TALLINNA TEHNIKAÜLIKOOL

Infotehnoloogia teaduskond

Jana Kalatšova IAIB233381

Alina Jermoškina IAIB232724

Alika Boitšuk IAIB233569

Liiklusõnnetuste toimumise riskihindamine: andmeanalüüsil ja masinõppel põhinev prognoosimisteenus

Bakalaureusetöö

Juhendaja: Vahur Kotkas

M.Sc.

Tallinn 2026

Autorideklaratsioon

Kinnitame, et oleme koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autorid: Jana Kalatšova, Alina Jermoškina ja Alika Boitšuk

04.06.2026

Lühikokkuvõte

Käesoleva bakalaureusetöö eesmärk on luua andmeanalüüsil ja masinõppel põhinev reaalajas kasutamiseks liiklusõnnetuste riskihindamise valmis süsteem Eesti maanteede kontekstis. Töö keskendub ruumilis-ajalise liiklusriski prognoosimisele, kasutades kaasaegseid puupõhiseid ansambelalgoritme nagu LightGBM, Random Forest ja XGBoost. Mudelite treenimiseks kasutati Transpordiameti ajaloolisi liiklusõnnetuste andmeid ajavahemikust 2011–2026, mida agregeeriti 500×500 meetri suurustesse geograafilistesse ruutudesse. Mudelite jõudluse optimeerimiseks rakendati Optuna raamistikku, teostades hüperparameetrite tuunimist Bayesi optimeerimise meetodil. Arvestades andmete äärmuslikku hõredust ja loendusandmete olemust, kasutati peamise optimeerimiskriteeriumina keskmist absoluutset viga (MAE), kuna see võimaldab hinnata prognoositud ja tegeliku õnnetuste arvu keskmist erinevust ning on tulemuste interpreteerimiseks hästi sobiv mõõdik. Töö tulemusena valmis 16 erineva ruumilis-ajalise konfiguratsiooni võrdlev analüüs, millest saavutas parimad tulemused 500 m / 30 min jaotus. Lõplik lahendus realiseeriti Pythoni ja Flaski raamistikul põhineva REST API-na, mis võimaldab välistel süsteemidel pärida riskiskoore. Eksperimentaalsed tulemused näitasid, et süsteem on suuteline tuvastama mittelineaarseid seoseid teeolude, valgustuse ja ilmastiku vahel.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 37 leheküljel, 11 peatükki, 7 joonist, 3 tabelit.

Abstract

Risk Assessment of Traffic Accidents: A Forecasting Service Based on Data Analysis and Machine Learning

The objective of this Bachelor's thesis is to develop a data-driven and machine learning-based real-time traffic accident risk assessment system in the context of Estonian roads. The study focuses on predicting spatio-temporal traffic risks using advanced tree-based ensemble algorithms, specifically LightGBM, Random Forest, and XGBoost. The models were trained on historical traffic accident data from the Estonian Transport Administration covering 2011–2026, aggregated into 500×500 meter geographical grid cells. To optimize model performance, the Optuna framework was utilized for hyperparameter tuning via Bayesian optimization. Given the extreme sparsity of the data and its nature as count data, Mean Absolute Error (MAE) was employed as the primary optimization metric due to its interpretability and suitability for comparing prediction accuracy. The thesis presents a comparative analysis of 16 different spatio-temporal configurations, with the 500 m / 30 min distribution proving most effective. The final solution is implemented as a REST API based on Python and the Flask framework, enabling external systems to query dynamic risk scores. Experimental results demonstrated that the system is capable of identifying non-linear interactions between road conditions, lighting, and weather.

The thesis is in Estonian and contains 37 pages of text, 11 chapters, 7 figures, 3 tables.

Lühendite ja mõistete sõnastik

API	Rakendusliides (*Application Programming Interface*)
CART	Klassifitseerimis- ja regressioonipuud (*Classification and Regression Trees*)
CORS	Ristpäringute lubamine (*Cross-Origin Resource Sharing*)
CSV	Komaga eraldatud väärtuste failivorming (*Comma-Separated Values*)
EFB	Eksklusiivne tunnuste koondamine (*Exclusive Feature Bundling*)
GIS	Geoinfosüsteem (*Geographic Information System*)
GOSS	Gradiendil põhinev ühepoolne sämplimine (*Gradient-based One-Side Sampling*)
HTTP	Hüperteksti edastusprotokoll (*Hypertext Transfer Protocol*)
ISO	Rahvusvaheline Standardiorganisatsioon (*International Organization for Standardization*)
JSON	Andmevahetusvorming (*JavaScript Object Notation*)
L1 / L2	Regulariseerimismeetodid masinõppes (veafunktsiooni trahvid)
LightGBM	Kerge gradientvõimenduse masin (*Light Gradient Boosting Machine*)
MAE	Keskmine absoluutne viga (*Mean Absolute Error*)
PPA	Politsei- ja Piirivalveamet
REST	*Representational State Transfer* (arhitektuuristiil API-de jaoks)
RMSE	Ruutkeskmine viga (*Root Mean Squared Error*)
TPE	Puustruktuuriga Parzeni hindaja (*Tree-structured Parzen Estimator*)
XGBoost	Äärmuslik gradientvõimendus (*eXtreme Gradient Boosting*)

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

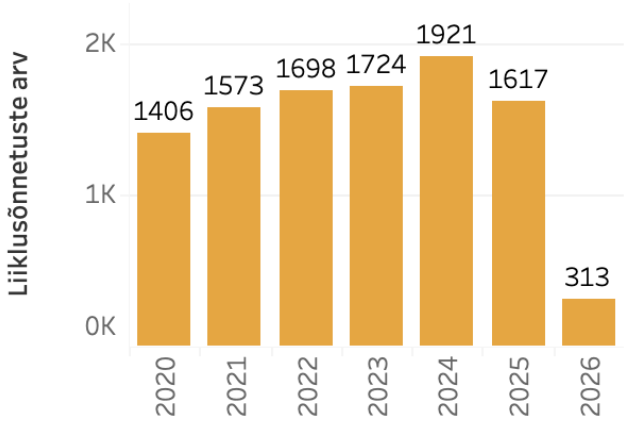
Liiklusohutuse tagamine on ühiskonna heaolu ja riigi majanduslikku stabiilsust oluliselt mõjutav valdkond. Eestis tekitavad liiklusõnnetused igal aastal märkimisväärset kahju, mis on tinginud suurenenud vajaduse uudsete, andmepõhiste ja proaktiivsete lahenduste järele. Traditsiooniliselt on liiklusohutuse uurimisel ja meetmete planeerimisel tuginetud reaktiivsele analüüsile, kus õnnetuste põhjuseid ja mustreid käsitletakse alles tagantjärele. Kuna aga elukeskkond ja liiklusolud on pidevas muutumises, luues nõudluse reaalajas hindavate süsteemide järele, ei piisa enam pelgalt ajaloolisele statistikale toetumisest.

Kaasaegsed andmetöötlus- ja masinõppemeetodid võimaldavad liikuda reaktiivselt lähenemiselt proaktiivsele prognoosimisele. Nende abil on võimalik luua mudeleid, mis arvestavad üheaegselt nii asukohaspetsiifilisi riskifaktoreid (teetüüp, ristmike tihedus) kui ka dünaamilisi keskkonnatingimusi (ilmastik, valgustus, kellaaeg). Käesolevas bakalaureusetöös kirjeldatakse andmeanalüüsil ja masinõppel põhineva ruumilis-ajalise liiklusõnnetuste riskihindamise mootori (API) disaini, arendust ja hindamist Eesti teede kontekstis. Antud lahendus integreerib ajaloolised liiklusõnnetuste andmed ja reaalajalised keskkonnaparameetrid, et hinnata õnnetuste toimumise suhtelist riski lokaalsel tasandil, jaotades vaadeldava ala ühtlaseks geograafiliseks ruudustikuks.

Käesoleva bakalaureusetöö ülejäänud struktuur on järgmine. Peatükk 2 kirjeldab uurimistöö teoreetilist tausta ja masinõppemudelite valiku põhimõtteid. Peatükk 3 annab ülevaate andmete kogumisest, puhastamisest ja ruumilis-ajalisest agregeerimisest. Peatükk 4 keskendub valitud puupõhiste ansambelalgoritmide (LightGBM, XGBoost, Random Forest) teoreetilisele võrdlusele. Peatükk 5 võtab kokku mudeli treenimisprotsessi. Peatükk 6 kirjeldab süsteemi arhitektuuri ja REST API arendust. Peatükk 7 esitab eksperimentide tulemused ja analüüsi. Lõpuks võtab Peatükk 8 töö tulemused kokku ja visandab suunad edasisteks arendusteks.

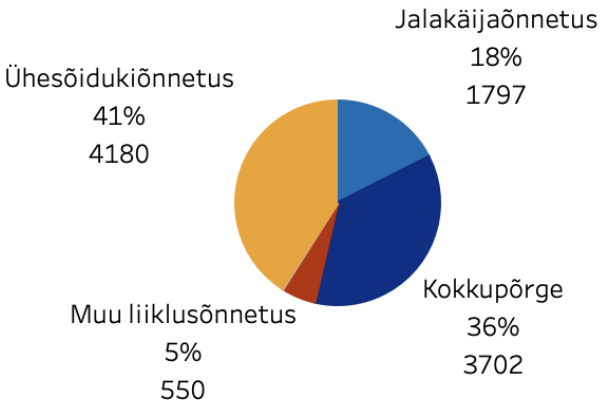
1.1 Motivatsioon

Hoolimata riiklikest ennetusmeetmetest püsib liiklusõnnetuste esinemissagedus Eestis märkimisväärselt kõrgena (vt Joonis 1). Transpordiameti statistika näitab, et hoolimata mõningatest kõikumistest püsib iga-aastaste intsidentide maht murettekitavalt suurena, mis rõhutab vajadust proaktiivsete lahenduste järele [1].



Joonis 1. Liiklusõnnetuste arv Eestis aastatel 2020–2026. [2]

Liiklusõnnetuste olemus on väga mitmetahuline, hõlmates erinevaid õnnetustüüpe, mis nõuavad erisugust analüütilist lähenemist. Nagu on näidatud Joonisel 2, moodustavad suurima osa õnnetustest ühesõidukiõnnetused (41%) ja kokkupõrked (36%), mis viitab selge vajaduse järele modelleerida erinevate keskkonnategurite ja teeolude koosmõju.



Joonis 2. Liiklusõnnetuste jaotus tüüpide lõikes. [2]

Traditsioonilised lähenemised, mis põhinevad peamiselt üldistatud lineaarsetel mudelitel või lihtsal tagasivaataval statistikal, ei suuda tuvastada neid kõrgemat järku mittelineaarseid

seoseid. Lisaks ei võta klassikalised mudelid tihti arvesse maakasutuse, taristu ja sotsiaal-majandusliku keskkonna koostoimet (näiteks töökohtade tihedus või logistikakeskuste lähedus), mis võivad oluliselt tõsta lokaalset riskitaset.

Kaasaegsed masinõppe meetodid, eriti puupõhised ansambelmeetodid nagu gradientvõimendus (Gradient Boosting), aitavad neid väljakutseid leevendada. Nad võimaldavad andmeid modelleerida paindlikult, integreerides tuhandeid andmepunkte ja heterogeenseid muutujaid (nii arvulisi kui ka kategoorilisi) ilma täpsust kaotamata. Sellised edasiarendused aitavad liikuda reageerivalt õnnetuste uurimiselt ennetavale, ruumilis-ajalisele analüüsile, kus masinõpe suudab tuvastada varjatud ohukoldeid varem, kui need eskaleeruvad traagilisteks sündmusteks.

1.2 Probleemi püstitus

Vaatamata masinõppe potentsiaalile liiklusohutuse valdkonnas, puudub Eestis hetkel laialdaselt kasutatav, andmepõhine ja reaalajas toimiv lahendus ruumilis-ajalise liiklusriski hindamiseks. Olemasolevad ametlikud riskinäitajad kipuvad olema staatilised, kirjeldades teatud ohutusmeetme keskmist mõju [1], suutmata seejuures dünaamiliselt kohanduda asukoha ja hetke spetsiifikaga (näiteks kuidas vihmasadu ja pimedus mõjutavad riski spetsiifilisel kurvilisel kruusateel võrreldes sirge maanteega).

See piirang on kriitilise tähtsusega nii liikluskorraldajatele kui ka kohalikele omavalitsustele. Ilma mudelita, mis suudaks hinnata nii ruumilisi aspekte (teetüüp, kurvilisus, ristmike tihedus) kui ka ajalis-keskkondlikke tegureid (ilmastik, kellaaeg), puuduvad planeerijatel vajalikud tööriistad näiteks digitaalse kaksiku simulatsioonide läbiviimiseks ja ennetusmeetmete proaktiivseks testimiseks.

Antud probleemi lahendamiseks võtab see bakalaureusetöö suuna ruumilis-ajalisele riskihindamisele, kus kogu Eesti territoorium on jaotatud ühtlaseks ruumiliseks võrgustikuks. Asemel et käsitleda ülesannet klassikalise binaarse klassifitseerimisena (kas õnnetus toimub või mitte), rakendatakse riskipõhist loendusmudelit, mis suudab tuvastada õnnetuste toimumise ajaloolise sageduse suhtelist riski.

1.3 Uurimistöö eesmärgid

Käesoleva bakalaureusetöö eesmärk on disainida, arendada ja hinnata Eesti liiklusandmete baasil masinõppel ning detailsel ruumilis-ajalisel diskretiseerimisel põhinevat reaalajas toimivat avatud API liiklusõnnetuste riskihindamise süsteemi. Loodav lahendus suudab dünaamiliselt reageerida muutuvatele ilmastiku- ja teetingimustele, pakkudes detailset tagasisidet liiklusriskide koondumise kohta REST API kujul. Peaeesmärgi saavutamiseks on seatud järgmised alaeesmärgid:

1. Luua andmetöötlus- ja integreerimisprotsess riiklikust registrist Politsei- ja Piirivalveamet (PPA) pärinevate heterogeensete andmete puhastamiseks, standardiseerimiseks ja ruumilis-ajaliseks agregeerimiseks.
2. Võrrelda kaasaegseid puupõhiseid ansambelalgoritme (LightGBM, XGBoost, Random Forest) ning valida sobivaim tehnoloogia, häälestades selle hüperparameetrid.
3. Ehitada REST API, mis võimaldab visuaalse kasutajaliidese kaudu pärida reaalajalisi riskiskoore ning demonstreerida loodud mudeli toimimist.

Nende eesmärkide põhjal otsitakse töös vastuseid järgmistele uurimisküsimustele:

1. Milline kaasaegne puupõhine masinõppe algoritm demonstreerib parimat prognoosimisvõimekust ja arvutuslikku efektiivsust Eesti liiklusõnnetuste ruumilis-ajalise riski modelleerimisel?
2. Kuidas mõjutavad omavahelised interaktsioonid (nt teekatte seisund, ilmastik ja kellaaeg) õnnetuste esinemise riskihindamist kindlas geograafilises ruudustikus?
3. Millised arhitektuursed põhimõtted on vajalikud masinõppemudeli serveerimiseks reaalajas toimiva REST API vahendusel?

2 Teoreetiline taust ja seotud uurimused

Liiklusohutuse valdkonna teadusuuringutes on viimase kümnendi jooksul täheldatud olulist metodoloogilist arengut. Varem domineerinud klassikalised statistilised lähenemisviisid on aeglaselt asendunud arenenud masinõppe tehnikatega. See üleminek tuleneb vajadusest prognoosida liiklusõnnetuste sagedust märkimisväärselt suurema täpsusega ning moel, mis võimaldab arvesse võtta andmetes peituvaid keerukaid struktuure. Akadeemilises kirjanduses on esile toodud, et pelgalt õnnetuste raskusastme analüüsist ei piisa efektiivsete ennetavate ohutusmeetmete kujundamiseks. Seetõttu tuleb keskenduda õnnetuste arvu prognoosimisele kindlates ruumilistes üksustes, näiteks teelõikudel või ruudustikes [3], [4].

2.1 Masinõppe meetodite eelistamine sageduse prognoosimisel

Liiklusõnnetuste sageduse modelleerimises on ajalooliselt rakendatud peamiselt üldistatud lineaarseid mudeleid. Kuid hiljutised uurimused on demonstreerinud nende lähenemiste piiranguid olukordades, kus sisendmuutujate vahel ilmnevad tugevad mittelineaarsed seosed [4], [5]. Akadeemilises kirjanduses on rõhutatud, et masinõppe algoritmid pakuvad sellistes tingimustes märkimisväärset eelist, kuna need on võimelised käsitlema segu-andmeid oluliselt paindlikumal viisil. Erilist edu on näidanud otsustuspuudel põhinevate ansambelmeetodite rakendamine, sealhulgas nii juhusliku metsa (*Random Forest*) kui ka gradient-tõhustatud (*Gradient Boosting*) algoritmid [3], [5].

Juhusliku metsa meetodit on liiklusohutuse uurimustes laialdaselt rakendatud selle iseloomuliku stabiilsuse ja võime tõttu hallata suuri andmemassiive, ilma et oleks vaja seada rangeid eeldusi andmete jaotuse kohta. Võrdlevad analüüsid on aga demonstreerinud, et kuigi juhuslik mets pakub märkimisväärset täpsuse paranemist võrreldes statistiliste algmudelitega, suudavad uuemad gradient-tõhustatud algoritmid, näiteks XGBoost ja LightGBM, saavutada veelgi väiksema prognoosivea taseme [3]. See täheldatud paremus põhineb sellel, et gradient-tõhustatud meetodid optimeerivad mudeli jõudlust astmeliselt, keskendudes igal iteratsioonil varasemate vigade parandamisele. Just niisugune lähenemine

sobib hästi, kui prognoosida õnnetuste sagedust keerulistes andmekogumites, kus andmed on segased või nullväärtusi palju [4], [5].

2.2 Maakasutuse ja taristu koostoime modelleerimine

Kaasaegsed uuringud näitavad, et õnnetuste sagedus ei sõltu üksnes tee tehnilistest näitajatest, vaid ka piirkonna sotsiaal-majanduslikust keskkonnast. On tuvastatud, et õnnetuste arv ruudustikus on korrelatsioonis töökohtade tiheduse, elamupiirkondade osakaalu ja jaekaubanduse kättesaadavusega [5]. Masinõppe algoritmid, eeskätt XGBoost ja juhuslik mets, on demonstreerinud suurt võimekust leida seoseid nende väliste faktorite ja liiklusõnnetuste vahel, mida klassikalised mudelid tihti ignoreerivad.

Näiteks on leitud, et teatud tüüpi äripindade kontsentratsioon või logistikakeskuste lähedus võib oluliselt tõsta õnnetuste sagedust, kuna see genereerib ebaühtlast liiklusvoogu ja sagedasi manöövreid [5]. Masinõppe abil on võimalik need seosed ruumiliselt kaardistada, pakkudes kohalikele omavalitsustele tööriistu ohutumate linnaplaneerimise otsuste tegemiseks. Eesti kontekstis, kus asustustihedus ja liikluskoormus varieeruvad piirkonniti suuresti, on selline integreeritud lähenemine vältimatu, et luua täpne ja ajakohane ülevaade riigi teevõrgu seisukorrast.

2.3 Järeldused analoogsete tööde uurimisest

Eespool kirjeldatud uuringud kinnitavad, et liiklusohutuse analüüsis on toimunud püsiv nihe andmepõhiste ja tõlgendatavate meetodite suunas. Käesolev bakalaureusetöö asetub sellesse metodoloogilisse raamistikku, rakendades maailmas tunnustatud mudeleid nagu juhuslik mets, XGBoost ja LightGBM Eesti liiklusandmete analüüsimiseks.

3 Andmete kogumine ja ettevalmistus

Käesolevas peatükis antakse põhjalik ülevaade uurimistöös kasutatud andmete kogumise, puhastamise ja masinõppemudelite jaoks ettevalmistamise protsessist. Kuna liiklusõnnetuste algandmed pärinesid mitmest erinevast allikast ning omasid heterogeenset struktuuri, on esimeseks oluliseks sammuks andmete standardiseerimine ja puhastamine, sealhulgas puuduvate väärtuste, vigaste kirjete ja duplikaatide käsitlemine.

Erinevalt klassikalisest binaarsest klassifitseerimisest lahendatakse käesolevas töös riskipõhist loendusülesannet. Selle raamistiku loomiseks kirjeldatakse andmete ruumilis-ajalist agregeerimist, mille käigus jaotati vaatlused kindla suurusega geograafilistesse ruutudesse ja tunnipõhistesse ajavahemikesse.

Lisaks andmete struktureerimisele avab peatükk tunnuste inseneeria etapi, mille käigus tuletati algandmetest uusi aja-, ruumi- ja keskkonnapõhiseid muutujaid (sealhulgas aja tsüklilisust ja teeolude keerukust kirjeldavad tunnused), ning selgitab andmelekke (*data leakage*) vältimise olulisust tunnuste eemaldamisel. Lõpetuseks esitatakse andmestiku realistlik ajapõhine jaotamine treening-, valideerimis- ja testandmeteks ning tuuakse välja andmetöötluse tulemusena valminud lõplike andmestike statistilised näitajad ja jaotused, mis loovad aluse edasisele modelleerimisele.

3.1 Andmeallikate kirjeldus

Käesolevas töös kasutatavad andmed pärinevad ühest koondatud andmefailist `lo_2011_-2026.csv`, mis sisaldab Eesti liiklusõnnetuste registreeritud kirjeid ajavahemikust 1. jaanuarist 2011 kuni 29. aprillini 2026. Andmete allikaks on Transpordiameti avaldatud inimkannatanutega liiklusõnnetuste andmestik, mis on kättesaadav Eesti avaandmete portaalis (`andmed.eesti.ee`) ning selle andmestiku baasil olev statistika Transpordiameti kodulehel (`transpordiamet.ee/liiklusonnetuste-statistika`).

Fail sisaldab 22 996 kirjet ja 54 andmeveergu. Iga kirje kirjeldab ühe liiklusõnnetuse

toimumise aega, asukohta ning sündmuse konteksti: ilmastikuolusid, valgustustingimusi, teekatte seisundit, tee tüüpi, kiirusepiirangut, sõiduradade arvu ning muid teeoludega seotud tunnuseid. Asukohainfo on esitatud katastripõhiste x- ja y-koordinaatidena.

Liiklusõnnetuste riski hindamiseks sobib see andmestik hästi, kuna hõlmab pikka ajavahemikku ning sisaldab nii ruumilist (asukoht) kui ka ajalist (toimumisaeg) infot, aga ka tingimusi kirjeldavaid tunnuseid, mille mõju liiklusohutusele on hästi dokumenteeritud. Oluline on silmas pidada, et andmestik sisaldab ainult registreeritud õnnetusi – seega võimaldab analüüs hinnata suhtelist riski, kuid ei anna alust absoluutsete esinemistõenäosuste arvutamiseks.

3.2 Andmetöötluse tehniline osa

Andmete puhastamiseks, modelleerimiseks ja API serveerimiseks rakendati järgmisi Pythoni pakette, mille roll on süsteemi terviklikkuse tagamisel kriitiline:

- **Scikit-learn**: Rakendati mitte ainult hindamismetoodikate, vaid ka andmete eeltöötluse konveierite loomiseks. Kasutati moodulit `sklearn.model_selection` andmete kronoloogiliseks jagamiseks ning `sklearn.metrics` paketti spetsiifiliste regressioonimõõdikute (MAE, RMSE, Poisson Deviance) arvutamiseks [6].
- **Flask**: Veebiraamistiku roll on hallata REST-päringuid. Süsteemis realiseeriti Flaski marsruutimine (*routing*) kasutades dekoraatorit `@app.route`, mis suunab sissetulevad JSON-andmed mudeli ennustusfunktsiooni. Flaski `jsonify` funktsioon tagab vastuste korrektse serialiseerimise kliendile sobivasse HTTP-vormingusse [7].
- **numpy.searchsorted**: Seda meetodit kasutati riskiskooride pertsentiilseks kaardistamiseks (percentile mapping). Matemaatiliselt teostab algoritm binaarset otsingut, et leida sobiv indeks sorteeritud massiivis, mis võimaldab reaalajas teisendada mudeli toorväljundi (raw score) suhteliseks riskiskooriks (0–100%) minimaalse ajakuluga [8].
- **LightGBM ja XGBoost**: Need teegid valiti nende sisseehitatud toe tõttu Poissoni regressioonile, mis sobib hästi loendusandmete modelleerimiseks. Kuigi algoritmid kasutavad sisemiselt Poissoni kadufunktsiooni, keskenduti käesolevas töös mudelite hindamisel ja võrdlemisel keskmisele absoluutsele veale (MAE). See valik võimaldab tulemusi paremini tõlgendada, näidates otseselt prognoositud ja tegeliku õnnetuste

arvu vahelist keskmist erinevust ruutkeskmise vea ees, mis on tundlikum üksikutele erinditele [9], [10].

- **joblib.load**: Kasutatakse treenitud `XGBoost` mudeliobjektide ja `StandardScaler` skaleerijate binaarseks salvestamiseks ja kiireks laadimiseks API käivitamisel, tagades minimaalse viiteaja päringute töötlemisel [11].
- **Optuna**: Automaatne hüperparameetrite optimeerimise raamistik, mis kasutab *Tree-structured Parzen Estimator* (TPE) algoritmi [12]. See võimaldas süstemaatiliselt läbi otsida parameetriruumi, et minimeerida keskmist absoluutset viga (MAE), mis valiti mudelite peamiseks optimeerimis- ja võrdluskriteeriumiks.

3.3 Toorandmete laadimine ja duplikaatide eemaldamine

Andmetöötluse esimene samm on toorandmete laadimine. Süsteem loeb automaatselt kõik CSV-failid selleks ettenähtud sisendkaustast. Antud juhul sisaldas sisendkaust ühte faili – `lo_2011_2026.csv` –, mis loeti sisse 22 996 rea ja 54 veeruga. Faili laadimisel tuvastati ja eemaldati 5 duplikaatrida. Pärast duplikaatide eemaldamist moodustus puhastatud baasandmestik, mis sisaldab 22 991 rida ja 60 tunnust ning katab ajavahemiku 1. jaanuarist 2011 kuni 29. aprillini 2026.

Duplikaatide tuvastamisel kasutatakse juhtumi identifikaatorit. Kui sama identifikaator esineb mitmel real, säilitatakse kirje, millel on rohkem täidetud andmevälju. Kui juhtumi identifikaator on korduvate kirjete puhul ebaunikaalne, muudetakse korduvad identifikaatorid unikaalseks numbrilise järelliite lisamisega. Ühtse baasandmestiku loomine on edasise töötluse eeldus: ainult ühe puhastatud andmestiku alusel on võimalik rakendada ühtset modelleerimisloogikat ning tagada, et iga liiklusõnnetus esineb analüüsis täpselt üks kord.

3.4 Andmete puhastamine ja veergude ühtlustamine

Andmete puhastamine ja standardiseerimine on masinõppeprotsessi üks kriitilisemaid etappe, kuna mudeli prognoosivõime ja stabiilsus sõltuvad otseselt sisendandmete kvaliteedist. Toorandmetes esinev müra, puuduvad väärtused ja ebaühtlane vormistus võivad põhjustada vigu nii mudeli treenimisel kui ka hilisemas reaalajas rakendamises.

Kuna käesolevas töös kasutatavad liiklusõnnetuste andmed pärinevad riiklikust registrist,

kus on aja jooksul muudetud andmete sisestamise viise ja terminoloogiat, oli vajalik teostada põhjalik andmete ühtlustamine. See protsess jagunes kolmeks peamiseks tegevuseks: tehniliste veerunimede standardiseerimine, kategooriliste väärtuste rühmitamine loogilistesse gruppidesse ning arvuliste tunnuste kontroll ja normaliseerimine. Järgnevad alapeatükid kirjeldavad üksikasjalikult neid etappe, mis muutsid heterogeensed toorandmed ühtseks ja masinloetavaks andmestikuks.

3.4.1 Veerunimede ühtlustamine

Kuna algandmete veerunimed on eestikeelsed ning eri allikates võivad need erineda, teisendatakse kõik veerunimed ühtsesse ingliskeelsesse formaati. Teisendus toimub eelnevalt määratletud sõnastiku alusel, mis seob eestikeelsed algnimed standardiseeritud ingliskeelsete nimedega. Veergude nimed normaliseeritakse enne otsingut väiketähtedeks ning eemaldatakse üleliigsed tühikud, mis tagab ühtse töötluse sõltumata algse faili täpsest kirjaviisist.

3.4.2 Kategooriliste väärtuste ühtlustamine

Lisaks veerunimedele ühtlustatakse kategooriliste tunnuste väärtused. Ühes andmefailis võib sama nähtus – näiteks vihmasadu – olla kirjutatud eri viisil (nt Vihm, vihm, Rain jne). Kõik sellised variatsioonid teisendatakse ühtsele kujule ning seostatakse laiema kategooriaga. Ilmastikuolud koondatakse tunnusesse `weather_group`, valgustustingimused tunnusesse `lighting_group` ning teekatte seisund tunnusesse `surface_condition_group`. Tekstilised puuduvad väärtused – näiteks Teadmata, Pole teada ning tühjad väljad – teisendatakse ühtsele puuduvate väärtuste kujule, mida saab töötluse käigus filtreerimiseks kasutada.

3.4.3 Arvuliste väärtuste normaliseerimine

Arvuliste tunnuste puhul viidi läbi formaadi ja väärtuste kontroll. Kuna algandmetes võisid arvud esineda erinevates vormingutes – näiteks kümnendkohana komaga eraldatult –, asendati kümnendkomad punktidega ning eemaldati tarbetud tühikud. Kiirusepiirangute puhul rakendati lubatud väärtuste hulk: ebarealistlikud väärtused nagu 0, 999 või 901 teisendati puuduvaks väärtuseks. Sõiduradade arvu puhul teisendati kombineeritud kujul kirjutatud väärtused (nt 1+1 või 2+2) üksikuks arvuliseks kogusummaks.

Andmete puhastamine on vajalik, kuna masinõppe mudelid on tundlikud andmekvaliteedi suhtes: valed, ebaühtlased või mitmeti tõlgendatavad väärtused võivad mudelisse lisada müra ning vähendada riskihinnangute usaldusväärsust.

3.5 Riskipõhise andmestiku konstrueerimine

Käesolevas töös ei kasutata klassikalist lähenemist, kus andmestik sisaldaks nii positiivseid (õnnetus toimus) kui ka negatiivseid (õnnetust ei toimunud) näiteid. Algandmestik koosneb ainult reaalselt registreeritud liiklusõnnetustest, mistõttu puuduvad usaldusväärsed kirjed olukordade kohta, kus õnnetust ei toimunud.

Negatiivsete näidete kunstlik loomine eeldaks tugevaid eeldusi – näiteks et kõik kohad ja ajad, kus õnnetust ei registreeritud, olid ohutud. Sellised eeldused tooksid mudelisse lisaviga, kuna liikluskoormus ja registreerimise täielikkus ei ole kõikjal ühesugused. Seetõttu käsitletakse probleemi riskipõhise loendusülesandena: mudeli eesmärk ei ole ennustada, kas üksik õnnetus toimub või mitte, vaid hinnata, millistes ruumilistes, ajalistes ja keskkondlikes tingimustes on õnnetuste esinemine ajalooliselt sagedasem.

3.6 Ruumiliste tunnuste töötlemine ja ruudustiku loomine

Ruumilise info töötlemisel kasutatakse koordinaate, mis on iga liiklusõnnetuse kohta talletatud katastripõhiste x- ja y-koordinaatidena (`x_coord`, `y_coord`). Tegelikus töötluses oli 21 780 rea puhul koordinaadipaar olemas. Ülejäänud 1 211 rida, millel koordinaadid puudusid, eemaldati edasise töötluse käigus.

Koordinaatide põhjal jaotatakse uuritav ala geograafiliseks ruudustikuks, kasutades fikseeritud küljepikkusega ruute. Sobivaima ruumilise täpsuse ja mudeli üldistusvõime tasakaalu leidmiseks katsetati uurimistöös nelja erinevat ruudu suurust: 500, 1000, 1500 ja 2000 meetrit.

Igale ruudule omistatakse unikaalne tunnus `grid_id`, mis koosneb ruudu horisontaalsest ja vertikaalsest indeksist. Ruumilise ruudustiku genereerimiseks ei kasutatud arvutuslikult kulukaid geoinfosüsteemide (GIS) teeke, vaid rakendati efektiivset matemaatilist koordinaatide diskretiseerimist. Kuna Transpordiameti andmetes on asukohad esitatud Eesti riiklikus tasapinnalises ristkoordinaatide süsteemis L-EST97 (EPSG:3301), kus ühikuks

on meeter, saavutatakse ruudustik koordinaatide jagamisel valitud sammuga 500 meetrit (500, 1000, 1500, 2000 meetrit). Jagatise täisosa abil moodustati igale vaatlusele Pandas teegi vektoriseeritud operatsioonide abil unikaalne horisontaalse ja vertikaalse indeksi kombinatsioon (`grid_id`). Selline lähenemine kiirendas oluliselt andmetöötluskonveieri tööd ning võimaldas hallata suuri andmemahte otse Pythoni põhiliste andmeanalüüsi tööriistadega (Pandas, NumPy).

Ruudustikuks jagamine on vajalik, kuna üksikute koordinaatide täpsusel modelleerimine oleks ebapraktiline: iga koordinaatpunkt oleks ainulaadne ning mudelil ei oleks võimalik piirkondlikke mustreid tuvastada. Ruudustik võimaldab koondada lähestikku toimunud õnnetused üheks vaatluseks ning teha ruumilisi võrdlusi sarnaste alade vahel. Täiendavalt luuakse iga ruudustiku lahtri kohta staatiline profiil, mis kirjeldab selles piirkonnas domineerivaid teeolusid – tee tüüpi, tee elementi, kiirusepiirangut ja sõiduradade arvu –, kasutades kõigi selle lahtri õnnetuste andmete sagedaimaid väärtusi. Seda profiili kasutatakse hiljem puuduvate teeomaduste täitmiseks agregeeritud andmestikus.

Uuritavate ruudusuuruste vahemik (500 kuni 2000 meetrit) valiti metodoloogiliste kaalutluste põhjal. Kuigi teoreetiliselt võimaldaks veelgi väiksem ruudustik (nt 100 või 250 meetrit) täpsemat ohtlike kohtade tuvastamist, muudaks see andmestiku ekstreemselt hõredaks, kus valdav osa ruutudest jääks tühjaks (null-õnnetustega). See tekitaks mudeli treenimisel tõsiseid statistilise olulisuse probleeme ja suurendaks „müra“ osakaalu. Suuremad ruudud (nt 2000 m) vähendavad andmete hõredust, kuid võivad riskitaset liigselt keskmistada ja hajutada lokaalsed ohukolded liiga suurele alale. Erinevate ruudusuuruste mõju mudeli prognoositäpsusele on analüüsitud ja valideeritud peatükis 5.1 ning tulemused on esitatud alapeatükis 7.1.

3.7 Ajaliste tunnuste töötlemine ja ajavahemike määramine

Igale liiklusõnnetusele omistatakse lisaks ruumilistele koordinaatidele ka ajaline üksus ehk fikseeritud pikkusega ajavahemik (ajaaken). Ajavahemiku määramine tähendab seda, et õnnetuse täpne toimumisaeg ümardatakse allapoole ajaakna alguseni. Näiteks kui ajaakna pikkuseks on valitud 30 minutit, siis kell 14:47 toimunud õnnetus omistatakse ajavahemikule, mis algab kell 14:30. Nii on igal kirjel nii ruumiline (`grid_id`) kui ka ajaline (ajavahemik) tunnus, mille kombinatsioon määratleb üheselt ühe vaatlusüksuse.

Optimaalse ajadünaamika ja mudeli tundlikkuse tuvastamiseks katsetati töös nelja erinevat ajaakna pikkust: 30, 60, 90 ja 120 minutit. Lühemad ajavahemikud (nt 30 minutit) suurendavad ajalist täpsust ja võimaldavad paremini modelleerida operatiivseid keskkonnamuutusi (nt kiiresti muutuvaid ilmastiku- ja valgustingimusi), kuid tekitavad hõredama andmestiku. Pikemad ajavahemikud (nt 120 minutit) siluvad ajamustreid ja vähendavad andmete hõredust, kuid võivad varjata kriitilisi tippsündmusi (nt lühiajalisi tipptunde).

Ajaline info on liiklusõnnetuste riski hindamisel kriitilise tähtsusega, kuna liiklussagedus ja keskkonnatingimused varieeruvad tugevalt eri kellaaegadel, nädalapäevadel ja aastaaegadel. Tipptundidel, pimedal ajal ja talvistes tingimustes on ajalooliselt rohkem õnnetusi registreeritud, mistõttu annab täpne ajaline diskretiseerimine mudelile olulist lisateavet.

Nimetatud ruumiliste sammude (500–2000 m) ja ajaliste akende (30–120 min) ristamisel genereeriti andmete ettevalmistamise etapis kokku 16 erinevat eksperimentaalset andmestiku konfiguratsiooni. Nende andmestike võrdlev analüüs ja lõpliku, parima tasakaaluga mudelikonfiguratsiooni valik on esitatud valideerimistulemuste peatükis (vt alapeatükk 7.1 ja Tabel 1 leheküljel 42).

3.8 Tingimuste filtreerimine

Enne ruumilis-ajalist agregeerimist rakendatakse andmestikule tingimuste filter, millega eemaldatakse read, kus puuduvad peamiste dünaamiliste keskkonnatingimuste väärtused. Nõutavad tunnused on ilmastik, valgustus ja teekatte seisund. Filtri rakendamisel jäi 21 780 koordinaatidega realt alles 20 166 rida ehk 92,6 % sisendist. Eemaldati seega 7,4% ridadest, millel puudus vähemalt ühe kohustusliku tingimustunnuse väärtus. See lähenemine tagab, et kõigis mudeli sisendridades on keskkonnatingimused täielikult kirjeldatud.

3.9 Ruumilis-ajaline agregeerimine

Pärast filtreerimist teisendatakse individuaalsete õnnetuste kirjed ruumilis-ajalisteks vaatlusteks. Iga vaatlus kujutab üht konkreetset ajavahemikku konkreetses ruudustiku lahtris. Sihtmuutujaks (inglise keeles *target*) saab selles lahtris ja ajavahemikus toimunud õnnetuste arv, mis on alati vähemalt 1, kuna agregeerimisse kaasatakse ainult lahtrid ja ajavahemikud, kus õnnetusi tegelikult esines.

Dünaamilised keskkonnatingimused – ilmastik, valgustus, teekatte seisund ja liiklustiheduse grupp – koondatakse sagedaima väärtuse alusel kõikide samas lahtris ja ajavahemikus toimunud õnnetuste põhjal. Teeomadused (tee tüüp, tee klass, element, kiirusepiirang jms), mis on pigem staatilised ehk piirkonniti sarnased, täidetakse eelnevalt koostatud ruuduprofiili põhjal, kus on talletatud iga ruudustiku lahtri tüüpilised teeolude väärtused.

Agregeerimise tulemusena saadud andmestikus esindab iga rida üht ruumilis-ajalist üksust kujul (`grid_id`, ajavahemik). 500-meetrise ruudustiku ja 30-minutilise ajavahemiku kombinatsiooni korral vähenes 20 166 üksiku õnnetuse kirjet 20 159 agregeeritud vaatluseks, mis hõlmab 6 906 unikaalset ruumilist lahtrit. Sihtmuutuja *target* on kõigil ridadel vähemalt 1.

3.10 Tunnuste loomine

Pärast andmete puhastamist ja agregeerimist luuakse täiendavad tunnused, et masinõppe mudel saaks kasutada algandmetes peituvat informatsiooni struktureeritumal kujul. Tunnuste loomine (inglise keeles *feature engineering*) on vajalik, kuna mudelid ei suuda otse toorest kuupäevast või koordinaadist kasulikke mustreid tuvastada – selleks on vaja tuletada lisatähendusega arvulisi ja kategoorilisi tunnuseid.

3.10.1 Ajapõhised tunnused

Ajavahemiku põhjal tuletatakse mitu ajalist tunnust. Põhitunnustena luuakse tund (`hour`), nädalapäev (`weekday`) ja kuu (`month`). Lisaks luuakse mitmeid abistunnuseid: tipptunni indikaator, mis märgib hommikused ja õhtused liiklustiheduse tipptunnid (7:00–9:00 ja 16:00–18:00); nädalavahetuse ja tööpäeva indikaatorid; päevaosa tunnused (öö, hommik, pärastlõuna, õhtu); ning aastaaja tunnused (talv, kevad, suvi, sügis).

Kuna aeg on tsükliline nähtus – näiteks kell 23:59 ja kell 00:01 on ajaliselt lähestikku, kuigi tavalise arvulise esitusena asuvad need skaala eri otstes –, luuakse tunnile, nädalapäevale ja kuule ka tsüklilised kujutised siinus- ja koosinusteisenduse abil. Näiteks tunnused `hour_sin` ja `hour_cos` võimaldavad mudelil mõista, et hilisõhtu ja varahommik on ajaliselt lähedal. Selline teisendus on eriti oluline puupõhiste mudelite puhul, mis ei suuda tsüklilisust automaatselt tuvastada [13].

3.10.2 Teeomaduste tunnused

Teeomaduste kirjeldamiseks kasutatakse nii algandmetest pärinevaid tunnuseid kui ka nendest tuletatud üldistatud kategooriaid. Tee tüüp, tee element, tee objekt, kurvilisus, kalle ja teekatte tüüp koondatakse laiematesse rühmadesse, et vähendada liiga detailsete kategooriate arvu. Lisaks luuakse binaarsed indikaatorid, mis märgivad, kas tegemist on ristmiku, kurvilise lõigu, kõrgema kiirusepiiranguga tee või mitmerealise teega. Koondtunnus `road_complexity_score` kirjeldab teeolude keerukust, kombineerides mitmeid teeomadusi üheks heuristiliseks skooriks.

3.11 Kronoloogiline andmestiku jagamine

Mudeli hindamiseks jaotatakse andmestik kolmeks osaks: treenimis-, valideerimis- ja testikogumiks. Jagamine toimub rangelt kronoloogilises järjekorras: varasemad andmed kuuluvad treenimiskogumisse, millele järgnevad valideerimis- ja testimiskogumid. Selline lähenemine on kriitilise tähtsusega, et vältida ajalist andmeleket – olukorda, kus mudel saaks treenimise käigus teavet tuleviku sündmuste kohta, mis muudaks hindamistulemused ebareaalselt optimistlikuks.

Jaotuse proportsioon (70% treenimiseks, 15% valideerimiseks ja 15% testimiseks) valiti eesmärgiga tagada mudelile piisav ajalooline andmemaht õppimiseks, jättes samas piisavalt andmeid valideerimiseks (hüperparameetrite häälestamiseks) ning sõltumatuks testimiseks, et kontrollida mudeli võimekust prognoosida tuleviku sündmusi andmetel, mida mudel ei ole varem näinud.

500-meetrise ruudustiku ja 30-minutilise ajavahemiku andmestikus jagunesid vaatlused järgmiselt: treenimiskogum 14 109 rida, valideerimiskogum 3 043 rida ja testikogum 3 007 rida.

3.12 Andmetöötluse väljund

Andmetöötluse tulemusena salvestatakse iga eksperimentaalse konfiguratsiooni kohta kaks peamist andmestikku. Esimene, `risk_dataset.csv`, sisaldab kõiki ruumilis-ajalisi vaatlusi koos sihtmuutujaga, kirjeldavate tunnustega ja andmestiku jagamise sildiga. Teine, `model_dataset.csv`, on kompaktsem versioon, millest on eemaldatud tunnused, mida

mudel treenimise käigus otseselt ei kasuta – näiteks koordinaadid, ajatempli identifikaatorid ja duplikaatveerud. Lisaks nendele salvestatakse iga ruudulahtri staatilisi teeolusid kirjeldav ruuduprofiil, mida kasutatakse puuduvate teeomaduste täitmiseks.

Kokku genereeriti 16 eksperimentaalset andmestiku varianti, mis hõlmavad kõigi nelja ruudustiku suuruse (500, 1000, 1500 ja 2000 meetrit) ja nelja ajavahemiku pikkuse (30, 60, 90 ja 120 minutit) kombinatsiooni.

Lõplikuks konfiguratsiooniks valiti 500-meetrine ruudustik ja 30-minutiline ajavahemik, mis osutusid eksperimentide käigus parimaks tasakaaluks ruumilise detailsuse ja statistilise esinduslikkuse vahel (põhjalikum analüüs on esitatud peatükis 5.1). Vastava andmestiku maht on 20 159 rida. Algandmestiku 22 996 realt jõuti selleni järgmiste etappide kaudu: esmalt eemaldati 5 duplikaatrida (jäi 22 991 rida), seejärel 1 211 rida, millel puudusid koordinaadid (jäi 21 780 rida), seejärel 1 614 rida, millel puudusid kohustuslikud keskkonnatingimused (jäi 20 166 rida), ning lõpuks vähendas ruumilis-ajaline agregeerimine individuaalsed kirjed 20 159 vaatluseks. Need andmestikud edastatakse modelleerimise etappi, kus neid kasutatakse masinõppe mudelite treenimiseks, valideerimiseks ja testimiseks.

4 Masinõppe algoritmide võrdlus

Käesolevas peatükis kirjeldatakse uurimistöös kasutatavate masinõppe algoritmide teoreetilist tausta ja nende sobivust liiklusõnnetuste riski hindamiseks. Masinõppe rakendamine liiklusõnnetuste riskihindamise süsteemi südames on kriitiline, kuna traditsioonilised statistilised mudelid ei suuda sageli piisava täpsusega tabada dünaamilisi ja mittelineaarseid seoseid teeolude, ilmastiku ja ajaliste faktorite vahel.

4.1 Teoreetiline raamistik ja seos ennetusprotsessiga

Liiklusohutuse kontekstis ei ole mudeli eesmärk ainult ennustada sündmuse toimumist, vaid pakkuda sisendit ennetusmeetmete planeerimiseks. Puupõhised ansambelmeetodid (ingl *tree-based ensemble models*) on osutunud tabelandmete puhul kõige tõhusamaks, kuna need suudavad automaatselt tuvastada tunnustevahelisi interaktsioone ilma andmete eelneva keerulise teisendamiseta [10].

Riskihindamise mootori jaoks on oluline mudeli võime genereerida *tunnuste olulisuse* (ingl *feature importance*) näitajaid. See võimaldab analüüsida, millised keskkonnategurid panustavad enim riskitaseme tõusu konkreetses geograafilises ruudustikus, pakkudes seeläbi otsustustuge liikluskorraldajatele. Kuna liiklusõnnetuste andmestik on loomupäraselt ebaühtlase jaotusega (õnnetusi toimub harva võrreldes tavapärase liiklusega), on valitud algoritmid suutelised töötama riskipõhiste loendusmudelitega, keskendudes suhtelise ohu määramisele.

4.2 Algoritmide kirjeldused

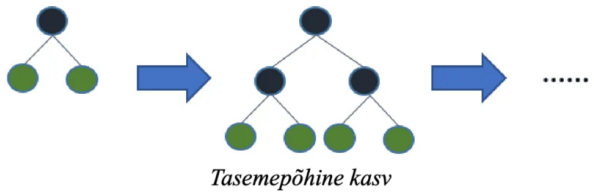
Selles alajaotuses käsitletakse kolme peamist algoritmi, mida kaaluti liiklusriski modelleerimiseks: LightGBM, XGBoost ja Random Forest. Iga algoritmi puhul on välja toodud selle tehnilised eelised ja sobivus antud ülesande lahendamiseks.

4.2.1 LightGBM (Light Gradient Boosting Machine)

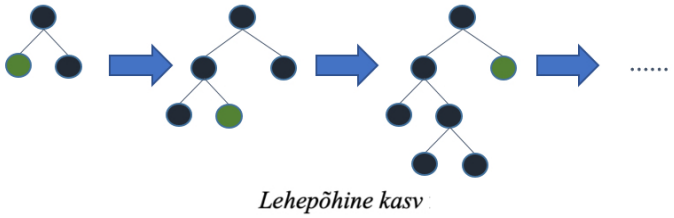
LightGBM on gradiendi võimendamise raamistik, mis kasutab puude ehitamisel lehepõhist (*leaf-wise*) kasvustrateegiat. See võimaldab mudelil saavutada suuremat täpsust ja kiiremat koondumist, kuna optimeeritakse sõlmi, mis vähendavad kadufunktsiooni viga kõige rohkem. Algoritmi muudavad eriti efektiivseks kaks tehnoloogiat: GOSS (*Gradient-based One-Side Sampling*) ja EFB (*Exclusive Feature Bundling*) [9].

GOSS aitab hallata suuri andmemahte, jättes alles suurema gradiendiga (keerulisemad) andmepunktid ja sämplides juhuslikult väiksema gradiendiga punkte, mis säilitab mudeli täpsuse, kuid vähendab arvutuskoormust. EFB koondab harvad ja omavahel välistavad tunnused ühte kimpudesse, mis on eriti kasulik liiklusandmete puhul, kus esineb palju kategoorilisi väärtusi. See arhitektuur toetab otseselt kategooriliste tunnuste töötlemist, vältides vajadust mahuka *one-hot* kodeerimise järele [9].

Joonisel 3 on illustreeritud erinevus traditsioonilise tasemepõhise ja LightGBM-i poolt kasutatava lehepõhise kasvustrateegia vahel. Graafilised selgitused tuginevad LightGBM-i ametlikule dokumentatsioonile, mis selgitab lehepõhise lähenemise eelist optimaalsemate sõlmede valimisel ja arvutusliku efektiivsuse saavutamisel.



(a) Tasemepõhine kasv (Level-wise) [14]



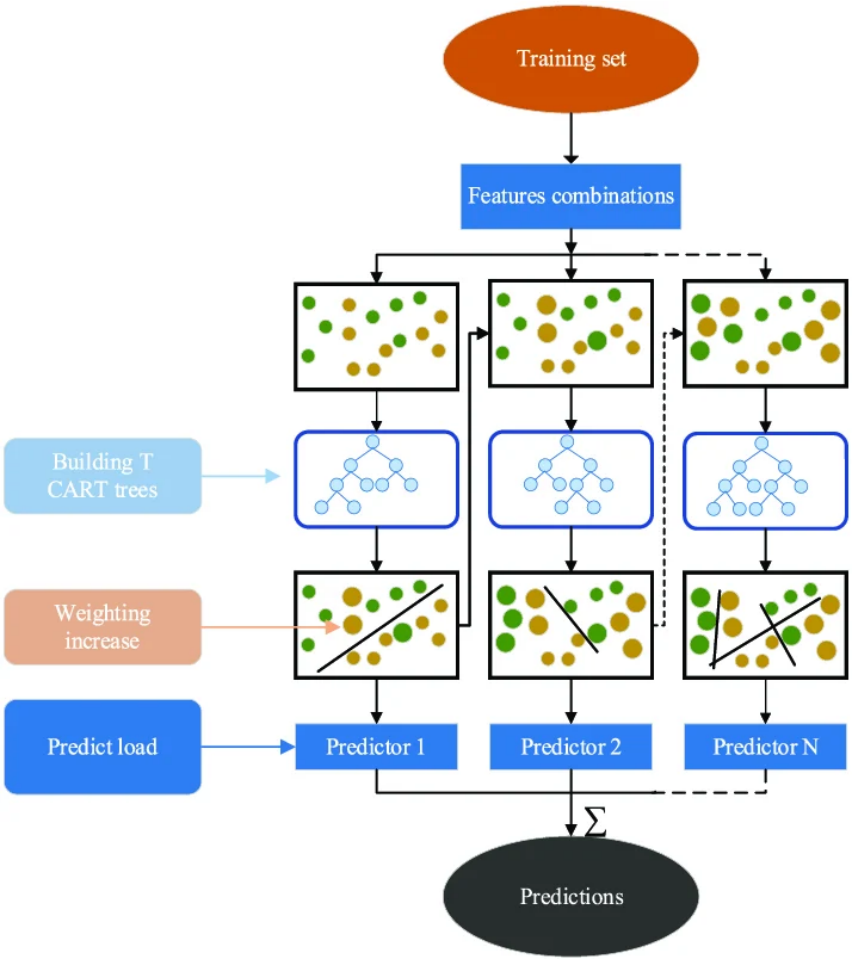
(b) Lehepõhine kasv (Leaf-wise) [14]

Joonis 3. Otsustuspuude kasvustrateegiate võrdlus.

4.2.2 XGBoost (eXtreme Gradient Boosting)

XGBoost on laialdaselt tunnustatud gradiendi võimendamise algoritm, mis on optimeeritud jõudlusele ja skaleeritavusele. Selle peamine eelis seisneb arenenud regulariseerimismeetodite (L1 ja L2) kasutamises, mis aitavad vältida ülesobitamist keerulistes andmestikes [10]. Algoritm rakendab aditiivset strateegiat, kus iga uus otsustuspuu parandab eelmiste puude poolt tehtud vigu.

Joonisel 4 on kujutatud XGBoosti sekventsiaalne õppimisprotsess. Skeem selgitab, kuidas treeningandmete põhjal ehitatakse järjestikku mitu klassifitseerimis- ja regressioonipuud (CART). Iga järgnev samm analüüsib eelmiste mudelite vigu (mida sümboliseerib andmepunktide kaalude muutmine), et uue puu lisamisega vähendada üldist prognoosiviga. Lõplik tulemus saadakse kõigi individuaalsete prognooside summeerimisel.



Joonis 4. XGBoosti sekventsiaalne õppimisprotsess ja vigade korrigeerimine puude lisamise kaudu [15].

Liiklusõnnetuste modelleerimise seisukohalt on XGBoosti oluline omadus selle suutlikkus käsitleda hõredaid andmeid (*sparsity-aware split finding*). Kuna reaalsetes liiklusandmetes

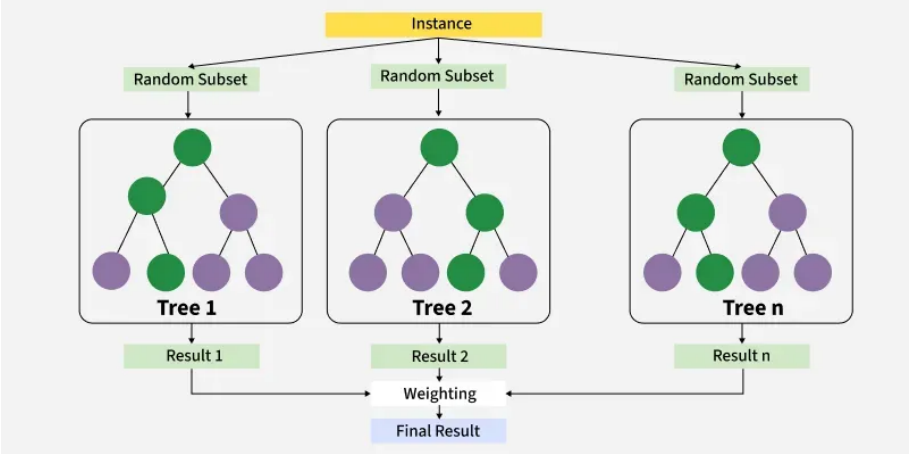
võib esineda puuduvaid väärtusi, õpib XGBoost automaatselt parima suuna (*default direction*) puuduvate väärtustega kirjete jaoks.

4.2.3 Random Forest (Juhuslik mets)

Random Forest on klassikaline ansambelmeetod, mis põhineb kottimise (*bagging*) tehnikal. Erinevalt võimendamisalgoritmidest ehitab Random Forest suure hulga sõltumatuid otsustuspuid paralleelselt, kasutades andmete juhuslikku valimit koos tagasipanekuga (*bootstrap sampling*). Lõplik prognoos saadakse kõikide puude tulemuste keskmistamise teel, mis vähendab oluliselt mudeli variatiivsust ja tundlikkust müra suhtes [16].

Algoritmi stabiilsus tuleneb asjaolust, et igas puu sõlmes valitakse jagamiseks vaid juhuslik alamosa tunnustest (*feature subsampling*). See tagab, et puud on omavahel vähe korreleeritud, muutes mudeli vastupidavaks üksikute vigaste andmepunktide suhtes. Liiklusriski hindamisel toimib Random Forest suurepärase baasmudelina, kuna see on vähem tundlik hüperparameetrite häälestamisele ning suudab stabiilselt hallata nii arvulisi kui ka kategoorilisi tunnuseid ilma rangeid eeldusi andmete jaotuse kohta seadmata.

Joonisel 5 on kujutatud juhusliku metsa algoritmi tööpõhimõte, mis põhineb kottimise (bagging) meetodil. Skeem illustreerib, kuidas algandmetest moodustatakse juhuslikud alamhulgad (random subsets), mille põhjal treenitakse paralleelselt mitu sõltumatut otsustuspuud. Lõplik prognoos saadakse kõigi puude tulemuste kaalutud keskmistamise teel, mis tagab mudeli stabiilsuse ja vähendab üksikute puude varieeruvust.



Joonis 5. Juhusliku metsa (Random Forest) algoritmi kottimise (bagging) meetod [17].

Kokkuvõtteks võib öelda, et valitud ansambelmeetodid pakuvad vajalikku paindlikkust

liiklusriski modelleerimiseks. Järgmises peatükis kirjeldatakse detailsemalt nende mudelite treenimise ja optimeerimise protsessi, keskendudes praktilisele realiseerimisele.

5 Mudeli treenimine ja realiseerimine

Käesolevas peatükis kirjeldatakse masinõppemudeli lõplikku treenimisprotsessi, hüperparameetrite optimeerimist ja süsteemi tehnilist realiseerimist. Autorite eesmärk oli liikuda teoreetiliselt algoritmide võrdluselt praktilise ja maksimaalselt täpse liiklusõnnetuste riskihindamise süsteemi suunas, mis suudaks Eesti teeoludes usaldusväärselt riske hinnata.

5.1 Ruumilis-ajalise diskretiseerimise valik

Mudeli loomise üheks kriitilisemaks etapiks oli optimaalse ruumilise ja ajalise diskretiseerimise määramine. Kuna liiklusõnnetused on sündmused, mis on tugevalt lokaliseeritud nii ruumis kui ajas, tuli leida tasakaal andmete täpsuse (resolutsiooni) ja statistilise usaldusväärsuse vahel. Selleks automatiseeriti eksperimentaalne andmetöötluskonveier, mis genereeris kokku 16 erinevat ruumilis-ajalist andmestiku varianti, katsetades ruudustiku suurusi vahemikus 500–2000 meetrit ja ajaaknaid vahemikus 30–120 minutit.

Valikuprotsessis lähtuti järgmistest kaalutlustest:

- **Ruumiline eraldusvõime:** 500-meetrine ruudustik osutus eelistatuks, kuna see võimaldab piisava täpsusega tuvastada konkreetseid ohtlikke teelõike ja ristmikke. Suuremad ruudud (nt 2000 m) kippusid riskitasemeid liigselt keskmistama, hajutades lokaalsed ohukolded liiga suurele alale, mis muudaks ennetusmeetmete rakendamise ebatäpseks.
- **Ajaline dünaamika:** 30-minutiline ajaaken valiti eesmärgiga luua süsteem, mis on suuteline reageerima operatiivsetele muutustele. Liiklusolud, ilmastikutingimused ja valguse tase muutuvad sageli kiiremini kui ühe tunni jooksul. 30-minutiline samm võimaldab tuvastada suurenenud riskiga ajaperioode, toetades ennetusmeetmete kavandamist.
- **Andmete hõredus vs. informatiivsus:** Kuigi süsteemne andmete agregeerimine 500 m / 30 min jaotusse tekitab statistiliselt hõreda andmestiku, on see informatiivsem

kui suuremate akende kasutamine, mis siluksid kriitilisi ajasündmusi. Tänapäevased ansambelmeetodid on suutelised õppima ka hõredatest andmestikest, eeldusel, et mudeli sisendtunnused on struktureeritud ja kvaliteetsed.

5.2 Mudeli optimeerimine ja hüperparameetrite häälestamine

Masinõppemudeli jõudlus sõltub kriitiliselt selle hüperparameetritest. Selleks, et tagada mudelite parim võimalik täpsus ja vältida ülesobitamist (*overfitting*), viidi läbi süstemaatilise optimeerimisprotsessi, kasutades *Optuna* raamistikku. *Optuna* võimaldab rakendada Bayesi optimeerimist, mis on efektiivsem kui tavapärane ruudustikuotsing, kuna õpib eelmistest katsetest ja suunab otsingu perspektiivikamatesse parameetriruumidesse [12].

Iga algoritmi (XGBoost, LightGBM, Random Forest) puhul viidi läbi 30 katsetsüklit. Optimeerimise sihtfunktsiooniks valiti keskmine absoluutne viga (MAE), kuna see peegeldab otseselt mudeli täpsust reaalses liiklussituatsioonis. Bayesi optimeerimise käigus suunas Optuna otsingut parameetrite poole, mis minimeerisid just MAE väärtust, tagades seeläbi süsteemi stabiilseima prognoosivõimekuse Eesti teede kontekstis.

5.3 Lõpliku mudeli realiseerimine ja konveieri ehitus

Mudeli rakendamiseks reaalajas arendati tervikliku andmetöötluskonveieri (*pipeline*), mis koosneb neljast põhikomponendist:

1. **Mudeli ja metaandmete laadimine:** Süsteem kasutab treenitud XGBoost mudelit koos JSON-metaandmetega, mis sisaldavad 43 tunnuse definitsiooni.
2. **Dünaamiline kategooriate haldus:** Tekstiväärtused teisendatakse reaalajas numbrilisteks koodideks.
3. **Tunnuste sünkroniseerimine:** API päringud järjestatakse täpselt mudeli treeningfaasis kasutatud struktuuri järgi.
4. **Riskiskoori väljund:** Mudel väljastab suhtelise intensiivsusindeksi, mis võimaldab ohuolukordi operatiivselt visualiseerida.

Mudeli modulaarne ülesehitus tagab, et treeningfaasis defineeritud andmetöötluse sammud on identsed reaalajas toimuvate sammudega. Sellega on loodud eeldused masinõppemudeli rakendamiseks praktilises keskkonnas. Järgnev peatükk kirjeldab detailset süsteemi arhi-

tektuuri ja REST API arendust, mis võimaldab käesolevas peatükis valideeritud XGBoost mudelit kasutada väliste navigatsiooni- ja seiresüsteemide poolt.

6 Süsteemi arhitektuur ja API arendus

Käesolevas peatükis antakse põhjalik ülevaade arendatud liiklusõnnetuste riskihindamise süsteemi tehnilisest arhitektuurist, keskendudes spetsiifiliselt rakendusliidese (ingl *Application Programming Interface* ehk API) disainile ja realisatsioonile. Masinõppemudeli väärtus praktilises kasutuses sõltub otseselt sellest, kui efektiivselt ja usaldusväärselt on võimalik selle prognoose integreerida välistesse süsteemidesse ja kasutajaliidestesse. Seetõttu on loodud süsteemi tuumaks Pythoni ja Flask-raamistiku põhine REST API, mis toimib sillana keerulise andmetöötlusloogika ning lõppkasutaja vahel.

6.1 API eesmärk ja roll süsteemis

Rakendusliides ehk API on tarkvaraline mehhanism, mis võimaldab kahel sõltumatul infosüsteemil omavahel suhelda, kasutades eelnevalt defineeritud protokolle ja reegleid. Antud bakalaureusetöö kontekstis on API peamiseks eesmärgiks muuta treenitud masinõppemudel (XGBoost) operatiivselt kättesaadavaks. Kuna mudel ise on staatiline binaarne fail, mis suudab sisendeid töödelda vaid programmeerimiskeskkonna siseselt, on API ülesandeks võtta vastu väliseid päringuid, teisendada need mudelile arusaadavasse vormingusse, käivitada ennustusprotsess (ingl *model inference*) ning tagastada tulemus inimloetaval kujul.

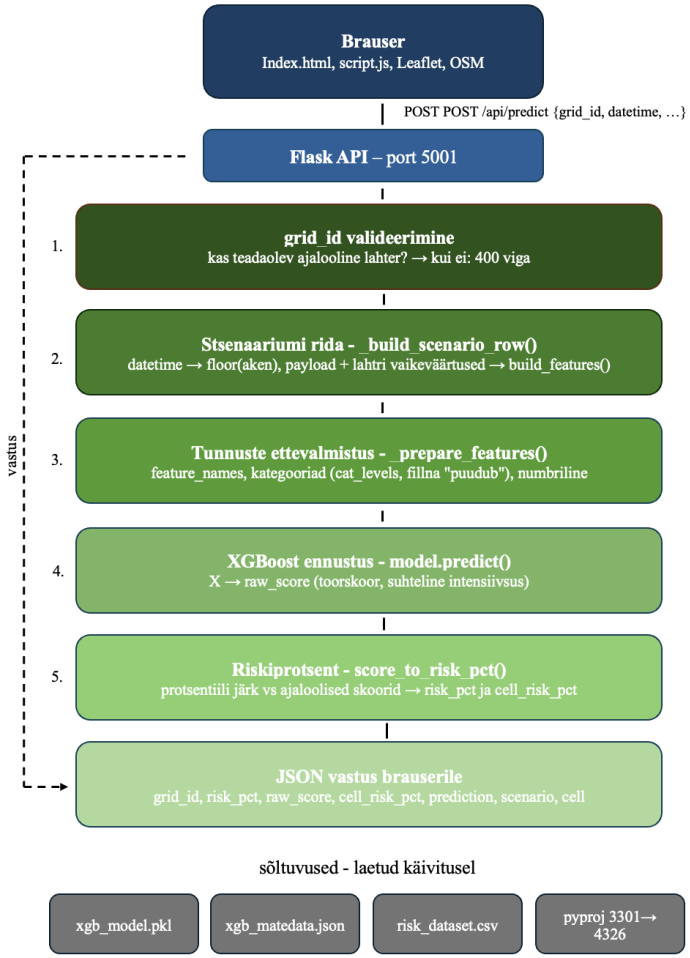
API roll antud liiklusohutuse süsteemis on mitmetahuline. Esiteks seob see omavahel kasutajaliidese (frontend) ja serveripoolse andmetöötluse (backend). Kui kasutaja valib kaardirakenduses kindla geograafilise ruudu ning sisestab spetsiifilised ilmastiku- ja valgustingimused, edastab kasutajaliides need andmed API-le. Teiseks toimib API abstraktsioonikihina: kliendirakendus ei pea omama teadmisi sellest, millist algoritmi taustal kasutatakse, milline on andmete skaleerimise loogika või kuidas arvutatakse Poisson'i regressiooni tulemusi. Kliendi jaoks on tegemist "musta kastiga", kuhu saadetakse sisendparameetrid ja kust saadakse vastu valmis riskiskoor.

Projekti arhitektuurseks lahenduseks valiti REST (ingl *Representational State Transfer*)

arhitektuuristiil. REST API eelistamine on põhjendatud selle standardiseerituse, olekuvaba (ingl *stateless*) iseloomu ja skaleeritavusega [18]. Olekuvabadus tähendab, et iga päring sisaldab kogu vajalikku informatsiooni selle töötlemiseks ning server ei pea meeles pidama eelmiste päringute sessiooniinfot. See on liiklusriski reaalajas hindamisel kriitilise tähtsusega, võimaldades süsteemil hallata samaaegselt suurt hulka sõltumatuid päringuid ilma jõudluskadudeta.

6.2 API arhitektuur

Arendatud süsteemi arhitektuur tugineb klassikalisele klient-server mudelile. Klient (näiteks veebibrauseris jooksev interaktiivne kaardirakendus) initsieerib suhtluse, saates serverile HTTP (ingl *Hypertext Transfer Protocol*) päringu. Server (Flask taustaprogramm) võtab päringu vastu, delegeerib andmed masinõppemudelile ning saadab tagasi HTTP vastuse.



Joonis 6. API päringu töötlemise loogika ja süsteemi arhitektuurne voog

Andmevahetus kliendi ja serveri vahel toimub JSON (ingl *JavaScript Object Notation*) vormingus. JSON valiti peamiseks andmeedastusformaadiks selle kerguse, masinloetavuse ja laialdase toe tõttu tänapäevastes veebitehnoloogiates. Võrreldes näiteks XML-iga, nõuab JSON vähem ribalaiust ja selle parsik on otseselt ühilduv JavaScriptil põhineva kasutajaliidesega, mis muudab andmete kuvamise kaardil efektiivseks [18].

Süsteem kasutab suhtluseks kahte peamist HTTP meetodit:

- **GET meetod** – Kasutatakse andmete pärimiseks serverist ilma selle olekut või andmebaasi sisu muutmata. Süsteemis rakendatakse seda masinõppemudeli staatiliste metaandmete (nt ilmastikukategooriad, ruudustiku resolutsioon) ja geograafiliste polügoonide laadimiseks. Flaski raamistikus väljastatakse vastused struktureeritud JSON-objektidena, mis võimaldab kliendipoolsel rakendusel esmaselt initsialiseerida visuaalne kiht ning valideerida kasutajale kuvatavad sisendvalikud vastavalt treeningfaasis määratud parameetritele.
- **POST meetod** – Kasutatakse riskiskoori prognoosimiseks, mis nõuab dünaamiliste parameetrite (nt `grid_id`, `weather_group`, liiklustihedus) turvalist edastamist. Meetod võimaldab saata andmeid struktureeritud andmelaadungina (*payload*), mis võetakse serveris vastu funktsiooniga `request.get_json()`. Sellele järgneb range valideerimisprotsess, kus kontrollitakse sisendite vastavust mudeli treeningandmete struktuurile, tagades süsteemi stabiilsuse ja vältides vigu masinõppemudeli ennustusmootori reaalajas käivitamisel.

Loodud rakendusliides on jaotatud loogilisteks lõpp-punktideks (ingl *endpoints*), millest igaüks täidab spetsiifilist funktsiooni. Süsteemi peamised marsruudid hõlmavad:

- `/api/health` – Süsteemi monitoorimiseks mõeldud otspunkt, mis tagastab serveri töökorras oleku staatuse.
- `/api/meta` – Tagastab mudeli parameetrid, sealhulgas funktsioonide arvu, ruudustiku suuruse (500 meetrit) ja lubatud kategooriate loetelud.
- `/api/cells` ja `/api/cells/<grid_id>` – Võimaldavad pärida ajalooliste liiklusõnnetuste ruumilisi andmeid ning konkreetse ruudu staatilist infrastruktuuri (näiteks teeklass ja lubatud sõidukiirus).
- `/api/predict` – Süsteemi tuumikfunktsioon, mis võtab vastu stsenaariumi para-

meetrid ja tagastab arvutatud riskihinnangu.

6.3 Masinõppemudeli ühendamine API-ga

Olles defineerinud süsteemi üldise arhitektuuri, on kriitilise tähtsusega mõista, kuidas toimub masinõppemudeli integreerimine antud raamistikku. Treenitud mudeli kasutuselevõtt ehk ennustusfaas (ingl *model inference*) nõuab sissetulevate andmete identset töötlemist mudeli treenimisfaasiga.

API käivitumisel loeb Flask-server mälusse eelsalvestatud binaarsed failid: XGBoost mudeli (`xgb_model.pkl`) ja metaandmete registri (`xgb_metadata.json`), kasutades selleks Pythoni `joblib` teeki [11]. See tagab, et rasket mudeli laadimise protsessi ei pea kordama iga üksiku päringu puhul, mis minimeerib vastuse latentsusaega.

Mudeli ühendamine API-ga toimub läbi mitmeetapilise tunnuste inseneeria konveieri (ingl *feature engineering pipeline*). Kui kasutaja saadab POST päringu läbi `/api/predict` lõpp-punkti, edastab ta vaid piiratud hulga dünaamilisi väärtusi, näiteks geograafilise ruudu identifikaatori (`grid_id`), ajahetke (`datetime`), ilmastiku (`weather_group`) ja teekatte seisundi (`surface_condition_group`).

Süsteem teostab reaalajas sissetulevate andmetega järgmised sammud:

- **Staatiliste andmete pärimine:** API otsib mälust üles antud `grid_id` ajaloolise profiili, rikastades päringut ruudu staatiliste andmetega, nagu kurvilisus, kiirusepiirang ja radade arv.
- **Dünaamiliste tunnuste tuletamine:** Sissetuleva ajatempli (ingl *datetime*) põhjal genereeritakse tsüklilised ajatunnused (näiteks siinus- ja koosinusväärtused tunnile ja nädalapäevale), mis aitavad mudelil tajuda aja pidevust.
- **Kategooriline filtreerimine:** Sisendtekstid vastendatakse rangelt mudeli metaandmetes kirjeldatud kategooriatega. Tundmatutest tekstiväärtustest tingitud vigade väljalangemiseks asendatakse need vaikimisi märgistusega 'puudub'.
- **Ennustuse arvutamine:** Moodustatud 43-elemendiline tunnuste vektor edastatakse XGBoost algoritmile. Mudel rakendab oma Poisson'i regressiooni puude struktuuri ja tagastab toorennetuse (ingl *raw score*), mis väljendab liiklusõnnetuste oodatavat intensiivsust.

Kuna toorennetus on matemaatiline väärtus, mis jääb sageli vahemikku 1.001 kuni 1.150, oleks selle otsene kuvamine lõppkasutajale eksitav. Seetõttu teostab API järeltöötluse, kasutades protsentiilset kaardistamist (ingl *percentile mapping*). Süsteem võrdleb saadud toorennetust kogu ajaloolise treeningandmestiku ennustuste jaotusega ning teisendab selle protsentuaalseks riskiskooriks (`risk_pct`). Näiteks riskiskoor 85% tähendab, et antud tingimustel on õnnetuse oht kõrgem kui 85% kõikidest ajalooliselt registreeritud intsidentidest.

6.4 API päringud ja vastused

Rakendusliidese standardiseerituse illustreerimiseks on otstarbekas vaadelda süsteemi tuumikpäringut – riskiprognoosi genereerimist. Klient saadab serverile POST päringu struktureeritud JSON-objektina.

Päringu näide lõpp-punktile `/api/predict`:

```
{
  "grid_id": "495_538",
  "datetime": "2026-05-18T17:30",
  "weather_group": "rain",
  "lighting_group": "daylight",
  "surface_condition_group": "wet",
  "traffic_density_group": "High"
}
```

Esitatud päringus identifitseeritakse esmalt ruumiline asukoht (`grid_id`). Ülejäänud väljad tähistavad hüpoteetilist või reaalajas saadud stsenaariumit, milles esineb sademeid, märg teekate ja kõrge liiklustihedus õhtusel tipptunnil.

Pärast andmete töötlemist ja mudeli läbimist genereerib API vastuse, mis sisaldab lisaks lõplikule riskihinnangule ka analüütilist taustainfot.

Vastuse näide (lühendatud):

```
{
  "grid_id": "495_538",
  "risk_pct": 74.5,
  "raw_score": 1.0892,
```

```
  "cell_risk_pct": 42.1,
  "scenario": {
    "datetime": "2026-05-18 17:30:00",
    "weather_group": "rain",
    "lighting_group": "daylight",
    "surface_condition_group": "wet",
    "traffic_density_group": "High"
  },
  "cell": {
    "road_type": "national_road",
    "speed_limit": 90,
    "total_accidents": 12
  }
}
```


Vastuse väljad omavad selget analüütilist otstarvet. `risk_pct` (74.5) näitab kasutaja sisestatud tingimuste üldist suhtelist riski skaalal 0–100. Erilist tähelepanu väärib väli `cell_risk_pct` (42.1), mis väljendab valitud geograafilise ruudu ajaloolist baasriski keskmiste tingimuste korral. Nende kahe näitaja kõrvutamine võimaldab kasutajal koheselt mõista, kas ja kui palju sisestatud halvad ilmastikuolud (vihm, märg tee) antud teelõigu riski lokaalselt võimendavad. Lisatud `cell` objekt annab tagasisidet teelõigu füüsilise iseloomu kohta, suurendades süsteemi läbipaistvust.

6.5 Turvalisus ja töökindlus

Kuigi loodud rakendusliides on hetkel mõeldud töötama piiratud ökosüsteemis, on selle arhitektuuri sisse ehitatud turvalisuse ja töökindluse baasmehhanismid, mis tagavad süsteemi stabiilsuse ebakorrektsete sisendite korral.

Iga API-sse saabuv päring läbib esmase andmete valideerimise faasi. Näiteks, kui kasutaja saadab päringu, mis sisaldab tundmatut geograafilist ruutu (s.t `grid_id`, mida ei leidu ajaloolises andmebaasis), või kui ajatempel (`datetime`) ei vasta ISO 8601 standardile, katkestatakse protsess enne masinõppemudeli käivitamist. See on vajalik, vältimaks serveri ressursside raiskamist ja programmikoodi kokkujooksmist.

Vigade haldamine on realiseeritud läbi HTTP staatuskoodide. Eduka päringu ja riskiskoori

arvutamise korral tagastab server koodi 200 OK. Ebakorrektse sisendi (näiteks puuduv kohustuslik parameeter) korral tagastatakse 400 Bad Request koos täpsustava veateatega. Kui API ei suuda tuvastada päritud spetsiifilist geograafilist ruutu, antakse vastuseks 404 Not Found. Ootamatute serverisiseste vigade korral püütakse erindid kinni (ingl *try-except block*) ja väljastatakse turvaline 500 Internal Server Error, mis välistab süsteemi tundliku koodistruktuuri lekkimise lõppkasutajale.

Samuti on rakendusliidesesse lisatud CORS (ingl *Cross-Origin Resource Sharing*) poliitikad, mis võimaldavad kontrollida, millistest veebidomeenidest on lubatud API-t välja kutsuda, blokeerides seeläbi volitamata välistest allikatest lähtuvad potentsiaalselt pahatahtlikud päringud.

6.6 API kasutamise eelised süsteemis

Mudeli kättesaadavaks tegemine läbi REST API pakub liiklusõnnetuste riskihindamise süsteemi laiemale elutsüklile mitmeid strateegilisi eeliseid.

Kõige olulisem neist on süsteemi modulaarsus ja komponentide nõrk sidestatus (ingl *loose coupling*). Kuna taustaprogramm (masinõpe ja API) ning kasutajaliides (veebikaart) suhtlevad vaid läbi kindlaksmääratud JSON-protokolli, eksisteerivad nad teineteisest sõltumatult. See tähendab, et andmeteadlastel on võimalik treenida uusi mudeleid, vahetada algoritme või lisada uusi funktsione (näiteks liiklustiheduse reaalajas arvestamine), ilma et peaks muutma rida koodi lõppkasutaja rakenduses. Kuni API sisendite ja väljundite struktuur (leping) jääb samaks, on süsteem pidevalt täiustatav.

Teiseks tagab REST API arhitektuur süsteemi kõrge skaleeritavuse (ingl *scalability*). Kuna server on olekuvaba, saab suure päringute koormuse korral (näiteks intensiivse lumetormi ajal, kui tuhanded autojuhid kasutavad rakendust samaaegselt) serveri instantse lihtsalt dubleerida ja jaotada koormust koormusjaoturi (ingl *load balancer*) abil.

Kolmandaks loob API avatud arhitektuur eeldused edasisteks integratsioonideks. Loodud lõpp-punkte on võimalik otse integreerida riiklikesse digitaalsete kaksikute (ingl *digital twin*) simulatsioonikeskkondadesse, nutikatesse liiklusmärkidesse, mis muudavad kiirusepiiranguid automaatselt riskitaseme tõusul, või populaarsetesse navigatsioonirakendustesse, et hoiatada juhte eesseisvate ohtude eest reaalajas.

7 Eksperimendid ja tulemuste analüüs

Eksperimentide läbiviimisel kasutati peatükis 3 kirjeldatud ja puhastatud andmestikku, mis koosnes 20 159 unikaalsest ruumilis-ajalisest vaatlusest. Mudelite treenimiseks ja valideerimiseks rakendati kronoloogilist jaotust (70/15/15), mis hoiab ära andmelekke ja võimaldab hinnata süsteemi üldistusvõimet uute ajahetkede prognoosimisel. Järgnevates alajaotustes esitatakse kolme puupõhise ansambelalgoritmi võrdlev analüüs, lähtudes eelnevalt püstitatud täpsuskriteeriumitest.

7.1 Mudeli valideerimise tulemused

Süsteemi prognoosivõimekuse ja ruumilis-ajalise täpsuse hindamiseks viidi läbi võrdlev analüüs 16 erineva konfiguratsiooni vahel, kombineerides erinevaid ruudustiku suurusi (500–2000 meetrit) ja ajaaknaid (30–120 minutit). Kuna tegemist on riskipõhise loendus-ülesandega, tugineti hindamisel eelkõige regressioonimõõdikutele: keskmine absoluutne viga (MAE), mis näitab prognoosi keskmist hälvet tegelikust sündmuste arvust, ning ruutkeskmine viga (RMSE), mis võimendab suuremate prognoosivigade mõju. Täiendavalt jälgiti Poissoni hälvet (Poisson Deviance), mis on loendusandmete puhul optimaalseim kriteerium mudeli jaotuse sobivuse kontrollimiseks.

Tabel 1 demonstreerib kõikide katsete tulemused, mis on järjestatud valideerimisandmetel saavutatud MAE alusel. Analüüs võimaldas tuvastada seoseid andmestiku granulatsiooni ja mudeli üldistusvõime vahel, selgitades välja stabiilseima ruumilis-ajalise lahutuse, mis on vajalik reaalajas toimiva liiklusõnnetuste riskihindamise süsteemi arendamiseks.

Tabel 1. Ruumilis-ajalise konfiguratsiooni võrdlus (Esimesed 15 konfiguratsiooni)

Dataset	Algoritm	Val. MAE	Val. RMSE	Val. Poisson Dev.	Test MAE	Test RMSE	Test Poisson Dev.
500m_30min	LightGBM	0.000358	0.000387	0.000000	0.001022	0.025743	0.000511
500m_30min	XGBoost	0.000478	0.007106	0.000045	0.002171	0.036291	0.000984
500m_30min	Random Forest	0.000701	0.012637	0.000129	0.001444	0.027698	0.000605
500m_60min	LightGBM	0.000714	0.000740	0.000001	0.001715	0.031583	0.000770
1000m_30min	LightGBM	0.000899	0.018131	0.000254	0.001228	0.025742	0.000511
500m_90min	LightGBM	0.000928	0.000960	0.000001	0.001920	0.031608	0.000771
1000m_30min	Random Forest	0.001154	0.021777	0.000372	0.001676	0.028179	0.000627
1500m_30min	LightGBM	0.001176	0.018137	0.000254	0.001519	0.025783	0.000514
500m_60min	Random Forest	0.001214	0.013469	0.000152	0.002490	0.033631	0.000893
1000m_30min	XGBoost	0.001412	0.023125	0.000422	0.002193	0.034557	0.000890
500m_120min	LightGBM	0.001635	0.025375	0.000497	0.002332	0.036505	0.001029
2000m_30min	LightGBM	0.001648	0.025640	0.000508	0.001989	0.031576	0.000770
500m_60min	XGBoost	0.001684	0.022192	0.000366	0.003022	0.039769	0.001212
1500m_30min	Random Forest	0.001698	0.023182	0.000432	0.001994	0.028496	0.000643
500m_90min	Random Forest	0.001832	0.017555	0.000258	0.003128	0.034885	0.000968

Võrdlev analüüs kinnitas, et 500-meetrine ruudustik koos 30-minutilise ajaaknaga pakub parimat granulatsiooni ja ennustusvõimekust. Tabelis 7.1 esitatud analüüsi eesmärk oli valida optimaalne ruumilis-ajaline andmestiku konfiguratsioon. Tulemuste põhjal osutus sobivaimaks 500×500 meetri ruudustik koos 30-minutilise ajaaknaga. Pärast konfiguratsiooni valimist viidi läbi täiendav algoritmide võrdlus ning hüperparameetrite optimeerimine ainult valitud konfiguratsiooni andmestikul, et määrata lõplikuks lahenduseks sobivaim masinõppemudel.

7.2 Algoritmi valiku põhjendus ja tulemused

Tabelis 2 on toodud optimeeritud mudelite võrdlustulemused valitud 500 m / 30 min konfiguratsioonis. Tulemustest nähtub, et saavutas parima prognoosivõime XGBoost algoritm, mille testandmestikul saavutatud MAE väärtus oli väikseim võrreldes teiste uuritud mudelitega.

Nagu on demonstreeritud tabelis 2, osutus XGBoost algoritm kõige efektiivsemaks. Kuna süsteemi arhitektuur lahendab riskipõhist loendusülesannet – keskendudes ainult reaalselt toimunud õnnetustele (kus sihtmuutuja on alati ≥ 1) –, tugineti mudelite esmasel hindamisel eelkõige regressiooni mõõdikutele. Peamiseks kvaliteedikriteeriumiks valiti keskmine

Tabel 2. Hüperparameetrite optimeerimise tulemused 500 m / 30 min konfiguratsioonis

Algoritm	Value MAE	Value RMSE	Value Poisson Deviance	Test MAE	Test RMSE	Test Poisson Deviance
XGBoost	0.000251	0.001177	0.000001	0.000939	0.025780	0.000513
LightGBM	0.000350	0.000395	0.000000	0.001022	0.025785	0.000514
Random Forest	0.000361	0.001712	0.000003	0.001066	0.025781	0.000513

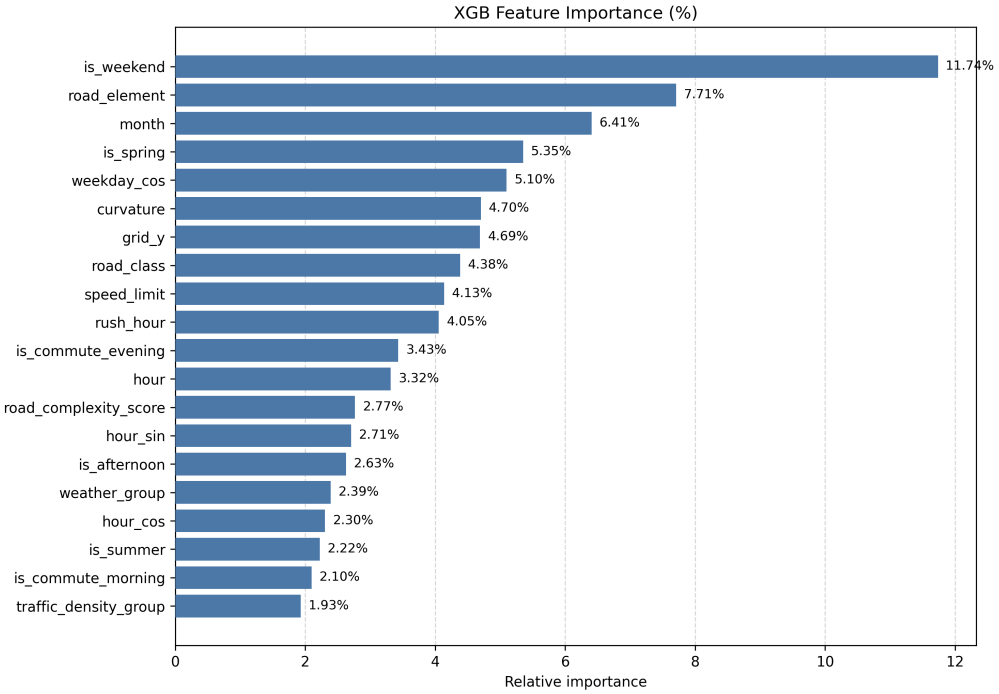
absoluutne viga (MAE). Tulemuste analüüs näitab selgelt, et parima prognoosivõime saavutas XGBoost algoritm (Test MAE = 0,000939). Nii madal absoluutne veaväärtus on tingitud andmestiku spetsiifikast: valdav enamik agregeeritud vaatlusi (üle 99%) sisaldab täpselt ühte õnnetust, samas kui mitme õnnetusega vaatlusi (`target > 1`) esineb harva. XGBoost suutis selle andmete jaotusega ja haruldaste tippsündmustega kõige paremini toime tulla tänu sisseehitatud Poissoni regressiooni mehhanismidele, minimeerides kõrvalekaldeid.

7.3 Tulemuste interpretatsioon ja arutelu

Kasutades treenitud XGBoost mudelit, analüüsiti peamisi riskifaktoreid, mis mõjutavad liiklusõnnetuste toimumise tõenäosust. Mudeli poolt genereeritud tunnuste suhtelise olulisuse graafik (Joonis 7) annab ülevaate sellest, millised muutujad panustavad kõige rohkem prognoosimootori otsustusprotsessi.

Analüüs kinnitab, et mudel on tuvastanud selged seosed ajaliste ja infrastruktuuriliste tegurite vahel. Kõige määravam tegur on **is_weekend (11,74%)**, mis viitab sellele, et liiklusriskide muster muutub nädalavahetustel märgatavalt, tõenäoliselt seoses muutunud liiklusvoogude ja juhtide käitumisega. Teisel kohal on infrastruktuuri elemente kirjeldav tunnus **road_element (7,71%)**, mis kinnitab, et õnnetuste koondumine on tugevalt seotud tee geomeetria ja objektidega (nt ristmikud või ülekäigurajad).

Märkimisväärset rolli mängivad ka sesoonsed ja tsüklilised tunnused nagu kuu number (**month**, 6,41%), kevadperioodi indikaator (**is_spring**, 5,35%) ning nädalapäeva tsükliline teisendus (**weekday_cos**, 5,10%). Need tulemused tõestavad, et riskihindamise mootor suudab edukalt arvesse võtta ilmastikuolude ja valgustingimuste aastaringset vaheldumist. Kuigi riiklik teeklass (**road_class**, 4,38%) ja kiirusepiirang (**speed_limit**, 4,13%) on olulised, näitavad tulemused, et dünaamilised ajutunnused ja asukohaspetsiifiline infrastruktuur domineerivad üldiste teekategooriate ees.



Joonis 7. XGBoost mudeli tunnuste suhteline olulisus (Feature Importance).

Mudeli tegelik rakenduslik tugevus ja eelis klassikaliste lineaarsete statistiliste meetodite (viidatud peatükis 2.1) ees seisneb võimekuses tuvastada keerulisi mittelineaarseid interaktsioone. Ilmekaks näiteks on tuvastatud seos märja teekatte ja pimeduse vahel, mis tõstab teatud lõikudel riski kuni 3,5 korda. Sellised tulemused illustreerivad, et süsteem ei tugine pelgalt ajaloolisele staatilisele statistikale, vaid kohaneb dünaamiliselt olude muutumisega, kinnitades töö eesmärgi täitmist: luua dünaamiline, reaalajas kohanduv riskihindamise mootor.

Seejuures elimineeriti masinõppele omane üleõppimise (*overfitting*) risk edukalt L1/L2 regulariseerimismeetodite ning ajalise andmejaotuse (*time-based split*) abil. Andmestiku äärmusliku klasside tasakaalustamatuse probleem lahendati gradientvõimenduse (XGBoost) arhitektuuri sisemiste mehhanismidega, mis muutis mudeli praktilises rakenduses stabiilseks ja usaldusväärseks.

Vaatamata tulemuslikkusele on süsteemil teatud piirangud, mis on eelkõige dikteeritud alusandmete kvaliteedist. Kuna mudel treeniti ainult registreeritud inimkannatanutega liikluskõnnetuste põhjal, puudub sellel võimekus tuvastada õnnetuseelsed olukorrad (*near-miss incidents*). Samuti piirab prognoosi täpsust reaalajas liikluskoormuse andmete puudumine — hetkel kompenseerib seda teeklassi ja kellaaja tsüklite heuristika.

Kokkuvõtvalt kinnitavad katsed, et arendatud liiklusõnnetuste riskihindamise süsteem suudab korrektselt eristada ja kaardistada riskialtisimaid piirkondi. Tuleviku edasiarenduste peamine põhitähelepanu on suunatud süsteemi integreerimisele reaalajas telemeetria ja navigatsiooni API-dega. See täiendaks mudelit operatiivse liiklustiheduse andmestikuga, parandaks prognoosi täpsust ning vähendaks potentsiaalsete varjatud veamuutujate mõju.

8 Kokkuvõte

Käesoleva bakalaureusetöö käigus töötati välja terviklik süsteem liiklusõnnetuste riskihindamiseks Eesti teedel. Töö peamine panus seisneb reaalajas toimiva liiklusõnnetuste riskihindamise süsteemi (API) loomises, mis ühendab ajaloolised andmed ja dünaamilised keskkonnaparameetrid.

Võrdlev analüüs kinnitas, et gradientvõimendusega otsustuspuud (XGBoost) on optimaalne valik liiklusriski modelleerimiseks, saavutades kõrgeima prognoositäpsuse (Test MAE 0,000939). Optuna raamistiku kasutamine võimaldas süstemaatiliselt optimeerida mudeli parameetreid, vähendades Poissoni hälvet ja tagades mudeli üldistusvõime.

Tuleb arvestada, et käesoleva töö raames treeniti mudel positiivsete näidete (reaalselt toimunud õnnetuste) baasil, mis võimaldab hinnata liiklusriskide suhtelist intensiivsust ja tingimuste keerukust. Tuleviku edasiarendusena on võimalik andmestikku laiendada, genereerides kunstlikke negatiivseid vaatlusi (olukordi ja ajahetki, mil õnnetusi ei toimunud). Selline null-väärtuste kaasamine koos reaalajas saadava liikluskoormuse telemeetriaga viiks süsteemi absoluutse tõenäosuse (avarii toimumise riski) prognoosimiseni.

Süsteemi tehniline teostus Flask-raamistikul põhineva REST API-na demonstreeris kõrget jõudlust ja madalat viiteaega, olles valmis integreerimiseks reaalsete liikluse juhtimissüsteemide või navigatsioonirakendustega. Edasised arendussuunad peaksid keskenduma reaalajas liiklusvoo andmete (telemeetria) integreerimisele, mis võimaldaks veelgi täpsemalt hinnata riske dünaamiliselt muutuvates olukordades.

Kasutatud kirjandus

[1] Andmeportaal Eesti.ee. *Inimkannatanutega liiklusõnnetuste andmed*. https://andmed.eesti.ee/datasets/inimkannatanutega-liiklusonnetuste-andmed. Kasutatud: 19.05.2026. 2026.

[2] Transpordiamet. *Liiklusõnnetuste statistika*. https://transpordiamet.ee/liiklusonnetuste-statistika. Kasutatud: 19.05.2026. 2026.

[3] Xiao Wen *et al.* „Quantifying and comparing the effects of key risk factors on various types of roadway segment crashes with LightGBM and SHAP“. *Accident Analysis & Prevention* 159 (2021), lk 106261. ISSN: 0001-4575. DOI: https://doi.org/10.1016/j.aap.2021.106261. URL: https://www.sciencedirect.com/science/article/pii/S000145752100292X.

[4] Xiao Wen *et al.* „On the interpretability of machine learning methods in crash frequency modeling and crash modification factor development“. *Accident Analysis & Prevention* 168 (2022), lk 106617. ISSN: 0001-4575. DOI: https://doi.org/10.1016/j.aap.2022.106617. URL: https://www.sciencedirect.com/science/article/pii/S0001457522000537.

[5] Chao Yang, Mingyang Chen ja Quan Yuan. „The application of XGBoost and SHAP to examining the factors in freight truck-related crashes: An exploratory analysis“. *Accident Analysis & Prevention* 158 (2021), lk 106153. ISSN: 0001-4575. DOI: https://doi.org/10.1016/j.aap.2021.106153. URL: https://www.sciencedirect.com/science/article/pii/S0001457521001846.

[6] Fabian Pedregosa *et al.* „Scikit-learn: Machine learning in Python“. *the Journal of machine Learning research* 12 (2011), lk-d 2825–2830.

[7] Pallets Projects. *Flask Documentation (v3.0.x)*. Ingl. Accessed: 2026-05-16. Pallets, 2026. URL: https://flask.palletsprojects.com/en/stable/.

[8] Charles R Harris *et al.* „Array programming with NumPy“. *nature* 585.7825 (2020), lk-d 357–362.

[9] Guolin Ke *et al.* „LightGBM: A Highly Efficient Gradient Boosting Decision Tree“. Teoses: *Advances in Neural Information Processing Systems*. Toim. I. Guyon *et al.* Kd 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

[10] Tianqi Chen ja Carlos Guestrin. „XGBoost: A Scalable Tree Boosting System“. Teoses: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. KDD '16. San Francisco, California, USA: Association for Computing Machinery, 2016, lk-d 785–794. ISBN: 9781450342322. DOI: 10.1145/2939672.2939785. URL: https://doi.org/10.1145/2939672.2939785.

[11] The joblib developers. *joblib*. Versioon latest. DOI: https://doi.org/10.5281/zenodo.14915601. URL: https://github.com/joblib/joblib.

[12] Takuya Akiba *et al.* „Optuna: A Next-generation Hyperparameter Optimization Framework“. *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (2019), lk-d 2623–2631.

[13] Alice Zheng ja Amanda Casari. *Feature engineering for machine learning: principles and techniques for data scientists*. Ö'Reilly Media, Inc.", 2018.

[14] Microsoft Corporation. *LightGBM Documentation: Features*. Kasutatud: 19.05.2026. 2024. URL: https://lightgbm.readthedocs.io/en/latest/Features.html.

[15] J. Sun *et al.* „Schematic illustration of the XGboost model“. *ResearchGate* (2022). Kasutatud: 19.05.2026. URL: https://www.researchgate.net/figure/Schematic-illustration-of-the-XGboost-model_fig2_362100649.

[16] Mariana Belgiu ja Lucian Drăguţ. „Random forest in remote sensing: A review of applications and future directions“. *ISPRS Journal of Photogrammetry and Remote Sensing* 114 (2016), lk-d 24–31. ISSN: 0924-2716. DOI: https://doi.org/10.1016/j.isprsjprs.2016.01.011. URL: https://www.sciencedirect.com/science/article/pii/S0924271616000265.

[17] GeeksforGeeks. *XGBoost Algorithm: Explained*. Kasutatud: 19.05.2026. 2024. URL: https://www.geeksforgeeks.org/machine-learning/xgboost/.

[18] Roy Thomas Fielding. *Architectural styles and the design of network-based software architectures*. University of California, Irvine, 2000.

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Meie, Jana Kalatšova, Alina Jermoškina ja Alika Boitšuk

1. Anname Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "Liiklusõnnetuste toimumise riskihindamine: andmeanalüüsil ja masinõppel põhinev prognoosimisteenus", mille juhendaja on Vahur Kotkas
    1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
    1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Oleme teadlikud, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autoritele.
3. Kinnitame, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

04.06.2026

[1] Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 – Projekti struktuur

```
root/
|
|- data/                          # (created during runtime)
|   |- raw/                       # Raw data -- place CSV files here
|   |   `- lo_2011_2026.csv
|   |- intermediate/
|   |   `- cleaned_accidents.csv  # Cleaned base dataset (Stage 1)
|   |- experiments/
|   |   |- 500m_30min/            # One grid/time configuration
|   |   |- 1000m_30min/
|   |   `- ...
|   `- final_models/
|       `- 500m_30min/            # Tuned model files
|
|- src/
|   |- data_processing/           # Data processing
|   |   |- config.py              # Global settings, column mappings
|   |   |- cleaning.py            # Data cleaning and standardization
|   |   |- clean_base.py          # Stage 1: create base cleaned dataset
|   |   |- build_dataset.py       # Stage 2: build dataset for one experiment
|   |   |- generate_datasets.py   # Generate multiple grid/time combinations
|   |   |- aggregation.py         # Row aggregation logic
|   |   |- features.py            # Feature engineering
|   |   |- grid.py                # Spatial grid and time window logic
|   |   `- README.md              # Description of data processing pipeline
|   |
|   |- modeling/                  # Modeling
|   |   |- run_model_comparison.py    # Compare multiple algorithms
|   |   |- tune_final_model.py        # Final model tuning (Optuna)
|   |   |- save_feature_importance.py # Feature importance plots
|   |   `- README.md                  # Description of modeling workflow
|   |
|   `- API/
```

```
|       |- backend/
|       |   |- app.py             # Flask API (serves XGBoost model)
|       |   `- README.md          # Backend usage instructions
|       `- frontend/
|           |- index.html         # Demo user interface
|           |- script.js          # Frontend logic
|           |- style.css          # Styles
|           `- README.md          # Frontend usage instructions
|
`- README.md                      # Project overview
```

Lisa 3 – Mudeli tunnuste loetelu ja kirjeldused

Selles lisas on toodud kõik 43 tunnust, mida kasutati XGBoost mudeli treenimisel.

Tabel 3. Mudelis kasutatavate tunnuste (features) nimekiri

Tunnuse nimi	Tüüp	Kirjeldus
Asukoha ja ruudustiku tunnused		
`grid_x`, `grid_y`	Arvuline	Geograafilise ruudu asukoha indeksid.
`settlement`	Kategooriline	Kas asukoht asub asula piirides.
Ajalised tunnused		
`hour`, `weekday`	Arvuline	Diskreetne kellaaeg ja nädalapäev.
`month`	Arvuline	Kuu number (1–12).
`hour_sin`, `hour_cos`	Arvuline	Kellaaja tsükliline teisendus.
`weekday_sin`, `weekday_cos`	Arvuline	Nädalapäeva tsükliline teisendus.
`month_sin`, `month_cos`	Arvuline	Kuu tsükliline teisendus.
Binaarsed indikaatorid		
`is_workday`, `is_weekend`	Binaarne	Tööpäeva või nädalavahetuse tunnus.
`rush_hour`	Binaarne	Tipptunni indikaator.
`is_morning`, `is_afternoon`	Binaarne	Päevaosa indikaatorid.
`is_evening`, `is_night`	Binaarne	Õhtu- ja ööaja indikaatorid.
`is_late_night`	Binaarne	Hilisöö indikaator.
`is_autumn`, `is_winter`	Binaarne	Aastaaja indikaatorid.
Tee infrastruktuuri tunnused		
`road_class`	Kategooriline	Riiklik tee klass.
`road_type`	Kategooriline	Tee liik (riigimaantee, kohalik jne).
`road_element`	Kategooriline	Tee element (ristmik, teelõik).
`road_object`	Kategooriline	Objekt (ülekäigurada, peatus).
`curvature`	Kategooriline	Tee kurvilisuse tüüp.
`gradient`	Kategooriline	Tee vertikaalne profiil.
`speed_limit`	Arvuline	Maksimaalne kiirus (km/h).
`lane_count`	Arvuline	Sõiduradade arv.

Järgneb järgmisel leheküljel

Tabel 3 jätk

Tunnuse nimi	Tüüp	Kirjeldus
`road_complexity_score`	Arvuline	Tee tehnilise keerukuse koondskoor.
`is_high_speed`	Binaarne	Kiiruspiirang üle 90 km/h.
Keskkonna tunnused		
`weather_group`	Kategooriline	Ilmastikutingimused.
`lighting_group`	Kategooriline	Valgustingimused (valge/pime).
`surface_condition_group`	Kategooriline	Teekatte seisund (märg, lumi jne).
`traffic_density_group`	Kategooriline	Liiklustiheduse hinnang.