TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Rasmus Sikk 164720IAPB

VIRTUAALNE PERSONAALTREENER MASINÕPPE MEETODIL

bakalaureusetöö

Juhendaja: Priit Järv
[Teaduskraad]

Tallinn 2020

Autorideklaratsioon

Kinnitan, et olen 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.

Autor: Rasmus Sikk

18.05.2020

Annotatsioon

Bakalaureusetöö eesmärgiks on luua masinõppe mudel mis modifitseerib kasutaja tulevase treeningu tuginedes tema varasematele treeningutele. Töös räägib autor lühidalt jõusaalitreeningute baastõdedest ja erinevate masinõppemudelite tööpõhimõtetest. Kirjeldatakse andmete olulisust igasuguse masinõppeprobleemi lahendamisel ja kuidas antud töö lahendamiseks andmekogu saadi. Töös räägitakse ka andmete analüüsist ja nende ettevalmistus ning puhastus protsessist. Autor kirjeldab erinevate lahenduste implementeermist, analüüsib tulemusi ning võrdleb erinevaid lahendusi. Kõik proovitud lahendused ennustasid küll piisava täpsusega, kuid esimeses produktsiooniversioonis kasutamiseks ei ole see piisav, mis tulenes töö koostamisel kasutatud andmekogust. Lõpptoote jaoks valmis töö käigus ka taga rakendus mida kasutatakse mobiilirakenduses mis on loodud kaastudengi, Reio Sedriku, bakalaureusetöös.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 41 leheküljel, seitse peatükki, 22 joonist ja kolm võrrandit.

Abstract

Virtual Personal Trainer Using Machine Learning

The goal of this thesis is to implement machine learning model that modifies user's next workout based on previous workouts. Author briefly describes background information related to weight training and machine learning methods used to accomplish the goal. Author talks about the importance of data in any machine learning project and descirbes the process of collecting the dataset for the thesis. The thesis will also cover the process of analysing and cleaning the data that is being used to train the models. Author describes the implementation of selected machine learning models and analyses the results and conducts a comparison between different models. Results show that predictions were relatively accurate, but the final result is not ready to be used in live product, mainly due to the dataset that was used in the training process. In addition to machine learning model, a back-end web-service was implemented, which will be used by mobile application to make predictions. Mobile application was implemented in another bachelors thesis which was created by co-student, Reio Sedrik.

The thesis is in Estonian and contains 41 pages of text, seven chapters, 22 figures and three equations.

Lühendite ja mõistete sõnastik

ANN	Artificial Neural Network
GDPR	General Data Protection Regulation
LSTM	Long-Short-Term-Memory
MAE	Mean Average Error
RMSE	Root Mean Squared Error
RNN	Reccurent Neural Network
SQL	Structured Query Language
WSGI	Web Server Gateway Interface

Sisukord

Jooniste loetelu

1 Sissejuhatus

Teadmine tervislikest eluviisidest ja selle olulisuse mõju inimese tervisele ühiskonnas aina kasvab. Regulaarne füüsiline treening aitab ennetada paljusid haigusi nii füüsilise kui ka vaimse tervise puhul. Raskustega treeningud on kogunud palju populaarsust eriti just algajate seas, kes pole varem regulaarselt füüsilist trenni teinud. Jõutreeninguteks vajalikud kohad, kas siis sise- või välijõusaalid, on praktiliselt igas linnas üle maailma, sellepärast on ka selline treeningviis kogunud hulga populaarsust ja uusi harrastajaid.

Töö autor on tegelenud spordiga terve oma elu, alustades noores eas jalgpalli mängimisega ja hetkel keskendudes jõusaalile. Autor tõdeb, et kui ta oleks jõusaalitreeningutega alustades saanud personaalset nõu treenerilt, siis oleks areng märksa kiirem olnud ja enesekindlus jõusaali minna ka suurem. Vähene enesekindlus on üks enim levinud põhjuseid, miks paljud inimesed ei alusta jõusaalitreeningutega kuigi nad tegelikult tahaksid [1]. Autori motivatsioon rakenduse loomiseks on aidata potentsiaalselt paljusid inimesi, kes tahaksid jõusaalitreeningutega alustada ja vajaksid selleks personaalset lähenemist.

Bakalaureusetöö eesmärk on koostada masinõppe rakendus, mille abil on võimalus juhendada personaalselt iga inimese jõusaalitreeninguid. Lõpp-produkt koosneb kahest osast: mobiilirakendus ja masinõppel põhinev soovitusrakendus. Mobiilirakenduse osa realiseerib autori kaastudeng Reio Sedrik, kelle loodud rakendus kasutab autori loodud soovitusrakendust, et kasutajatele personaalselt kohandada treeningkava igaks treeninguks. Bakalaureusetööst kaugemale mõeldes on plaanis teha sellest tootest *start-up* ettevõte ja hakata toodet reaalselt müüma. Jõusaalitreeningute personaalsele modifitseerimisele on plaanis ka arendada juurde toitumise osa, mis aitab kasutajal järgida korrapärast toitumist, mis toetaks tema treeninguid ning taastumist.

Bakalaureusetöö on jaotatud seitsmesse osasse. Esimeses osas räägib autor sissejuhatavalt selle valdkonna olulisusest, töö kirjutamise motivatsioonist ning seatud eesmärgist.

Teises osas käsitletakse kirjanduse ülevaadet antud tööga seotud teemadel. Uuritakse

teadusartikleid ja spordimeditsiini uuringuid jõusaalitreeningute kohta ning nende mõjust inimese füüsilisele ja vaimsele tervisele. Uuritakse ka tööd toetavat kirjandust masinõppe meetodite kohta, mida autor kasutab probleemi lahendamiseks - rekurrentne närvivõrk (RNN), lineaarne regressioon ja logistiline regressioon.

Kolmandas osas käsitletakse treeningandmete kogumist, analüüsi ja töötlemist. Masinõppe alustala on kvaliteetsed andmed, küllaltki suures mahus. Andmeid on vaja masina õpetamiseks, mis tähendab, et masin proovib õppida inimeste treeningmustreid ja seeläbi teha soovitusi järgmiseks treeninguks. Kuna tegu on inimeste andmetega, siis on need struktureerimata ja korrapäratud. Andmete analüüs ja puhastamine on oluline ja kriitiline samm, mis määrab suuresti, milliseid erinevaid masinõppe meetodeid on võimalik kasutada ja kui hästi on võimalik masin õppima panna.

Neljandas osas käsitletakse erinevate masinõppemeetodite implementeerimist ja kirjeldamist. Masinõppe valdkonnas on keeruline täpselt öelda milline mudel või lahendus annab parima tulemuse ilma katseid tegemata. Töös kasutatakse kolme erinevat mudelit - rekurrentne närvivõrk, lineaarne regressioon ja logistiline regressioon.

Viiendas osas käsitletakse implementeeritud mudelite tulemuste analüüsi ja valideerimist. Analüüsitakse iga kasutatud mudeli tulemusi ja võrreldakse neid omavahel. Tuuakse välja iga kasutatud mudeli positiivsed ja negatiivsed küljed antud probleemi lahendamiseks. Tehakse järeldused milline mudel sobib antud probleemi lahendamiseks kõige paremini ja kas masin on piisavalt hästi õppinud, et seda kasutada lõpptootes.

Kuuendas osas kirjeldatakse arhitektuurilist poolt, kuidas töötavad koos autori loodud rakendus ja kaastudengi, Reiko Sedriku, poolt loodud mobiilirakendus. Autor räägib lühidalt ka loodud taga rakendusest mis on töö põhiosa, masinõpe, kasutamiseks lõpptootes vajalik.

Seitsmendas ja viimases osas annab autor ülevaate kokkuvõtte vormis ning teeb omalt poolt järeldusi. Kirjeldatakse ka edasisi tulevikuplaane ja arendustöid mis tuleb teha enne rakenduse turu valmidust.

2 Kirjanduse ülevaade

Käesolevas peatükis annab autor ülevaate relevantsest kirjandusest tööga seonduvatel teemadel, mis on aluseks bakalaureusetöö eesmärgi saavutamiseks. Uuritud informatsioon võetakse aluseks probleemi lahendamisele ja järelduste tegemisele.

See peatükk on jagatud kahte suuremasse kategooriasse, millest esimeses tutvustatakse kahte levinud ja efektiivset jõutreeningute kontseptsiooni, milleks on progressiivse ülekoormuse printsiip ja maha laadimise periood. Teises osas käsitletakse valitud masinõppe meetodeid tutvustavat kirjandust.

2.1 Spordimeditsiin ja teadusuuringud jõusaali treeningute kohta

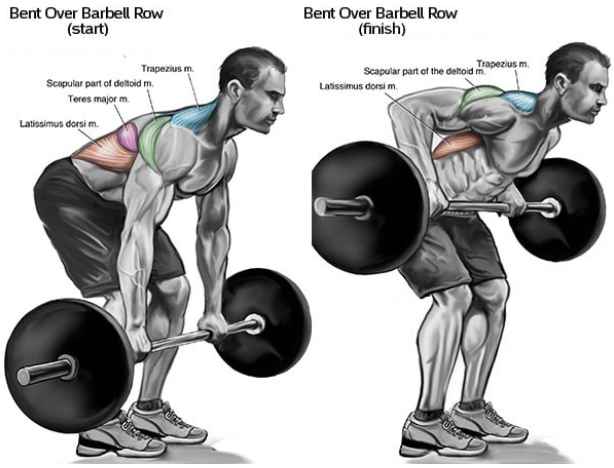
2.1.1 Progressiivse ülekoormuse printsiip

Progressiivne ülekoormus (ingl k. *progressive overloading*) on üks tähtsamaid kontseptsioone raskustreeningute puhul. See tähendab seda, et treenides peaksid pidevalt suurendama koormust ja vastupanu treenitavale lihasele, et lihas areneks. Nii teoorias kui praktikas on progressiivse ülekoormuse printsiip lihtsasti arusaadav ja teostatav, kuid paljud treenijad ei kasuta seda, kas siis teadmatusest või tulenevalt struktureerimata treeningkavast. [2]

Kui treenida järjepidevalt, näiteks neli kuni viis korda nädalas, sooritades täpselt samasid harjutusi, raskusi ja kordusi muutmata, siis jõuab treenija kindlasse arengupunkti füüsilise vormiga, millest edasi progressi ei toimu. Progress peatub seetõttu, et treenija lihased on sellise vastupanuga harjunud ja keha suudab probleemideta toimida antud koormuse all, mis kaotab põhjuse arenemiseks. Koormuse tõstmiseks jõutreeningutel on mitu erinevat viisi. Neist tuntumad ja kõige lihtsamad on kas korduste arvu tõstmine, raskuse tõstmine või seeriate arvu tõstmine. See paneb lihased suurema stressi alla ning sunnib neid kasvama ja tugevamaks saama. Parimate tulemuste saamiseks peaks tõstma nii raskusi kui ka kordusi. Teadusuuringud on leidnud, et kõige efektiivsem korduste vahemaa on 8 - 12 korduse vahel. [3]

Edasijõudnumad viisid kuidas suurendada koormust lihasele on näiteks harjutuse sooritamise ulatus, harjutuse sooritustehnika parandamine, harjutuste regulaarne modifitseerimine, aeg seeria sooritamisel, puhkuseperioodi vähendamine jt. Harjutuse sooritusulatuse (ingl k. *range of motion*) limiteeritus ja vale tehnika esinevad rohkem algajate seas. Näiteks kui sooritada rinnalt surumise harjutust, siis algaja tihtipeale ei soorita täies ulatuses antud harjutust - kang vastu rinda ja surudes käed praktiliselt sirgeks jättes küünarnukkidest end lukustamata. Seetõttu on tema lihased lühema aja raskuse all, mis vähendab üldist stressi lihasele. Regulaarne harjutuste modifitseerimine on edasijõudnutele hea viis kuidas kehale avaldada suuremat koormust, sest keha harjub liigutusega kiiresti ja see limiteerib lihasmassi kasvu. Modifitseerida harjutust on võimalik mitut moodi, muutes harjutuse sooritamisel kehaasendi nurka või haarde laiust. Näiteks sooritades kangiga seljatõmbe harjutust (ingl k. b*arbell row*) koguaeg kummardudes ette ja hoides ülakeha paralleelselt maaga, siis modifitseerida saab keha ülakeha asendit (vt Joonis 1 ja Joonis 2)[4] [5]. See avaldab lihastele varem tundmatut koormust ja stressi, mis soodustab lihaste arengut. [6]

Joonis 1. Parallel barbell row



Joonis 2. Incline barbell row

2.1.2 Maha laadimise periood

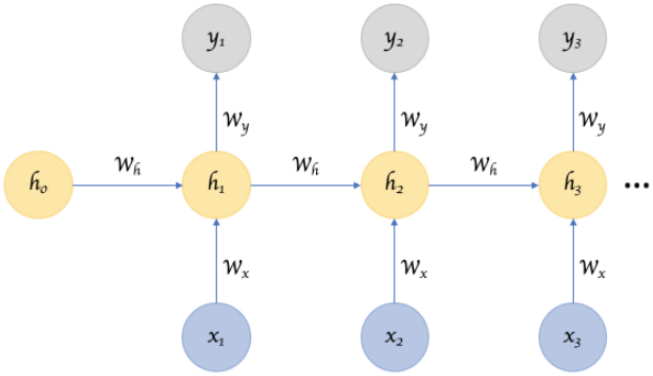
Eelmises peatükis kirjeldatud progressiivse ülekoormuse rakendamine on tõhus strateegia kuidas lihasmassi kasvatada, kuid sellega käsikäes käib ka maha laadimise periood (ingl k. *deload*). On selge, et igavesti ainult koormust tõsta ei ole võimalik ja üks hetk tuleb ette platoo, kust edasi arengut ei toimu. Selle jaoks kasutatakse maha laadimist, mis kestab ühe nädala või ühe treeningkava täisrotatsiooni. Mahalaadimisnädal aitab vähendada vigastusi, lihasvalusid, ületreenimist ja lisaks annab see ka taastumisaega lihaskudedele, mis on olnud regulaarselt stressi all. Jõutreeningud mõjuvad tugevalt kesknärvisüsteemile, millele on ka vaja anda puhkust. Mahalaadimisnädalal tehakse treeninguid väiksemate raskustega, hea tava on kasutada 60% tavapärasest raskusest, seeläbi väheneb kogu koormus kehale märgatavalt ja soodustab füüsilise kui ka vaimse tervise taastumist. [7][8]

2.2 Rekurrentsed närvivõrgud ja regressioonimudelid

Igasuguse masinõppe mudeli eesmärk on sisendile leida väljund, kasutades statistikat ja kindlaid algoritme. Masin proovib leida suurest hulgast struktureeritud andmetest mustreid ja seeläbi konfigureerida enda siseseid parameetreid vastavalt algoritmist ja mudelist, et tundmatute andmete peal teha ennustusi või klassifitseerimist vastavalt probleemile.

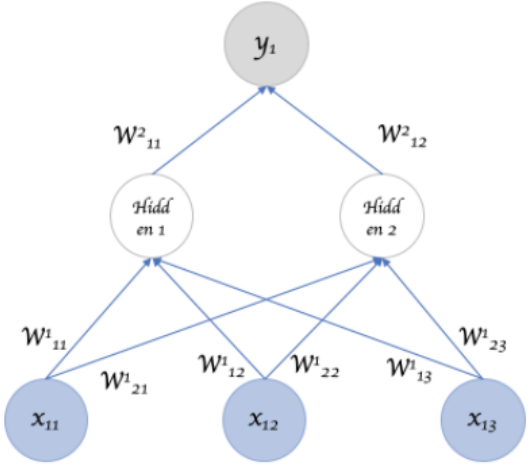
2.2.1 Rekurrentsed närvivõrgud

Rekurrentsed närvivõrgud (ingl k. *recurrent neural networks/RNN*) (vt Joonis 3) [9] on osa masinõppe teadusest, mis kuulub süvaõppe (ingl k. *deep learning*) hulka. Süvaõpe on tehnika, mis annab masinale võimaluse leida ja võimendada isegi kõige väiksemaid mustreid, mis andmetes sisaldub. Süvaõppeks nimetatakse seda tüüpi õppemeetodit just tema arhitektuuri pärast. Need mudelid koosnevad paljudest kihtidest, mis omakorda koosnevad seotud neuronitest, mis töötavad sümbioosis andmetest mustrite otsimisel. [10]



Joonis 3. Rekurrentne närvivõrk peidetud olekuga

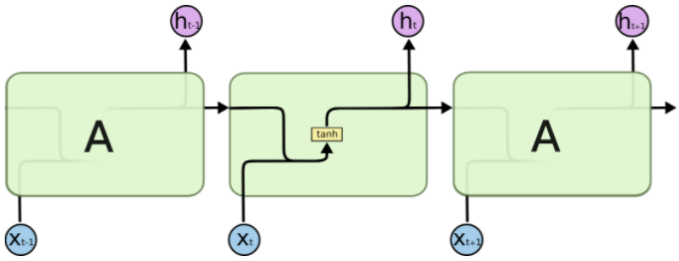
Rekurrentsed närvivõrgud on edasiarendus tehislikest närvivõrkudest (ingl k. a*rtificial neural network/ANN*) (vt Joonis 4)[11]. Põhierinevus tehislikel ja rekurrentse närvivõrgu vahel on rekurrentse närvivõrgu omadus töödelda ajalisi või jadana andmeid. Idee on andmete jada vahelisi seoseid kasutada, milles tihti leidub olulist informatsiooni. Näiteks lause ennustuse probleemi puhul on oluline iga sõna kontekst. Tehisnärvivõrguga (vt Joonis 4) ei ole võimalik jadalist andmehulka protsessida, see tähendaks lihtsalt rohkem sisendeid, küll aga rekurrentsete närvivõrkude puhul on võimalik edasi kanda tähendusrikast seost sisendite vahel. Rekurrentne närvivõrk hoiab meeles eelmisi tulemusi ja tema tulevased otsused on mõjutatud sellest, mida ta on varem õppinud. Rekurrentse närvivõrgu väljund on mõjutatud lisaks kaaludele, mis on rakendatud sisenditele, ka peidetud olekuvektorit, mis hoiab endas konteksti tuginedes eelmistele sisenditele ja väljunditele. [12]



Joonis 4. Tehislik närvivõrk kolme sisendiga, ühe peidetud kihi ja ühe väljundiga

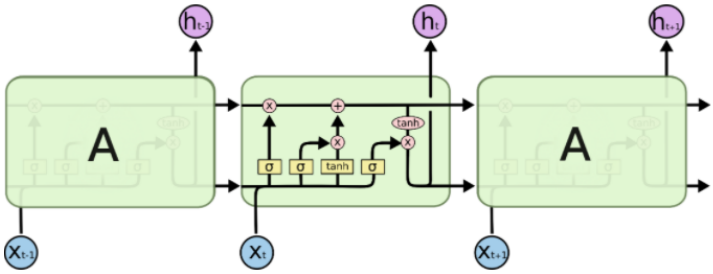
Bakalaureusetöö valitud probleem on ennustada treenijale järgmiseks treeninguks raskused ja kordused põhinedes tema eelmisele sooritusele. Teoorias sobib selle probleemi lahendamiseks rekurrentne närvivõrk hästi, kuna ennustusi tehes mängivad treenija varasemad treeningud suurt rolli. Sellele tuginedes otsustas autor kasutada töös rekurrentse närvivõrgu edasiarendust – *Long-short-term-memory* (LSTM) närvivõrku.

LSTM on spetsiifiline rekurrentse närvivõrgu tüüp, mis on disainitud mustrite ja konteksti leidmisest pikas ajas. Tavaline rekurrentne närvivõrk suudab ka minevikku siduda hetkeseisuga, aga kui see vahemaa muutub liiga pikaks, siis ei saa ta enam hakkama. LSTM erinevus tavalisest rekkurentsest närvivõrgus seisneb tema korduvuse mooduli struktuurist. RNN korduvmooduli struktuur on lihtsa ehitusega näiteks üksik tanh kiht (vt Joonis 5)[13]. [14].



Joonis 5. RNN ühekihiline korduv moodul

LSTM kasutab sarnast ahelalikku struktuuri kuid erinevus seisneb korduvmooduli struktuuris, ühe kihi asemel on neli kihti mis käituvad väga spetsiifilisel viisil (vt Joonis 6)[15]. Selle joonise diagrammi iga joon kannab endas tervet vektorit, alates ühe sõlme väljundist kuni teiste sisenditeni. Roosad ringid tähistavad suunalisi operatsioone, näiteks vektorite lahutamine. Joonisel olevad kollased ristkülikud on juba õpitud närvivõrgu kihid. Ühendavad jooned joonisel tähistavad liitumist, laiali minevad jooned tähistavad kantava vektori kopeerimist ja koopiate suunamist erinevatesse asukohtadesse. LSTM erilisus peitub “raku” olekus (ingl k.. *cell state*), mida väljendab horisontaalne joon Joonise 6 ülaosas [15].



Joonis 6. LSTM neljakihiline korduv moodul

2.2.2 Regressioonimudelid

Regressioon on statistika tööriist leidmaks suhet tulemuse ja ühe või rohkema sõltumatu muutuja vahel. See on laialdaselt kasutusala leidnud ennustamise probleemide lahendamisel ja seoste otsimiseks muutujate vahel. Lineaarne regressioon on tavaline regressiooni analüüsi vorm, mis eeldab, et tulemuse ja sõltumatute muutujate vahel on lineaarne seos. Regressiooni eesmärk on leida võrrand, mis kirjeldab kõige paremini tulemuse ja muutujate suhet.[16]

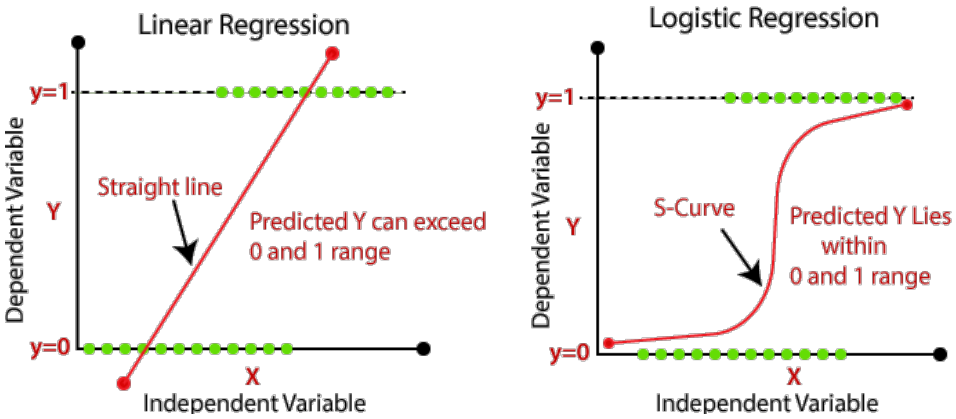
Lineaarne regressioon on arendatud välja statistika valdkonnas ja on kasutusele võetud masinõppe mudelite koostamiseks oma lihtsuse tõttu. Lineaarse regressiooni mudel on lineaarvõrrand, mis kombineerib sisendväärtused selliselt, et lahend on ennustatav tulemus. Lineaarse regressiooni puhul on nii sisend kui ka väljund numbrilisel kujul. Lihtne näide lineaarsest regressioonivõrrandist on näiteks:

Võrrand 1. Lineaarne regressioonivõrrand

$$Y = \beta 0 + \beta 1 x$$

Lineaarvõrrandis määratakse igale sisendile faktor, mida nimetatakse koefitsendiks ja kirjutatakse Kreeka tähena Beta. Lisatakse ka üks lisa koefitsent mis annab sirge võrrandile määratud vabaduse. Mudeli suutlikkuse hindamiseks kasutatakse näiteks keskmist absoluutset viga (ingl k. *mean absolute error/MAE*) ja ruutjuur ruutkeskmisest veast (ingl k. *root mean squared error/RMSE*). [16]

Sarnaselt lineaarsele regressioonile (vt Joonis 7)[17] on ka logistiline regressioon (vt Joonis 7)[17] esmalt kasutusele võetud statistikas ja laenatud masinõppe mudelite koostamiseks. Logistilise regressiooni puhul on ennustus binaarsel kujul - 0 või 1 ja lineaarnse regressiooni puhul võivad väärtused 0 ja 1 välja minna.[18]

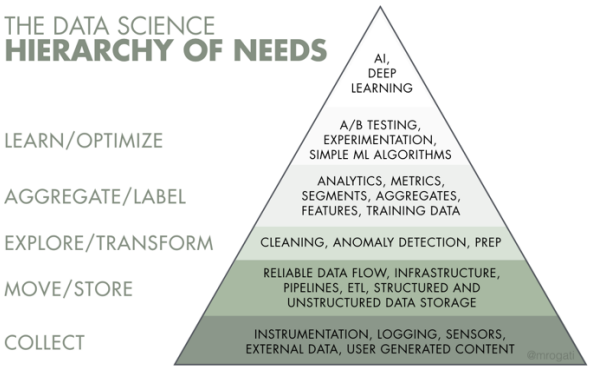


Joonis 7. Lineaarne- ja logistiline regressioon

3 Andmed

Andmed on igasuguse masinõppe algoritmi alustala. Andmed on hoiavad endas informatsiooni, seoseid, mustreid ja tähendust, mida algoritm proovib sealt leida ja mille põhjal algoritm ennast õpetab. Kui andmeid ei ole piisavalt, s.t pole kaetud kõik või enamus juhtumeid mingi probleemi lahendamisel, siis mudel jääb selliste ennustuste puhul hätta. Andmed peavad sisaldama endas piisavalt mustreid ja konteksti antud probleemi kirjeldamiseks, s.t andmete rikkalikkus peab olema piisav suutliku mudeli loomiseks.

Enne algoritmide ja mudelite koostamist, tuleb teha korralik eeltöö andmetele. Kõige olulisem protsess on andmete kogumine (vt Joonis 8) [19]. See tähendab andmete vajaduse defineerimist andmekogu loomiseks ja selle kättesaadavust. Kogumisele järgneb esmane analüüs muutujatele andmestikus, et mõista erinevaid parameetreid, näiteks mis tüüpi parameetriga on tegu või millistes erinevates vahemikkudes andmed on, kas on puudulikke andmed jne. Järgmine loogiline samm on andmestikust välja filtreerida mõjuvõimsad muutujad ja ülejäänud müratekitajad eemaldada. Levinud on ka uute muutujate loomine olemasoleva andmestiku pealt, hea näide on pildi klassifikatsiooni probleemi puhul võimalus sama pilti pöörata, suurendada ja muid parameetreid muuta, et saada masina jaoks ühest pildist mitu erinevat pilti. Kui eelnevad sammud on tehtud, siis alles nüüd on aeg hakata töötama algoritmide ja mudelite kallal. [20]



Joonis 8. Andmeteaduse vajaduste püramiid

3.1 Andmete kogumine

Bakalaureusetöö vajalikud andmed on inimeste jõusaalitreeningute andmed, mis sisaldavad endas minimaalselt järgmisi muutujaid:

- Missugust harjutust sooritatakse
- Korduste arv ühe seeria jooksul
- Seeriate arv ühe harjutuse jooksul
- Korduse jooksul kasutatud raskus

Nende muutujate pealt on võimalik arvutada treeningu kogu maht (ingl k. *training volume*) mis arvutatakse järgnevalt:

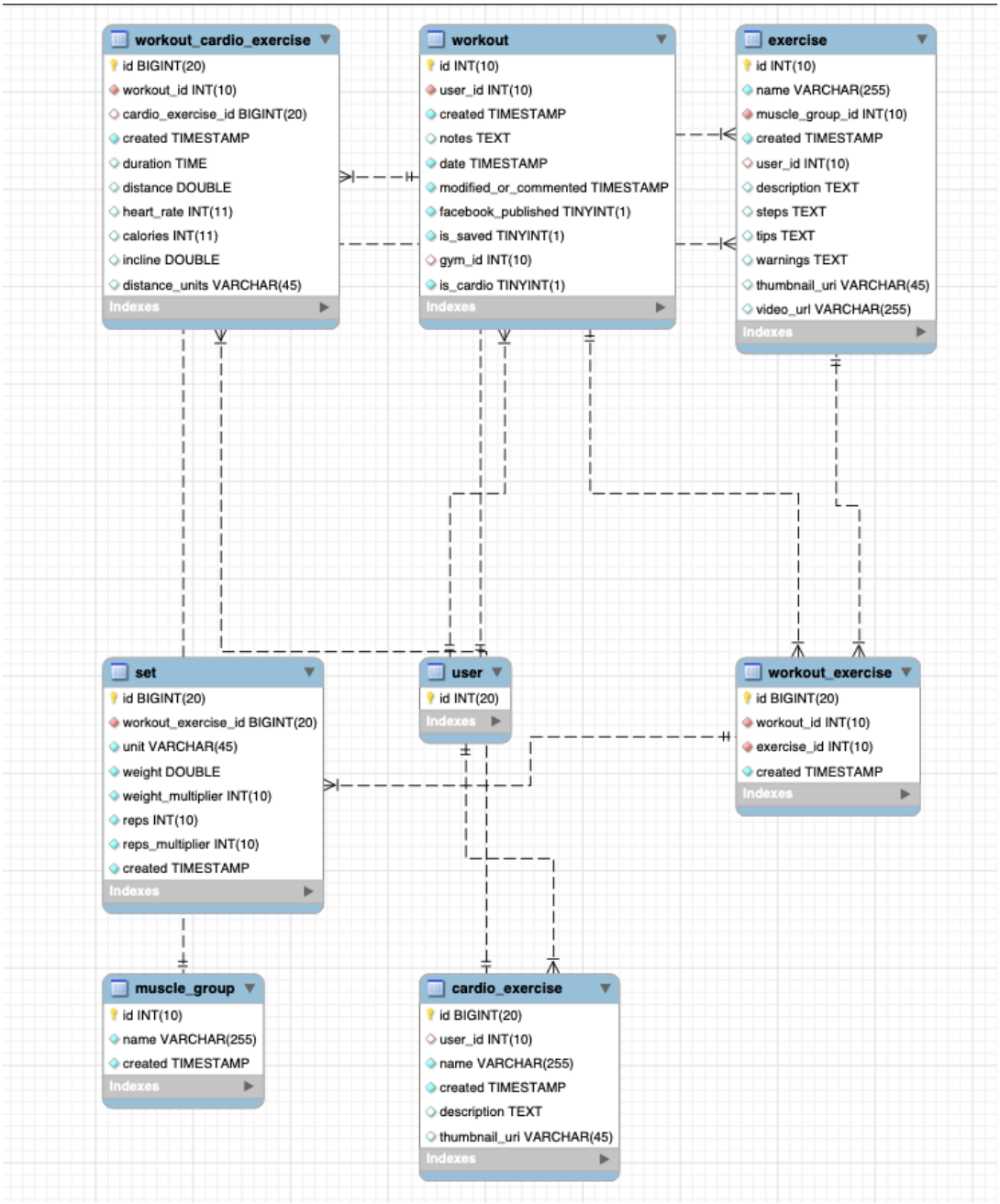
Võrrand 2. Treeningmahu arvutamise valem

$$Maht = kordused * raskused * seeriad$$

See on oluline osa mida treenimise juures jälgida just eelnevalt kirjeldatud progressiivse ülekoormuse printsiibi seisukohast. Ideaalses andmekogus antud probleemi lahendamiseks peaks olema lisaks eelnevalt nimetatud muutujatele ka järgmised:

- Treenija vanus
- Treenija kogemus jõusaali treeningutel
- Treenija kehakaal
- Treenija pikkus
- Treenija sugu
- Puhkuse aeg seeriate vahel

Kvaliteetsete andmete puudus ja kättesaamatus on üks peamisi põhjuseid, miks paljud masinõppel põhinevad projektid ebaõnnestuvad [21]. Suure tõenäosusega kvaliteetsed ja vajalikus mahus andmehulgad internetiavarustes ei vedele, eriti kui tegu on spetsiifilise valdkonnaga. See väide osutus tõeks ka selle bakalaureusetöö puhul, kus vabavaralisi andmekogusid asjakohaste andmetega saada ei olnud. Andmekogu bakalaureusetöö koostamisel saadi ettevõttelt Gymwolf, mis on Eestis loodud ettevõte jõusaalitreeningute järjepidamiseks, ehk virtuaalne treeningpäevik [22]. Gymwolf projektijuht ja asutaja olid nõus jagama kogutud treeningandmeid, mis sisaldasid endas eelnevalt nimetatud minimaalseid vajalikke muutujaid (vt Joonis 9)[23].



Joonis 9. Gymwolf andmebaasi skeem

Esialgne andmekogu ei sisaldanud informatsiooni treenijate kohta, s.t sugu, vanust, kaalu ja pikkust, kuna see oleks rikkunud General Data Protection Regulation (GDPR) andmekaitse seadust [24]. Esialgne andmekogu koosnes ~37 tuhande inimese treeningutest, ~200 tuhandest treeningust ja ~2,4 miljonist sooritatud seeriast.

3.2 Andmete analüüs ja töötlemine

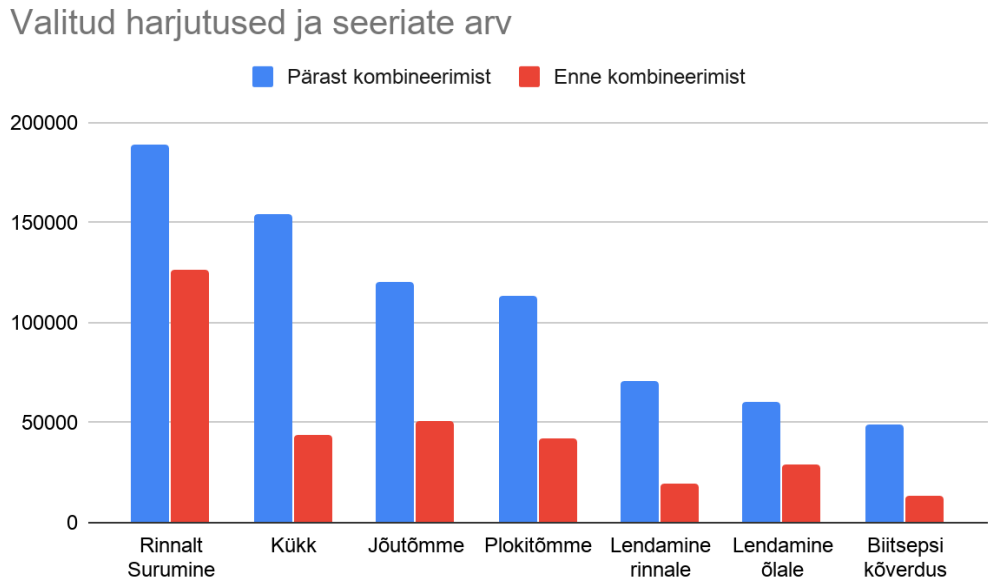
Inimeste andmed on oma loomult korrapäratud, mis suurendab oluliselt mustrite leidmise raskustaset. Korrapäratud on nad inimloomuse käitumise tõttu, antud kontekstis

korrapäratused tekivad näiteks pauside tõttu regulaarse treenimise vahel, ei märgita järjepidevalt treeningus sooritatut, märkimisel tekivad inimlikud vead jne. Autor otsustas kõikide harjutuste ennustamise asemel valida välja enamlevinud harjutused, mille kohta on antud andmekogus kõige rohkem informatsiooni. Valitud andmestikku iseloomustab Joonis 10. [25]

Andmete analüüsiks ja puhastamiseks kasutas autor MySQL andmebaasi ja SQL andmebaasi keelt. MySql on vabavaraline relatsioonilise andmebaasi manageerimissüsteem ja SQL on struktureeritud päringukeel (ingl k. S*tructured Query Language*). Algset andmebaasi majutati Amazon'i pilvepõhisel majutamiskeskkonnas Amazon Web Services, Relational Database service [26].

Esiteks eemaldas autor andmekogust kindlate parameetrite järgi vigased andmed. Kaks põhilist parameetrit, mille järgi puhastamine sooritati, oli korduste arv ja raskus. Kui korduste arv ületas 40 kordust, siis määras autor selle seeria vigaseks - nii suure arvu korduste juures on enamasti tegu vigase sisendiga või väga spetsiifilise treeningstiiliga, mida andmetes esineb vähe ning sealt mustreid õppida ei ole võimalik, pigem tekitab see segavat müra. Teiseks parameetriks oli seeria raskus, autor määras vigaseks andmetest kõik seeriad mille ühe korduse raskus oli suurem kui 200 kg, samadel põhjustel nagu ülisuurte korduste arvu juures, see on müra tekitaja.

Kuna Gymwolf rakenduses on võimalus ise harjutuste nimesid sisestada mida treeningul sooritada (järeldus tehtud andmebaasi andmete põhjal), siis oluline samm oli kombineerida kõik harjutused, mis viitasid samale harjutusele. Seda tehti kõikide harjutuste kohta ja selle järgi valiti ka välja seitse harjutust (vt Joonis 10), mille kohta oli kõige rohkem andmeid.

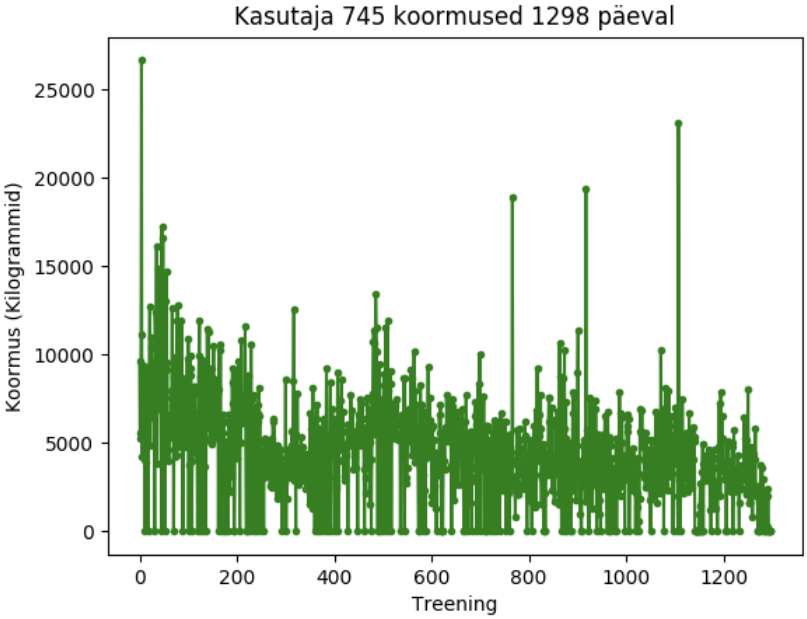


Joonis 10. Kasutatud harjutused ja seeriate arv, enne ja pärast harjutuste kombineerimist

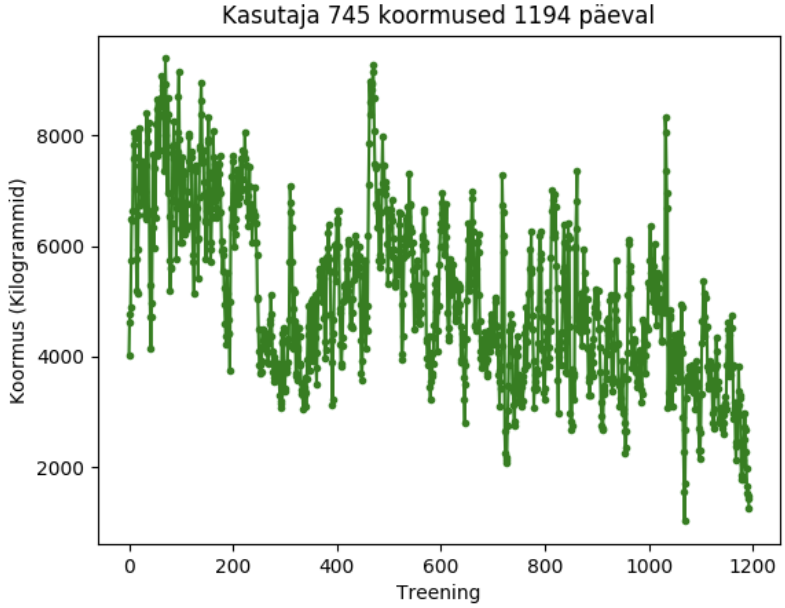
Sügavamat analüüsi igale harjutusele tegi autor kasutades programmeerimiskeelt Python ja seal olevaid andmeteaduse pakette. Python on üks populaarsemaid tehnoloogiaid andmeteaduses mitmel põhjusel: lihtsasti loetav kood, kõrge taseme keel, laialdane teekide ja lisapakettide valik just teadusvaldkonnas jne. Mõned teegid mida autor kasutas olid Pandas, Matplotlib, Sklearn ning Numpy [27-30].

Andmeid analüüsiti süvitsi iga kasutaja koha ja aluseks oli treeningmaht (vt Võrrand 2). Kuna kahe järjestiku treeningu mahud on sarnased, siis otsustas autor kasutada jooksvat viie päeva keskmist mahtu. See lähenemine aitas eemaldada andmetes müra, mis muudab mustrite leidmise lihtsamaks. Autor eemaldas iga kasutaja kohta ka treeningud, mille maht jäi alla 25% piiri või üle 75% piiri võttes arvesse kasutaja treeningmahtude amplituudi.

Järgnevatel joonistel on selgelt näha, et töötlemata andmetes (vt Joonis 11)[31] on palju korrapäratust ja müra. Sellise andmekogu pealt on masinal väga raske mustreid leida ja õppida. Pärast andmete töötlemist (vt Joonis 12)[32] tuleb välja selgem muster ja andmetes ei ole enam nii suuri korrapäratusi.



Joonis 11. Töötlemata rinnalt surumise harjutuse andmed kasutaja 745 kohta



Joonis 12. Töödeldud rinnalt surumise harjutuse andmed kasutaja 745 kohta

Autor otsustas vigased andmed andmekogust eemaldada, kuid üks võimalus oleks veel vigased andmed interpoleerida. See lahendus oleks kindlasti olnud valiidne, kuid tekitaks võimaluse, et andmetesse ehitatakse mustreid mida seal tegelikult ei olnud. Selle tõttu otsustas autor vigaste andmete interpoleerimisele need andmekogust eemaldada.

3.3 Andmete ettevalmistamine

Järgmine samm töödeldud andmetele on nende ettevalmistamine masina jaoks. Mida standardsemal ja normaliseeritumal kujul on sisendandmed masinale, seda kiirem on õppeprotsess ja täpsem tulemus, sest masina jaoks on olulised numbrid ja nende vahelised seosed, mitte iseloomustavad atribuudid mida inimene tajub (värvid, heli, lõhnad jms).

Kuna andmed on jadaandmed, siis on väga oluline, et masinale sisendiks antakse need õiges järjekorras. Sisendiks kõikidele kasutatud mudelitele andis autor 15 päeva tegelikud skaleeritud mahud lõigendatud iga kasutaja kaupa ja väljundiks ennustati järgmise treeningu maht (vt Joonis 13)[33]. Andmeid skaleeriti nulli ja ühe suurusjärgu vahele kasutades Sklearn [26] teegi MinMaxScalar [34] funktsiooni. Üks oluline avastus mis autor tegi just andmete skaleerimisel oli see, et kui skaleerida iga kasutaja andmed eraldi, siis lõpptulemus oli võrdlemisi parem kui skaleerides kõikide kasutajate andmed koos. Põhjuseks võib olla see, et esimene skaleerimisviis võimendas kasutajapõhiseid mustreid, mis kogu andmekogu skaleerimisel korraga kadusid ära.

M = treeningu maht
n - treeningute arv

Sisend:		Ennustus:
[M1, M2, M3, … , M14, M15]	——>	[M16]
[M2, M3, M4, … , M15, M16]	——>	[M17]
[M3, M4, M5, … , M16, M17]	——>	[M18]
…		
[Mn-15, Mn-14, Mn-13, … , Mn-1, Mn]	——>	[Mn+1]

Joonis 13. Illustreeriv sisend

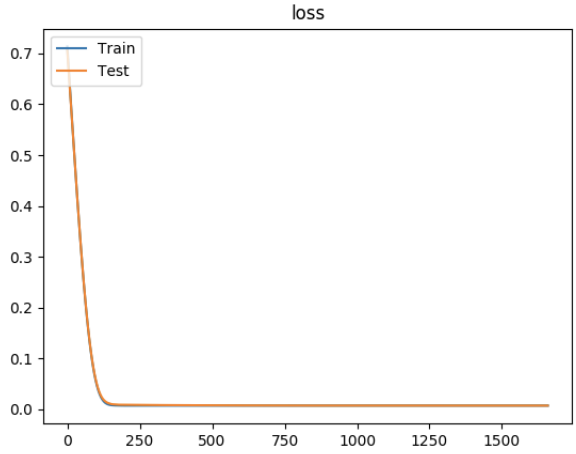
4 Masinõppe mudelid

See peatükk on jagatud kolmeks osaks, mis igaüks kirjeldab autori valitud mudeleid, nende ehitust ja iseloomu. Valitud mudelid olid rekurrentne närvivõrk LSTM, lineaarne regressioon ja logistiline regressioon. Autor toob siinkohal välja olulise nüansi tulemustele konteksti loomiseks. Järgnevalt kirjeldatud mudelid on treenitud tavakasutajate andmete põhjal, mis ei pruugi olla kvaliteetsed. Kvaliteetsed andmed oleksid sertifitseeritud personaaltreeneri jälgimisel sooritatud treeningute andmed. Kui autor kirjutab töös, et mudeli eksimus on näiteks 5% siis väljendab see mudeli suutlikust õppida antud karakteristiga andmekogult, mitte et 5% jäi ideaalsest tulemusest puudu, mis oleks soovitatud personaaltreeneri poolt.

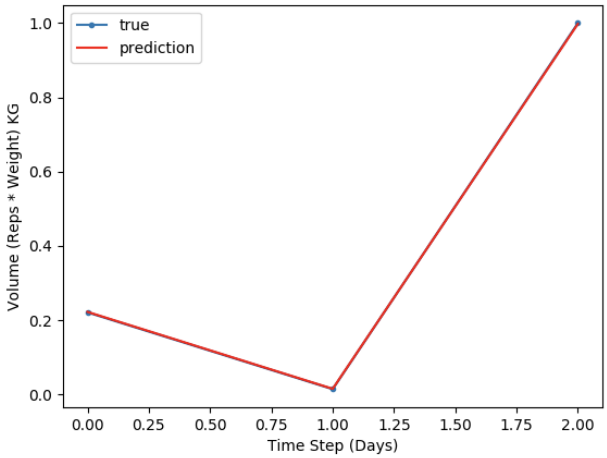
4.1 Long-Short-Term-Memory (LSTM) närvivõrk

Esimene mudel mida autor proovis bakalaureusetöö probleemi lahendamiseks oli *Long-Short-Term-Memory* (LSTM) [14] rekurrentne närvivõrk, mis on tuntud oma suutlikkuse tõttu sarnaste probleemide lahendamisel. Selle mudeli koostamisel kasutati Google arendatud masinõppe raamistikku Tensorflow 2.0 ja selle alamteeki Keras [35].

Seda närvivõrgu arhitektuuri on kasutatud näiteks uuringus “Modeling Heart Rate and Activity Data for Personalized Fitness Recommendation”, mis on antud tööga sarnases valdkonnas [36]. Mudeli koostamisel kasutas autor Andreij Karpathy poolt kirjeldatud süstemaatilist protsessi [37]. Esiteks võeti minimaalne andmekogu, näiteks üks kuni kolm treeningut ühe inimese andmetest ja prooviti üle sobitada (ingl k. *overfitting*) minimaalne mudel, mis koosnes ainult ühest LSTM kihist ja ühest väljundkihist. Oodatav tulemus oli saada viga (ingl k. *loss*) nullilähedaseks ning ennustatav tulemus ideaalselt täpne, ennustuse tegemisel kasutati täpselt sama andmekogu, millega mudelit treeniti. See valideerib mudeli omaduse õppida antud andmetelt (vt Joonis 14 ja Joonis 15)[38][39].



Joonis 14. Närvivõrgu vea minimaliseerimise

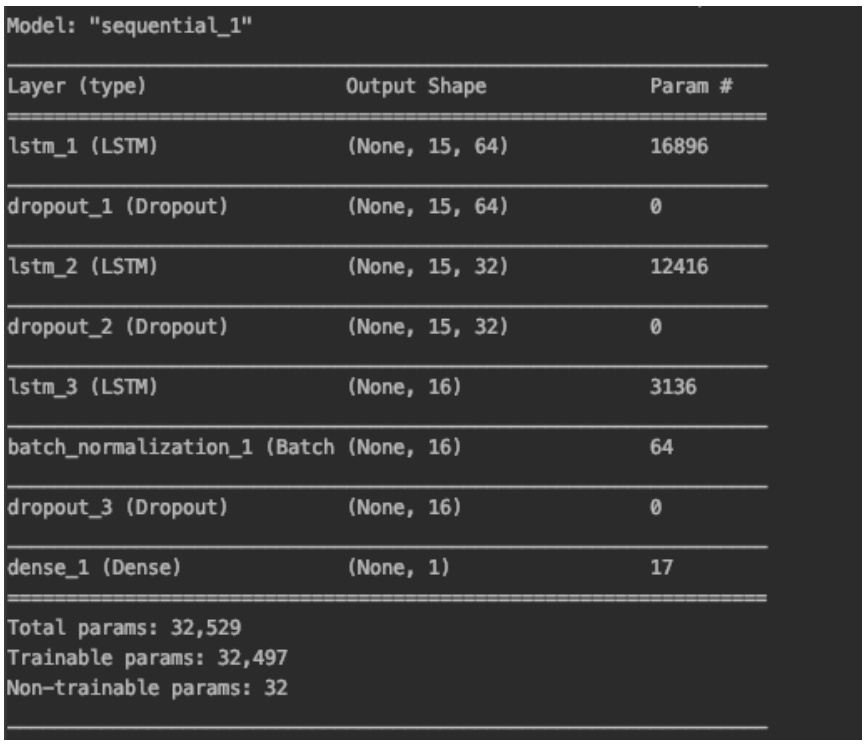


Joonis 15. Ideaalne ennustus üle sobitatud mudeliga

Kui ideaalne ennustus on saavutatud ja mudeli viga on null, siis järgmine samm on lisada järk-järgult rohkem andmeid ja suurendada mudeli keerukust, lisades rohkem kihte ja suurendada neuronite arvu igas kihis.

Kuna andmed on jadalisel kujul siis on ka mudeli tüüp peab olema järjestikune (ingl k. *sequential*). Lõplik LSTM mudel (vt Joonis 16) [40] koosneb kolmest LSTM kihist ja ühest tihedalt seotud väljundi kihist. Esimeses LSTM kihis on 64 neuronit, teises 32 neuronit ja viimases LSTM kihis on 16 neuronit. Iga LSTM kihi järel on väljakukkumis kiht (ingl k. d*ropout*). Väljakukkumis kiht võimaldab eemaldada kindla arvu neurone treenimisel, mis aitab vältida üle sobitamist (ingl k. *overfitting*). Ülesobitumine tähendab seda, et mudel sobitab oma sisesed parameetrid väga tihedalt vastavusse treeningandmetega. See annab meile valesid indikaatoreid, et justkui mudel on täpne, kuid reaalsuses on ta kohandatud treeningandmete järgi. Uute, varem nägemata andmete

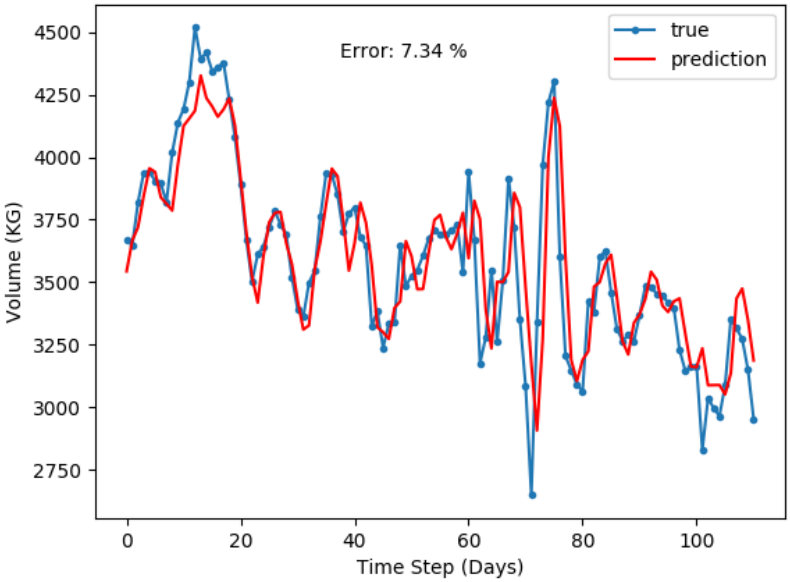
peal väheneb täpsus oluliselt kui mudel on üle sobitatud. Mudeli treeningu suutlikkuse valideerimisel kasutab autor ADAM optimeerijat [41]. ADAM optimeerija on esimese ja teise järgu hetkedel põhinev sohhastiline laskumise meetod (ingl k. *Gradient Descent*) [42]. Vea arvutamisel kasutab autor mudelis mediaani ruudu valemit (ingl k. *Mean Squared Error/MSE*), mis on antud probleemi lahendamiseks efektiivne, kuna see meetod karistab suurte eksimuste korral rohkem kui näiteks tavaline keskmise mediaani vea (ingl k. *Mean Average Error/MAE*) valem [43].



```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm_1 (LSTM)                (None, 15, 64)            16896
_________________________________________________________________
dropout_1 (Dropout)          (None, 15, 64)            0
_________________________________________________________________
lstm_2 (LSTM)                (None, 15, 32)            12416
_________________________________________________________________
dropout_2 (Dropout)          (None, 15, 32)            0
_________________________________________________________________
lstm_3 (LSTM)                (None, 16)                3136
_________________________________________________________________
batch_normalization_1 (Batch (None, 16)                64
_________________________________________________________________
dropout_3 (Dropout)          (None, 16)                0
_________________________________________________________________
dense_1 (Dense)              (None, 1)                 17
=================================================================
Total params: 32,529
Trainable params: 32,497
Non-trainable params: 32
_________________________________________________________________
```

Joonis 16. LSTM mudeli lõplik kokkuvõte

Autor katsetas kahte erinevat lahendust lõpliku mudeli koostamiseks, esiteks üks mudel mis oskab ennustada valitud seitsme jaoks ja igale harjutusele eraldi mudel. Efektiivsemaks osutus koostada igale harjutusele eraldi mudel, kokku seitse mudelit, mis ennustavad igaüks kindlat harjutust. Esimese lahenduse korral oli keskmiselt ennustuse möödapanek tegelikust 15% ja teise lahenduse (vt Joonis 17)[44] korral 8%.



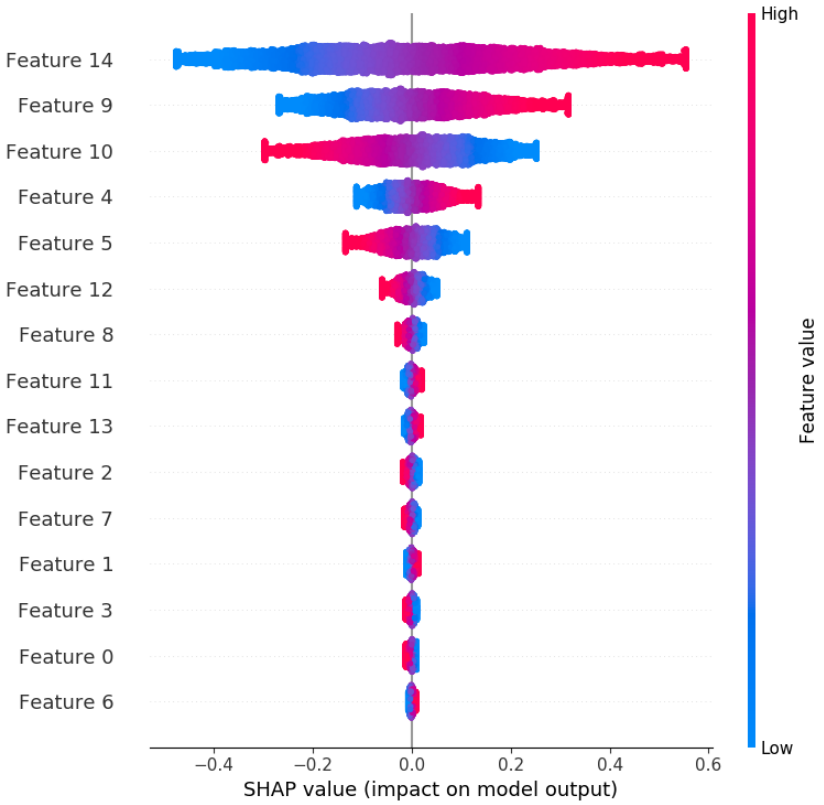
Joonis 17. Lõpliku LSTM mudeli jõutõmbe harjutuse ennustus ja tegelik tulemus

4.2 Lineaarne regressioon

Teine masinõppe mudel mida autor otsustas bakalaureusetöö probleemi lahendamiseks kasutada on lineaarsel regressioonil põhinev lahendus. Autor valis selle mudeli, kuna lineaarne regressioon on üks lihtsamatest mudelitest oma ehituse kui ka tööpõhimõtte poolest. Erinevalt tavalistest ja rekurrentsetest närvivõrkudest, kus tööpõhimõte ja arhitektuur on oluliselt keerulisem, on lineaarne regressioon enamasti kergesti mõistetav ja üheselt tõlgendatav. Just selle lahenduse keerukuse madalus oli põhjus, miks autor otsustas seda lahendust proovida. Implementeerimisel kasutas autor Sklearn [29] Pythoni programmeerimiskeele teeki, kus sisaldub lineaarse regressiooni mudel, mille saab luua teoreetiliselt mõne rea koodiga. Sisendiks katsetas autor erinevaid ajavahemikke 30 päevast kuni 15 päevani ja selgus, et kõige olulisem regressioonimudeli jaoks on 15 päeva treeningmahud viie päeva jooksva keskmisega (vt Joonis 18)[45] ja väljundiks on järgmise treeningu maht. Seega sisend ja väljund on täpselt samasugune nagu LSTM mudeli puhul (vt Joonis 13).

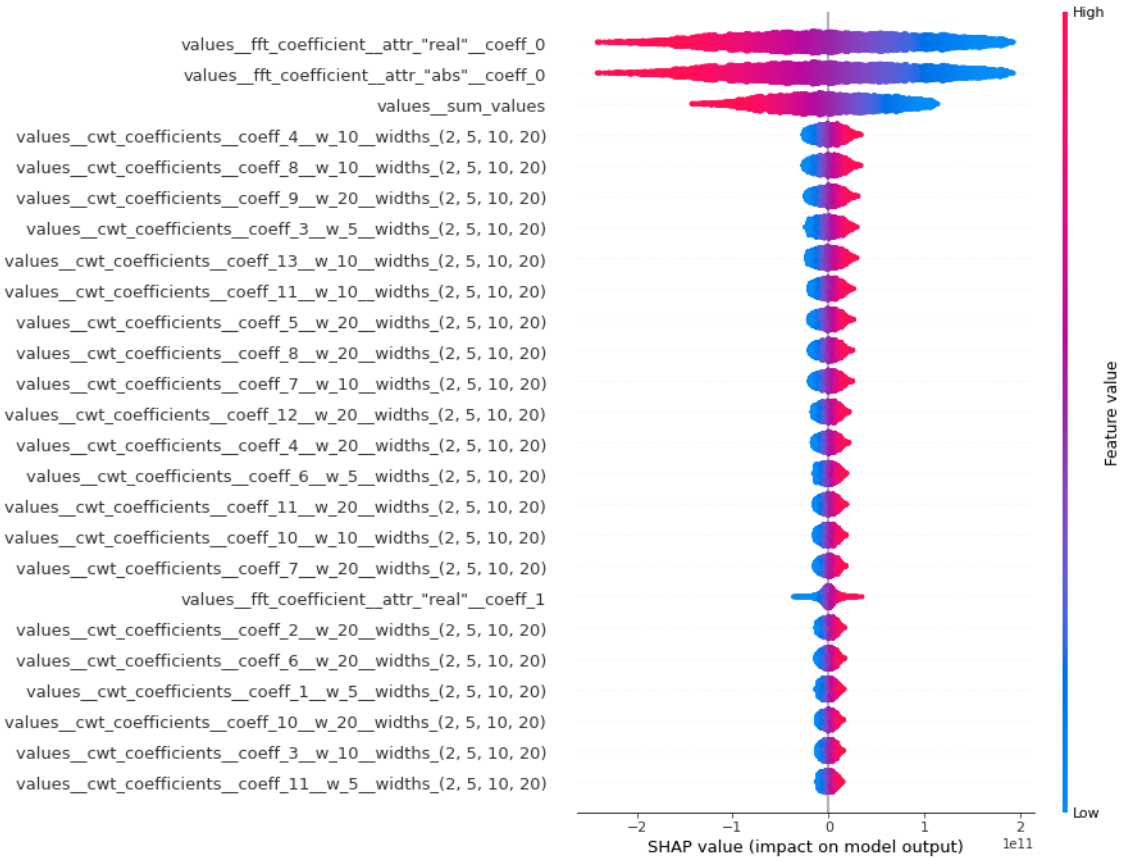
Lineaarse regressiooni puhul on oluline analüüsida ja leida informatiivsed tunnused (ingl k *features*) mudelile, mille pealt on võimalik õppida, lihtsalt eelnevate treeningute kogumahust jääb väheks. Treenides mudelit ainult 15 eelmise treeningu mahu pealt sai autor tulemuse, mille erinevus tegelikust oli 11%.

Esmane tunnuste analüüs tehti SHAP [46] Pythoni teegi abi, millega on võimalik arvutada, kui suure kaaluga on etteantud tunnused lõpptulemuse arvutamisel (vt Joonis 18)[45]. Joonisel 18 on tunnus ühe treeningu maht vastavalt selle numbrile, näiteks "Feature 14" tähendab 15nda treeningu mahtu antud kontekstis. Jooniselt on selgelt näha, et mida kaugemale jääb treenigu aeg, seda vähem mõjutab see ennustust.



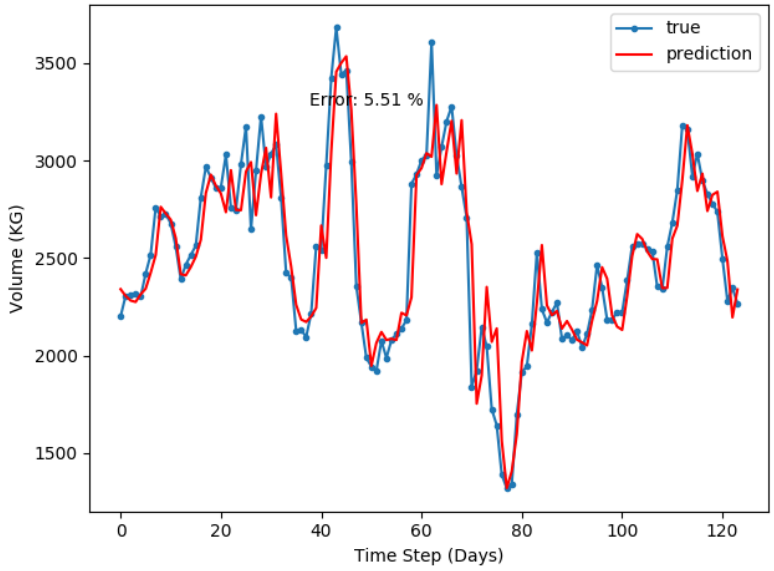
Joonis 18. Tunnuste tähtsus väljundi arvutamisel

Hea tava on olemasolevate andmete pealt arvutada erinevaid väärtusi, mida kasutada mudeli treenimisel, näiteks mingi perioodi keskmine, mediaan, maksimaalne väärtus, minimaalne väärtus jne. Ka selle operatsiooni jaoks on Pythonis olemas teek, mis aitab seda teha, Tsfresh [47]. Tsfresh automatiseerib jadaandmetest tunnuste arvutamise ning omab erinevaid meetodeid, kuidas hinnata tunnuste olulisust ennustuste tegemisel. Lisades tunnuste hulka Tsfresh teegi abil arvutatud väärtused, on sisendis kokku 349 tunnust 15 asemel. Kõik arvutatud väärtused ei ole kindlasti mudeli jaoks olulised, tuleb välja pruukida ainult need, mis on suure kaaluga. Pärast tunnuste filtreerimist jäi järele 196 tunnust, kõige olulisemad neid on kujutatud Joonisel 19 [48]. On näha mitmeid Fourier'i [49] transformatsiooni koefitsiente (fft_coefficient) ja ka Waveleti transformatsiooni koefitsiente [50] (cwt_coefficient).



Joonis 19. Tunnused järjestatud tähtsusejärjekorras

Pärast tunnuste lisamist ja filtreerimist paranes ennustuse täpsus ligikaudu 5% protsenti. Seda protsessi tuleks korrata mitmeid kordi ja modifitseerida oluliste tunnuste hüperarameetreid, kuid ajalise piirangu ja bakalaureusetöö mahu tõttu autor otsustas jääda antud tulemuse juurde, milleks on keskmine ennustuse viga 6% (vt Joonis 20)[51].



Joonis 20. Lõpliku regressioonimudeli ennustus rinnalt surumise harjutuse kohta

4.3 Logistiline regressioon

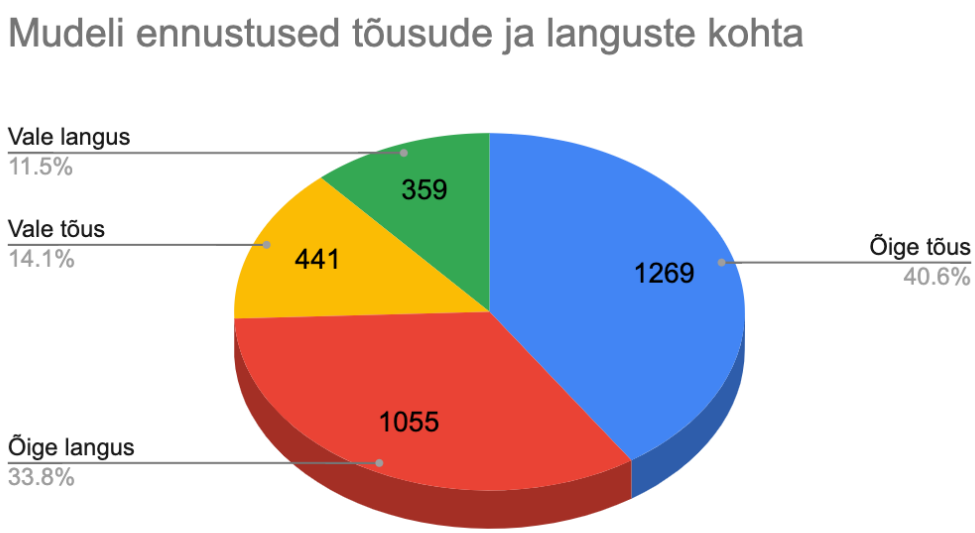
Viimane masinõppe mudel mida autor bakalaureusetöö probleemi lahendamiseks proovis, oli logistilisel regressioonil põhinev mudel. Sarnaselt lineaarsele regressioonile on ka logistiline regressioon oma olemuselt lihtsam kui rekurrentne närvivõrk. Logistilist regressiooni kasutatakse enamasti klassifitseerimise probleemide lahendamiseks, kuna selle mudeli tulemus on binaarne, s.t 0/1 stiilis. Selle mudeliga ei saa ennustada kui suurel määral peaks treeningmahtu muutma, kuid võimalik on leida vastus küsimusele, kas mahtu tõsta või langetada. Selle mudeli kasutamine lõpp rakenduses eeldab lisaarendust, kuna tuleb luua lahendus mis leiaks, kui palju otsustatud suunas mahtu muuta. Teoreetiliselt see ei ole takistav faktor mudeli kasutamiseks, sest kui mõelda jõusaali treeningute peale, näiteks rinnalt surumine, siis raskuste inkremendid on kindla raskusega. Standardsed väärtused hakkavad 1,25 kg ketastest, st minimaalne raskuse inkrement sellisel juhul on 2,5 kg kui paigutada kangi mõlemale poole üks ketas. Üks puudujääk mis sellise lahendusega kaasneb, on võimalus raskus samaks jätta, mis tegelikkuses on normaalne juhtum.

Autor kasutas mudeli implementeerimisel Sklearn [29] Pythoni teeki, mis sisaldab juba valmis logistilisel regressioonil põhinevat masinõppe mudelit. Sisend mudelile on sama nagu kirjeldatud logistilise regressiooni põhjal, 15 päeva treeningmahud ja juurde arvutatud tunnused. Sisendandmetele vastavusse arvutati tõusu või languse vaste arvestades keskmisi tulemusi, võrreldi 15 treeningu sisendandmete keskmist mahtu ja järgmise viie päeva keskmist mahtu. Kui järgneva viie päeva keskmine oli suurem kui eelneva 15 päeva keskmine, siis õige tulemus oli mahtu tõsta ja vastupidi (vt Võrrand 3). Tunnuste leidmise ja filtreerimise protsess on täpselt sama nagu lineaarse regressiooni puhul ning tulemused olid samad (vt Joonis 19).

Võrrand 3. Tõusu või languse sildi arvutamise võrrand

$$Y = \sum_{i=0}^{14}(x_i) > \sum_{i=14}^{v19}(x_i)$$

Mudeli suutlikus ennustamisel, kas raskust tõsta või langetada, oli keskmiselt ~75% täpsusega (vt Joonis 21)[52]. Parema tulemuse võinuks anda andmete spetsiifiline töötlemine, kus kasutaja treeningajalugu jagada gruppidesse, kus on tõusud ja langused, kuid autor otsustas seda mitte teha limiteeritud aja ja bakalaureusetöö mahu tõttu.



Joonis 21. Lõpliku logistilise regressioonimudeli ennustus tulemused rinnalt surumise harjutuse kohta

5 Tulemuste analüüs

Bakalaureusetöö probleemi lahendamisel proovis autor kasutada kolme erinevat masinõppe mudelit: rekurrente närvivõrk LSTM, lineaarne regressioon ja logistiline regressioon. Kõige rohkem aega kulus LSTM mudeli koostamisele ja optimeerimisele, kuna see on oma olemuselt kõige keerulisem. Iga mudeliga prooviti treenida mudelit kõigi valitud seitsme harjutuse kohta ja iga harjutuse kohta koostada eraldi mudel. Harjutuse kohta koostatud mudelid olid keskmiselt 7% täpsemad kui üks mudel, mis ennustas kõikide harjutuste kohta korraga.

Kirjeldatud vigade ja proportsionaalsete eksimuste juures tuleb arvesse võtta, et mudeleid on õpetatud tavakasutajate andmete pealt, kus suur osa on algajad ja ei soorita harjutusi korrektselt ning nende treeningkava ei ole optimaalne. Bakalaureusetöös koostatud mudelid näitavad, et selline ennustamine on võimalik ja õppimine inimeste treeningandmete pealt töötab. Hüpoteetiliselt saaks kasutada kõiki mudeleid ka lõpptootes, aga seda esialgu autor teha ei plaani. Mudelite treenimisel oleks vaja suurt andmestikku, kus on personaaltreeneri monitoorimisel sooritatud treeningute andmed. Sellisel juhul oleks võimalik mudelil õppida professionaalsete andmete peal ja ennustused järgiksid personaaltreenerite soovitusi, aga kuna sellisele andmekogule ligipääsu töö koostamise ajal ei olnud, siis jääb see vaid oletuseks.

5.1 LSTM mudeli tulemus

Rekurrentse närvivõrgu LSTM mudeli keskmine viga võrreldes tegelikkusega oli keskmiselt 8%. Kui konverteerida see viga ümber reaalsetesse väärtustesse, võttes aluseks näiteks rinnalt surumise harjutuse, raskuseks 60 kg, korduste arvuks kaheksa ja seeriate arvuks neli, annab see kokku 1920 kg mahtu ühe treeningu kohta rinnalt surumise harjutusele. 8% eksimus tähendab sellisel juhul ~ 2,5 kordusega möödapanekut 32st kordusest, mis tegelikult ei ole üldse halb.

5.2 Lineaarse regressioonimudeli tulemus

Lineaarse regressiooni keskmine viga oli 6% mis eespool kirjeldatud loogika alusel annaks proportsionaalse eksimuse ~2 kordust 32st. Lineaarse regressiooni mudelit on võimalik veel optimeerida ja vähendada eksimuse suurust, näiteks on kindlasti võimalik veel optimeerida sisend tunnuste hulka ja iseloomu. Ajalise puuduse tõttu oli 6% eksimus hetkel parim tulemus mis saavutati.

5.3 Logistilise regressioonimudeli tulemus

Logistilise regressiooni mudel ennustas keskmiselt 75% kordadest õigesti, kas tuleks raskust suurendada või vähendada. Seda tulemust otseselt teiste kasutatud mudelitega võrrelda on keeruline, kuna see sõltuks funktsionaalsusest mis defineerib suuruse kui palju tuleks koormust suurendada või vähendada vastavalt mudeli otsusele. Üks idee, mida autor kavatseb proovida, on kombineerida logistilise regressiooni ennustus lineaarse regressiooni mudeliga. Idee oleks selles, et logistiline mudel annab vastuse kas raskust tuleks tõsta või langetada ja lineaarne mudel ennustaks kui suur oleks muudetav raskus. Ajapuuduse tõttu seda lahendust bakalaureuse töö jooksul proovida ei olnud võimalik.

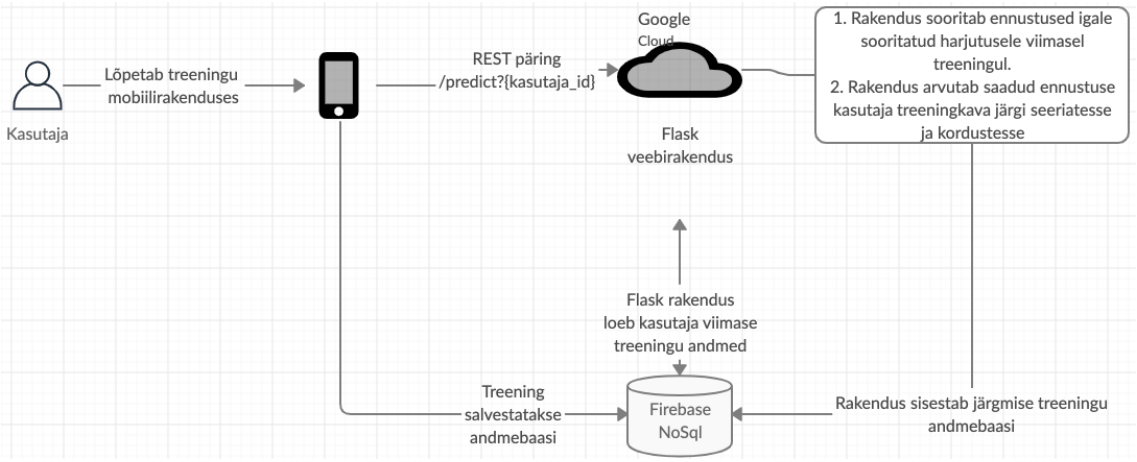
6 Rakenduse arhitektuur ja tehnoloogiad

Autori bakalaureusetöö skoobis oli luua ka taga rakendus, millega kaastudengi, Reio Sedriku, loodud mobiilirakendus saab suhelda kasutades REST veebiteenuse protokolli. Mobiilirakenduses on kasutajal võimalik koostada treeningkava ning seda kasutada treenimisel. Pärast iga treeningu salvestamist rakendusse toimub sooritatud treeningkavala uute raskuste ennustamine. Kuna meil on teada milline on kasutaja kava, kordused ja seeriad, siis saame masinalt ennustatud väljundi mahuna (vt Võrrand 2) transleerida vastava kasutaja kavasse (vt Joonis 22).

Kuna masinõppe mudelid on koostatud Pythoni programmeerimiskeeles, siis otsustas autor kasutada ka taga rakenduse loomisel sama programmeerimiskeele veebiarenduse raamistikke. Taga rakendus on loodud kasutades Pythoni teeki Flask. Flask on lihtne ja õhuke veebiserveri lüüsi liidese (ingl. k *Web Server Gateway Interface/WSGI*) veebirakenduste loomise raamistik. Flask annab võimaluse valida ise tehnoloogiad ja raamistikud millega skaneerida rakendust, st jääb vabadus valida majutuskeskkond, andmebaasi tüüp, kasutajaliidese raamistik jne. Lisaks sellele on Flaski kogukond üpris suur, mis muudab arendamise kiiremaks, kuna probleemide lahendamiseks vastuseid leida on lihtne. [53]

Lõpprakenduse andmebaasiks on Firestore NoSql. Firestore on Google arendatud pilvepõhine andmebaas. NoSql andmebaas on võrreldes relatsiooniliste andmebaasidega skaleeritavam kuna NoSql andmebaasi lugemispäringu hind on tunduvalt madalam kui relatsioonilise andmebaasi lugemispäring. Sarnaste rakenduste puhul ületab lugemise käskude arv kordades teiste operatsioonide arvu [54].

Autori loodud rakendus on majutatud Google Cloud pilveteenuses. Rakendus on pakitud Dockeri konteinerisse ja see omakorda tarnitud Google Cloud majutusteenusesse. Google Cloud võimaldab jooksutada Dockeri konteinereid, mis muudab tarnete, uuenduste tegemise ja versioonihalduse mugavaks. [55]



Joonis 22. Rakenduste vaheline suhtlus

7 Kokkuvõte

Bakalaureusetöö eesmärk oli koostada rakendus mis kasutab masinõppe mudelit ennustamisel kasutajale järgmist jõusaalitreeningut. Bakalaureusetöös valminud osa loodeti kasutada mobiilirakenduses, mis valmib kaastudengi, Reiko Sedriku, bakalaureusetöös. Kahest bakalaureusetööst kokku valmib terviklik rakendus, mida plaanitakse ka turustada ja müüa kui lisaarendused on valmis, mis tööde skoopi ei mahtunud. Rakenduse eesmärk on olla virtuaalne personaaltreener, mis aitab kasutajaid jõusaalitreeningutel.

Kriitilise tähtsusega oli Gymwolf rakenduse haldjatelt saadud andmekogu, mis andis aluse masinõppe mudelite disainimisele ja implementeerimisele. Gymwolf rakendus on jõusaalitreeningute virtuaalne päevik mis on turul olnud juba kümme aastat.

Autor proovis kasutada probleemi lahendamiseks kolme erinevat masinõppe mudelit: rekurrentne närvivõrk LSTM, lineaarne regressioon ja logistiline regressioon. Kahjuks andmekvaliteedi tõttu ei ole võimalik bakalaureusetöös valminud mudeleid kasutada lõpptoote esimeses versioonis, kuna ennustuste täpsused ei ole piisavad, et rakendus täidaks oma ülesannet. Küll aga tõestas bakalaureusetöö jooksul loodud lahendus, et masinõpet on võimalik kasutada kasutajate treeningute ennustamiseks ja masin on võimeline õppima mustreid. Tulevikus kui on kogutud rohkem andmeid, mis on jäädvustatud personaaltreeneri monitoorimisel või pika treeningkogemusega treenijatelt, on võimalik masinõpet kasutada ka valmistootes.

Hetkel alternatiivne idee, mida kasutada toote esimeses produktsiooni versioonis, on implementeerida eelseadistatud funktsionaalsus, mis ei kasuta masinõpet ja põhineb puhtalt teadusuuringutel, kasutaja treeningute modifitseerimiseks.

7.1 Edasised plaanid

Bakalaureusetöös valmis rakenduse minimaalse valmidusega toode, mida turule veel ei ole võimalik tuua. Plaanis on teha mitmeid lisaarendusi mobiilirakenduse poolel ja ka

soovitusmehhanismis, nagu eelnevalt kirjeldatud ehk implementeerida robustne teadusuuringutel põhinev funktsionaalsus mis esialgu ei kasuta masinõpet. Rakenduse esimese produktsiooni versiooni plaanivad autorid valmis saada käesoleva aasta augustikuu lõpuks.

Kaugemale tulevikku mõeldes on rakenduse autoritel plaanis lisada treeningu funktsionaalsusele veel mitmeid osasid, mis eristavad rakendust hetkel turul oelvatest samalaadsetest toodetest. Mõned ideed mida kindlasti proovitakse realiseerida on näiteks toitumisega seotud funktsionaalsus, kus on ka võimalik kasutada masinõpet. Idee on kasutaja jaoks teha võimalikult mugavaks oma toitumise jälgimine. Selle funktsionaalsuse juures saab kasutada masinõppel põhinevat soovitust kas kindla toidukorra kohta või näiteks teha toidutuvastus algoritm, mis põhineb pildiklassifitseerimisel, et anda kasutajale võimalus teha mobiiltelefoniga oma toidust pilt ja saada rakenduselt hinnang, kui palju kaloreid ja erinevaid toitained see toit sisaldab.

Veel üks idee mida kindlasti proovitakse rakendada on treeningutel harjutuse soorituse hindamine. Idee on selles, et kasutajal oleks võimalik näiteks filmida ennast harjutust sooritamas. Seejärel saada rakenduselt hinnang tehnikale ja soovitusi kuidas tehnikat paremaks muuta. Ka selle funktsiooni puhul saaks kasutada masinõpet.

Viimane potentsiaalne tunnus rakenduses, mis autoritel hetkel mõttes on, on luua robotvastaja, millega kasutaja saab suhelda ja küsida nõu treeningute või toitumisega seotud teemadel. Näiteks kui kasutaja on trenni tegemas ja tema kavas on mingi kindel harjutus, aga antud hetkel on see masin hõivatud, siis saab ta pöörduda robotvastaja poole, et saada nõu, mis harjutusega võiks ta kavas oleva harjutuse vahetada.

Kasutatud kirjandus

[1] “Gym anxiety: what is it and how to get over it”, [Võrgumaterjal]. Available: https://www.precor.com/en-us/resources/gym-anxiety-how-get-over-it . Kasutatud [10 mai 2020].

[2] Kavanaugh, Ashley. "The Role of Progressive Overload in Sports Conditioning." *Conditioning Foundamentals. NSCA's Performance Training Journal* 6.1 (2007).

[3] “Progressive Overload”, [Võrgumaterjal]. Available: https://www.bodybuilding.com/content/progressive-overload-the-concept-you-must-know-to-grow.html . [Kasutatud 10 mai 2020].

[4] Joonis 1. https://bodybuilding-wizard.com/bent-over-barbell-row-guide/

[5] Joonis 2. https://standupbestrong.com/fitness/barbell-row/

[6] “Ten Rules of Progressive Overload”, [Võrgumaterjal]. Available: https://bretcontreras.com/progressive-overload/ . [Kasutatud 10 mai 2020].

[7] Bruton, Anne. "Muscle plasticity: response to training and detraining." *Physiotherapy* 88.7 (2002): 398-408.

[8] Peterson, Mark D., et al. "Progression of volume load and muscular adaptation during resistance exercise." *European journal of applied physiology* 111.6 (2011): 1063-1071.

[9] Joonis 3. https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce

[10] “Machine learning”, [Võrgumaterjal]. Available: https://www.technologyreview.com/2018/11/17/103781/what-is-machine-learning-we-drew- you-another-flowchart/ . [Kasutatud 11 mai 2020].

[11] Joonis 4. https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce

[12] “Reccurrent Neural networks”, [Võrgumaterjal]. Available: https://towardsdatascience.com/recurrent-neural-networks-d4642c9bc7ce . [Kasutatud 11 mai 2020].

[13] Joonis 15. http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[14] “LSTM”, [Võrgumaterjal]. Available: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ . [Kasutatud 11 mai 2020].

[15] Joonis 6. http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[16] “Linear Regression In Machine Learning”, [Võrgumaterjal]. Available: https://www.mygreatlearning.com/blog/linear-regression-in-machine-learning/ . [Kasutatud 11 mai 2020].

[17] Joonis 7. https://www.javatpoint.com/linear-regression-vs-logistic-regression-in-machine-learning

[18] Guido, Joseph J., Paul C. Winters, and Adam B. Rains. "Logistic regression basics." *MSc University of Rochester Medical Center, Rochester, NY* (2006).

[19] Joonis 8. https://hackernoon.com/the-ai-hierarchy-of-needs-18f111fcc007

[20] "The Pyramid of Data Needs", [Võrgumaterjal]. Available: https://medium.com/@hugh_data_science/the-pyramid-of-data-needs-and-why-it-matters-for-your-career-b0f695c13f11. [Kasutaud 11 mai 2020].

[21] "AI Training Data issues", [Võrgmaterjal]. Available: https://dataconomy.com/2019/07/why-96-of-enterprises-face-ai-training-data-issues/ . [Kasutatud 11 mai 2020]

[22] "Gymwolf", [Võrgumaterjal]. Available: https://www.gymwolf.com/ . [Kasutatud 11 mai 2020].

[23] Joonis 9. Autori koostatud

[24] "General Data Protection Regulation", [Võrgumaterjal]. Available: https://gdpr-info.eu/ . [Kasutatud 11 mai 2020].

[25] Joonis 10. Autori koostatud

[26] "Amazon web services", [Võrgumaterjal]. Available: https://aws.amazon.com/rds/?hp=tile&so-exp=below . [Kasutatud 11 mai 2020].

[27] "Pandas", [Võrgumaterjal]. Available: https://pandas.pydata.org/ . [Kasutatud 12 mai 2020].

[28] "Matplotlib", [Võrgumaterjal]. Available: https://matplotlib.org/ . [Kasutatud 12 mai 2020].

[29] "Sklearn", [Võrgumaterjal]. Available: https://scikit-learn.org/stable/ . [Kasutatud 12 mai 2020].

[30] "Numpy", [Võrgumaterjal]. Available: https://numpy.org/. [Kasutatud 12 mai 2020].

[31] Joonis 11. Autori koostatud

[32] Joonis 12. Autori koostatud

[33] Joonis 13. Autori koostatud

[34] "MinMaxScalar", [Võrgumaterjal]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html . [Kasutatud 12 mai 2020].

[35] "Tensorflow 2.0", [Võrgumaterjal]. Available: https://www.tensorflow.org/guide/effective_tf2 [Kasutatud 1 märts - 10 mai 2020]

[36] "Modeling Heart Rate and Activity Data for Personalized Fitness Recommendation", [Võrgumaterjal], Available: https://cseweb.ucsd.edu/~jmcauley/pdfs/www19.pdf [Kasutatud 5 märts 2020].

[37] "Andreij Karpathy", [Võrgumaterjal], Available: https://karpathy.github.io/2019/04/25/recipe/ [Kasutatud 3 aprill 2020].

[38] Joonis 14. Autori koostatud

[39] Joonis 15. Autori koostatud

[40] Joonis 16. Autori koostatud

[41] "ADAM", [Võrgumaterjal], Available:https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Adam [Kasutatud 11 mai 2020].

[42] "ADAM", [Võrgumaterjal], Available: https://arxiv.org/abs/1412.6980 [Kasutatud 12 mai 2020].

[43] "Loss Functions", [Võrgumaterjal], Available: https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e9d4b0 . [Kasutatud 12 mai 2020]

[44] Joonis 17. Autori koostatud

[45] Joonis 18. Autori koostatud

[46] "SHAP", [Võrgumaterjal], Available: https://shap.readthedocs.io/en/latest/ [Kasutatud 12 mai 2020].

[47] "Tsfresh", [Võrgumaterjal], Available: https://tsfresh.readthedocs.io/en/latest/ [Kasutatud 12 mai 2020].

[48] Joonis 19. Autori koostatud

[49] "Fourier coefficient" , [Võrgumaterjal], Available: https://tsfresh.readthedocs.io/en/latest/api/tsfresh.feature_extraction.html#tsfresh.feature_extraction.feature_calculators.fft_coefficient . [Kasutatud 12 mai 2020]

[50] "Wavelet coefficient", [Võrgumaterjal], Available: https://tsfresh.readthedocs.io/en/latest/api/tsfresh.feature_extraction.html#tsfresh.feature_extraction.feature_calculators.cwt_coefficients . [Kasutatud 12 mai 2020].

[51] Joonis 20. Autori koostatud

[52] Joonis 21. Autori koostatud

[53] "Flask", [Võrgumaterjal], Available: https://flask.palletsprojects.com/en/1.1.x/ .[Kasutatud 12 mai 2020].

[54] "Firebase", [Võrgmaterjal], Available: https://firebase.google.com/ . [Kasutatud 12 mai 2020].

[55] "Google Cloud", [Võrgumaterjal], Available: https://cloud.google.com/ [Kasutatud 12 mai 2020].