyrauja tokios savybės kaip Veiklos Pastovumas ir DEX Transakcijų Dažnumas. Šis metodas puikiai tinka automatizuotų veiksmų lokalizavimui specifinėse duomenų erdvės vietose. Tačiau šio metodo ribotumas atsiskleidžia analizuojant retesnius žmonių piniginių elgesio modelius, kuriems identifikuoti reikalingas globalesnis požiūris.

K-means metodas, priešingai, išsiskiria gebėjimu aptikti tiek botų, tiek žmonių anomalijas platesniame duomenų kontekste. Pagrindinės botų anomalijos savybės, tokios kaip Instrukcijų Kompleksiškumas ir Fee-to-Amount Ratio, leidžia identifikuoti globalius modelius, kurie gali būti

nepastebėti tankio metoduose. Žmonių piniginių atveju K-means pabrėžia sudėtingesnius elgesio bruožus, tokius kaip Transakcijų Dažnis ir Gavėjų Skaičius, kurie rodo didelę elgesio įvairovę.

Isolation Forest metodas vaidina esminį vaidmenį patvirtinant DBSCAN ir K-means aptiktas anomalijas. Šis metodas išryškina tas pačias savybes, kurios buvo svarbios ir kitų metodų analizėje, tačiau taip pat suteikia daugiau pasitikėjimo dėl anomalijų patikimumo. Isolation Forest ypač efektyvus identifikuojant subtilias žmonių veiklos anomalijas, kurios gali būti praleistos naudojant tik klasterizacijos metodus.

Taikomų metodų tarpusavio papildomumas atsiskleidžia jų taikymo skirtumuose. DBSCAN suteikia tikslią lokalizaciją tankiuose regionuose, K-means užtikrina globalų požiūrį, o Isolation Forest sustiprina rezultatus ir užtikrina jų patikimumą Apibdenrinus skirtingi metodai gali būti taikomi anomalijų aptikimui Solana blokų grandinėje. Palyginimas atskleidė, jog kiekvienas metodas turi savitų privalumų – nuo tankių anomalijų regionų aptikimo (DBSCAN), globalių elgesio modelių nustatymo (K-means) iki subtilių žmonių veiklos anomalijų identifikavimo (Isolation Forest). Holistinis požiūris pabrėžia metodų tarpusavio papildomumą, leidžiantį užtikrinti patikimesnį ir išsamesnį anomalijų aptikimo procesą.

3.6 Ekspermento vertinimas

Šiame poskyryje siekiama ne tik nuosekliai ir išsamiai įvertinti atlikto eksperimento rezultatus, pritaikytus klasterizacijos bei anomalijų aptikimo metodus, bet ir nustatyti patikimiausią botų aptikimo sprendimą Solana blokų grandinėje. Vertinimas atliekamas remiantis apskaičiuotomis metrikomis: klasterizacijos kokybė matuojama pagal Silhouette Score, Davies–

Bouldin ir Calinski–Harabasz rodiklius, o anomalijų aptikimo efektyvumui įvertinti taikomi ROC-AUC, tikslumo (Precision), atkūrimo (Recall) ir F1 rodikliai. Šis duomenų analizės metodų palyginimas leidžia nustatyti jų gebėjimą išskirti netipinius duomenis (galimus botus) iš normalių.

Sprendžiant botų identifikavimo iššūkius, aktualu atsižvelgti į bendrą Solana blokų grandinės ekosistemos sudėtingumą. Sudėtingas duomenų srautas, didelis operacijų kiekis ir įvairialypė vartotojų elgsena reikalauja efektyvių, patikimų ir greitų klasterizacijos bei anomalijų aptikimo metodų. Sėkmingai pritaikius šiuos metodus, galima užtikrinti didesnį tinklo saugumą, nustatyti neįprasto elgesio šaltinius ir optimizuoti veiklos procesus. Analizuojant tiek klasterizacijos, tiek anomalijų aptikimo rezultatus, atsiskleidžia atitinkamų metodų pranašumai ir trūkumai, taip pat jų derinimo galimybės.

24 lentelė

Anomalijų aptikimo vertinimo metrikų rezultatai pagal metodus

Metodas	**Tikslumas (Precision)**	**Atsikartojamumas (Recall)**	**F1-rezultatas (F1-Score)**	**ROC-AUC**	**Silueto balas (Silhouette Score)**
DBSCAN	0.675	0.900	0.771	0.835	0.6308
K-means	0.375	0.500	0.429	0.625	0.4511
Isolation Forest	0.575	0.767	0.657	1.000	0.5842

Šaltinis: sudaryta autoriaus.

Iš lentelėje pateiktų rodiklių matyti, kad tarp trijų analizuotų metodų (DBSCAN, K-means ir Isolation Forest) ryškiausiai išsiskiria DBSCAN ir Isolation Forest, o K-means rodikliai lieka žemiausi visose vertintose kategorijose. DBSCAN, remiantis nurodytomis tikslumo, atkūrimo ir F1 reikšmėmis, demonstruoja pakankamai subalansuotą anomalijų aptikimo efektyvumą, kur ypač didelė atkūrimo vertė rodo gebėjimą aptikti didžiąją dalį netipinių atvejų. Toks metodas yra naudingas, kai svarbiausia nepalikti nepastebėtų kritinių anomalijų (pavyzdžiui, potencialiai botų veiklos), tačiau aukštas atkūrimo rodiklis kartu lemia ir didesnį klaidingų pavojaus signalų skaičių.

K-means, vertinant visus lentelėje nurodytus rodiklius, reikšmingai atsilieka nuo kitų dviejų metodų. Nepakankamas tiek tikslumas, tiek atkūrimas sąlygoja mažesnę bendrą anomalijų aptikimo kokybę, o santykinai žemas silueto balas atskleidžia prastoką gebėjimą suformuoti

aiškius klasterius. Šios išvados patvirtina, kad K-means gali būti ne pats tinkamiausias metodas sprendžiant botų aptikimo uždavinius, ypač jei duomenys neturi sferinės struktūros, o tikėtinų klasterių skaičius iš anksto nėra aiškus.

Tuo tarpu Isolation Forest, vertinant jo reikšmes pagal lentelėje nurodytus tikslumo, atkūrimo ir ROC-AUC rodiklius, pasižymi stipriu gebėjimu atskirti anomalijas nuo normalių duomenų, nors tarpinės F1 bei silueto balo reikšmės rodo, kad išlieka tam tikras disbalansas tarp nustatomų anomalijų kiekio ir klaidingų priskyrimų. Vis dėlto itin aukštas ROC-AUC leidžia daryti prielaidą, kad, tinkamai pasirinkus ribinę vertę (angl. threshold), šis metodas galėtų pasižymėti ypač aukštu tikslumu, o tai labai svarbu identifikuojant botų veiklą, kuri gali būti maskuojama sudėtingoje ir didelės apimties Solana blokų grandinės ekosistemoje.

Atsižvelgiant į bendrą rezultatyvumą, galima teigti, kad DBSCAN, dėl didelio atkūrimo rodiklio ir santykinai gero bendro efektyvumo, yra tinkamas variantas, kai prioritetas teikiamas maksimaliam potencialiai kenksmingų atvejų aptikimui. Isolation Forest, savo ruožtu, pasižymi išskirtiniu gebėjimu atskirti tipiškus ir netipinius atvejus, kas pasimato iš jo itin aukšto ROC-AUC, todėl, subtiliai parinkus modelio parametrus, jis gali pranokti DBSCAN išvengiant pernelyg dažno klaidingų pavojaus signalų generavimo. K-means galėtų būti svarstomas nebent kaip papildoma priemonė duomenų struktūrai preliminariai įvertinti, tačiau remiantis lentelės rodikliais, jis vargiai bus efektyviausias pasirinkimas botų aptikimui.

25 lentelė

Kombinuotų metodų anomalijų aptikimo vertinimo metrikų rezultatai

Metodų kombinacija	Tikslumas (Precision)	Atsikartojamumas (Recall)	F1-rezultatas (F1-Score)
DBSCAN + Isolation Forest	0.676	0.833	0.746
K-means + Isolation Forest	0.474	0.600	0.529
DBSCAN + K-means	0.526	0.667	0.588

Šaltinis: sudaryta autoriaus.

Pateiktos metrikos rodo, kad skirtingų metodų deriniai lemia pakitusią anomalijų aptikimo kokybę, vertinant tikslumą, atsikartojamumą ir F1-rezultatą. Bendras šių rodiklių kontekstas išryškina, jog ne visos kombinacijos sėkmingai subalansuoja jautrumą retoms anomalijoms ir gebėjimą išvengti perteklinių klaidingai teigiamų atvejų. Analizuojant tokius rezultatus, aktualu

atsižvelgti į duomenų pobūdį bei jų struktūrą: tankių sričių identifikavimą, nenuoseklų elgesį, reikalaujantį specializuoto aptikimo, bei bendrą algoritmų tarpusavio sąveiką.

Pagal lentelėje nurodytas reikšmes, geriausiai pasirodo kombinacija, jungianti DBSCAN ir Isolation Forest. Šis derinys pasižymi aukštu atsikartojamumo rodikliu, kas liudija gebėjimą aptikti didesniąją dalį anomalijų, taip pat pakankamai gera tikslumo reikšmė padeda palaikyti patikimą bendrą F1-rezultatą. DBSCAN, identifikuodamas tankesnes sritis, iš anksto atrenka galimai problemines duomenų dalis, kurias vėliau sėkmingai išskiria Isolation Forest, sutelktas į retų ar atitolusių nuo daugumos taškų suradimą. Toks metodų papildomumas pasirodo itin veiksmingas, ypač kai duomenys kompleksiški ir pasižymi įvairialypėmis savybėmis, kaip yra Solana blokų grandinės ekosistemoje.

Taip pat matyti, kad K-means derinimas su Isolation Forest neduoda tokio pastebimo pagerėjimo, kokio būtų galima tikėtis. Viena priežasčių – K-means ribotumas, susijęs su būtinybe iš anksto nurodyti klasterių skaičių ir prielaida apie sferines duomenų grupes, kuri realiomis sąlygomis dažnai nepasitvirtina. Dėl to, net ir pritaikius patikimą anomalijų aptikimo algoritmą, galima netiksliai apdoroti duomenis pirmajame žingsnyje, o gautas rezultatas išlieka vidutinis. Analogiškai, DBSCAN ir K-means kombinacija taip pat nesudaro žymesnės sinergijos, nes tankumu pagrįstas DBSCAN jau savaime gana efektyviai tvarkosi su nevienalytėmis duomenų grupėmis, taigi K-means pranašumai čia neatsiskleidžia.

Galutinė įžvalga rodo, kad derinant kelis metodus, svarbiausia atsižvelgti į jų tarpusavio papildomumą. Tarpininkavimas tarp tankiais regionais pasižyminčio klasterizacijos algoritmo ir specializuoto anomalijų aptikimo sprendimo leidžia sėkmingiau išaiškinti retas, bet potencialiai svarbias anomalijas. Šiuo požiūriu DBSCAN ir Isolation Forest derinys demonstravo aukščiausią efektyvumą, todėl jį rekomenduotina taikyti praktinėse situacijose, kuriose reikia ypač kruopštaus ir patikimo neįprastos veiklos nustatymo. Kitos kombinacijos, nors gali turėti tam tikrų privalumų specifiškose situacijose, bendrai nėra tokios veiksmingos aptikti subtilių duomenų elgsenos nukrypimų, būdingų sparčiai besivystančiai blokų grandinės aplinkai.

Metodas	Stiprybės	Trūkumai

DBSCAN	Aptinka tankias botų grupes, gerai tvarkosi su nesferinėmis duomenų struktūromis	Gali praleisti atskiras ar retesnes anomalijas dėl parametrų jautrumo
K-means	Efektyvus globalių klasterizacijos modelių nustatymui	Blogai tvarkosi su nesferinėmis grupėmis, didelis klaidingų teigiamų anomalijų skaičius
Isolation Forest	Geba izoliuoti anomalijas remiantis svarbiausiais ypatumais, aukštas tikslumas anomalijoms aptikti	Sunku išskirti žmogaus elgesį nuo žmogų imituojančių botų, gali atsirasti klaidingų neigiamų rezultatų
DBSCAN + Isolation Forest	Subalansuotas tikslumas ir atsikartojamumas, sumažina klaidingų signalų skaičių	Didesnis skaičiavimų sudėtingumas
K-means + Isolation Forest	Pagerina klaidingų signalų valdymą, tačiau neženkliai	Sumažėję rezultatai lyginant su kitomis kombinacijomis
Visų metodų kombinacija	Teorinis potencialas identifikuoti sudėtingus modelius	Didelis sudėtingumas ir klaidingų signalų skaičius, per daug parametrų optimizavimui

26 lentelė

Metodų stiprybės ir trūkumai

Šaltinis: sudaryta autoriaus.

DBSCAN pasižymi gebėjimu aptikti tankias botų grupes bei efektyviai tvarkytis su nesferinėmis duomenų struktūromis, tačiau yra linkęs praleisti retesnes anomalijas, jei jo parametrų reikšmės nėra kruopščiai suderintos. K-means išlieka efektyvus įprastoms, globalioms klasterizacijos užduotims, tačiau prastai reaguoja į heterogenišką duomenų erdvę ir dėl to padidina klaidingų teigiamų atvejų skaičių. Isolation Forest pranašumai atsiskleidžia aukštame tikslume, ieškant retų elgsenos šablonų, bet, nesant aiškios ribos tarp žmogaus elgesio ir žmogų imituojančių botų, rizikuoja leisti praslysti daliai svarbių anomalijų. Apjungus DBSCAN su Isolation Forest, išryškėja darnaus rodiklių balanso potencialas, nes šis derinys sumažina klaidingų pavojaus signalų gausą, nors tuo pačiu didina bendrą skaičiavimų sudėtingumą. Tuo tarpu K-means ir Isolation Forest sinergija tik nežymiai pagerina klaidingų signalų valdymą, o bendri rezultatai dažnai prastesni nei kitų kombinacijų. Galiausiai, visų metodų sujungimas, nors ir teoriškai galintis identifikuoti sudėtingus modelius, praktiškai susiduria su per dideliu parametrų optimizavimo krūviu, didesniu klaidingų signalų skaičiumi ir sunkiai valdomu sistemos sudėtingumu, kas verčia kruopščiai pasverti tokios integracijos naudą.

27 lentelė

Botų aptikimo metodų rezultatų palyginimas su ankstesniais tyrimais

Tyrimas	Metodas	Precision	Recall	F1-rezultatas	Pavadinimas
Lukas Palionis (2025)	DBSCAN + Isolation Forest	0.676	0.833	0.746	"Automatizuotų sandorių aptikimo metodas Solanos tinkle"
	K-means + Isolation Forest	0.474	0.6	0.529	"Automatizuotų sandorių aptikimo metodas Solanos tinkle"
	DBSCAN + K-means	0.526	0.667	0.588	"Automatizuotų sandorių aptikimo metodas Solanos tinkle"
Niedermayer et al. (2024)	DBSCAN, Hierarchical Clustering, K-means	0.728	0.85	0.755	"Detecting financial bots on the Ethereum blockchain"
Zwang et al. (2018)	DBSCAN, K-means	0.612	0.793	0.671	"Detecting Bot Activity in the Ethereum Blockchain Network"
Li, Guan & Lee (2023)	Isolation Forest, KNN, Random Forest	0.637	0.814	0.714	"Towards understanding and characterizing the arbitrage bot scam in the wild (Ethereum)"

Šaltinis: sudaryta autoriaus.

Pateiktoje lentelėje palyginami įvairūs tyrimai, nagrinėjantys automatizuotų sandorių bei botų aptikimą skirtingose blokų grandinėse. Iš mūsų atlikto tyrimo rezultatų matyti, kad DBSCAN ir Isolation Forest derinys efektyviau atskiria netipines transakcijas Solanos tinkle. Šis derinys leidžia aptikti tiek tankias botų grupes, tiek retesnius anomalinius elgesio modelius. Taip pat pastebėta, kad K-means metodas, dėl savo polinkio į sferines struktūras, dažniau susiduria su iššūkiais analizuojant kompleksines Solanos duomenų struktūras, todėl rezultatai yra mažiau patikimi.

Kiti tyrimai, tokie kaip Niedermayer et al. (2024) ir Zwang et al. (2018), orientuojasi į Ethereum blokų grandinę. Šiuose darbuose akcentuojamas DBSCAN metodo pranašumas, kai jis derinamas su hierarchiniais klasterizacijos metodais ar K-means algoritmu. Tai leidžia pasiekti didesnį F1-rezultatą, išryškinant tankias botų grupes ir jų elgesio modelius. Pavyzdžiui, Niedermayer et al. (2024) darbe pateikti rezultatai rodo, kad hierarchiniai metodai veiksmingai papildo DBSCAN gebėjimą atskirti anomalijas, o tai užtikrina aukštą tikslumo ir atpažinimo rodiklių balansą.

Li et al. (2023) tyrimas gilinasi į Ethereum arbitražo botų veikimą. Naudojant Isolation Forest kartu su KNN ir Random Forest, autoriai pabrėžia šių metodų potencialą atskirti sudėtingus automatizuotus modelius, kurie gali manipuliuoti rinkos likvidumu. Šis tyrimas išryškina poreikį derinti kelis algoritmus, kad būtų galima efektyviai identifikuoti tiek akivaizdžius, tiek subtilius anomalijų atvejus.

Apibendrinant norint efektyviai aptikti botus ir automatizuotas perlaidas, svarbu atsižvelgti į konkrečias blokų grandinės savybes. Šiame darbe taikytų metodų analizė rodo, kad derinant kelis algoritmus, kaip DBSCAN ir Isolation Forest, galima pasiekti gerus rezultatus net ir skirtingose ekosistemose. Tačiau kiekvienas metodas turi savo ribotumus, todėl jų tobulinimas ir pritaikymas specifiniams scenarijams, kaip manipuliaciniai arbitražo modeliai ar decentralizuotų finansinių (DeFi) platformų apsauga, yra būtinas siekiant užtikrinti tinklų stabilumą ir saugumą.

28 lentelė

Ekspermento metodų pritaikymo galimybės Solana ekosistemoje

Pritaikymo sritis	Naudojami metodai	Galimas efektas
Automatizuotų perlaidų identifikavimas	DBSCAN, Isolation Forest	Tankių botų grupių aptikimas, retų anomalijų nustatymas, sumažėjęs netinkamų transakcijų skaičius.
DeFi platformų apsauga nuo manipuliacinių sandorių	DBSCAN, Isolation Forest, K-means (ribotai)	Neįprastų rinkos judėjimų atskleidimas, laiku perduodami perspėjimai, galimybė stabdyti piktavališkus veiksmus.
Vertės (angl. *value*) srautų kontrolė dinamiškuose tinkluose	DBSCAN + Isolation Forest	Greitas srautų struktūrizavimas, botų sukurtų nukrypimų nustatymas realiuoju laiku, prisitaikymas prie skirtingų duomenų kiekių.
Parametrų optimizavimas bei testavimas skirtingose ekosistemose	DBSCAN, Isolation Forest	Galimybė universalizuoti algoritmus, užtikrinant efektyvų naudojimą kitose blokų grandinėse (Ethereum, BSC).

Šaltinis: sudaryta autoriaus.

Pagrindinės nagrinėtų metodų pritaikymo sritys, atsižvelgiant į jų specifiką ir galimą poveikį (žr. 25 lentelę). DBSCAN ir Isolation Forest metodai išsiskiria gebėjimu aptikti tankias botų grupes bei retas anomalijas, todėl yra itin tinkami automatizuotų perlaidų ir manipuliacinių sandorių identifikavimui Solana ekosistemoje. K-means metodas, nors ir ribotai taikomas, gali būti naudingas analizuojant globalias rinkos tendencijas, tačiau jo efektyvumas yra žemesnis. Be to,

derinant DBSCAN ir Isolation Forest, galima realiuoju laiku nustatyti botų sukurtus nukrypimus ir struktūrizuoti sudėtingus duomenų srautus. Šių metodų optimizavimas bei testavimas kitose blokų grandinėse, tokiose kaip Ethereum ar Binance Smart Chain, suteikia galimybę kurti universalius algoritmus, pritaikomus skirtingose ekosistemose, taip stiprinant decentralizuotų finansų saugumą.

Apibendrinant galima konstatuoti, kad atliktas eksperimentas pasiekė savo pirminius tikslus: išbandyti ir įvertinti kelis klasterizacijos bei anomalijų aptikimo metodus automatizuotų perlaidų ir galimų botų veiklos nustatymui Solana blokų grandinėje. Gauti rezultatai rodo, kad DBSCAN metodas efektyviau aptinka tankias ir nereguliarų aktyvumą rodančias grupes, tuo tarpu Isolation Forest padeda atskleisti retesnes, bet strategiškai svarbias anomalijas. Nors K-means pasirodė prasčiau įvairiapusėse, nesferinėse duomenų struktūrose, eksperimente išryškėjo metodų kombinacijų nauda, ypač derinant DBSCAN ir Isolation Forest. Šios įžvalgos ne tik patvirtina eksperimentinę hipotezę, kad tinkamai parinkti algoritmai gali reikšmingai padidinti botų ir automatizuotų perlaidų aptikimo tikslumą, bet ir suteikia praktinio pritaikymo gairių, kaip toliau tobulinti parametrų derinimą bei sprendimų priėmimo procesus Solanos ir kitose blokų grandinėse.

IŠVADOS

1. Apibendrinus automatizuotų sandorių (botų) veiklos tipus ir jų poveikį blokų grandinės sistemoms, nustatyta, kad botai daro reikšmingą įtaką tiek teigiamais, tiek neigiamais aspektais. Jie gali pagerinti rinkos efektyvumą, užtikrinti likvidumą ir vykdyti sudėtingas operacijas, kurios prisideda prie decentralizuotų finansų ekosistemų vystymosi. Tačiau jų veikla taip pat kelia rimtų iššūkių rinkos skaidrumui ir tinklo stabilumui. Pagrindiniai botų tipai, tokie kaip aukšto dažnio prekybos botai, arbitražo botai ir rinkos formavimo botai, kartu su nesąžiningoms manipuliacijoms naudojamais botais, tokiais kaip pump-and-dump ir sandwich atakos, pasižymi plačiu funkcionalumu, tačiau dažnai iškraipo rinkos dinamiką, mažina skaidrumą bei kelia saugumo riziką. Botų aktyvumas ne tik dirbtinai padidina likvidumą, bet ir apsunkina kainų analizę, apkrauna tinklą bei mažina sandorių efektyvumą. Dėl šių priežasčių botų manipuliacijos prisideda prie investuotojų pasitikėjimo mažėjimo ir kelia grėsmę decentralizuotų finansų ekosistemų tvarumui.

2. Sukurta metodologija, skirta piniginių elgsenos analizei ir anomalijų aptikimui Solanos tinkle, sudaryta iš duomenų rinkimo, apdorojimo, požymių inžinerijos ir klasterizacijos bei anomalijų aptikimo etapų. Duomenys buvo surinkti naudojant Solanos RPC API, apdoroti .NET pagrindu veikiančiame backend'e, normalizuoti ir saugomi SQLite duomenų bazėje. Analizei buvo pritaikyti K-means ir DBSCAN klasterizacijos algoritmai naudojantis Python porgramavimo kalba, leidžiantys identifikuoti panašias piniginių elgsenas bei tankius sandorių grupių modelius, taip pat Isolation Forest algoritmas, skirtas reikšmingai nuo normos nukrypusioms anomalijoms aptikti. Požymių inžinerijos procesas papildė duomenis reikšmingais požymiais, tokiais kaip sandorių dažnis, vidutinė sandorio suma, mokesčių ir sumos santykis, aktyvumo nuoseklumas ir programų pasiskirstymo balas, kurie yra esminiai botų veiklos aptikimui. Naudotų algoritmų efektyvumas buvo įvertintas Silhouette Score, ROC-AUC, Precision, Recall bei F1-score metrikomis, užtikrinant tikslumą ir patikimumą. Sukurta metodologija pademonstravo galimybes išsamiai analizuoti Solanos tinklo transakcijas, identifikuoti automatizuotų sandorių veiklos modelius ir aptikti anomalijas, prisidedant prie tinklo saugumo didinimo ir rinkos skaidrumo gerinimo.

3. Tyrimas parodė, kad taikant K-means, DBSCAN ir Isolation Forest algoritmus galima sėkmingai atskirti tipines ir netipines piniginių elgsenos grupes Solanos tinkle, tačiau

jų efektyvumas ženkliai priklauso nuo duomenų struktūros ir parametrų suderinimo. K-means gerai veikia, kai klasterių forma yra pakankamai „sferinė“, tačiau esant sudėtingesnėms struktūroms pasirodo kukliau. DBSCAN pasižymi gebėjimu aptikti tankius anomalijų židinius, todėl ypač naudingas nustatant organizuotus botų veiklos modelius. Isolation Forest veiksmingai išskiria retesnes, bet dažnai strategiškai reikšmingas elgsenos nukrypimų formas, kurios nepastebimos taikant įprastą klasterizaciją. Tokiu būdu, apjungus šių trijų algoritmų rezultatus, gauta visapusiška įžvalga apie tipišką ir atipinį piniginių elgesį, suteikianti vertingą pamatą rizikų vertinimui bei automatizuotų perlaidų stebėsenai.

4. Gautų rezultatų analizė rodo, kad efektyviausias sprendimas Solanos tinklo saugumui ir rinkos skaidrumui didinti yra DBSCAN derinys su Isolation Forest, užtikrinantis aukštesnį tiek tikslumo, tiek atkūrimo (atsikartojamumo) rodiklį. Ši kombinacija leidžia ne tik aptikti dažniausiai sutinkamus manipuliacijų atvejus, bet ir užčiuopti retesnius, tačiau potencialiai pavojingesnius anomalijų scenarijus. Remiantis eksperimento išvadomis, rekomenduotina nuosekliai tikrinti parametrų reikšmes, kaip MinPts ar medžių skaiči, kad būtų pasiektas optimalus jautrumo ir specifiškumo santykis. Be to, siekiant užtikrinti nuolatinę tinklo apsaugą, vertėtų reguliariai atnaujinti treniravimo duomenų rinkinį, įtraukiant naujausias transakcijų elgsenos tendencijas. Toks holistinis požiūris suteiks galimybę anksti aptikti ir pažaboti neteisėtus veiksmus, taip padidinant Solanos ekosistemos patikimumą.

LITERATŪRA

1. Zheng, Z., Xie, S., Dai, H. N., Chen, X., & Wang, H. (2018). Blockchain challenges and opportunities: A survey. International Journal of Web and Grid Services, 14 (4), 352-375.
2. Nakamoto, S. (2008). Bitcoin: A peer-to-peer electronic cash system. Decentralized Business Review, 21260.
3. Xie, J., Yu, F. R., Huang, T., Xie, R., Liu, J., & Liu, Y. (2019). A survey on the scalability of blockchain systems. IEEE Network, 33(5), 166-173.
4. Wüst, K., & Gervais, A. (2018, June). Do you need a blockchain?. In 2018 Crypto Valley Conference on Blockchain Technology (CVCBT) (pp. 45-54). IEEE.
5. Bach, L. M., Mihaljevic, B., & Zagar, M. (2018, May). Comparative analysis of blockchain consensus algorithms. In 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO) (pp. 1545-1550). IEEE.
6. Mingxiao, D., Xiaofeng, M., Zhe, Z., Xiangwei, W., & Qijun, C. (2017, October). A review on consensus algorithm of blockchain. In 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC) (pp. 2567-2572). IEEE.
7. Drescher, D. (2017). *Blockchain basics: A non-technical introduction in 25 steps* (pp. 57–62) Apress.
8. Casino, F., Dasaklis, T. K., & Patsakis, C. (2019). A systematic literature review of blockchain-based applications: current status, classification and open issues. Telematics and Informatics, 36, 55-81.
9. Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. Information processing & management, 45(4), 427-437.
10. Yli-Huumo, J., Ko, D., Choi, S., Park, S., & Smolander, K. (2016). Where is current research on blockchain technology?—a systematic review. PloS one, 11(10), e0163477.
11. Ferdous, M. S., Chowdhury, M. J. M., Hoque, M. A., & Colman, A. (2020). Blockchain consensus algorithms: A survey. arXiv preprint arXiv:2001.07091.
12. Reid, F., & Harrigan, M. (2013). An analysis of anonymity in the bitcoin system. In Security and privacy in social networks (pp. 197-223). Springer, New York, NY.

13. Möser, M., Soska, K., Heilman, E., Lee, K., Heffan, H., Srivastava, S., ir Christin, N. (2018, October). An empirical analysis of traceability in the monero blockchain. In Proceedings on Privacy Enhancing Technologies (Vol. 2018, No. 3, pp. 143-163). Sciendo.

14. Yakovenko, A. (2018). Solana: A new architecture for a high performance blockchain v0. 8.13. Whitepaper.

15. Yakovenko, A. (2017). Proof of History. Decentralized Systems Workshop.

16. Yakovenko, A. (2019). Gulf Stream: Solana's mempool-less transaction forwarding protocol. Medium article.

17. Solana Documentation. (2023). Solana Developers. https://docs.solana.com/developers

18. Kahmann, F., Honecker, F., Dreyer, J., Fischer, M., & Tönjes, R. (2023). Performance Comparison of Directed Acyclic Graph-Based Distributed Ledgers and Blockchain Platforms. Computers, 12(12), 257.

19. Xu, B., Luthra, P., Cole, Z., & Blakely, N. (2018). EOS: An architectural, performance, and economic analysis. arXiv preprint arXiv:1808.04774.

20. EigenPhi. (2022). EigenPhi. https://eigenphi-1.gitbook.io/classroom

21. Federico Cernera, Massimo La Morgia, Alessandro Mei, Alberto Maria Mongardini, and Francesco Sassi. Ready, Aim, Snipe! Analysis of Sniper Bots and their Impact on the DeFi Ecosystem. In Companion Proceedings of the ACM Web Conference 2023, pages 1093–1102, Austin TX USA, April 2023. ACM.

22. Stefan Kitzler, Friedhelm Victor, Pietro Saggese, and Bernhard Haslhofer. Disentangling Decentralized Finance (DeFi) Compositions. ACM Transactions on the Web, 17(2):10:1–10:26, March 2023.

23. Julien Piet, Jaiden Fairoze, and Nicholas Weaver. Extracting Godl [sic] from the Salt Mines: Ethereum Miners Extracting Value, March 2022. arXiv:2203.15930 [cs].

24. Kaihua Qin, Liyi Zhou, Pablo Gamito, Philipp Jovanovic, and Arthur Gervais. An Empirical Study of DeFi Liquidations: Incentives, Risks, and Instabilities. In Proceedings of the 21st ACM Internet Measurement Conference, pages 336–350, November 2021. arXiv:2106.06389 [cs, q-fin].

25. Ye Wang, Yan Chen, Haotian Wu, Liyi Zhou, Shuiguang Deng, and Roger Wattenhofer. Cyclic Arbitrage in Decentralized Exchanges. In Companion Proceedings of the Web Conference 2022, WWW '22, pages 12–19, New York, NY, USA, August 2022. Association for Computing Machinery.

26. Sam Werner, Daniel Perez, Lewis Gudgeon, Ariah Klages-Mundt, Dominik Harz, and William Knottenbelt. SoK: Decentralized Finance (DeFi). In Proceedings of the 4th ACM Conference on Advances in Financial Technologies, AFT '22, pages 30–46, New York, NY, USA, July 2023. Association for Computing Machinery.

27. Anatoly Yakovenko. Solana: A new architecture for a high performance blockchain v0. 8.13. Whitepaper, 2018.

28. Morit Zwang, Shahar Somin, Alex Pentland, and Yaniv Altshuler. Detecting Bot Activity in the Ethereum Blockchain Network, October 2018. arXiv:1810.01591 [cs].

29. Malaiya, R., Kwon, D., Suh, S., ir Kim, H. (2019). An empirical evaluation of deep learning for network anomaly detection. *IEEE Access*, PP(99), 1-1.

30. Chen, W., Zheng, Z., Cui, J., Ngai, E., Zheng, P., ir Zhou, Y. (2018). Detecting Ponzi schemes on Ethereum: Towards healthier blockchain technology. In Proceedings of the 2018 World Wide Web Conference (pp. 1409-1418).

31. Wu, J., Liu, J., Zhao, Y., ir Zheng, Z. (2021). Analysis of cryptocurrency transactions from a network perspective:

32. Miftahuddin, Y., & Al-Ghifary, M. H. (2024). X bot detection using one-class classification methods with isolation forest algorithm. *International Journal on Advanced Science Engineering and Information Technology*, 14(4), 1233–1239.

33. Nidhi, ir Patel, K. A. (2016). An efficient and scalable density-based clustering algorithm for normalized data. *Procedia Computer Science*, 89, 552–561.

34. Niedermayer, T., Saggese, P., ir Haslhofer, B. (2024). Detecting financial bots on the Ethereum blockchain. *arXiv*.

35. Chengsheng, T., Huacheng, L., & Bing, X. (2017). AdaBoost typical algorithm and its application research. *MATEC Web of Conferences, 139*, 00222.

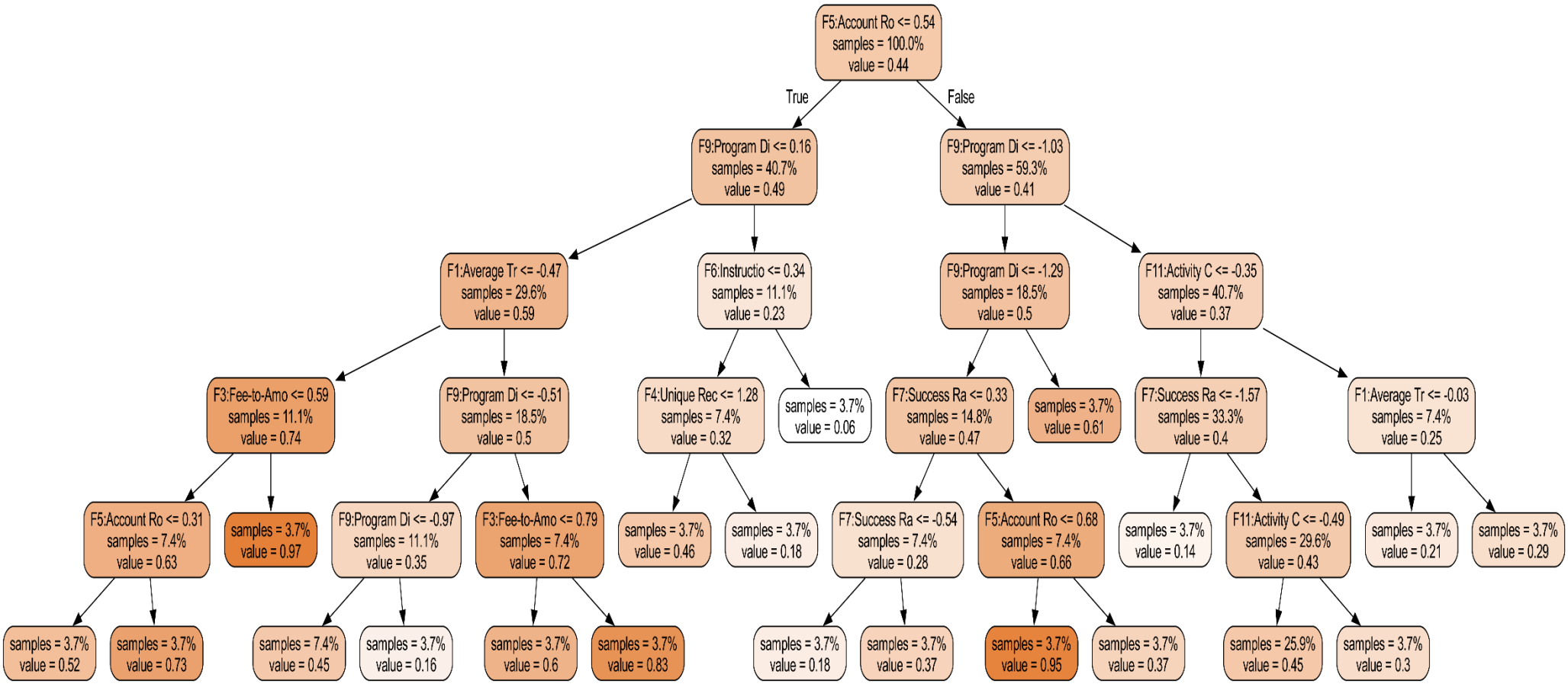
36. Li, K., Guan, S., & Lee, D. (2023). Towards understanding and characterizing the arbitrage bot scam in the wild. *Proceedings of the ACM on Measurement and Analysis of Computing Systems, 7*(3), Article 52, 1–29.

37. Davis, J., & Goadrich, M. (2006, June). The relationship between Precision-Recall and ROC curves. In Proceedings of the 23rd international conference on Machine learning (pp. 233-240).

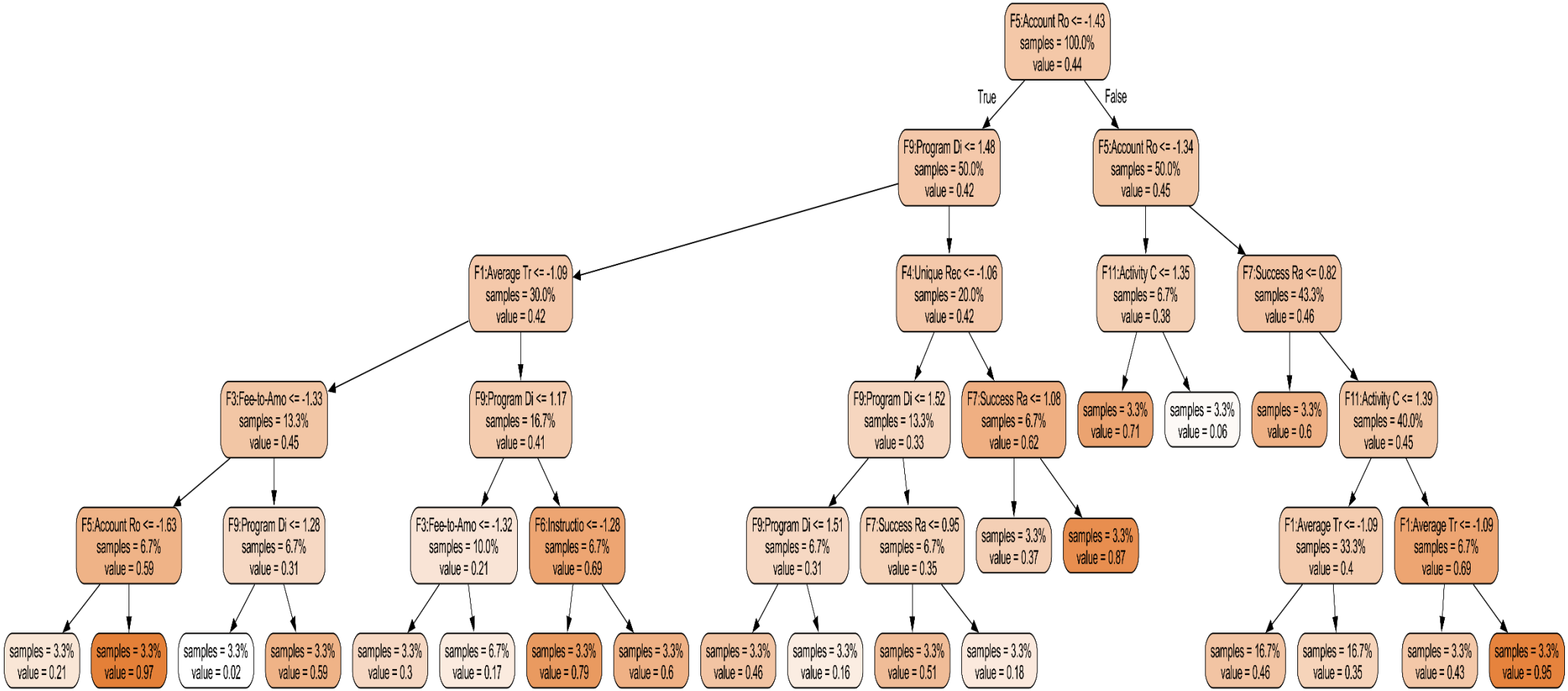
PRIEDAI

1 PRIEDAS

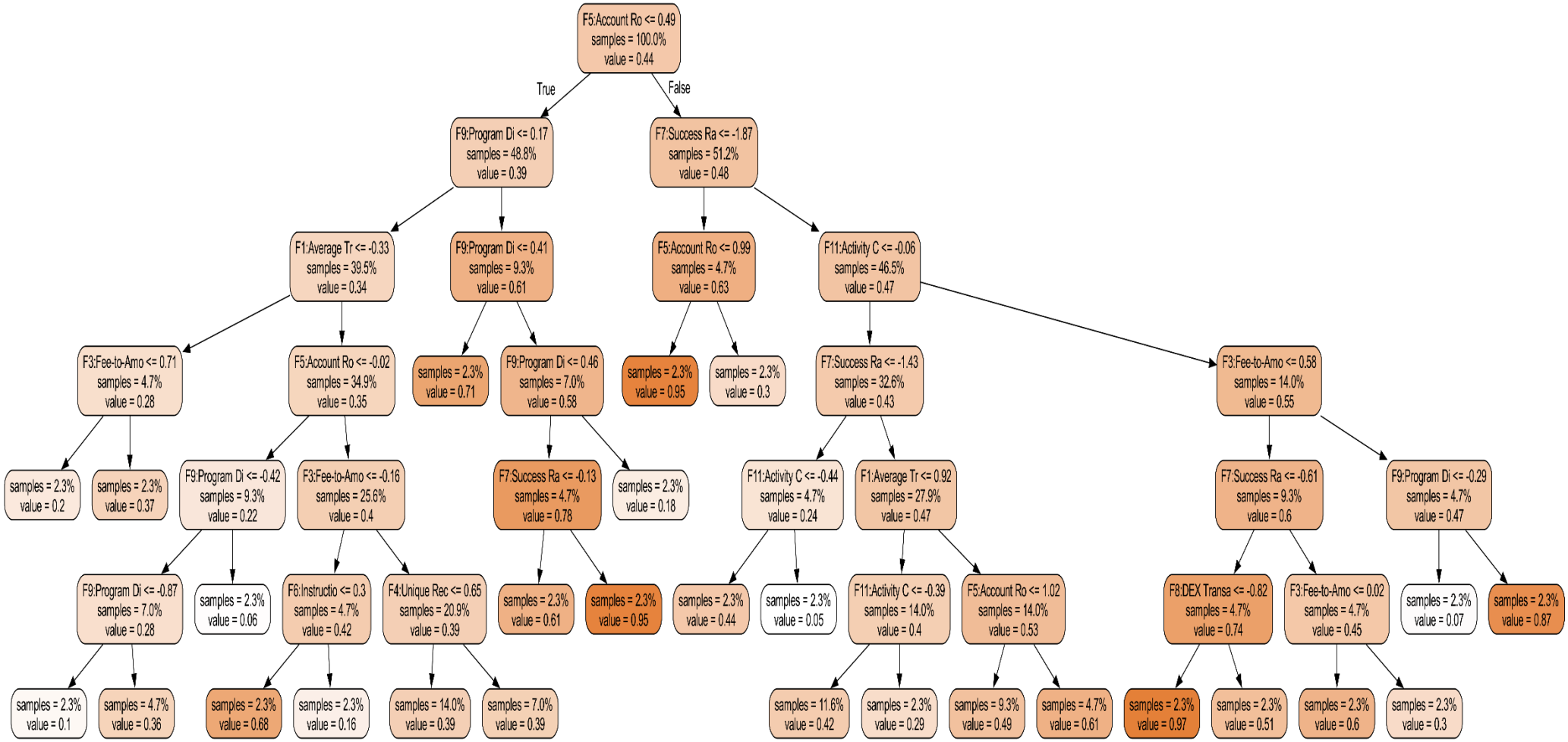
K MEANS KASTERIO 0 ANOMALIJŲ IZOLIACIJOS MIŠKO MEDIS



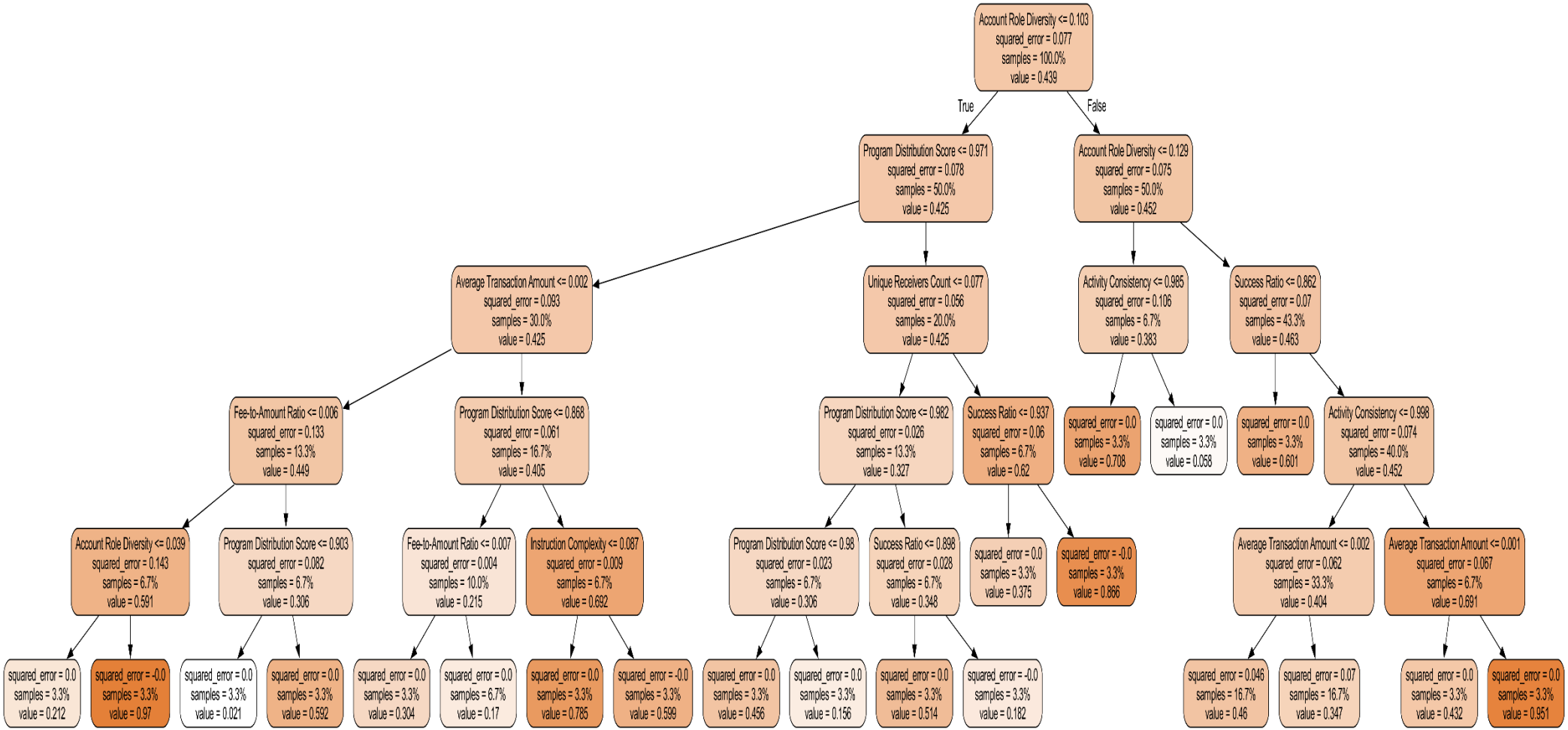
K MEANS KASTERIO 1 ANOMALIJŲ IZOLIACIJOS MIŠKO MEDIS



K MEANS KASTERIO 2 ANOMALIJŲ IZOLIACIJOS MIŠKO MEDIS

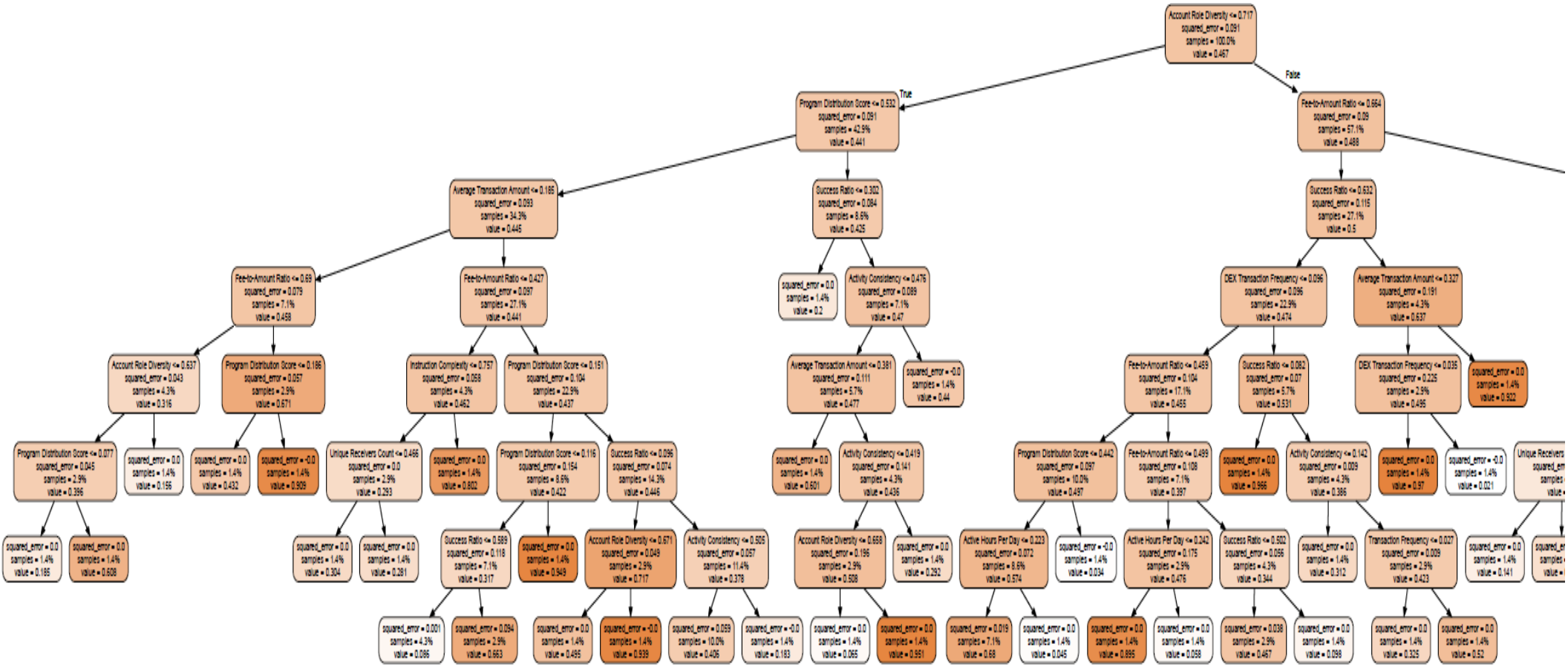


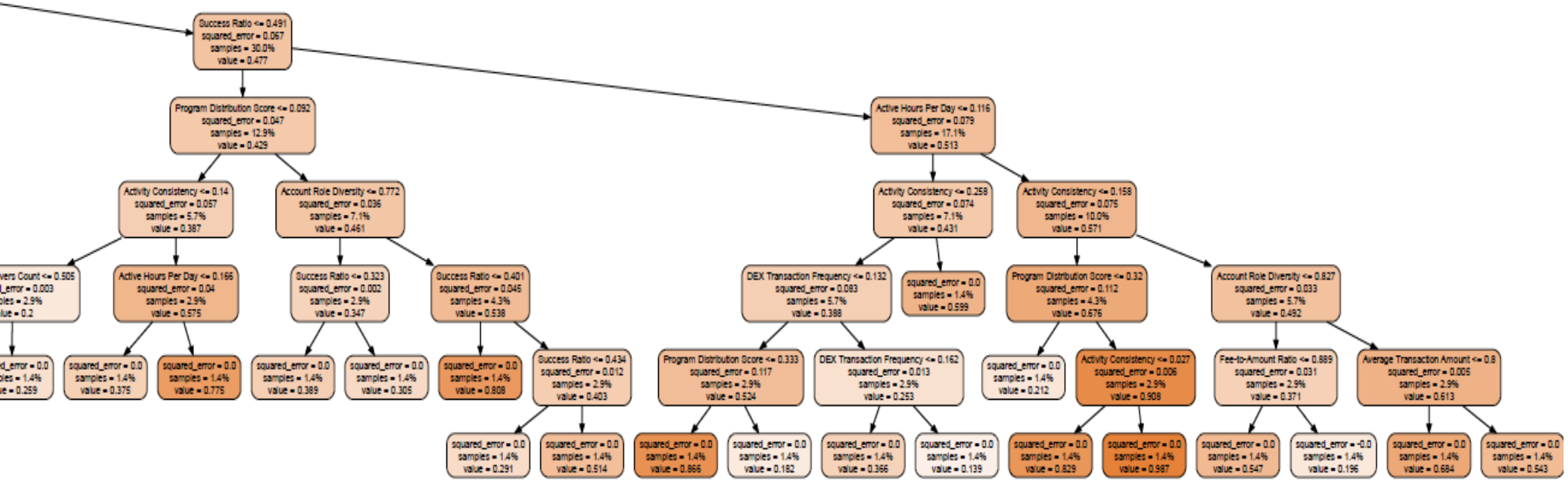
DBSCAN KASTERIO 0 ANOMALIJŲ IZOLIACIJOS MIŠKO MED



5 PRIEDAS

DBSCAN KASTERIO 1 ANOMALIJŲ IZOLIACIJOS MIŠKO MEDIS




Success Ratio <= 0.491
squared_error = 0.067
samples = 30.0%
value = 0.477
Program Distribution Score <= 0.092
squared_error = 0.047
samples = 12.9%
value = 0.429
Active Hours Per Day <= 0.116
squared_error = 0.079
samples = 17.1%
value = 0.513
Activity Consistency <= 0.14
squared_error = 0.057
samples = 5.7%
value = 0.387
Account Role Diversity <= 0.772
squared_error = 0.036
samples = 7.1%
value = 0.461
Activity Consistency <= 0.258
squared_error = 0.074
samples = 7.1%
value = 0.431
Activity Consistency <= 0.158
squared_error = 0.075
samples = 10.0%
value = 0.571
Active Hours Per Day <= 0.166
squared_error = 0.04
samples = 2.9%
value = 0.575
Success Ratio <= 0.323
squared_error = 0.002
samples = 2.9%
value = 0.347
Success Ratio <= 0.401
squared_error = 0.045
samples = 4.3%
value = 0.538
DEX Transaction Frequency <= 0.132
squared_error = 0.083
samples = 5.7%
value = 0.388
squared_error = 0.0
samples = 1.4%
value = 0.599
Program Distribution Score <= 0.32
squared_error = 0.112
samples = 4.3%
value = 0.676
Account Role Diversity <= 0.827
squared_error = 0.033
samples = 5.7%
value = 0.492
Success Ratio <= 0.434
squared_error = 0.012
samples = 2.9%
value = 0.403
Program Distribution Score <= 0.333
squared_error = 0.117
samples = 2.9%
value = 0.524
DEX Transaction Frequency <= 0.162
squared_error = 0.013
samples = 2.9%
value = 0.253
squared_error = 0.0
samples = 1.4%
value = 0.212
Activity Consistency <= 0.027
squared_error = 0.006
samples = 2.9%
value = 0.908
Fee-to-Amount Ratio <= 0.889
squared_error = 0.031
samples = 2.9%
value = 0.371
Average Transaction Amount <= 0.8
squared_error = 0.005
samples = 2.9%
value = 0.613

APRAŠAMOSIOS STATISTIKOS SKAIČIAVYMŲ PYTHON KODAS

Pirmas žingsnis

```python
import pandas as pd
import numpy as np

def calculate_central_tendency(df):
    metrics = {}

    # Analizuojamos funkcijos
    key_features = [
        'Transaction Frequency',
        'Average Transaction Amount',
        'Active Hours Per Day'
    ]

    for feature in key_features:
        metrics[feature] = {
            'Mean': np.mean(df[feature]),
            'Median': np.median(df[feature]),
            'Mode': df[feature].mode()[0]
        }

    return pd.DataFrame(metrics)

# Duomenų įkėlimas
df = pd.read_csv(r"C:\Users\Lukas\WalletData\Solana_Bots_Dataset.csv")

# Apskaičiuoti rodiklius
results = calculate_central_tendency(df)
print(results)
```

Antra

```python
def calculate_variability_measures(df):
    metrics = {}

    # Analizuojamos funkcijos
    key_features = [
        'Transaction Frequency',
        'Average Transaction Amount',
        'StdDev Transaction Amount',
        'Active Hours Per Day'
    ]

    for feature in key_features:
        metrics[feature] = {
            'Standard_Deviation': np.std(df[feature]),
            'Range': np.ptp(df[feature]),  # Peak to peak (max - min)
            'IQR': np.percentile(df[feature], 75) - np.percentile(df[feature], 25)
        }

    return pd.DataFrame(metrics)

# Duomenų įkėlimas
df = pd.read_csv(r"C:\Users\Lukas\Downloads\Solana_Bots_Dataset.csv")

# Apskaičiuoti rodiklius
results = calculate_variability_measures(df)
print("\nVariability Measures:")
print(results)
```

Penktas

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

def plot_distributions(df):
    # Pagrindinės analizuojamos funkcijos
    features = [
        'Transaction Frequency',
        'Average Transaction Amount',
        'Active Hours Per Day',
        'Activity Consistency',
        'Fee-to-Amount Ratio',
        'StdDev Transaction Amount'
    ]

    # Sukurti histogramų subplotus
    plt.figure(figsize=(15, 10))
    for i, feature in enumerate(features, 1):
        plt.subplot(2, 3, i)
        sns.histplot(data=df, x=feature, kde=True)
        plt.title(f'{feature} Distribution')
        plt.xticks(rotation=45)
    plt.tight_layout()
    plt.show()

    # Sukurti langelių skylypus
    plt.figure(figsize=(15, 6))
    plt.subplot(1, 2, 1)
    sns.boxplot(data=df[features])
    plt.xticks(rotation=45)
    plt.title('Distribution Patterns Box Plots')
    plt.tight_layout()
    plt.show()

# Įkelti duomenis
df = pd.read_csv(r"C:\Users\Lukas\WalletData\Solana_Bots_Dataset.csv")


# # Sukurti plots
plot_distributions(df)

# # Papildomas box plot konkretiems palyginimams
def plot_detailed_boxplots(df):
    fig, axes = plt.subplots(2, 2, figsize=(15, 10))

    # Transaction patterns
    sns.boxplot(data=df, y='Transaction Frequency', ax=axes[0,0])
    axes[0,0].set_title('Transaction Frequency Distribution')

    sns.boxplot(data=df, y='Average Transaction Amount', ax=axes[0,1])
    axes[0,1].set_title('Transaction Amount Distribution')

    # Activity patterns
    sns.boxplot(data=df, y='Active Hours Per Day', ax=axes[1,0])
    axes[1,0].set_title('Active Hours Distribution')

    sns.boxplot(data=df, y='Activity Consistency', ax=axes[1,1])
    axes[1,1].set_title('Activity Consistency Distribution')

    plt.tight_layout()
    plt.show()

plot_detailed_boxplots(df)
```

Trečias

```python
import pandas as pd
import numpy as np
from scipy.stats import skew, kurtosis

def calculate_distribution_shape(df):
    metrics = {}

    # Analizuojamos funkcijos
    key_features = [
        'Transaction Frequency',
        'Average Transaction Amount',
        'Active Hours Per Day',
        'Activity Consistency'
    ]

    for feature in key_features:
        metrics[feature] = {
            'Skewness': skew(df[feature]),
            'Kurtosis': kurtosis(df[feature])
        }

    return pd.DataFrame(metrics)

# Duomenų įkėlimas
df = pd.read_csv(r"C:\Users\Lukas\WalletData\Solana_Bots_Dataset.csv")

# Apskaičiuoti rodiklius
results = calculate_distribution_shape(df)

# Pavaizduoti rezultatus
print("\nDistribution Shape Measures:")
for feature in results.columns:
    print(f"\n{feature}:")
    print(f"Skewness: {results[feature]['Skewness']:.2f}")
    print(f"Kurtosis: {results[feature]['Kurtosis']:.2f}")
```

Ketvirtas

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

def calculate_correlations(df):
    # Pasirinkite atitinkamas funkcijas
    features = [
        'Transaction Frequency',
        'Average Transaction Amount',
        'StdDev Transaction Amount',
        'Fee-to-Amount Ratio',
        'Active Hours Per Day',
        'Activity Consistency'
    ]

    # Apskaičiuokite koreliacijos matricą
    corr_matrix = df[features].corr()

    # Sukurti šilumos žemėlapį
    plt.figure(figsize=(10, 8))
    sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
    plt.title('Feature Correlations')
    plt.show()

    return corr_matrix

# Duomenų įkėlimas
df = pd.read_csv(r"C:\Users\Lukas\WalletData\Solana_Bots_Dataset.csv")

# Apskaičiuoti ir atvaizduoti
correlations = calculate_correlations(df)

# {ateikti tam tikras koreliacijas}
important_pairs = [
    ('Transaction Frequency', 'Average Transaction Amount'),
    ('Transaction Frequency', 'Active Hours Per Day'),
    ('Active Hours Per Day', 'Activity Consistency'),
    ('Transaction Frequency', 'Fee-to-Amount Ratio')
]

print("\nKey Correlation Pairs:")
for pair in important_pairs:
    correlation = correlations.loc[pair[0], pair[1]]
    print(f"{pair[0]} vs {pair[1]}: {correlation:.3f}")
```

ISOLATION FOREST METODAS

```python
import pandas as pd
import numpy as np
from sklearn.ensemble import IsolationForest
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.manifold import TSNE
from sklearn.neighbors import LocalOutlierFactor
from sklearn.svm import OneClassSVM
from sklearn.cluster import KMeans
from sklearn.covariance import EllipticEnvelope
from sklearn.metrics import precision_recall_curve, roc_curve, auc, precision_score, recall_score
from scipy import stats

# Pirminė funkcijos svarbos funkcija
def analyze_feature_importance(model, features, data):
    importances = []
    base_score = model.decision_function(data)
    for i in range(len(features)):
        temp_data = data.copy()
        temp_data[:, i] = np.random.permutation(temp_data[:, i])
        new_scores = model.decision_function(temp_data)
        importance = np.mean(np.abs(base_score - new_scores))
        importances.append(importance)
    return pd.Series(importances, index=features)

# Naujos analizės funkcijos
def analyze_temporal_patterns(df):
    hourly_patterns = df.groupby('Active Hours Per Day')['Anomaly'].mean()
    consistency_patterns = df.groupby('Activity Consistency')['Anomaly'].mean()
    return hourly_patterns, consistency_patterns

def analyze_feature_correlations(df, features):
    correlation_matrix = df[features + ['Anomaly']].corr()
    plt.figure(figsize=(12, 8))
    sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm')
    plt.title('Feature Correlations with Anomaly Scores')
    plt.show()

def compare_ensemble_methods(X, contamination=0.1):
    methods = {
        'IsolationForest': IsolationForest(contamination=contamination),
        'LOF': LocalOutlierFactor(contamination=contamination, novelty=True),
        'OCSVM': OneClassSVM(nu=contamination),
        'EE': EllipticEnvelope(contamination=contamination)
    }
    results = {}
    for name, method in methods.items():
        method.fit(X)
        results[name] = method.predict(X)
    return results

def analyze_clusters(X, y, n_clusters=3):
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    clusters = kmeans.fit_predict(X)
    cluster_anomaly_rates = pd.Dataframe({
        'Cluster': range(n_clusters),
        'Anomaly_Rate': [np.mean(y[clusters == i]) for i in range(n_clusters)]
    })
    return cluster_anomaly_rates

def calculate_performance_metrics(y_true, y_scores):
    precision, recall, thresholds = precision_recall_curve(y_true, y_scores)
    return {
        'precision': precision,
        'recall': recall,
        'thresholds': thresholds
    }

def profile_anomalies(df, features):
    anomaly_profiles = df[df['Anomaly'] == 1][features].describe()
    normal_profiles = df[df['Anomaly'] == 0][features].describe()
    return pd.Dataframe({
        'Anomaly_Profile': anomaly_profiles.mean(),
        'Normal_Profile': normal_profiles.mean(),
        'Difference': anomaly_profiles.mean() - normal_profiles.mean()
    })

def statistical_analysis(df, features):
    results = {}
    for feature in features:
        normal = df[df['Anomaly'] == 0][feature]
        anomaly = df[df['Anomaly'] == 1][feature]
        stat, pvalue = stats.mannwhitneyu(normal, anomaly)
        results[feature] = {'statistic': stat, 'p-value': pvalue}
    return pd.Dataframe(results)

def plot_roc_curve(y_true, y_score):
    fpr, tpr, _ = roc_curve(y_true, y_score)
    roc_auc = auc(fpr, tpr)

    plt.figure()
    plt.plot(fpr, tpr, color='darkorange', lw=2,
             label=f'ROC curve (AUC = {roc_auc:.2f})')
    plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver Operating Characteristic (ROC) Curve')
    plt.legend(loc="lower right")
    plt.show()

def plot_feature_distributions(df, features):
    fig, axes = plt.subplots(3, 4, figsize=(20, 15))
    for feature, ax in zip(features, axes.ravel()):
        sns.kdeplot(data=df, x=feature, hue='Anomaly', ax=ax)
        ax.set_title(feature)
    plt.tight_layout()
    plt.show()

def plot_threshold_analysis(scores, labels):
    thresholds = np.linspace(scores.min(), scores.max(), 100)
    metrics = []
    for threshold in thresholds:
        pred = (scores < threshold).astype(int)
        metrics.append([
            precision_score(labels, pred),
            recall_score(labels, pred)
        ])
    metrics = np.array(metrics)
    plt.figure(figsize=(10, 6))
    plt.plot(thresholds, metrics[:, 0], label='Precision')
    plt.plot(thresholds, metrics[:, 1], label='Recall')
    plt.xlabel('Threshold')
    plt.ylabel('Score')
    plt.title('Threshold Analysis')
    plt.legend()
    plt.show()

def analyze_feature_importance(model, features, data):
    importances = []
    base_score = model.decision_function(data)
    for i in range(len(features)):
        temp_data = data.copy()
        temp_data[:, i] = np.random.permutation(temp_data[:, i])
        new_scores = model.decision_function(temp_data)
        importance = np.mean(np.abs(base_score - new_scores))
        importances.append(importance)
    return pd.Series(importances, index=features)

# Įkelkite ir paruoškite duomenis
df = pd.read_csv(r"C:\Users\Lukas\WalletData\Solana_Bots_Dataset.csv")

features = [
    "Transaction Frequency",
    "Average Transaction Amount",
    "StdDev Transaction Amount",
    "Fee-to-Amount Ratio",
    "Unique Receivers Count",
    "Account Role Diversity",
    "Instruction Complexity",
    "Success Ratio",
    "DEX Transaction Frequency",
    "Program Distribution Score",
    "Active Hours Per Day",
    "Activity Consistency"
]

# Standartizuoti funkcijas
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df[features])

#  Isolation Forest apmokymas
isolation_forest = IsolationForest(
    n_estimators=100,
    contamination=0.1,
    random_state=42
)

# Prognozės ir galutinė prognozės
df['Anomaly'] = isolation_forest.fit_predict(scaled_data)
df['Anomaly'] = df['Anomaly'].map({1: 0, -1: 1})
df['Anomaly Score'] = isolation_forest.decision_function(scaled_data)

# Pagrindinė statistika
print("\nBasic Statistics:")
print("-" * 50)
print(f"Total Records: {len(df)}")
print(f"Anomalies Detected: {df['Anomaly'].sum()}")
print(f"Normal Points: {len(df) - df['Anomaly'].sum()}")

# Funkcijų svarbos analizė
importances = analyze_feature_importance(isolation_forest, features, scaled_data)
print("\nFeature Importance Analysis:")
print("-" * 50)
print(importances.sort_values(ascending=False))

# Visualizations
plt.style.use('default')  # Using default style instead of seaborn

# 1. Anomalijų balų pasiskirstymas
plt.figure(figsize=(10, 6))
plt.hist(df[df['Anomaly'] == 0]['Anomaly Score'], bins=50, alpha=0.5, label='Normal', color='blue')
plt.hist(df[df['Anomaly'] == 1]['Anomaly Score'], bins=50, alpha=0.5, label='Anomaly', color='red')
plt.title("Distribution of Anomaly Scores")
plt.xlabel("Anomaly Score")
plt.ylabel("Frequency")
plt.legend()
plt.show()

# 2. Funkcijos svarbos grafikas
plt.figure(figsize=(12, 6))
importances.sort_values().plot(kind='barh')
plt.title("Feature Importance in Anomaly Detection")
plt.tight_layout()
plt.show()

# 3. PCA vizualizacija
pca = PCA(n_components=2)
reduced_data = pca.fit_transform(scaled_data)
plt.figure(figsize=(10, 6))
scatter = plt.scatter(reduced_data[:, 0], reduced_data[:, 1],
                      c=df['Anomaly'], cmap="coolwarm", s=50)
plt.title("PCA Visualization of Anomalies")
plt.colorbar(scatter)
plt.show()

# 4. t-SNE vizualizacija
tsne = TSNE(n_components=2, random_state=42)
tsne_results = tsne.fit_transform(scaled_data)
plt.figure(figsize=(10, 6))
plt.scatter(tsne_results[:, 0], tsne_results[:, 1],
            c=df['Anomaly'], cmap="coolwarm", s=50)
plt.title("t-SNE Visualization of Anomalies")
plt.colorbar()
plt.show()

# Palyginti su kitais metodais
print("\nComparison with Other Methods:")
print("-" * 50)

# LOF
lof = LocalOutlierFactor(contamination=0.1, novelty=True)
lof.fit(scaled_data)
lof_labels = lof.predict(scaled_data)
lof_scores = -lof.score_samples(scaled_data)

# One-Class SVM
ocsvm = OneClassSVM(nu=0.1)
ocsvm.fit(scaled_data)
ocsvm_labels = ocsvm.predict(scaled_data)
ocsvm_scores = ocsvm.decision_function(scaled_data)

# Išsaugoti išsamius rezultatus
results_df = pd.Dataframe({
    'Isolationforest_Score': df['Anomaly Score'],
    'Isolationforest_Label': df['Anomaly'],
    'LOF_Score': lof_scores,
    'LOF_Label': lof_labels,
    'OCSVM_Score': ocsvm_scores,
    'OCSVM_Label': ocsvm_labels
})

results_df.to_csv('detailed_anomaly_detection_results.csv', index=False)

# Išsaugoti išsamią analizės ataskaitą
with open('anomaly_detection_report.txt', 'w') as f:
    f.write("Anomaly Detection Analysis Report\n")
    f.write("=" * 50 + "\n\n")
    f.write(f"Total Records: {len(df)}\n")
    f.write(f"Anomalies Detected: {df['Anomaly'].sum()}\n")
    f.write(f"Normal Points: {len(df) - df['Anomaly'].sum()}\n\n")
    f.write("Feature Importance:\n")
    f.write(str(importances.sort_values(ascending=False)))

print("\nAnalysis complete. Results saved to files.")

print("\nPerforming Additional Analyses...")
print("-" * 50)

# Laiko modelioi
hourly_patterns, consistency_patterns = analyze_temporal_patterns(df)
print("\nTemporal Patterns Analysis Complete")

# Funkcijų koreliacijos
analyze_feature_correlations(df, features)
print("\nFeature Correlation Analysis Complete")

# Ansamblio palyginimas
ensemble_results = compare_ensemble_methods(scaled_data)
print("\nEnsemble Method Comparison Complete")

# Klasterinė analizė
cluster_analysis = analyze_clusters(scaled_data, df['Anomaly'])
print("\nCluster Analysis Complete")
print(cluster_analysis)

# Veiklos rodikliai
performance_metrics = calculate_performance_metrics(df['Anomaly'], df['Anomaly Score'])
print("\nPerformance Metrics Calculated")

# Anomalijų profiliavimas
anomaly_profiles = profile_anomalies(df, features)
print("\nAnomaly Profiling Complete")
print(anomaly_profiles)

# Statistinė analizė
stat_analysis = statistical_analysis(df, features)
print("\nStatistical Analysis Complete")
print(stat_analysis)

# ROC kreivė
plot_roc_curve(df['Anomaly'], -df['Anomaly Score'])

# Funkcijų pasiskirstymai
plot_feature_distributions(df, features)

# Slenksčio analizė
plot_threshold_analysis(-df['Anomaly Score'], df['Anomaly'])

# Atnaujinti ataskaitą su naujomis analizėmis
with open('enhanced_anomaly_detection_report.txt', 'w') as f:
    f.write("Enhanced Anomaly Detection Analysis Report\n")
    f.write("=" * 50 + "\n\n")
    f.write(f"Total Records: {len(df)}\n")
    f.write(f"Anomalies Detected: {df['Anomaly'].sum()}\n")
    f.write(f"Normal Points: {len(df) - df['Anomaly'].sum()}\n\n")
    f.write("Feature Importance:\n")
    f.write(str(importances.sort_values(ascending=False)) + "\n\n")
    f.write("Cluster Analysis:\n")
    f.write(str(cluster_analysis) + "\n\n")
    f.write("Anomaly Profiles:\n")
    f.write(str(anomaly_profiles) + "\n\n")
    f.write("Statistical Analysis:\n")
    f.write(str(stat_analysis))
print("\nEnhanced analysis complete. Results saved to files.")
```

DBSCAN METODAS

```python
import pandas as pd
import numpy as np
from sklearn.preprocessing import RobustScaler
from sklearn.cluster import DBSCAN
from sklearn.metrics import silhouette_score, davies_bouldin_score, calinski_harabasz_score
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
import seaborn as sns
from matplotlib.gridspec import GridSpec

# Load the data
df = pd.read_csv(r"C:\Users\Lukas\WalletData\Solana_Bots_Dataset.csv")

# Select features for clustering
features = [
    'Transaction Frequency',
    'Average Transaction Amount',
    'StdDev Transaction Amount',
    'Fee-to-Amount Ratio',
    'Unique Receivers Count',
    'Account Role Diversity',
    'Instruction Complexity',
    'Success Ratio',
    'DEX Transaction Frequency',
    'Program Distribution Score',
    'Active Hours Per Day',
    'Activity Consistency'
]

# Function to remove outliers
def remove_outliers(df, columns, n_std=3):
    df_clean = df.copy()
    for column in columns:
        mean = df_clean[column].mean()
        std = df_clean[column].std()
        df_clean = df_clean[abs(df_clean[column] - mean) <= (n_std * std)]
    return df_clean

# Remove outliers and scale features
df_clean = remove_outliers(df, features)
scaler = RobustScaler()
X_scaled = scaler.fit_transform(df_clean[features])

# Perform PCA for visualization
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

# Function to optimize DBSCAN parameters
def optimize_dbscan(X, eps_range=np.arange(0.1, 2.0, 0.05), min_samples_range=range(5, 15)):
    best_score = -1
    best_params = None
    best_labels = None
    results = []

    for eps in eps_range:
        for min_samples in min_samples_range:
            dbscan = DBSCAN(eps=eps, min_samples=min_samples)
            labels = dbscan.fit_predict(X)

            n_clusters = len(np.unique(labels[labels != -1]))
            if n_clusters >= 2:
                try:
                    sil_score = silhouette_score(X, labels)
                    ch_score = calinski_harabasz_score(X, labels)
                    db_score = davies_bouldin_score(X, labels)

                    if sil_score > best_score:
                        best_score = sil_score
                        best_params = (eps, min_samples)
                        best_labels = labels

                    results.append({
                        'eps': eps,
                        'min_samples': min_samples,
                        'silhouette': sil_score,
                        'ch_score': ch_score,
                        'db_score': db_score,
                        'n_clusters': n_clusters,
                        'noise_points': sum(labels == -1)
                    })
                except:
                    continue

    return best_params, best_score, best_labels, pd.DataFrame(results)

# Function to plot clustering results
def plot_clustering_results(X_scaled, labels, pca, X_pca, features, results_df, best_params, best_score):
    fig = plt.figure(figsize=(20, 15))
    gs = GridSpec(3, 2, figure=fig)

    # 1. PCA visualization
    ax1 = fig.add_subplot(gs[0, 0])
    scatter = ax1.scatter(X_pca[:, 0], X_pca[:, 1], c=labels, cmap='viridis', alpha=0.7)
    ax1.set_title('DBSCAN Clustering (PCA Visualization)', pad=20)
    ax1.set_xlabel('First Principal Component')
    ax1.set_ylabel('Second Principal Component')
    plt.colorbar(scatter, ax=ax1, label='Cluster Label')

    # 2. Feature Importance
    ax2 = fig.add_subplot(gs[0, 1])
    importances = np.abs(pca.components_[0]) + np.abs(pca.components_[1])
    feat_importance = pd.DataFrame({'Feature': features, 'Importance': importances})
    feat_importance = feat_importance.sort_values('Importance', ascending=True)

    colors = plt.cm.viridis(np.linspace(0, 1, len(features)))
    ax2.barh(range(len(feat_importance)), feat_importance['Importance'], color=colors)
    ax2.set_yticks(range(len(feat_importance)))
    ax2.set_yticklabels(feat_importance['Feature'])
    ax2.set_title('Feature Importance in Clustering', pad=20)

    # 3. Cluster sizes
    ax3 = fig.add_subplot(gs[1, 0])
    cluster_sizes = pd.Series(labels).value_counts().sort_index()
    colors = plt.cm.viridis(np.linspace(0, 1, len(cluster_sizes)))

    bars = ax3.bar(range(len(cluster_sizes)), cluster_sizes.values, color=colors)
    ax3.set_xticks(range(len(cluster_sizes)))
    ax3.set_xticklabels([f'Cluster {i}' for i in cluster_sizes.index])
    ax3.set_title('Cluster Size Distribution', pad=20)
    ax3.set_ylabel('Number of Samples')

    # Add percentage labels
    for bar in bars:
        height = bar.get_height()
        ax3.text(bar.get_x() + bar.get_width()/2., height,
                 f'{height/len(labels)*100:.1f}%',
                 ha='center', va='bottom')

    # 4. Parameter optimization results
    ax4 = fig.add_subplot(gs[1, 1])
    scatter = ax4.scatter(results_df['eps'], results_df['silhouette'],
                          c=results_df['n_clusters'], s=100*results_df['noise_points'],
                          alpha=0.6, cmap='viridis')
    ax4.set_xlabel('Epsilon (ε)')
    ax4.set_ylabel('Silhouette Score')
    ax4.set_title('Parameter Optimization Results', pad=20)

    # Add best parameter marker
    ax4.scatter(best_params[0], best_score, color='red', s=200, marker='*',
                label='Best Parameters')
    ax4.legend()

    # 5. Validation metrics
    ax5 = fig.add_subplot(gs[2, :])
    metrics_df = results_df.groupby('eps')[['silhouette', 'ch_score', 'db_score']].mean()
    metrics_df.plot(ax=ax5)
    ax5.set_xlabel('Epsilon (ε)')
    ax5.set_ylabel('Score')
    ax5.set_title('Validation Metrics vs Epsilon', pad=20)
    ax5.legend(['Silhouette Score', 'Calinski-Harabasz Score', 'Davies-Bouldin Score'])

    plt.tight_layout()
    return fig

# Optimize DBSCAN parameters and perform clustering
best_params, best_score, best_labels, results_df = optimize_dbscan(X_scaled)
final_dbscan = DBSCAN(eps=best_params[0], min_samples=best_params[1])
labels = final_dbscan.fit_predict(X_scaled)

# Print basic clustering results
print("\nOptimization Results:")
print(f"Best parameters: eps={best_params[0]}, min_samples={best_params[1]}")
print(f"Best silhouette score: {best_score:.4f}")

print("\nClustering Metrics:")
print(f"Number of clusters: {len(np.unique(labels)) - (1 if -1 in labels else 0)}")
print(f"Number of noise points: {sum(labels == -1)}")
if len(np.unique(labels)) > 1:
    print(f"Silhouette Score: {silhouette_score(X_scaled, labels):.4f}")
    print(f"Davies-Bouldin Index: {davies_bouldin_score(X_scaled, labels):.4f}")
    print(f"Calinski-Harabasz Index: {calinski_harabasz_score(X_scaled, labels):.4f}")

# Generate and save visualizations
fig = plot_clustering_results(X_scaled, labels, pca, X_pca, features, results_df, best_params, best_score)
plt.savefig('dbscan_clustering_results.png', dpi=300, bbox_inches='tight')
plt.show()

# Detailed cluster analysis
print("\nDetailed Cluster Analysis:")
for cluster in sorted(set(labels)):
    # Get indices for current cluster
    cluster_indices = np.where(labels == cluster)[0]
    cluster_data = X_scaled[cluster_indices]

    print(f"\n{'Noise Points' if cluster == -1 else f'Cluster {cluster}'}")
    print("-" * 50)
    print(f"Sample Count: {len(cluster_indices)}")
    print(f"Proportion: {len(cluster_indices)/len(labels)*100:.2f}%")

    # Create DataFrame for readable statistics
    cluster_df = pd.DataFrame(cluster_data, columns=features)
    print("\nFeature Statistics:")
    print(cluster_df.describe().round(3))
    print("\nFeature Medians:")
    print(cluster_df.median().round(3))

# Save the results
df_clean['Cluster'] = labels
df_clean.to_csv('dbscan_clustering_results.csv', index=False)
```

K MEANS METODAS

```python
import pandas as pd
import numpy as np
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score, calinski_harabasz_score
from sklearn.model_selection import KFold
import matplotlib.pyplot as plt

# Load the dataset
file_path = r"C:\Users\Lukas\WalletData\Solana_Bots_Dataset.csv"
df = pd.read_csv(file_path)

# Features to include in clustering
features = [
    "Transaction Frequency",
    "Average Transaction Amount",
    "StdDev Transaction Amount",
    "Fee-to-Amount Ratio",
    "Unique Receivers Count",
    "Account Role Diversity",
    "Instruction Complexity",
    "Success Ratio",
    "DEX Transaction Frequency",
    "Program Distribution Score",
    "Active Hours Per Day",
    "Activity Consistency"
]

# Standardize the features
scaler = StandardScaler()
scaled_data = scaler.fit_transform(df[features])

# Apply PCA
pca = PCA(n_components=0.8)
reduced_data = pca.fit_transform(scaled_data)
print(f"PCA reduced dimensions to: {reduced_data.shape[1]}")

# Cross-validation function
def cross_validate_kmeans(data, n_clusters, n_splits=5):
    """
    Perform k-fold cross-validation for K-means clustering
    """
    kf = KFold(n_splits=n_splits, shuffle=True, random_state=42)
    silhouette_scores = []
    calinski_scores = []
    inertia_scores = []

    for train_idx, val_idx in kf.split(data):
        # Split data into training and validation sets
        train_data = data[train_idx]
        val_data = data[val_idx]

        # Fit K-means on training data
        kmeans = KMeans(n_clusters=n_clusters, random_state=42, n_init=20)
        kmeans.fit(train_data)

        # Predict on validation data
        val_labels = kmeans.predict(val_data)

        # Calculate validation metrics
        if len(np.unique(val_labels)) > 1:  # Check if more than one cluster
            silhouette_scores.append(silhouette_score(val_data, val_labels))
            calinski_scores.append(calinski_harabasz_score(val_data, val_labels))
            inertia_scores.append(kmeans.inertia_)

    return {
        'mean_silhouette': np.mean(silhouette_scores),
        'std_silhouette': np.std(silhouette_scores),
        'mean_calinski': np.mean(calinski_scores),
        'std_calinski': np.std(calinski_scores),
        'mean_inertia': np.mean(inertia_scores),
        'std_inertia': np.std(inertia_scores)
    }

# Evaluate different K values with cross-validation
k_range = range(2, 11)
cv_results = {}

for k in k_range:
    print(f"\nEvaluating K={k}")
    cv_scores = cross_validate_kmeans(reduced_data, k)
    cv_results[k] = cv_scores
    print(f"Mean Silhouette Score: {cv_scores['mean_silhouette']:.4f} ± {cv_scores['std_silhouette']:.4f}")
    print(f"Mean Calinski-Harabasz Score: {cv_scores['mean_calinski']:.4f} ± {cv_scores['std_calinski']:.4f}")

# Plot cross-validation results
plt.figure(figsize=(15, 5))

# Plot Silhouette Scores with error bars
plt.subplot(1, 3, 1)
means = [cv_results[k]['mean_silhouette'] for k in k_range]
stds = [cv_results[k]['std_silhouette'] for k in k_range]
plt.errorbar(list(k_range), means, yerr=stds, marker='o')
plt.title('Cross-validated Silhouette Score')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('Mean Silhouette Score')

# Plot Calinski-Harabasz Scores with error bars
plt.subplot(1, 3, 2)
means = [cv_results[k]['mean_calinski'] for k in k_range]
stds = [cv_results[k]['std_calinski'] for k in k_range]
plt.errorbar(list(k_range), means, yerr=stds, marker='o')
plt.title('Cross-validated Calinski-Harabasz Score')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('Mean Calinski-Harabasz Score')

# Plot Inertia with error bars
plt.subplot(1, 3, 3)
means = [cv_results[k]['mean_inertia'] for k in k_range]
stds = [cv_results[k]['std_inertia'] for k in k_range]
plt.errorbar(list(k_range), means, yerr=stds, marker='o')
plt.title('Cross-validated Inertia')
plt.xlabel('Number of Clusters (K)')
plt.ylabel('Mean Inertia')

plt.tight_layout()
plt.show()

# Train final K-Means model with the chosen optimal number of clusters
optimal_k = 3  # Adjust based on your cross-validation or analysis
kmeans = KMeans(n_clusters=optimal_k, random_state=42, n_init=20)
df["Cluster"] = kmeans.fit_predict(reduced_data)

# Save the clustered dataset with the new "Cluster" column for further analysis
output_file = "refined_clustered_wallets.csv"
df.to_csv(output_file, index=False)
print(f"Clustered dataset with cluster labels saved to {output_file}")

# Optional: Print the first few rows to verify
print(df.head())

# Visualize final clustering
plt.figure(figsize=(8, 6))
plt.scatter(reduced_data[:, 0], reduced_data[:, 1], c=df["Cluster"], cmap="viridis", s=10)
plt.title("Final K-Means Clustering")
plt.xlabel("First Principal Component")
plt.ylabel("Second Principal Component")
plt.colorbar(label="Cluster")
plt.show()
```

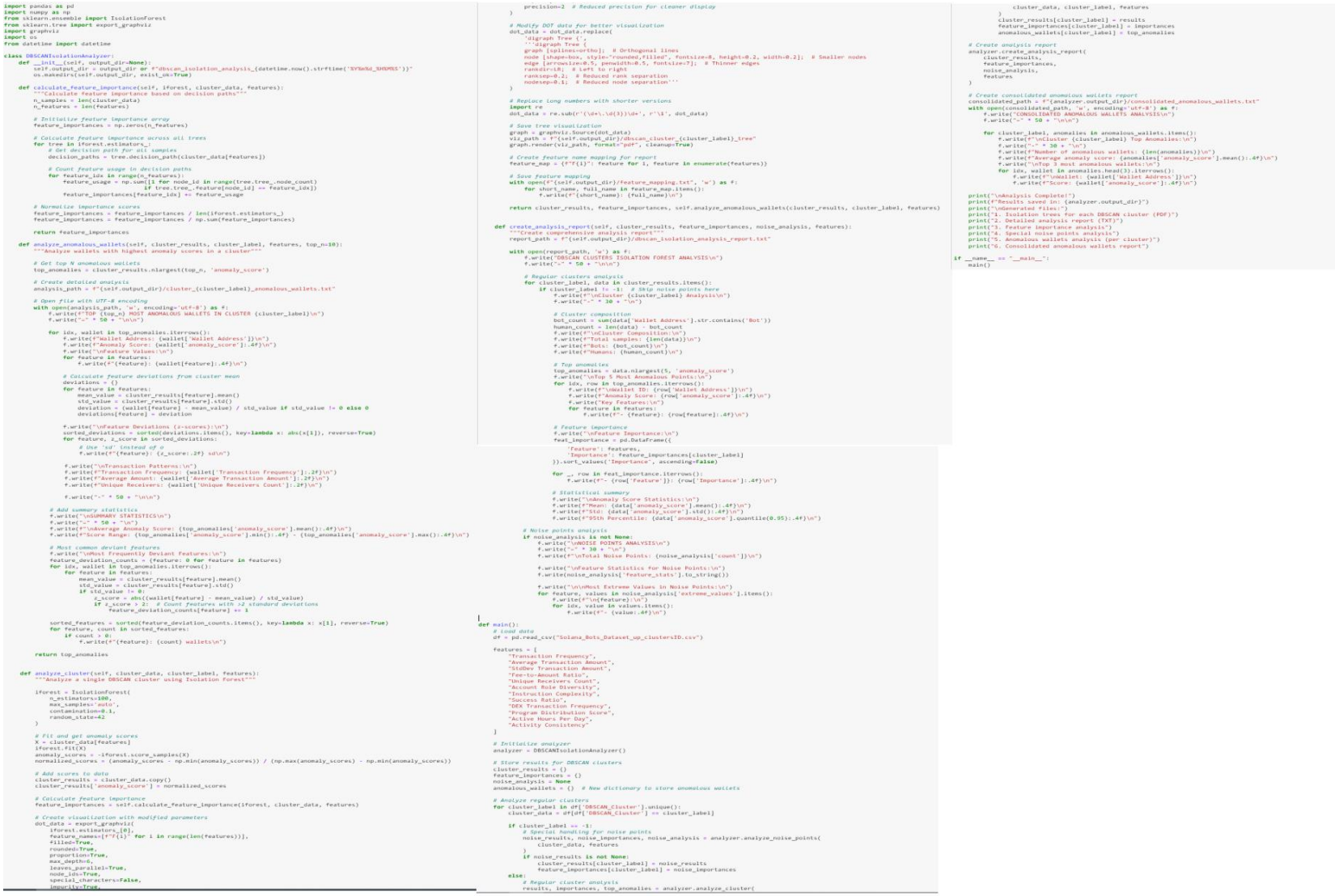
```python
# Analyze cluster centroids
print("\nCluster Centroids Analysis:")
print("==========================")
# Transform cluster centers back to original feature space
cluster_centers = pd.DataFrame(
    scaler.inverse_transform(pca.inverse_transform(kmeans.cluster_centers_)),
    columns=features
)

# Print detailed centroid analysis
for i in range(optimal_k):
    print(f"\nCluster {i} Characteristics:")
    print("-------------------------")
    for feature in features:
        print(f"{feature}: {cluster_centers.iloc[i][feature]:.2f}")

    # Calculate feature importance for this cluster
    cluster_data = df[df['Cluster'] == i]
    feature_means = cluster_data[features].mean()
    overall_means = df[features].mean()
    relative_importance = (feature_means - overall_means).abs()

    print("\nTop Distinguishing Features:")
    top_features = relative_importance.nlargest(3)
    for feature, importance in top_features.items():
        print(f"- {feature}: {importance:.2f} units from mean")

# Analyze cluster distributions
cluster_sizes = df['Cluster'].value_counts()
print("\nCluster Size Distribution:")
print("==========================")
for cluster in range(optimal_k):
    size = cluster_sizes[cluster]
    percentage = (size / len(df)) * 100
    print(f"Cluster {cluster}: {size} samples ({percentage:.1f}%)")

# Visualize feature distributions within clusters
plt.figure(figsize=(15, 10))
for i, feature in enumerate(features[:6], 1):  # First 6 features
    plt.subplot(2, 3, i)
    for cluster in range(optimal_k):
        cluster_data = df[df['Cluster'] == cluster][feature]
        plt.hist(cluster_data, alpha=0.5, label=f'Cluster {cluster}', bins=20)
    plt.title(feature)
    plt.legend()
plt.tight_layout()
plt.show()

# Create 3D visualization of clusters
from mpl_toolkits.mplot3d import Axes3D
fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111, projection='3d')
scatter = ax.scatter(reduced_data[:, 0],
                     reduced_data[:, 1],
                     reduced_data[:, 2],
                     c=df['Cluster'],
                     cmap='viridis')
plt.colorbar(scatter)
ax.set_xlabel('PC1')
ax.set_ylabel('PC2')
ax.set_zlabel('PC3')
plt.title('3D Cluster Visualization')
plt.show()

# Calculate and print cluster separation metrics
print("\nCluster Separation Analysis:")
print("============================")
for i in range(optimal_k):
    for j in range(i+1, optimal_k):
        center_i = kmeans.cluster_centers_[i]
        center_j = kmeans.cluster_centers_[j]
        distance = np.linalg.norm(center_i - center_j)
        print(f"Distance between Cluster {i} and {j}: {distance:.2f}")

# Calculate within-cluster variation
print("\nWithin-Cluster Variation:")
print("=========================")
for i in range(optimal_k):
    cluster_points = reduced_data[df['Cluster'] == i]
    center = kmeans.cluster_centers_[i]
    variation = np.mean([np.linalg.norm(point - center) for point in cluster_points])
    print(f"Cluster {i}: {variation:.2f}")

# Feature correlation analysis within clusters
print("\nFeature Correlation Analysis:")
print("============================")
for i in range(optimal_k):
    print(f"\nCluster {i} Significant Correlations:")
    cluster_data = df[df['Cluster'] == i][features]
    corr_matrix = cluster_data.corr()
    # Get significant correlations
    significant_corr = np.where(np.abs(corr_matrix) > 0.7)
    for idx in range(len(significant_corr[0])):
        feat1, feat2 = features[significant_corr[0][idx]], features[significant_corr[1][idx]]
        if feat1 < feat2:  # Avoid printing duplicates
            corr = corr_matrix.loc[significant_corr[0][idx], significant_corr[1][idx]]
            print(f"{feat1} -- {feat2}: {corr:.2f}")

# Save detailed analysis to file
with open('cluster_analysis_report.txt', 'w') as f:
    f.write("Cluster Analysis Report\n")
    f.write("======================\n\n")
    f.write(f"Number of clusters: {optimal_k}\n")
    f.write(f"Total samples: {len(df)}\n\n")

    f.write("Cluster Sizes:\n")
    for cluster in range(optimal_k):
        size = cluster_sizes[cluster]
        percentage = (size / len(df)) * 100
        f.write(f"Cluster {cluster}: {size} samples ({percentage:.1f}%)\n")

    f.write("\nCluster Centroids:\n")
    f.write(cluster_centers.to_string())

print("\nDetailed analysis has been saved to 'cluster_analysis_report.txt'")
```

K MEANS HIERACHINE ANALIZĖ ANOMALIJŲ IZOLIACIJOS MIŠKO MEDŽIŲ GENERAVIMAS

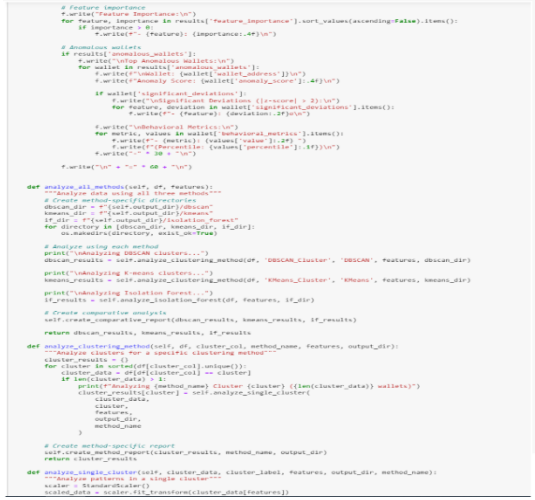
```
port pandas as pd
port numpy as np
om sklearn.cluster import KMeans
om sklearn.ensemble import IsolationForest
om sklearn.preprocessing import StandardScaler
om sklearn.tree import export_graphviz
port graphviz
port os
om datetime import datetime

ass IsolationForestVisualizer:
  def __init__(self, max_depth=None, output_format="pdf"):
      self.max_depth = max_depth
      self.output_format = output_format

  def create_tree_visualization(self, estimator, feature_names, cluster_id, output_path):
      dot_data = export_graphviz(
          estimator,
          feature_names=feature_names,
          filled=True,
          rounded=True,
          max_depth=self.max_depth,
          proportion=True,
          precision=2,
          impurity=False,
          class_names=['normal', 'anomaly']
      )

      graph = graphviz.Source(
          dot_data,
          directory=output_path,
          format=self.output_format
      )

      output_file = f"{output_path}/isolation_tree_cluster_{cluster_id}"
      graph.render(filename=f"isolation_tree_cluster_{cluster_id}",
                   cleanup=True)

      return f"{output_file}.{self.output_format}"

f format_wallet_info(row, features):
  """Format wallet information in a readable way"""
  wallet_info = []

  # Format feature values
  feature_values = {feature: row[feature] for feature in features}
  # Sort features by absolute value to identify most significant ones
  sorted_features = sorted(feature_values.items(), key=lambda x: abs(x[1]), reverse=True)

  # Add top 5 most significant features
  for feature, value in sorted_features[:5]:
      wallet_info.append(f"{feature}: {value:.4f}")

  return wallet_info

f analyze_anomalies(df, anomaly_scores, cluster_id, features, output_dir, timestamp):
  """Analyze and save information about the most anomalous wallets"""

  # First, let's print the actual columns to debug
  print("Available columns:", df.columns.tolist())

  # Assuming the wallet address column might be named "Wallet Address_1" or similar
  wallet_col = [col for col in df.columns if 'Wallet' in col][0]
  print(f"Using wallet column: {wallet_col}")

  # Convert anomaly scores to a more interpretable format
  normalized_scores = -anomaly_scores

  # Add scores to dataframe
  df_cluster = df.copy()
  df_cluster['anomaly_score'] = normalized_scores

  # Sort by anomaly score (highest anomalies first)
  df_anomalies = df_cluster.sort_values('anomaly_score', ascending=False)

  # Select top 10 most anomalous wallets
  top_anomalies = df_anomalies.head(10)

  # Create detailed analysis file
  analysis_file = f"{output_dir}/cluster_{cluster_id}_anomalies_{timestamp}.txt"

  with open(analysis_file, 'w') as f:
      f.write(f"ANOMALY ANALYSIS FOR CLUSTER {cluster_id}\n")
      f.write("=" * 50 + "\n\n")

      f.write("Top 10 Most Anomalous Wallets:\n")
      f.write("-" * 30 + "\n\n")

      for idx, row in top_anomalies.iterrows():
          f.write(f"Wallet ID: {row[wallet_col]}\n")  # Using detected wallet column
          f.write(f"Anomaly Score: {row['anomaly_score']:.4f}\n")
          f.write("Most Significant Features:\n")

          wallet_info = format_wallet_info(row, features)
          for info in wallet_info:
              f.write(f"- {info}\n")
          f.write("\n")

      # Add statistical summary
      f.write("\nStatistical Summary of Anomaly Scores:\n")
      f.write("-" * 30 + "\n")
      f.write(f"Mean Score: {df_cluster['anomaly_score'].mean():.4f}\n")
      f.write(f"Median Score: {df_cluster['anomaly_score'].median():.4f}\n")
      f.write(f"Standard Deviation: {df_cluster['anomaly_score'].std():.4f}\n")

      # Calculate and write percentile information
      percentiles = [95, 90, 75]
      f.write("\nPercentile Analysis:\n")
      for p in percentiles:
          score = np.percentile(df_cluster['anomaly_score'], p)
          f.write(f"{p}th percentile: {score:.4f}\n")

  # Save top anomalous wallets to CSV with formatted data
  output_csv = f"{output_dir}/cluster_{cluster_id}_top_anomalies_{timestamp}.csv"
  top_anomalies.to_csv(output_csv, index=False)

  return top_anomalies

f create_latex_table(all_anomalies, features, output_dir, timestamp):
  """Create LaTeX table for top anomalies across clusters"""
  latex_file = f"{output_dir}/anomaly_latex_tables_{timestamp}.tex"

  # Get wallet column name
  wallet_col = [col for col in next(iter(all_anomalies.values())).columns if 'Wallet' in col][0]

  with open(latex_file, 'w') as f:
      for cluster, anomalies in all_anomalies.items():
          # Table header
          f.write("\\begin{table}[htbp]\n")
          f.write("\\centering\n")
          f.write("\\caption{Top 5 Most Anomalous Wallets in Cluster " + str(cluster) + "}\n")
          f.write("\\begin{tabular}{|l|c|c|}\n")
          f.write("\\hline\n")
          f.write("Wallet ID & Anomaly Score & Key Features \\\\\n")
          f.write("\\hline\n")

          # Add top 5 wallets
          for idx, row in anomalies.head().iterrows():
              wallet_id = row[wallet_col]  # Using detected wallet column
              score = f"{row['anomaly_score']:.4f}"
```

```
                  # Get top 3 most significant features
                  feature_values = {f: row[f] for f in features}
                  top_features = sorted(feature_values.items(), key=lambda x: abs(x[1]), reverse=True)[:3]
                  features_str = "; ".join([f"{f[:15]}: {v:.2f}" for f, v in top_features])

                  f.write(f"{wallet_id} & {score} & {features_str} \\\\\n")

              f.write("\\hline\n")
              f.write("\\end{tabular}\n")
              f.write("\\label{tab:anomalies_cluster_" + str(cluster) + "}\n")
              f.write("\\end{table}\n\n")

def main():
      timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")

      # Load dataset
      df = pd.read_csv("Solana_Bots_Dataset.csv")

      # Define features
      features = [
          "Transaction Frequency",
          "Average Transaction Amount",
          "StdDev Transaction Amount",
          "Fee-to-Amount Ratio",
          "Unique Receivers Count",
          "Account Role Diversity",
          "Instruction Complexity",
          "Success Ratio",
          "DEX Transaction Frequency",
          "Program Distribution Score",
          "Active Hours Per Day",
          "Activity Consistency"
      ]

      short_features = [f"F{i}:{name[:10]}" for i, name in enumerate(features)]
      feature_map = dict(zip(short_features, features))

      # Preprocess data
      scaler = StandardScaler()
      scaled_features = scaler.fit_transform(df[features])

      # Apply clustering
      n_clusters = 3
      kmeans = KMeans(n_clusters=n_clusters, random_state=42)
      df['Cluster'] = kmeans.fit_predict(scaled_features)

      output_dir = f"appendix_visualizations_{timestamp}"
      os.makedirs(output_dir, exist_ok=True)

      visualizer = IsolationForestVisualizer(max_depth=None, output_format="pdf")

      # Store all anomaly results
      all_anomalies = {}
      tree_paths = []

      for cluster in df['Cluster'].unique():
          print(f"\nProcessing Cluster {cluster}...")

          cluster_mask = df['Cluster'] == cluster
          cluster_data = df[cluster_mask]
          cluster_features = scaled_features[cluster_mask]

          iforest = IsolationForest(
              n_estimators=100,
              max_samples='auto',
              random_state=42,
              contamination=0.1
          )

          iforest.fit(cluster_features)
          anomaly_scores = iforest.score_samples(cluster_features)

          top_anomalies = analyze_anomalies(
              cluster_data,
              anomaly_scores,
              cluster,
              features,
              output_dir,
              timestamp
          )
          all_anomalies[cluster] = top_anomalies

          tree_path = visualizer.create_tree_visualization(
              iforest.estimators_[0],
              short_features,
              cluster,
              output_dir
          )
          tree_paths.append(tree_path)

          print(f"Tree visualization saved: {tree_path}")
          print(f"Anomaly analysis saved for cluster {cluster}")

      # Create LaTeX tables
      create_latex_table(all_anomalies, features, output_dir, timestamp)

      # Create summary report
      summary_file = f"{output_dir}/anomaly_summary_{timestamp}.txt"
      with open(summary_file, 'w') as f:
          f.write("SUMMARY OF ANOMALOUS WALLETS ACROSS CLUSTERS\n")
          f.write("=" * 50 + "\n\n")

          for cluster, anomalies in all_anomalies.items():
              f.write(f"\nCluster {cluster} - Top 5 Most Anomalous Wallets:\n")
              f.write("-" * 40 + "\n")
              for idx, row in anomalies.head().iterrows():
                  f.write(f"Wallet ID: {row['Wallet Address']}\n")  # Fixed case here
                  f.write(f"Anomaly Score: {row['anomaly_score']:.4f}\n")
                  wallet_info = format_wallet_info(row, features)
                  f.write("Key Characteristics:\n")
                  for info in wallet_info:
                      f.write(f"- {info}\n")
                  f.write("\n")

      print("\nAnalysis Complete!")
      print(f"Summary report saved to: {summary_file}")
      print("\nFiles generated:")
      print("1. Tree visualizations (PDF)")
      print("2. Detailed anomaly analysis per cluster (TXT)")
      print("3. Top anomalous wallets per cluster (CSV)")
      print("4. Overall summary report (TXT)")
      print("5. LaTeX tables for publication")

if __name__ == "__main__":
      main()
```

DBSCAN HIERACHINE ANALIZĖ ANOMALIJŲ IZOLIACIJOS MIŠKO MEDŽIŲ GENERAVIMAS



METODŲ PALYGINIMAS



VERTINIMAS