

VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
PROGRAMŲ SISTEMŲ KATEDRA

Baigiamasis bakalauro darbas

**Privačios informacijos išsaugojimas taikant dirbtinio intelekto
technologijas**
(Privacy-preserving AI)

Atliko: 4 kurso 3 grupės studentas

Paulius Milmantas (parašas)

Darbo vadovas:

dr. Linas Petkevičius (parašas)

Recenzentas:

Vilnius
2021

Turinys

Išvadas	2
1 Asmens duomenų privatumas mašininio mokymo kontekste	3
2 Duomenų pažeidžiamumo metrikos	6
2.1 Apibrėžimas	6
2.2 Skaičiavimas praktikoje	7
2.3 Iliustacinis pavyzdys	8
3 Modelių palyginimo metodai	9
3.1 Lyginimas pagal duomenų nuokrypį	9
3.2 Skaičiavimo pavyzdys	9
3.3 Algoritmo verifikavimas	10
4 Homomorfinis šifravimas	12
5 Federuotas mašininis mokymas	13
5.1 Metodas	13
5.2 Komunikacijos kaštų mažinimas	13
5.3 Įrenginių heterogeniškumas	13
5.4 Analizės heterogeniškumas	13
5.5 Saugos problemos	13
6 Blokų-grandinių technologija	14
7 Saugus skirtingų pusių skaičiavimas	15
8 Bendro įspūdžio privatus agregavimas	16
9 Tyrimas	17
9.1 Modelių palyginimas	17
9.2 DMDK metrikos tyrimas	17
9.3 Tyrimo išvados	17

Įvadas

Mašininis mokymas yra dirbtinio intelekto sritis, kuri pasitelkia statistinius algoritmus, kad apibrėžtų duomenų generavimo mechanizmą, ar egzistuojančius sąryšius, priklausomybes. Modelis dažnai turi didelį kiekį nežinomų parametrų, kuriuos reikia įvertinti iš duomenų, todėl modelio apmokymui dažniausiai reikia turėti daug duomenų. Kai kurie uždaviniai reikalauja duomenų, kurie nėra laisvai prieinami ir yra privatūs. Mašininio mokymo tyrimų srityje yra kilusi problema dėl jų saugojimo. Vienas iš faktorių, kuris lėmė šį susidomėjimą yra 2016 metais Europos Sąjungoje priimtas duomenų apsaugos reglamentas (GDPR). Pagal jį, fizinių asmenų duomenys turi būti saugomi naudojantis tam tikromis taisyklėmis ir negali būti atskleisti trečiosioms šalims, be asmens sutikimo [116].

Šią problemą išspręsti siekia įvairūs tyrimai ir naujai pasiūlyti metodai privatumą saugančio dirbtinio intelekto srityje. Šią problemą galima išskaidyti į kelias atskiras sritis:

- Analizuojamų duomenų privatumas [Tha20]. Algoritmas apmoko modelį atpažinti duomenis. Turint sukurtą modelį, neturi būti galima atgaminti duomenų, pagal kuriuos jis buvo mokomas, bei negali būti identifikuoti asmenys. Taip nukentėtų žmonių privatumas ir būtų pažeistas Europos duomenų apsaugos reglamentas. Šio pažeidimo pavyzdys gali būti ir paprastas teksto atkūrimo modelis. Duodama sakinio pradžia, modelis nuspėja jo pabaigą. Jeigu suvedus tam tikras detales modelis užbaigia sakinį naudodamas asmeninius duomenis, kurie atskleidžia žmonių tapatybę, šis modelis nėra saugus.
- Duomenų įvesties privatumas. Trečios šalys neturi matyti įvedamų duomenų. Tai gali būti tinklo saugumo spragos, duomenų surinkimo aplikacijų spragos ir t.t...
- Modelio išvesties privatumas. Modelio išvesties neturi matyti asmenys, kuriems šie duomenys nepriklauso. Šis punktas yra sąlyginis, priklauso nuo modelio svarbos. Jeigu tai yra svarbūs asmeniniai duomenys, negalima rizikuoti. Tačiau jeigu tai yra viešai prieinami duomenys, šis punktas negalioja.
- Modelio apsauga. Sukurtas modelis negali būti niekieno pasisavintas. Šis punktas yra skirtas apsaugoti programos kūrėją.

Darbo tikslas - ištirti ir palyginti privatumą saugančius dirbtinio intelekto algoritmus pagal jų saugumą, našumą ir panaudojamumą, bei pateikti rekomendacijas.

Darbo uždaviniai:

- Išanalizuoti esamus algoritmus pagal jų saugumą ir panaudojamumą.
- Ištirti kurie algoritmai yra realizuoti realizuoti algoritmus, kurie nėra atvirai prieinami.
- Palyginti algoritmus pagal našumą.

1 Asmens duomenų privatumas mašininio mokymo kontekste

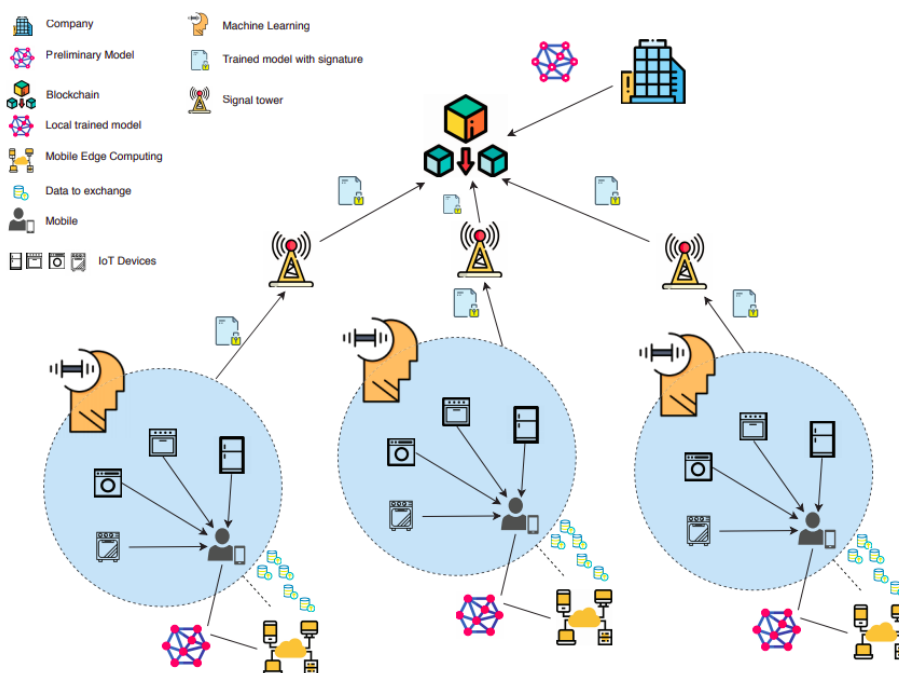
Vienas iš būdų apsaugoti duomenis yra decentralizuoti modelį ir naudoti paskirstyto mokymo algoritmus. Paprastai mašininiam mokymui yra naudojamas centralizuotas serveris. Yra surenkami duomenys į vieną vietą ir modelį moko vienas kompiuteris. Jeigu norima pridėti daugiau duomenų arba papildomą klasifikatorių, modelį reikia apmokyti iš naujo. Taip pat, kas kuria šį modelį, turi visą prieigą prie duomenų, šie modeliai nėra saugūs.

Šiai bėdai išspręsti, vienas iš metodų yra federuotas mašininis mokymas (“Federated machine learning”). Šis metodas padeda išspręsti kaikurias bėdas, vykdamas modelio mokymą decentralizuotai [Bha19]. Prie bendro tinklo gali prisijungti daug įrenginių. Visi jie turi savo unikalų duomenų rinkinį. Kiekvienas įrenginys pasirenka geriausią statistinį metodą modelio mokymui ir pagal tą metodą sukuria modelį. Taip yra sukuriamas daug skirtingų modelių, realizuotų su skirtingais duomenų rinkiniais. Visi šie rinkiniai vėliau yra surenkami į vieną vietą ir toliau naudojami duomenų analizei. Taip surinkus atskirų įrenginių sukurtus modelius, neturime prieigos prie pradinių duomenų ir daugiau žmonių gali prisidėti prie modelio kūrimo, nematant pilno duomenų rinkinio. Tačiau naudojant šį metodą ne visos problemos yra išsprendžiamos. Duomenų saugumas nėra garantuojamas [Bha19]. Jeigu duomenys nėra užšifruojami, jie gali būti pavogti. Tai gali būti padaryta, jeigu yra naudojamas nesaugus interneto ryšys, arba jeigu yra bandoma analizuoti atskirų įrenginių atsiųstus modelius. Šis metodas išsprendžia tik kelias problemas. Likusios yra: komunikavimo kaštai, įrenginių heterogeniškumas, statistinis heterogeniškumas ir saugumo problemos [LSTS19a].

Visos šios problemos yra nagrinėjamos ir joms spręsti kuriami nauji metodai. Komunikacijos kaštams mažinti, yra kuriami algoritmai, kaip turi būti perduodamas modelis tinkle, kad būtų mažas tinklo apkrova. Įrenginių heterogeniškumui spręsti, yra parenkami tam tikri įrenginių rodikliai ir pagal tai nustatoma, kokia bus įrenginio komunikacija: ar jis naudos asinchroninį komunikavimą, kaip dažnai tai vyks ir kokia yra įrenginio klaidos tikimybė [LSTS19a]. Statistikos heterogeniškumui pataisyti kuriami yra metodai, kaip meta duomenų-mokymas ir keletos užduočių mokymas (“Multi-task learning”).

Straipsnyje apie privatumą išsaugančius blokų-grandinių tinklus (“Privacy-Preserving Blockchain-Based Federated Learning for IoT Devices”) yra pateikiamas naujas paskirstyto mokymosi algoritmas, kuris yra paremtas blokų grandinių technologijomis [ZZJ⁺20]. Šis modelis yra grafiškai pateiktas pav.1. Šią architektūrą pavyzdyje sudaro 3 komponentai: gamintojai, klientai ir blokų gradinių tinklas. Šiame pavyzdyje gamintojas pareiškia užklausą, kad reikia atlikti apklausą. Klientai, kurie sutinka su užklausa, išsiunčia savo sukurtą modelį. Blokų grandinių tinklas elgiasi kaip centralizuotas serveris ir surenka visus klientų modelius. Tuomet, pasirinktas kompiuteris, kuris atlieka visą darbą (“miner”) atlieka galutinį modelių sujungimą.

Minėti decentralizuoti būdai apjungia skirtingus sukurtus modelius į vieną ir visi įrenginiai kurie kuria modelius, naudoja savo unikalius duomenis. Saugus skirtingų pusių skaičiavimo metodas (“Secure multiparty computation”) yra kriptografinis protokolas, kuris leidžia įrenginiams, su unikaliais duomenimis, skaičiuoti funkcijos reikšmę, nematant kitų įrenginių reikšmių [6]. Su šiuo



1 pav. Paskirstytas mokymas [ZZJ⁺20]

metodu, yra sukuriamas vienas modelis tarp įvairių įrenginių. Pagrindinis įrenginys, kuris ruošia skaičiavimo užklausą, gali suskaidyti pradinis duomenis, užšifruoti juos ir taip paskirstyti tarp kitų įrenginių. Taip sukurti įrenginio rezultatai negali būti atšifruoti ir pasisavinti, o minėto federuoto mokymo metodo sukurti rezultatai, jeigu nėra gerai apsaugoti, gali būti pasisavinti ir atgauti pradiniai duomenys su kuriais modelis buvo sukurtas.

Kitas būdas apsaugoti modelį yra naudoti homomorfinį šifravimą. Pagal šį metodą, modelis yra mokomas naudojant užšifruotais duomenimis. Jie mokymo metu, nėra atšifruojami. Šis metodas leidžia keliems įrenginiams vienu metu atlikti skaičiavimus su duomenimis, kurių jie nemato. Yra daug metodo varijacijų. Kaikurios yra pažeidžiamos ir gali atskleisti visos infrastruktūros duomenis [Sen13]. Dažniausiai homomorfinės sistemos yra labiau pažeidžiamos nei nehomomorfinės.

2016 metais buvo pasiūlytas naujas metodas paspartinti pilnai homomorfinės šifravimo sistemos [CKKS16]. Homomorfinės sistemos, kurios naudoja šį metodą yra laikomos ketvirtos kartos. Šis metodas leidžia aproksimuoti užšifruoto teksto sudėtį, daugybą ir pakeisti atšifruoto teksto proporcijas. Proporcijų pakeitimo procedūra suskaido užšifruotą tekstą į dalis, to pasekoje yra apvalinamas neužšifruotas teksas. Šio metodo pagrindinė idėja yra pridėti triukšmo prie pagrindinės žinutės. Šis triukšmas yra pridedamas prie neužšifruoto teksto dėl saugumo priežasčių ir jis yra laikomas kaip aproksimavimo paklaida. Tuo pasekoje, metodas veikia su tam tikra paklaida

Bendro išspūdžio agregavimo metodas yra naujausias iš visų minėtų. Jis leidžia įrenginiui su savo duomenimis sukurti modelį naudojant betkokius metodus ir taip prisidėti prie bendro progreso [6]. Visi modeliai yra surenkami ir leidžiami duomenys per šiuos modelius. Kiekvienas modelis skiria savo balsą, jie surenkami ir padaromas bendras sprendimas. Šis metodas leidžia lengvai plėsti

modelį. Taip pat yra saugomas ir duomenų privatumas. Jeigu keli modeliai, kurie nesidalina duomenimis, teigia vienodai, tai reiškia, kad negalima atkurti duomenų ir jie yra saugūs. Skaičiavimas vykdomas pasirenkant optimalią strategiją ir imant mažiausią galimą žingsnių skaičių.

2 Duomenų pažeidžiamumo metrikos

2.1 Apibrėžimas

Norint apsaugoti duomenis, sukurtam duomenų aptikimo modeliui reikia atlikti analizę, kaip tikėtina, kad modelio pradiniai duomenys bus atkurti. Tai išanalizuoti yra daug būdų, vienas iš jų yra pateiktas straipsnyje [LSTS19b]. Šis straipsnis ieško metrikos reikšmės, kuri parodo, kaip pradinis duomenis atsimena modelis.

Tarkime turime duomenų rinkinį $s[r]$. Šiam rinkiniui pirma reikia apskaičiuoti rangą [CKKS16].

$$rangas(s[r]) = |\{r' \in R : Px(s[r']) \leq Px(s[r])\}| \quad (1)$$

čia funkcija Px – yra .., $|\cdot|$ – yra .., $s[\cdot]$ – yra ...

Šis rangas nurodo, kurioje vietoje yra šis inicijuotas duomenų rinkinys sąraše, tarp visų galimų testinių rinkinių kombinacijų. Pavyzdžiui, norima sukurti natūralios kalbos atpažinimo modelį. Jo mokymui, yra paduodamas atsitiktinai sugeneruotas sakinyss “Atsitiktinis skaičius 125” ir apskaičiuojama jo entropija. Tada galima išrašyti visus galimus sakinius, kuriuos yra leidžiama siųsti, ir juos surūšiuoti pagal entropijos laipsnį. Tada reikia apskaičiuoti, kaip dažnai pasitaiko panašūs sakiniai. Taip yra gaunamas rangas. Kadangi šis pavyzdinis sakinyss yra sudėtingas, su atsitiktiniais skaičiais ir jo entropija yra aukšta, tikriausiai aukštesnė nei dauguma mokymo duomenų, jo rangas bus artimas vienetui. Darant prielaidą, kad šio sakinio entropija yra pati aukščiausia iš galimų testavimo duomenų, galima teigti, jog šio sakinio rangas yra lygus vienetui.

Rangas tiesiogiai nepasako kokia yra tikimybė, kad bus sugeneruotas toks testavimo rinkinys. Jo skaičiavimas paima daug resursų, nes reikia sugeneruoti visas galimas kombinacijas [CKKS16].

Kitas nagrinėjamas kintamasis yra “atvirumo” metrika. Tai aproksimacija, kuri nusako, kaip tikėtina, jog testavimo duomenys bus atgaminti iš modelio. Ši metrika pasako ką naujo sužinome apie pradinis duomenis, kai per modelį paleidžiame atsitiktinai sugeneruotus duomenis. Todėl, reikia skaičiuoti prognozuojamą entropiją, pagal pateiktas modelio prognozes. Prognozės entropija, tai spėjimų skaičius $E(X)$, kuris reikalingas atspėti diskretų, atsitiktinai sugalvotą parametą X .

Jeigu kintamasis r yra pasirenkamas atsitiktinai $r \in R$, tai reiškia, kad reikia generuoti atsitiktinius skaičius tol, kol bus gauta r reikšmė. Iš to seka, kad spėjimų turėtų būti lygus [2] lygčiai.

$$E(s[r])_\theta = \frac{1}{2}|R| \quad (2)$$

Turint suskaičiuotus galimus rinkinių rangus, bei entropijas, galima naudoti patobulinta skaičiavimo taktiką. Visi galimi duomenys yra surūšiuojami pagal entropiją arba sudėtingumą. Sąrašo pradžioje yra elementas su mažiausia entropija. Jo atspėjimo tikimybė yra viena iš didžiausių. To pasekoje, gauname formulę [3].

$$E(s[r]|f_\theta) = rank_\theta(s[r]) \quad (3)$$

Ši formulė padidina skaičiavimų spartumą ir tai galima apskaičiuoti padalinus vieną formulę iš kitos. Tai pateiktas [4] lygtys.

$$\frac{E(s[r])}{E[s[r]|f_\theta]} = \frac{\frac{1}{2}|R|}{rank_\theta(s[r])} \quad (4)$$

Rangai tiksliai neapibrėžia kokia yra tikimybė, jog elementas bus atspėtas. Jis tik lygina visas galimas kombinacijas ir skaičiuoja jų entropijas. To pasekoje, visi šie skaičiavimai yra tik apytikslis spėjimas. Kadangi tai nėra tikslu ir norima sužinoti tik bendrą vaizdą apie algoritmą, galima naudoti logaritmus. Tai atlikta lygtys [5].

$$\log_2\left(\frac{E(s[r])}{E[s[r]|f_\theta]}\right) = \log_2\left(\frac{\frac{1}{2}|R|}{rank_\theta(s[r])}\right) \quad (5)$$

Šias lygtis supaprastinus, gaunama lygtis [6].

$$\log_2|R| - \log_2(rank_\theta(s[r])) - 1 \quad (6)$$

Dėl paprastumo yra pridedamas vienetas ir taip gaunama atvirumo metrika, pateikta lygtys [7].

$$atvirumas(s[r])_\theta = \log_2|R| - \log_2 rangas_\theta(s[r]) \quad (7)$$

2.2 Skaičiavimas praktikoje

Pažeidžiamumo metrika “Atvirumas” yra naudojama tikrinant mašininio mokymo algoritmų duomenų įsiminimą. Atlikus šiuos skaičiavimus, programuotojai gali spręsti dėl tolimesnių veiksmų: tęsti kūrimą ar dirbti ties duomenų anonimizavimu.

Mašininio mokymo algoritmo tikrinimui su atvirumo metrika, pirma reikia pasiruošti duomenų rinkinį $s[r]$. Sudarius šį rinkinį, reikia sukurti mašininio mokymo modelį. Turint gautą modelį ir duomenis, kurie buvo naudojami modelio gavimui, galima skaičiuoti atvirumo metriką. Siekiant daugiau sužinoti apie modelį, galima savo duomenis įterpti daugiau nei vieną kartą. Pavyzdžiui, jeigu vienus duomenis įterpsime kelis kartus, o kitus duomenis šimtą kartų, galima analizuoti atvirumo metriką su šiais duomenimis. Taip galima gauti abstrakčią koreliaciją tarp duomenų kiekio ir modelio atsiminimo. Kad šie eksperimentai būtų tikslūs, būtina kiekvieno modelio kūrimo metu naudoti tuos pačius parametrus: optimizavimo funkcijas, hiperparametrus ir kitus duomenis. Taip su kiekvienu pavyzdiniu duomenų rinkiniu, turi būti atlikta atskira analizė. Jeigu galima, turi būti didinamas vienodų duomenų kiekis ir tikrinama atvirumo ir žinomų duomenų kiekio koreliacija.

Atvirumui sužinoti, reikia skaičiuoti rangą. Tikslaus jo radimo procedūros nėra. Todėl reikia naudotis analitiniais metodais. Vienas iš jų yra aproksimacija pagal distribucijos modelius. Tarkime, turime duomenis, kurių entropija yra didesnė, nei $s[r]$ duomenų. Reikia rasti kiek yra tokių duomenų, kurių entropija yra mažesnė. Taip pat, tarkime, kad $s[r]$ entropija yra $\rho(\cdot)$ pasiskirstymo

distribucijos. Atvirumą galima aproksimuoti skaičiuojant šio skirstinio plotą grafike, iki testavimo duomenų, ir dauginant iš logaritmo. Taip gauname formulę [8].

$$atvirumas(s[r])_{\theta} = -\log_2 \int_0^{Px_{\theta}(s[r])} \rho(x) dx \quad (8)$$

2.3 Iliustacinis pavyzdys

Metodo panaudojamumą pademonstruoti, pasinaudokime mašininio modeliu, pagrįstu cukriniu diabetu sergančių moterų duomenimis. Modelis priima 5 parametrus: nėštumų skaičių, gliukozės kiekį kraujyje, kraujo spaudimą, BMI ir amžių. Visų šių parametrų ribos yra žinomos, todėl turint atsitiktinių duomenų rinkinį, galima apskaičiuoti rangus. Todėl bus naudojama lygtis (8), turinti rangus.

$$atvirumas(s[r])_{\theta} = \log_2 |R| - \log_2 rangas_{\theta}(s[r]) \quad (9)$$

Tarkime, kad šių duomenų galimos ribos yra tokios: nėštumai: [0-30], gliukozės kiekis kraujyje: [0-180], kraujo spaudimas: [0-250], BMI: [1-70], amžius: [1-110]. Suskaičiuoti kiek iš viso yra galimų variantų, galima sudauginus visas šias ribas, tai padaryta lygtyje (10).

$$R = 30 \cdot 180 \cdot 250 \cdot 69 \cdot 109 = 10153350000 \quad (10)$$

Tarkime, turime dviejų žmonių testinių duomenų rinkinį. Pirmo žmogaus duomenys: [2, 60, 98, 25, 25], antro: [0, 90, 128, 27, 18]. Skaičiuojame rangus (11) ir (12). Vėliau pagal apskaičiuotus rangus, ieškoma atvirumo metrika (13) ir (14).

$$s[1] = 2 \cdot 180 \cdot 250 \cdot 69 \cdot 109 + 60 \cdot 250 \cdot 69 \cdot 109 + 98 \cdot 69 \cdot 109 + 25 \cdot 109 + 25 = 790444808 \quad (11)$$

$$s[2] = 90 \cdot 250 \cdot 69 \cdot 109 + 128 \cdot 69 \cdot 109 + 27 \cdot 109 + 18 = 170188149 \quad (12)$$

$$atvirumas(s[1]) = \log_2 10153350000 - \log_2 790444808 = 3.68314726 \quad (13)$$

$$atvirumas(s[2]) = \log_2 10153350000 - \log_2 170188149 = 5.89868142 \quad (14)$$

Pagal rezultatus, galima teigti, kad yra labiau tikėtina, kad bus atskleisti antro žmogaus duomenys. Turint daugiau parametrų, būtų dar mažesnė atvirumo metrikos reikšmė, nes būtų dar mažiau tikėtina, kad bus atkartoti duomenys. Turint kelius modelius, galima lyginti modelių atvirumo metrikas ir spręsti apie modelių saugumą.

3 Modelių palyginimo metodai

3.1 Lyginimas pagal duomenų nuokrypį

Siekiant palyginti, kaip modeliai gerai saugo privačius duomenis, apibrėžiame metriką DMDK (didžiausias modelio duomenų nuokrypis). Kuo metrika yra mažesnė, tuo tiksliau galima nuspėti, kokie duomenys buvo naudojami modelio mokymui. Kuo metrika didesnė, tuo yra sunkiau nuspėti, kokie duomenys buvo naudojami mokymui. Ši metrika leidžia lyginti skirtingus modelius su skirtingais duomenimis.

DMDK yra išvestas lygtyje [?]. Prieš DMDK skaičiavimą reikia paimti visus modelio mokymui skirtus duomenis ir kiekvienai duomenų eilutei apskaičiuoti modelio išvestį. Skaičiavimus reikia atlikti tik su tomis eilutėmis, su kuriomis modelis išvedė teisingą atsakymą. Turint tik tas eilutes, su kuriomis modelis išvedė teisingą atsakymą, galima į nelygybę įstatyti kintamuosius. Lygtyje yra naudojami tokie kintamieji: m - duomenų eilučių skaičius, h - parametrų skaičius (stulpeliai), ϵ - ieškomas didžiausias galimas kintamasis, su kuriuo modelis nepakeičia išvesties rezultatų, $D_{eilut.:n, stulp.:k}$ - duomenys n eilutėje ir k stulpelyje.

$$DMDK = \sum_{n=0}^m \left(\sum_{k=0}^h (max(|\epsilon| + D_{eilut.:n, stulp.:k}) : \epsilon \in R) \right) / h / m \quad (15)$$

3.2 Skaičiavimo pavyzdys

Tarkime egzistuoja modelis, kuris turi 2 parametrus: gliukozės kiekis kraujyje ir KMI. Pagal šiuos du parametrus modelis išveda rezultatą: ar žmogus serga cukriniu diabetu ar ne. Paimamos visos mokymo duomenų eilutės, su kuriomis modelis išveda teisingą rezultatą. Tarkime, jos yra aprašytos 5 lentelėje.

Gliukozė	KMI	Išvestis
148	33.6	1
85	26.6	0
183	23.3	1

1 lentelė. Pavyzdiniai duomenys

Taikant jau sukurtą modelį, įvertiname ar keičiasi išvestis didinant kiekvienos eilutės kiekvieno stulpelio reikšmes, pridėdant ϵ . Kiekvienai reikšmei yra atskirai didinamas ϵ , kol gaunamas tos eilutės tam tikro stulpelio maksimalus ϵ . Skaičiavimams parodyti, tarkime, kad gautos maksimalios ϵ reikšmės yra pateiktos lentelėje [2].

DMDK skaičiavimas pateiktas lygtyje [16].

$$DMDK = \frac{3+2}{2} + \frac{6+1}{2} + \frac{4+0.1}{2} = 8.05 \quad (16)$$

Apskaičiavus kitų modelių DMDK reikšmes, galima lyginti, kuris modelis efektyviau saugo

Gliukozė	KMI
3	2
6	1
4	0.1

2 lentelė. Pavyzdinės epsilon reikšmės

duomenis, naudotus mokymosi metu. Jeigu kito modelio apskaičiuota DMDK reikšmė būtų didesnė už 8.05, reikštų, kad tas modelis labiau saugo duomenų privatumą.

3.3 Algoritmo verifikavimas

Siekiant įrodyti, kad algoritmas teisingai įvertina skirtingus modelius ir galima juos palyginti tarpusavyje, paimkime kelis skirtingus modelių pavyzdžius.

Tarkime, kad pirmas modelis turi vieną parametą - KMI. Pagal šį parametą, modelis prognozuoja, ar žmogus serga cukriniu diabetu ar ne. Modelio tikslumas yra 54%, jis visą laiką prognozuoja, kad žmogus serga cukriniu diabetu. Skaičiuojant DMDK, reikia pasirinkti tik tuos duomenis, su kuriais buvo išvesta teisinga prognozė. Modelis visą laiką prognozuoja, kad žmogus serga cukriniu diabetu, todėl, nėra tokios reikšmės, kurią pridėjus prie duomenų, pasikeis modelio prognozė. Todėl DMDK reikšmė yra ∞ ir tai reiškia, kad modelis negali būti saugesnis. Kai modelio prognozė visą laiką yra tapati, neįmano atgaminti pradinių duomenų, su kuriais modelis buvo mokomas.

Antras pavyzdinis modelis prognozuoja ar žmogus serga cukriniu diabetu, pagal 2 parametrus: KMI ir gimdymų skaičiumi. Modelio tikslumas yra 72%, modelio DMDK reikšmė yra 0,00134. Sprendžiant pagal DMDK, modelis yra nesaugus ir yra labai priklausomas nuo pradinių duomenų, su kuriais modelis buvo mokomas. Modelio nesaugumui įrodyti, paimkime kelias savo sugalvotas duomenų eilutes ir pabandykime atgaminti pradinius duomenis. Tarkime, kad mūsų sugalvotos duomenų eilutės yra lentelėje [3].

KMI	Gimdymų skaičius
25	3
23	2

3 lentelė. Sugalvotos duomenys reikšmės modelio tikrinimui

Kiekvienai sugalvotai duomenų eilutei, analizuojame kiekvieną stulpelį. Prie stulpelio reikšmės pridedame ϵ ir didiname ϵ reikšmę tol, kol pasikeičia modelio rezultatas. Analizuojame kiekvieną stulpelį ir pasirenkame stulpelį su mažiausia ϵ reikšme. Pakeičiame analizuojamos duomenų eilutės stulpelio reikšmę, su kuria gauname mažiausią ϵ reikšmę. Išanalizuokime pirmą duomenų eilutę lentelėje [3]. Po pirmos iteracijos, gauname pakeistą duomenų eilutę: KMI - 26, gimdymų skaičius - 3. Atliekame iteracijas tol, kol gaunamų naujų reikšmių skirtumas tampa labai mažas. Atlikus visų duomenų analizę, gauname duomenis, kurie buvo gauti iš sugalvotų duomenų eilučių. Jeigu modelio DMDK yra mažas, šie duomenys turi būti artimi pradiniais duomenimis, kurie buvo naudojami modelio mokymui. Atlikus analizę buvo gauti duomenys lentelėje [4]. Duomenys

yra labai panašūs pradiniam modelio duomenims. Tai reiškia, kad DMDK matavimo algoritmas pasiteisino.

KMI	Gimdymų skaičius
26	1
22	2

4 lentelė. Sugalvotos duomenys reikšmės modelio tikrinimui

4 Homomorfinis šifravimas

Homomorfinis šifravimas, yra šifravimo algoritmų klasė, kuri yra grindžiama principu, leidžiančiu atlikti skaičiavimus su užšifruotais duomenimis, jų neatšifruojant. Šie algoritmai yra šiek tiek panašūs į viešo šifravimo algoritmus, kurie yra paplitę tinklalapių apsaugoje. Yra naudojamas viešas ir privatus raktas. Su viešu raktu duomenys yra užšifruojami, o su privačiu, atšifruojami. Homomorfinis šifravimas skiriasi nuo viešo šifravimo tuo, kad naudoja algebrinę sistemą, kuri leidžia atlikti skaičiavimus su užšifruotais duomenimis.

Algebrinė sistema yra duomenų rinkinys kartu su savo matematinėmis operacijomis.

Yra trijų tipų homomorfiniai šifrai: dalinai homomorfiniai (PHE), šiek tiek homomorfiniai (SHE) ir pilnai homomorfiniai šifrai (FHE). Dalinai homomorfinės sistemos leidžia atlikti skaičiavimus visiems duomenims, tik su pasirinkta operacija. Tai gali būti arba sudėtis arba daugyba. Operacijos duomenims gali būti atliktos neribotą kartą. Šiek tiek homomorfinės sistemos (SHE) naudoja vieną operaciją iki tam tikro sudėtingumo. Šios operacijos gali būti atlikto ribotą kartų skaičių. Pilnai homomorfinės sistemos leidžia naudoti sudėtį ir daugybą vienu metu, neribotą kartų.

Homomorfinių sistemų saugumas yra paremtas žiedinio mokymo su klaidomis problema (“Ring learning with errors”).

5 Federuotas mašininis mokymas

5.1 Metodas

5.2 Komunikacijos kaštų mažinimas

5.3 Įrenginių heterogeniškumas

5.4 Analizės heterogeniškumas

5.5 Saugos problemos

6 Blokų-grandinių technologija

7 Saugus skirtingų pusių skaičiavimas

8 Bendro įspūdžio privatus agregavimas

9 Tyrimas

9.1 Modelių palyginimas

Tyrimo tikslams, buvo sukurti modeliai ir, skaičiuojant DMDK metriką, jie lyginami tarpusavyje. Gauti rezultatai pateikti lentelėje [5].

Modelis	Nuostolių f-jos reikšmė (MSE)	Tikslumas	DMDK
PyTorch neuroninis tinklas	103.9837813	81.1%	87.419
Gradientinio nuolydžio algoritmas	2272.06	63.78%	0.051792
Pallier homomorfiniu šifravimu grįstas modelis	2185.37	63.52%	0.1416

5 lentelė. Sugaltotos duomenys reikšmės modelio tikrinimui

Pagal gautus rezultatus, pateiktus lentelėje [5], gautos išvados:

- PyTorch neuroninis tinklas greičiau artėja prie nuostolių funkcijos minimumo, nei kiti modeliai. Esant 70% modelių tikslumui, PyTorch modelis yra saugiausias. Mažiausiai saugus yra paprastu gradientiniu nuolydžiu grįstas modelis.
- Palliers kriptografija grįstas modelis yra saugesnis už paprastą gradientinio nuolydžio metodą.
- Didėjant PyTorch modelio tikslumui, jis pradeda labiau prisirišti prie duomenų ir vidutinė galima maksimalaus nuokrypio reikšmė smarkiai krenta – modelis tampa vis mažiau saugus. Jeigu uždavinio tikslas yra sukurti labai tikslų modelį, vertėtų apgalvoti ar Pallier sistema grįstas modelis nėra geriau.
- Su visais modeliais, išskyrus PyTorch neuroninį tinklą, didėjant modelio tikslumui, didėja vidutinė maksimalaus nuokrypio reikšmė – modelis tampa saugesnis, o su PyTorch neuroniniu tinklu, mažėja - modelis tampa mažiau saugus.

9.2 DMDK metrikos tyrimas

9.3 Tyrimo išvados

Literatūros sąrašas

- [116] Europos parlamento ir tarybos reglamentas, Apr 2016.
- [6]
- [Bha19] Dr. Santanu Bhattacharya. The new dawn of ai: Federated learning, Jan 2019.
- [CKKS16] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers, 2016.
- [LSTS19a] Tian Li, Anit Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions, 08 2019.
- [LSTS19b] Tian Li, Anit Sahu, Ameet Talwalkar, and Virginia Smith. Federated learning: Challenges, methods, and future directions, 08 2019.
- [Sen13] Jaydip Sen. Homomorphic encryption: Theory & applications, 07 2013.
- [Tha20] Patricia Thaine. Perfectly privacy-preserving ai, Jan 2020.
- [ZZJ⁺20] Yang Zhao, Jun Zhao, Linshan Jiang, Rui Tan, Dusit Niyato, Zengxiang Li, Lingjuan Lyu, and Yingbo Liu. Privacy-preserving blockchain-based federated learning for iot devices, 2020.