TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Tõnis Piip
156215IAPB

KÜBERRÜNNAKUTE TUVASTAMINE LIHTSAMATE KLASSIFIKAATORITE ABIL

Bakalaureusetöö

Juhendaja: PhD Sven Nõmm
Kaasjuhendaja: PhD Hayretdin Bahsi

Tallinn 2019

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: Tõnis Piip

20.05.2019

Annotatsioon

Käesoleva töö eesmärgiks on ühes kindlas süsteemis oleva elektrisüsteemi vastu tehtavate küberrünnakute tuvastamine primitiivsemate klassifikaatorite abil ning seejärel välja õpetada võimalikult täpne klassifikaator, mis võiks tulla kas vähemate muutujate arvuga või täpsem kui Hinki, Beaveri ja teiste töö tulemusena sama klassifikaator.

Töö käigus analüüsitakse andmehulkade muutujate Fisheri väärtusi (*score*) ning nende reastamisel võetakse klassifikaatorite treenimiseks N parimat muutujat. Töös kasutatakse viit erinevat klassifikaatori mudelit - k-lähimat naabrit, tugivektormasinaid, otsustuspuud, logistilist regressiooni ning lineaarse diskriminandi analüüsi. Tulemusi kontrollitakse ristvalideerimise abil ning lõpuks võrreldakse saadud tulemusi Hinki ja teiste tulemustega.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 28 leheküljel, 6 peatükki, 8 joonist, 9 tabelit.

Abstract

Detecting Cyberattacks Using Simple Classifiers

The main goal of this thesis is detecting cyberattacks against a power system within the system itself in a very specific network using primitive classifiers. The author also aims to train either a more accurate classifier or a classifier of the same accuracy but using less features than the resulting same or similar classifier model in Hink and others' work. The classifiers used in this thesis include support vector machine, k-nearest neighbours, decision tree, logistic regression and linear discrimant analysis.

Two-, three- and multiclass datasets were created by Uttam Adhikari, Shengyi Pan and Tommy Morris in collaboration with Raymond Borges and Justin Beaver, that include measurements from four Intelligent Electronic Devices (IEDs), log data and classifications. The main features used in this thesis, measured by the IEDs, are ranked using their Fisher's scores. Different classifiers are trained using the best N features to see, which model is best suited with the given data. Later, feature selection is optimized with different methods, such as intersecting best features within one class system.

The thesis resulted in a support vector machine classifier that correctly detects attack, natural and null events on average about 72% of the time within the three class dataset. The author managed to achieve one of the two desired outcomes, having developed a support vector machine classifier with the same accuracy of 72% but using considerably less features than the one in Hink and others' work. The results are as good or better because of feature selection optimization, which is practically never done in cyber security.

The thesis is in Estonian and contains 28 pages of text, 6 chapters, 8 figures, 9 tables.

Lühendite ja mõistete sõnastik

ICS	*Industrial Control System*
IES	Intelligentne Elektrooniline Seade
KNN	*K-Nearest Neighbour*
Kt	Kordustäpsus
S	Saagis
SCADA	*Supervisory Control and Data Acquisition*
SVM	*Support Vector Machine*

Sisukord

Jooniste loetelu

Tabelite loetelu

1 Sissejuhatus

Tööstus- või mõne muu sektori digitaliseerumisega kaasneb alati küberrünnakute oht ja seda eelkõige juhtivate või kontrollivate masinate suhtes. Tööstuskontrolli- või -juhtimissüsteem (ingl. k *Industrial Control System* ehk ICS) on kollektiivne termin, mida kasutatakse erinevate juhtimissüsteemide ning nendega seotud instrumentide kirjeldamiseks. Need hõlmavad seadmeid, süsteeme, võrke ja juhtnööre, mida kasutatakse tööstusprotsesside juhtimiseks või kontrollimiseks. ICS-id ehitatakse ülesannete tõhusaks haldamiseks ning nende toimimine on erinev ja sõltub tööstusharust. Tänapäeval on ICS-id kasutusel pea igas tööstussektoris ja kriitilises infrastruktuuris, nagu näiteks tootmis-, transpordi, energia- ja veepuhastustööstuses. ICS-e on erinevaid tüüpe, kõige levinuimad on näiteks järelvalvekontrolli ja andmete kogumise (ingl. k *Supervisory Control and Data Acquisition*) ehk SCADA süsteemid ning jaotatud juhtimissüsteemid (ingl. k *Distributed Control Systems*) ehk DCS. Kohalikke operatsioone kontrollivad tavaliselt niinimetatud välisseadmed, mis saavad järelvalvekäske kaugjaamadest [1].

1.1 Juhtimissüsteemide küberünnakute tõus

Kaspersky 2018. aastal läbi viidud uuring leidis, et 51% vastanutest ei olnud kogenud ühtegi kontrollsüsteemiga seotud juhtumit. Kuigi juhtumite kogemiste arv oli natuke langenud, leidis Kaspersky, et võrreldes 2017. aastaga organisatsioonides tööstuskontrollsüsteemiga seotud küberrünnakute tõenäosus tõusis. Kui eelnevalt oli sellise rünnaku ohvriks langemise väga või üsna tõenäolisuse protsent ülemaailmselt 74, siis 2018. aastal oli see arv 77, samas tõusis väga tõenäolisuse protsent 7 võrra [2].

IBM Security 2015. aastal läbi viidud uuringus avastati ajavahemikus 01.01.2013

kuni 20.08.2015 IBM Managed Security Services'i jälgitavate süsteemide vastu tehtavate rünnakute arvu tõus. Kui 2013 aastal oli rünnakute arv veidi üle 600, siis kahel järgneval aastal, 2014 ja 2015 oli see arv juba ligikaudu 1300 [3]. Samas aruandes on ka mainitud Delli 2015. aasta aruannet, kus on märgitud SCADA vastu toimunud rünnakute tõusu. 2012. ja 2014. aasta vahel suurenes rünnakute arv SCADA süsteemide vastu 636%, olles 2012 aastal 91 676 ja jõudes 2014. aastal 675 186 piirini. Viimasest arvust toimusid 202 322 rünnakut Soomes, 69 656 Suurbritannias ja 51 258 Ameerika Ühendriikides [4].

IBM Security avaldatud artiklis kirjeldab artikli autor, et vastavalt IBM Managed Security Services andmetele tõusis 2016. aastal tööstuskontrollisüsteemide vastu tehtavate rünnakute arv võrreldes 2015. aasta andmetega 110%. Täpsemalt oli tõus seotud toore jõuga sisse murdmiste osas, mille abil on ründavatel osapooltel võimalik süsteemi seestpoolt jälgida või isegi kontrollida [5].

1.2 Lõputöö eesmärk

Eelnevalt mainitud aruannete tulemuste põhjal on näha, et küberrünnakute arv juhtimissüsteemide vastu on viimaste aastatega peamiselt ainult tõusnud, mille tõttu on vaja juhtimissüsteemidele head küberkaitset. Selline kaitse võib blokeerida autoriseerimata isikutel ja masinatel süsteemi sisenemise, kuid võib ka juba süsteemi sisse saanud isikuid või masinaid takistada.

Antud lõputöö uurib ühes kindlas võrgus oleva elektrisüsteemi vastu tehtavaid rünnakuid ning kas ja kuidas saaks neid antud mõõtetulemuste põhjal võimalikult edukalt klassifikaatorite abil tuvastada. Rünnakute tuvastamine toimub süsteemisiseselt, saades rünnakust aru kui süsteemis toimuvad ebaloomulikud muutused. Klassifikaatorite sisendmuutujate valimeid ka optimiseeritakse, mida küberturbes praktiliselt kunagi ei tehta. Muutujate valimi optimiseerimisel võib klassifikaatori täpsus suureneda, kuid võib ka juhtuda, et klassifikaatori täpsus tuleb samane, mis optimiseerimata muutujate hulgaga. Lõpetuseks võrreldakse antud töös saadud tulemusi eelnevalt samade andmetega tehtud Hinki ja teiste töö tulemustega. Vaadatakse ka seda, kas autor suutis välja töödelda vähemate

muutujatega, kuid sama täpse või lihtsalt täpsema klassifikaatori, kui seda tegid Hink ja teised [6].

2 Andmed

Käesoleva töö jaoks saadud andmed pärinevad Tommy Morrise, Uttam Adhikari, Shengyi Pani, Raymond Borgesi ja Justin Beaveri koostöös loodud kolmest elektrisüsteemi andmekogumikust [7]. Andmed on jagatud kolme grupi vahel vastavalt klasside arvule - kaheklassiline, kolmeklassiline ja mitmeklassiline andmekogumik. Igas andmekogumikus on kokku 15 erinevat andmefaili (data1.csv kuni data15.csv), milles igas ühes on umbes 5000 rida katseandmeid.

Kogu töö käigus valminud kood on kirjutatud Pythonis, kasutades erinevaid lisapakette, nagu näiteks Scikit-learn, NumPy ning Matplotlib.

2.1 Töövoog

Töö alustuseks tutvus autor andmetega, millega kõik edasine töö toimus. See oli vajalik, et aru saada, milliste muutujatega võiks või peaks edaspidine töö toimuma. Järgnevalt oli vaja andmed Pythonisse sisse lugeda ning iga faili iga muutuja Fisheri väärtus arvutada. Pärast muutujate Fisheri väärtuse leidmist oli vaja hakata nende samade väärtuste põhjal parimaid muutujaid valima. Valitud muutujate arv valiti esialgu võimalikult väike. Iga uue treenimisega valiti ühe võrra suurem muutujate arv.

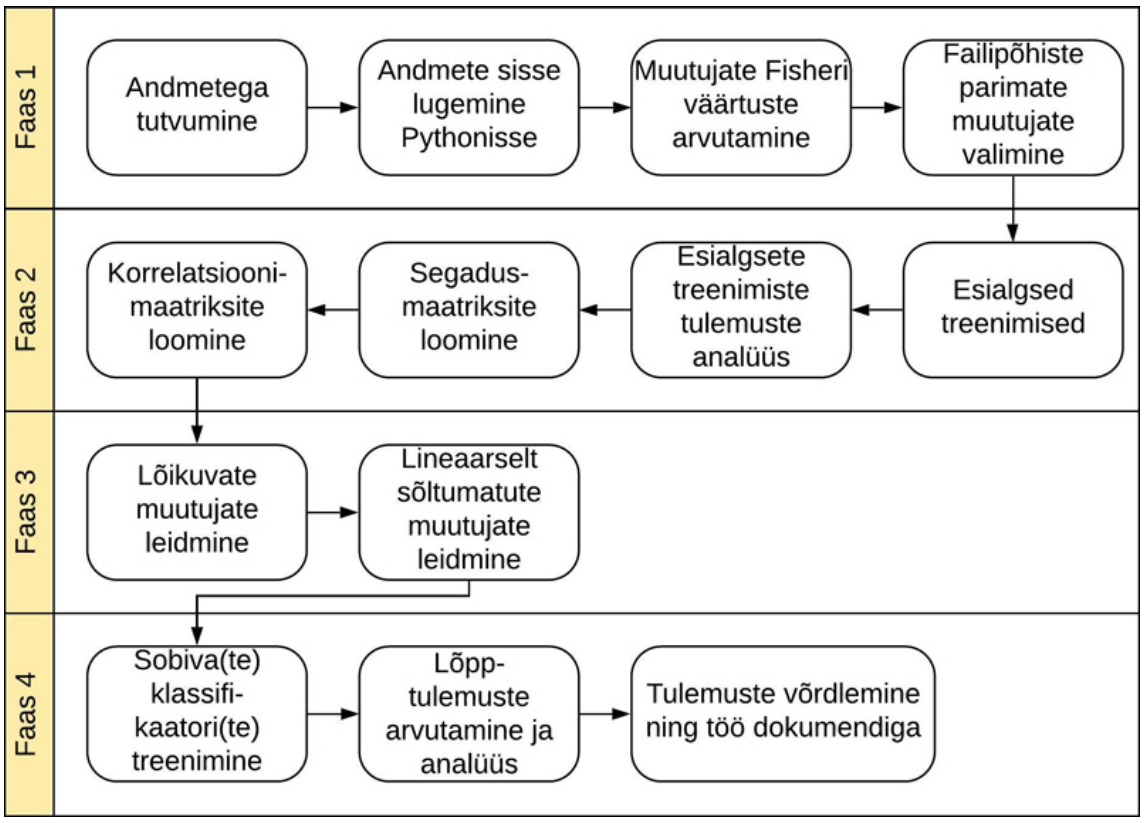
Esmased treenimised toimusid Fisheri väärtuste põhjal kolmeklassiliste ning mitmeklassiliste andmetega, jättes esialgu kaheklassilised andmed kõrvale. Seejärel, pärast esimesi klassifikaatorite treeninguid, analüüsiti saadud tulemusi, milles kajastus vastavate klassisüsteemide andmetega sobivaim klassifikaatori mudel. Esmaste treenimistulemuste analüüsi järel loodi valitud klassifikaatorite mudelite abil nii segadus- kui ka korrelatsioonimaatriks.

Pärast maatriksite loomist ja analüüsimist otsiti igas klassisüsteemis failide vahel lõikuvad muutujad parimate failipõhiste muutujate seast, Fisheri väärtuste põhjal. Lisaks jäeti ühest muutujate valimist välja lineaarselt sõltuvad muutujad, mis leitigi korrelatsioonimaatriksi abil.

Eelviimaste treenimiste jaoks võeti kõige sobivama klassifikaatori mudeli sisenditeks eelnevalt mainitud muutujate valimid, kaasa arvatud parimad failipõhiste muutujate valim, erinevate muutujate arvuga. Nende käigus leiti minimaalseim muutujate hulk, mis tagaks võrreldava tööga võimalikult sarnase klassifikaatori täpsuse.

Lõpptulemuste arvutamiseks kasutati leitud minimaalset muutujate arvu ning sobivaimat klassifikaatori mudelit iga klassisüsteemi andmetega. Saadud tulemusi analüüsiti ning võrreldi Hinki ja teiste töö tulemustega [6]. Lõpetuseks toimus edasine töö dokumendi loomisega

Kogu andmetega seotud töövoog on kokkuvõtva graafina kujutatud Joonisel 1.



Joonis 1: Töövoo graaf

2.2 Andmete klassifikatsioon

Kaheklassilistes andmetes on nime kohaselt ainult kaks võimalikku klassi ehk juhtumit, tavaline juhtum või rünnak (ingl. k. *Natural* ja *Attack*). Kolmeklassilistes andmetes on lisaks eelnevalt mainitud juhtumitele lisandunud veel null-sündmus (ingl. k. *NoEvents*), mille puhul lihtsalt normaalne töökoormus muutub. Mitmeklassilistes süsteemis on sõnalised klassid asendunud täisarvudega 1-30 ning 35-41, mida on võimalik ka esitada nii kaheklassilises kui ka kolmeklassilises süsteemis. Tabelis 1 on näidatud mitmeklassiliste sündmuste jagamine kaheklassilises süsteemis. Mitmeklassilise süsteemi sündmuste jaotus kolmeklassilise süsteemi klasside vahel on näidatud tabelis 2 [8].

Tabel 1: Mitmeklassilised juhtumid kaheklassilises süsteemis.

	Rünnak	Tavaline
Mitmeklassi juhtum/klass	7, 8, 9, 10, 11, 12, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 35, 36, 37, 38, 39, 40	1, 2, 3, 4, 5, 6, 13, 14, 41

Tabel 2: Mitmeklassilised juhtumid kolmeklassilises süsteemis.

	Rünnak	Tavaline	Null-sündmus
Mitmeklassi juhtum/klass	7, 8, 9, 10, 11, 12, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 35, 36, 37, 38, 39, 40	1, 2, 3, 4, 5, 6, 13, 14	41

Siinkohal peab ka ära märkima, et antud andmed on suuresti tasakaalustamata. Igas failis on rünnakuid palju rohkem kui tavalisi juhtumeid, mille tõttu võib klassifikaatorite tulemustes esineda iseärasusi.

2.3 Muutujate olemus ja nimetused

Töös kasutatud tunnused või muutujad tulevad neljast erinevast Intelligentsest Elektroonilisest Seadmest (ingl. k. *Intelligent Electronic Device*) ehk IESist ning on erinevates suurusjärkudes arvuliste väärtustega. Iga IES mõõdab 29 erinevat muutujat, kokku mõõdetakse 116 muutujat. Igas failis on lisaks veel ka Snorti, simuleeritud kontrollpaneeli ning releede logide muutujad, mis lisab juurde 12 muutujat. Snort on võrgu sissetungi avastamise ja sissetungi peatamise tarkvara [9]. Viimases tulbas on määratud mõõte tulemuse klassifikatsioon. Kokkuvõtvalt on igas failis ilma klassifikatsioonita 128 muutujat.

Iga IESi muutuja on nimelisel kujul "R#-Signaali Viide", mis näitab "R#"-ga määratud Intelligentse Elektroonilise Seadme mõõte tüüpi, kus "#" on vastava IESi number. Signaali viited ja neile vastavad kirjeldused on toodud tabelis 3 [8]. Näiteks R2-PA2:VH on Intelligentse Elektroonilise Seadme R2 mõõdetud faasi B pinge faasi nurk.

Tabel 3: Muutujate nimetused ja kirjeldused

Muutuja	Kirjeldus
PA1:VH - PA3:VH	Faasi A-C pinge faasi nurk
PM1:V - PM3:V	Faasi A-C pinge suurusjärk
PA4:IH - PA6:IH	Faasi A-C voolu faasi nurk
PM4:I - PM6:I	Faasi A-C voolu suurusjärk
PA7:VH - PA9:VH	Pos.-Neg.-Null pinge faasi nurk
PM7:V - PM9:V	Pos.-Neg.-Null pinge suurusjärk
PA10:VH - PA12:VH	Pos.-Neg.-Null voolu faasi nurk
PM10:V - PM12:V	Pos.-Neg.-Null voolu suurusjärk
F	Releede sagedus
DF	Releede sageduste delta (dF/dt)
PA:Z	Reelede nähtav ilmne takistus
PA:ZH	Releede nähtava ilmse takistuse nurk
S	Releede oleku lipp

2.4 Fisheri väärtus (*Fisher's score*)

Fisheri väärtus on loomu poolest mõeldud arvuliste muutujate jaoks. Selle abil on võimalik mõõta keskmise klassidevahelise eraldamise suhet klassisisese keskmise eraldamisega. Mida kõrgem Fisheri väärtus mingil muutujal on, seda suurem on selle muutuja diskrimineeriv võime ja seda paremini sobib see muutuja klassifikaatori sisendiks. Fisheri väärtuse valem on vastavalt

$$F = \frac{\sum_{j=1}^{k} p_j(\mu_j - \mu)^2}{\sum_{j=1}^{k} p_j(\sigma_j)^2} \tag{1}$$

kus k on erinevate klasside arv, μ_j and σ_j on ühe muutuja klassi j kuuluvate andmete aritmeetiline keskmine ja standardhälve, p_j on klassi j klassi kuuluvate andmete murdosa kõigist selle muutuja andmetest ja μ on antud muutuja andmete globaalne keskmine [10, p.290].

2.5 Muutujate Fisheri väärtused

Eelnevalt mainitud valemit kasutati nii kaheklassiliste, kolmeklassiliste kui ka mitmeklassiliste andmefailide muutujate Fisheri väärtuste arvutamiseks. Antud töös ei tulnud Fisheri väärtused väga suured, kuid sellegi poolest sai muutujaid Fisheri väärtuste järgi korralikult järjestada ja seejärel neist valimeid koostada.

Kaheklassilise süsteemi failide muutujate Fisheri väärtused olid oodatult kõige madalamad. Data1 parimad kaks muutujad omasid Fisheri väärtusi vastavalt 0.0715 ning 0.0668, kuid selles failis toimus pärast teist parimat muutujat suur langus Fisheri väärtuses. Juba kolmas parim Fisheri väärtusega muutuja omas vastavat väärtust ümardatult 0.0113, mis on ligikaudu viiekordne langus Fisheri väärtuses. Teiste failide puhul sama järskusi langusi ei toimunud. Teiste failide puhul jäävad parimad Fisheri väärtused 0.0336 ja 0.0059 vahele. Kaheklassiliste andmefailide parima seitse muutuja Fisheri väärtused on toodud tabelis 4.

Tabel 4: Kaheklassiliste andmete 7 parimat Fisheri väärtust.

Fail	I	II	III	IV	V	VI	VII
Data1	0.0715	0.0668	0.0113	0.0110	0.0110	0.0109	0.0087
Data2	0.0127	0.0117	0.0107	0.0107	0.0101	0.0099	0.0098
Data3	0.0070	0.0067	0.0065	0.0064	0.0060	0.0059	0.0059
Data4	0.0309	0.0304	0.0276	0.0276	0.0118	0.0118	0.0114
Data5	0.0212	0.0210	0.0208	0.0203	0.0194	0.0172	0.0166
Data6	0.0081	0.0081	0.0081	0.0080	0.0038	0.0029	0.0028
Data7	0.0316	0.0314	0.0280	0.0274	0.0272	0.0262	0.0104
Data8	0.0336	0.0321	0.0312	0.0310	0.0303	0.0300	0.0296
Data9	0.0117	0.0115	0.0111	0.0102	0.0099	0.0084	0.0061
Data10	0.0080	0.0080	0.0076	0.0066	0.0066	0.0063	0.0054
Data11	0.0146	0.0145	0.0144	0.0143	0.0101	0.0058	0.0031
Data12	0.0089	0.0083	0.0078	0.0072	0.0067	0.0067	0.0065
Data13	0.0223	0.0222	0.0204	0.0203	0.0191	0.0190	0.0143
Data14	0.0059	0.0055	0.0054	0.0049	0.0049	0.0046	0.0044
Data15	0.0186	0.0186	0.0179	0.0179	0.0177	0.0173	0.0173

Kolmeklassilises süsteemis on näha kaheklassiliste failide muutujatega võrreldes paremaid Fisheri väärtusi. Kahe faili puhul, Data4 ning Data7, on Fisheri väärtus esimese viie muutujaga üle 0.1, esikohad vastavalt 0.105 ja 0.1085. Erinevalt kaheklassilisest, toimub kolmeklassilises süsteemis väärtuse langustes hüppeid tihedamalt. Ülejäänud failide muutujate esikohalised Fisheri väärtused jäävad vahemikku 0.0114 ning 0.0854. Kolmeklassiliste failide seitse parimat muutujate Fisheri väärtust on välja toodud Tabelis 5.

Tabel 5: Kolmeklassiliste andmete 7 parimat Fisheri väärtust

Fail	I	II	III	IV	V	VI	VII
Data1	0.0854	0.0791	0.0558	0.0516	0.0294	0.0278	0.0269
Data2	0.0332	0.0311	0.0190	0.0188	0.0182	0.0181	0.0167
Data3	0.0120	0.0101	0.0100	0.0098	0.0098	0.0097	0.0082
Data4	0.1050	0.1048	0.1043	0.1041	0.0375	0.0374	0.0368
Data5	0.0574	0.0570	0.0565	0.0565	0.0550	0.0550	0.0506
Data6	0.0226	0.0225	0.0220	0.0216	0.0208	0.0208	0.0191
Data7	0.1085	0.1084	0.1081	0.1062	0.1044	0.1031	0.1028
Data8	0.0610	0.0609	0.0606	0.0605	0.0601	0.0591	0.0548
Data9	0.0298	0.0291	0.0252	0.0250	0.0215	0.0215	0.0194
Data10	0.0199	0.0196	0.0193	0.0193	0.0174	0.0174	0.0161
Data11	0.0330	0.0326	0.0325	0.0320	0.0314	0.0312	0.0312
Data12	0.0114	0.0111	0.0110	0.0106	0.0106	0.0097	0.0096
Data13	0.0424	0.0424	0.0408	0.0407	0.0388	0.0388	0.0353
Data14	0.0265	0.0260	0.0258	0.0258	0.0258	0.0256	0.0220
Data15	0.0415	0.0415	0.0413	0.0413	0.0408	0.0407	0.0405

Tabelis 6 on kirjeldatud mitmeklassiliste andmefailide seitse parimat muutujat Fisheri väärtuste põhjal. Antud väärtuste puhul on näha nende äkilist tõusu võrreldes eelnevate klassisüsteemide muutujate Fisheri väärtustega. Kui eelmistes süsteemides olid absoluutselt parimad Fisheri väärtused 0.1 läheduses, siis mitmeklassiliste andmete muutujate parimad Fisheri väärtused on keskmiselt 0.32 läheduses. See tuleneb sellest, et andmed on normaliseerimata. Esikohalised Fisheri väärtuses jäävad vahemikku 0.2290 ning 0.6053. Failis Data10 on näha neljanda ja viienda muutuja Fisheri väärtustes suuremat langust, kuid ülejäänud failides sellist olukorda ei esine.

Tabel 6: Mitmeklassiliste andmete 7 parimat Fisheri väärtust

Fail	I	II	III	IV	V	VI	VII
Data1	0.3071	0.3054	0.3053	0.3047	0.2858	0.2839	0.2482
Data2	0.2290	0.2234	0.2232	0.2217	0.2198	0.2155	0.2127
Data3	0.3179	0.3177	0.3159	0.3158	0.3043	0.2938	0.2919
Data4	0.2461	0.2437	0.2435	0.2415	0.2179	0.2028	0.1991
Data5	0.2677	0.2677	0.2621	0.2519	0.2460	0.2376	0.2327
Data6	0.3981	0.3960	0.3659	0.3644	0.3594	0.3528	0.3152
Data7	0.3277	0.3189	0.3048	0.3044	0.2964	0.2963	0.2916
Data8	0.3217	0.3198	0.3190	0.3169	0.2847	0.2709	0.2585
Data9	0.4471	0.2702	0.2673	0.1960	0.1959	0.1905	0.1884
Data10	0.6053	0.6031	0.6030	0.6024	0.2695	0.2560	0.2329
Data11	0.2727	0.2564	0.2557	0.2411	0.2378	0.2326	0.2324
Data12	0.2685	0.2675	0.2537	0.2168	0.2168	0.2157	0.2157
Data13	0.2739	0.2649	0.2642	0.2635	0.2634	0.2633	0.2618
Data14	0.2941	0.2910	0.2857	0.2857	0.2827	0.2771	0.2769
Data15	0.2301	0.2294	0.2164	0.2099	0.2077	0.2053	0.2015

Fisheri väärtuste analüüsi käigus selgus, et Snorti, simuleeritud juhtpaneeli ning releede logide muutujate kohad väärtuste järjestuses olid enamasti viimaste hulgas. Selle põhjuseks saab lugeda seda, et antud muutujad hoiustavad logide arve, mitte detaile või statistikat. Erandiks on releede logide muutujad, mis mõnede andmefailide puhul olid lausa esikümne seas. Tõenäoliselt selle pärast, et juhtpaneeli ning Snorti logide muutujate summa on mitme tuhande andmerea kohta väga väike, samas kui releede logide muutujate summad on kahesaja ulatuses. Kolmeklassiliste andmete puhul oli releede logid mõnedes failides esikümne seas, ühes failis isegi kõige parem. Mitmeklassiliste andmete puhul on märgatav releede logide koha numbrite langus, olles maksimaalselt 29. Selle tõttu, et logide muutujad ei omanud mingit suuremat tähtsust klassifitseerimisel ning nende kohad Fisheri väärtuste järjestustes ei olnud väga head, jäeti need lõplikust muutujate valikust välja.

3 Masinõppe metoodika

Antud töös võeti kasutusele viis erinevat klassifkaatori mudelit - k-lähima naabri, tugivektormasina, otsustuspuu, logistilise regressiooni ja lineaarse diskriminandi analüüsi mudelid. Valik kujunes klassikaliste ja lihtsamate klassifikaatorite valimisest. Klassifikaatorite valimisse kaasati erinevat tüüpi mudeleid, et testimiste ja treenimiste käigus selguks, milline klassifikaatori mudel sobib antud töös kasutatud andmete jaoks kõige paremini.

3.1 K-Lähima naabri

K-lähima naabri meetod ehk KNN teeb ennustusi otseselt treeningandmeid kasutades. Ennustusi tehakse uuele juhtumile, vaadates läbi kogu treeningandmete hulk, et sealt leida K kõige sarnasemat näidet ehk K lähimat naabrit. Nende K naabri abil tehakse kokkuvõte vastava juhtumi naabrite väljundmuutuja kohta. Klassifitseerimisel on väljundmuutujaks antud töös naabrite seas levinuim klassi väärtus. Naabrite leidmisel kasutatakse enamasti eukledilist kaugust, kuid levinud on ka Hammingi, Manhattani ja Minkowski kaugused [11]. Neist viimane on antud töö KNN mudelis kasutusel vastavalt klassifikaatorimudeli vaikesätetele.

3.2 Tugivektormasin

Tugivektormasinad (ingl. k. *Support Vector Machines* ehk SVM-id) on kontrollitud õppemeetodide kogum, mida kasutatakse klassifitseerimisel, regressioonil ning ka näiteks kõrvalekallete avastamisel. SVM konstrueerib hüpertasandi või hüpertasandite hulga lõpmatu või mitmedimensioonilises ruumis. Hea eraldus on saavutatud siis, kui hüpertasandi kaugus igasse klassi kuuluvast hüpertasandile

lähimast punktist on suurim. Uue juhtumi klassifitseerimiseks vaatab SVM kummale poole hüpertasandit vastav punkt jääb [12].

3.3 Otsustuspuu

Otsustuspuud ehk *Decision Trees* on mitteparameetrilised kontrollitud õppemeetodid, mida kasutatakse klassifitseerimiseks ja regressiooniks. Otsustuspuu eesmärgiks on luua mudel, tavaliselt binaarne puu, mis suudab ennustada sihtmuutuja väärtuse, õppides selleks lihtsamaid otsustusreegleid üldandmete põhjal [13].

3.4 Logistiline regressioon

Logistiline regressioon, erinevalt tavalisest regressioonist ei ürita ennustada mingi muutuja väärtust sisendite hulga abil, vaid ennustuse väljundiks on tõenäosus, et sisendpunkt kuulub mingisse klassi. Logistiline regressioon eeldab, et antud sisendid on võimalik lineaarse piiriga jagada iga klassi jaoks erinevaks regiooniks. Antud piir on kahemõõtmelises ruumi sirge ja kolmemõõtmelises ruumis tasand, vastavalt kahe ja kolme klassiga süsteemis [14].

3.5 Lineaarse diskriminandi analüüs

Logistilisest regressioonist edasi aretatud lineaarse diskriminandi analüüs on Bayesi klassifikaator, mis üritab minimeerida valesti klassifitseerimise tõenäosust. Lineaarne diskriminant on logistilises regressioonis nimetatud piir, mis on lineaarne oma funktsiooni poolest ning see aitab eristada (ingl. k *discriminate*) erinevatesse klassidesse kuuluvaid punkte [15].

3.6 Kasutatud erinevad treenimise viisid

Klassifikaatorite treenimine on tugevalt seotud treenimiseks kasutatavate andmete kogusest, mille tõttu kasutati selles töös paari erinevat treenimise meetodit. Kõige esimesena kasutati vastava faili n parimat Fisheri väärtusega muutujat - edaspidi failipõhised muutujad. Teise meetodina analüüsiti iga andmefaili suhtes eraldi neis olevate muutujate lineaarset sõltumatust ning valiti ainult need muutujad, mis ei olnud ühegi teise muutujaga lineaarselt täiesti sõltuv. Neist muutujatest valiti omakorda veel ainult n parimat muutujat. Viimase meetodina uuriti igas klassisüsteemis olevate andmefailide lõikuvaid muutujaid ning valiti n kokkulangevat muutujat. Iga meetodi puhul läbisid andmed erinevate muutujate arvuga treenimised, kasutades kõiki eelnevalt nimetatud klassifikaatorite mudeleid.

3.7 Andmete jagamine ja tulemuste valideerimine

Klassifikaatorite treenimiseks pidi kõigepealt andmeid jagama Scikit-learn *Model selection* paketi *train_test_split* meetodi abil, mis jagab andmed juhuslikeks treening- ning testalamhulkadeks [16].

Klassifikaatorite täpsuste tulemusi valideeriti k-korda ristvalideerimisega (ingl. k. *k-fold cross-validation*), valides k=10, samaselt võrreldava tööga. K-korda ristvalideerimise puhul jaotatakse tulemused suvaliselt k erinevasse gruppi ehk osavalimiks (ingl. k. *fold*), mis on enam-vähem ühesuurused. Üks moodustatud gruppidest valitakse valideerimiskogumiks ja ülejäänud k-1 kogumi peal treenitakse mudelit. Järgnevalt arvutatakse valideerimiskogumi andmetega keskmine ruutviga (ingl. k. *mean squared error*) MSE_1. Protseduuri korratakse k korda, valides igal korral uue valideerimiskogumi, mida ei ole eelnevalt juba kasutatud. Protsessi tulemusena saadakse k testivea hinnangut, $\mathrm{MSE}_1 ... \mathrm{MSE}_k$. Nende hinnangute keskmine väärtus on k-korda ristvalideerimise hinnang ja on antud töös täpsuse lõplik määraja [17, p.181].

3.8 Kordustäpsus, saagis ja f-mõõt

Lisaks üldisele klassifikaatori täpsusele arvutati lõpus välja antud klassifikaatori kordustäpsus (ingl. k. *precision*), saagis (ingl. k. *recall*) ning f-mõõt (ingl. k. *f-measure*). Kordustäpsus näitab klassifikaatori kohta seda, milline osa positiivsetest tuvastustest on õige ehk kui palju kõigist ühte klassi määratud elementidest olid päriselt ka sellest klassist. Kõrge kordustäpsus näitab, et klassifikaatori valepositiivsete määr on väike. Saagis näitab, milline osa tegelikult positiivsetest elementidest tuvastati õigesti ehk kui palju tegelikult ühte klassi kuuluvatest elementidest klassifitseeriti vastavasse klassi. Valemite kujul arvutatakse kordustäpsust (Kt) vastavalt

$$Kt = \frac{TP}{TP + VP} \tag{2}$$

ning saagist (S) vastavalt

$$S = \frac{TP}{TP + VN} \tag{3}$$

kus TP on tõsipositiivsete, VP on valepositiivsete ning VN on valenegatiivsete tuvastuste arv.

F-mõõt tuleb otseselt kordustäpsuse ning saagise kaalutud keskmisest ehk siis näiteks kordustäpsusega 1.0 ja saagisega 0.5 on f-mõõt 0.0, samas kui tavaline keskmine on 0.5. Madal f-mõõt näitab seda, et tõsipositiivsete, valepositiivsete või mõlemate tuvastuste arvud on madalad. F-mõõtu arvutatakse valemi 4 järgi.

$$F\text{-}m\tilde{o}\tilde{o}t = 2 \cdot \frac{Kt \cdot S}{Kt + S} \tag{4}$$

4 Esimesed treenimised

Klassifikaatorite treenimistega alustati esialgu nii kolmeklassiliste kui ka mitmeklassiliste andmete peal. Esimese asjana uuriti erinevate klassifikaatorite mudelite sobivust antud andmetega. Seejärel prooviti erinevaid muutujate valiku konfiguratsioone - vastava faili põhised, süsteemisiseselt lõikuvad ning lineaarselt mittesõltumatud parimad N muutujat. Muutujate valik sõltus lisaks teistest valikumeetodidest alati ka vastavate muutujate Fisheri väärtuste järjestusest.

4.1 Sobiva mudeli leidmine

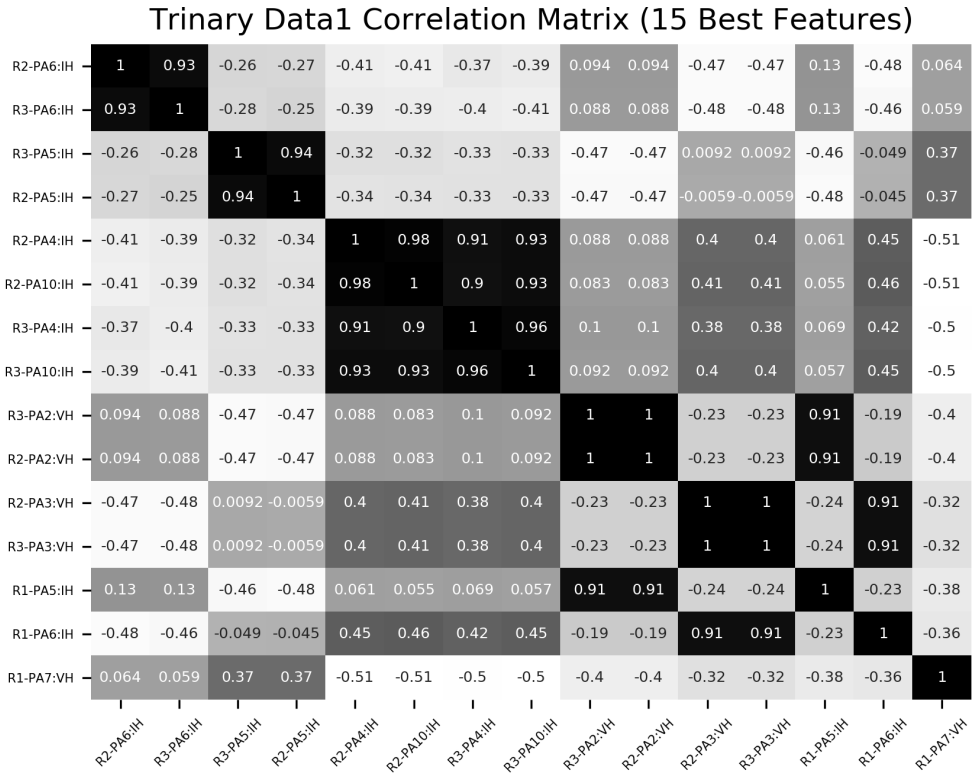
Mudeli valiku jaoks treeniti kõiki viit klassifikaatori mudelit, andes neile iga faili puhul selle faili N parimat muutujat Fisheri väärtuste põhjal, kus N=2...15. Kokku treeniti mudeli valimise jaoks 2100 klassifikaatorit - 2 klassisüsteemi, 15 faili süsteemi kohta, 5 erinevat klassifikaatori mudelit ning 14 muutuja kombinatsiooni iga faili ja mudeli kohta. Selle tulemusena leiti, et mõlema klassisüsteemi, kolmeklassiliste ja mitmeklassiliste andmete puhul tulid parimad esialgsed täpsused enamasti tugivektormasina mudeliga, kuid oli ka paar tulemust, mis näitasid otsustuspuu ülekaalu. Ülejäänud kasutatud mudelid ei saavutanud piisavalt häid tulemusi. Tabelis 7 on toodud mudeli valimise tulemused koos mudelite täpsustega.

Tabel 7: Mudeli valimise tulemused

Fail	Kolmeklassiline parim	Täpsus	Mitmeklassiline parim	Täpsus
Data1	Otsustuspuu	91.9	Tugivektormasin	77.6
Data2	Tugivektormasin	89.7	Tugivektormasin	78.6
Data3	Otsustuspuu	90.2	Tugivektormasin	78.1
Data4	Tugivektormasin	91.0	Otsustuspuu	75.7
Data5	Tugivektormasin	91.1	Tugivektormasin	78.1
Data6	Tugivektormasin	91.8	Tugivektormasin	78.2
Data7	Tugivektormasin	91.9	Tugivektormasin	78.2
Data8	Tugivektormasin	92.1	Tugivektormasin	78.9
Data9	Otsustuspuu	91.7	Otsustuspuu	85.8
Data10	Tugivektormasin	92.0	Otsustuspuu	79.3
Data11	Tugivektormasin	91.5	Otsustuspuu	78.4
Data12	Otsustuspuu	94.1	Otsustuspuu	79.2
Data13	Tugivektormasin	94.6	Tugivektormasin	79.7
Data14	Tugivektormasin	91.5	Tugivektormasin	77.4
Data15	Tugivektormasin	90.5	Tugivektormasin	79.0

4.2 Korrelatsioonimaatriks

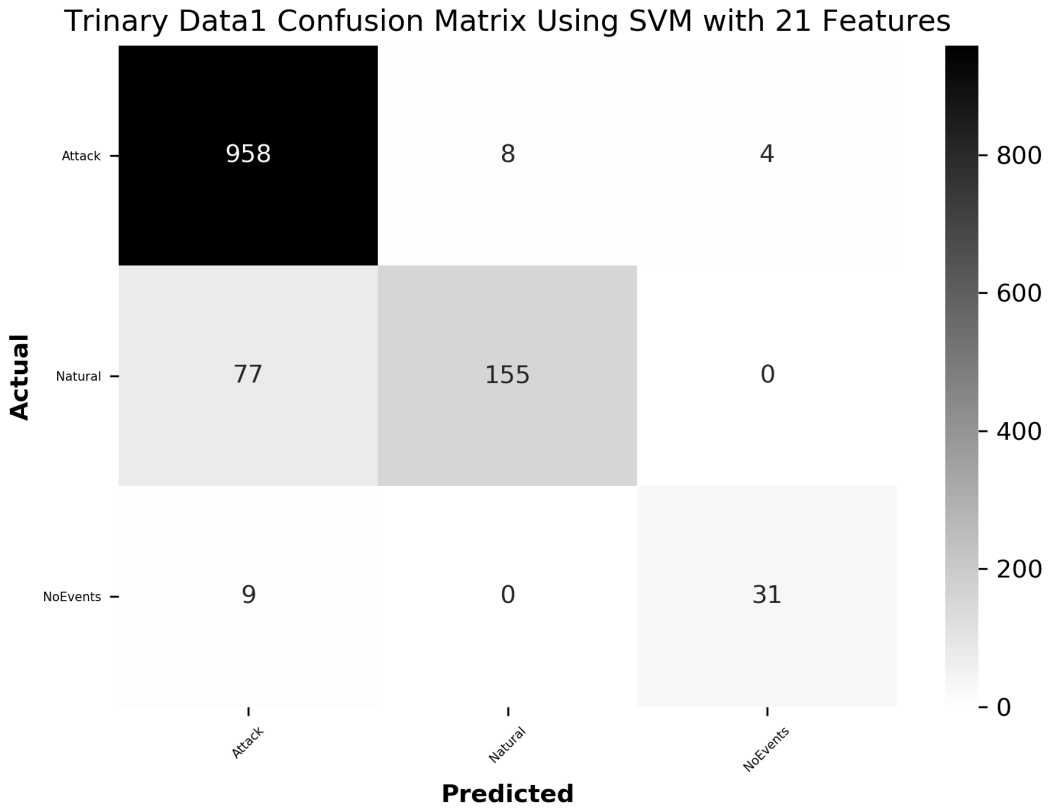
Lisaks lihtsalt Fisheri väärtuste põhjal muutujate valimisele, otsustati lisaks veel vaadata iga faili puhul selle faili muutujate lineaarseid sõltuvusi. Sellele aitas kaasa muutujate kohta korrelatsioonimaatriksi loomine, mis näitab otseselt vastavate muutujate lineaarset sõltuvust üksteise suhtes. Joonisel 2 on näitena toodud kolmeklassilise süsteemi faili Data1 15 parima muutuja korrelatsiooninäitajad. Mida suurem on näitaja muutujate lõikumispunktis, seda suurem on kummagi muutuja lineaarne sõltuvus üksteisest, olles minimaalselt -1 ning maksimaalselt 1. Kui sõltuvusnäitaja on väärtusega 1 kahe erineva muutuja korral, jäeti need muutujad lineaarselt mittesõltumatute muutujate valikust välja.



Joonis 2: Korrelatsioonimaatriksi näide

4.3 Segadusmaatriks

Klassifikaatori edukuse näitajaks oli vaja veel luua segadusmaatriks, mille abil on võimalik välja arvutada klassifikaatori kordustäpsus, saagis ning f-mõõt. Segadusmaatriks näitab maatrikskujul klassifikaatori ennustuste tulemusi. Sellelt maatriksilt on näha, kui täpsed klassifikatsioonid testandmetega tulid, andes peadiagonaalil õigesti ning ülejäänud maatriksil valesti klassifitseeritud andmete arvu. Segadusmaatriksi näidis antud töös kasutatud andmete põhjal on toodud joonisel 3.



Joonis 3: Segadusmaatriksi näide

4.4 Lõpptulemuste jaoks valitud sätted

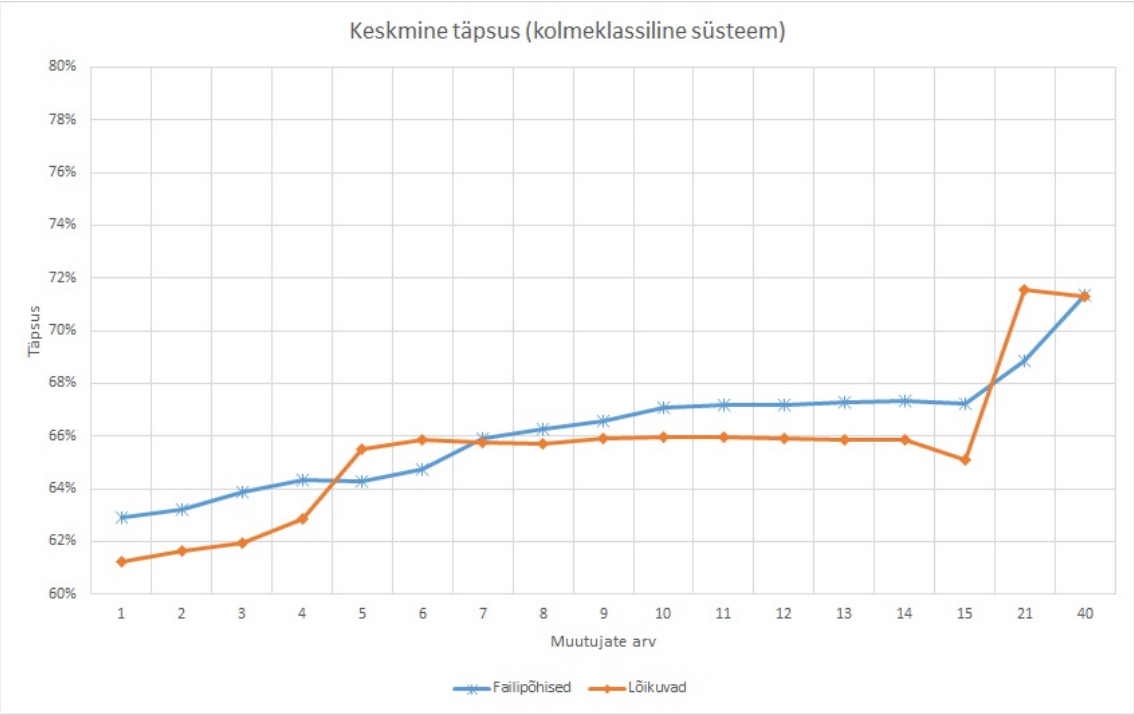
Klassifikaatori mudelite esmaste treenimiste tulemuste põhjal otsustati valida tugivektormasina mudel peamiseks klassifikaatoriks. Muutujate valikute suhtes otsustati kasutada Fisheri väärtuste põhjal parimaid failipõhiseid ning süsteemisiseselt lõikuvaid muutujaid. Lineaarselt sõltuvaid muutujaid ei leitud piisavalt, et tagada sellise muutujate valiku kasutamist.

5 Masinõppe tulemused

Pärast sobiva klassifikaatori mudeli leidmist jätkati tööd valitud muutujate konfiguratsioone kasutades vastava mudeli treenimisega. Esimese etapina treeniti tugivektormasina klassifikaatorit kolmeklassiliste andmega, et leida sobiv muutujate arv. Muutujate arvu otsimisel vaadati eelkõige, et see arv oleks võimalikult väike ning tunduvalt väiksem võrreldavas töös kasutatud muutujate arvust. Pärast sobiva muutuja arvu leidmisel treeniti vastavat mudelit ka kaheklassiliste ja mitmeklassiliste andmetega.

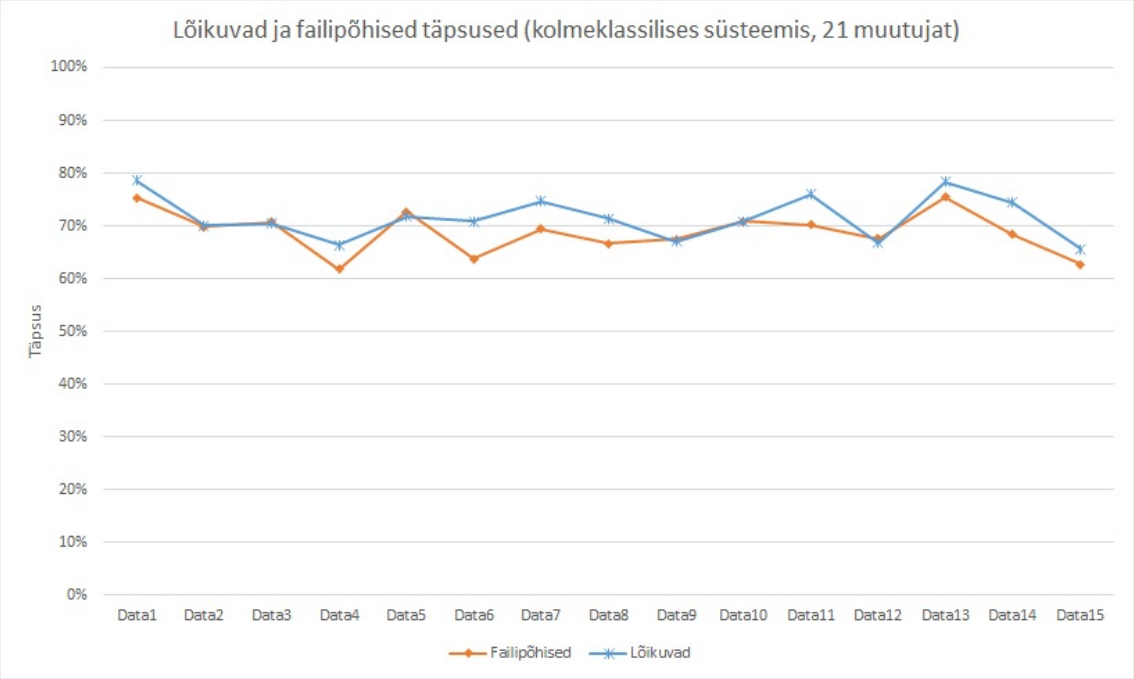
5.1 Kolmeklassiliste andmete treenimine

Lõpliku muutujate arvu leidmiseks treeniti tugivektormasina klassifikaatori mudelit, kasutades kolmeklassilisi andmeid. Sisenditeks anti nii parimad lõikuvad kui ka failipõhised muutujad, alustades 1 muutujast ning minnes järk-järgult kõrgema muutujate arvuni. Sobiv minimaalne muutuja arv leiti muutujate arvu kasvamisel olevat 21. Joonisel 4 on toodud kolmeklassiliste andmete keskmine täpsus sõltuvalt muutujate arvust ning sellelt on näha, et kuni 15 muutujani väga suuri hüppeid täpsuses ei esine. Küll aga on 15 muutujalt 21 muutujale minnes näha mitmeprotsendist tõusu, samas kui 21 ja 40 muutuja vahel ei toimunud täpsuses mingit suuremat tõusu. Vastupidiselt 40 lõikuva muutuja valikul täpsus hoopis langes. Ümardatult saavutati 21 lõikuva muutujaga 72% ning 21 failipõhiste muutujatega 69% täpsus.



Joonis 4: Kolmeklassiliste andmete keskmiste täpsuste graaf

Parima muutujate arvu leidmisel sooritati tugivektormasina treenimised vastava muutujate arvu ning kahe valitud muutujate valiku konifguratsioonidega. Joonisel 5 on vastavad tulemused iga andmefaili kohta.



Joonis 5: Kolmeklassiliste andmefailide täpsuste graaf 21 muutujaga

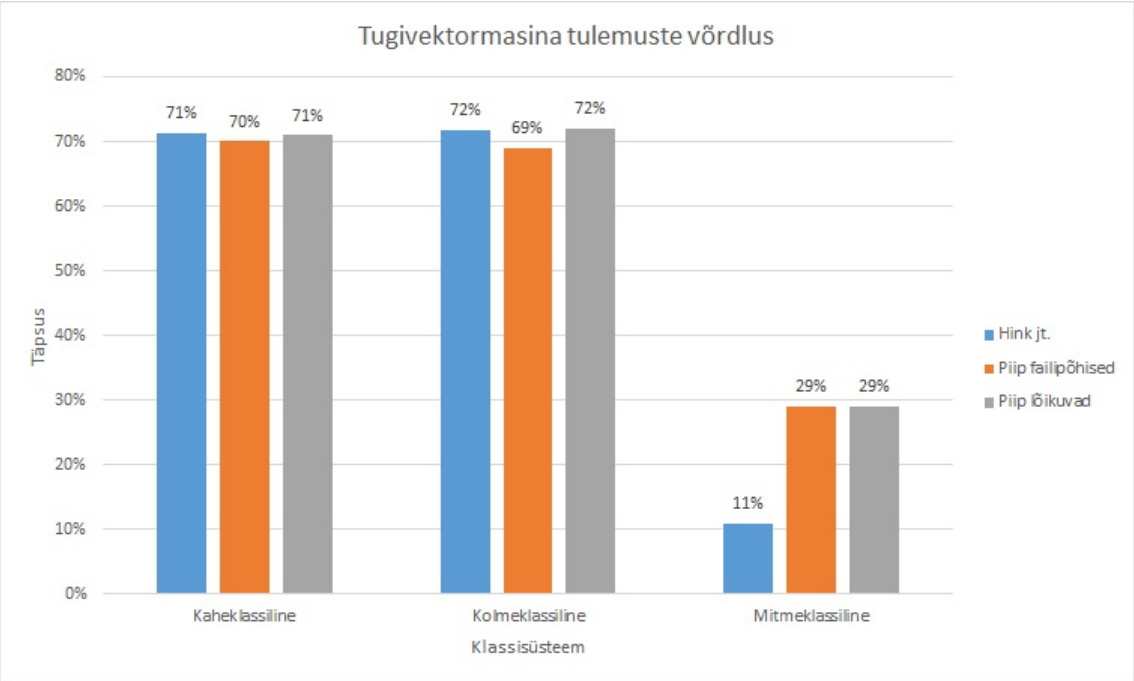
5.2 Treenimine kõigi klassisüsteemidega

Saades kolmeklassiliste andmetega hinnanguliselt head tulemused, otsustati samade muutujavalikutega ning klassifikaatori mudeliga teostada treenimised ka kaheklassiliste ja mitmeklassiliste andmetega.

Kaheklassiliste andmetega treenitud klassifikaator saavutas lõikuvate muutujatega ümardatult 71% täpsuse, kuid failipõhiste muutujatega õige pisut halvema, 70% täpsuse. Mitmeklassiliste andmetega saavutas tugivektormasina mudel nii lõikuvate kui ka failipõhiste muutujatega 29% täpsuse.

5.3 Tulemuste võrdlemine Hinki ja teiste töö tulemustega

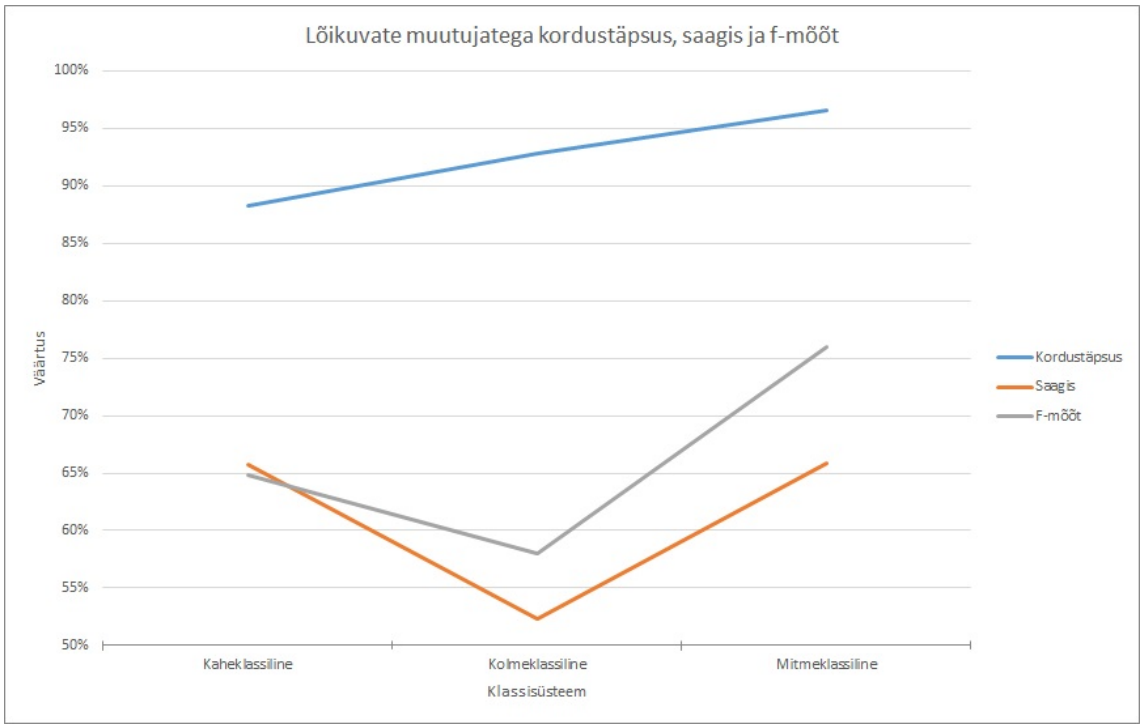
Eelnevas peatükis mainitud tugivektormasina tulemusi on võimalik võrrelda Hinki ja teiste töö tulemustega. Antud lõputöös saavutati keskmiselt sarnase täpsusega tugivektormasina klassifikaator nii kaheklassilises kui ka kolmeklassilises süsteemis. Mitmeklassilises süsteemis saadi antus töös kordades parem tulemus. Hinki ja teiste töö tugivektormasina tulemused on ligikaudu 71%, 72% ning 11% vastavalt kaheklassilises, kolmeklassilises ja mitmeklassilises süsteemis [6]. Joonisel 6 on toodud võrreldava ning käesoleva töö tulemusi võrdlev graaf. Kuigi täpsuse suhtes ei saadud üldiselt paremat tulemust, saadi samas suurusjärgus täpsus vähemate muutujatega. Võrreldavas töös kasutati 40 paremat muutujat informatsiooni kasumi kasvu (ingl. k. *information gain*) alustel, kuid antud töös kasutati 21 Fisheri väärtuste ning lõikuvuse alusel olevaid muutujaid [6].



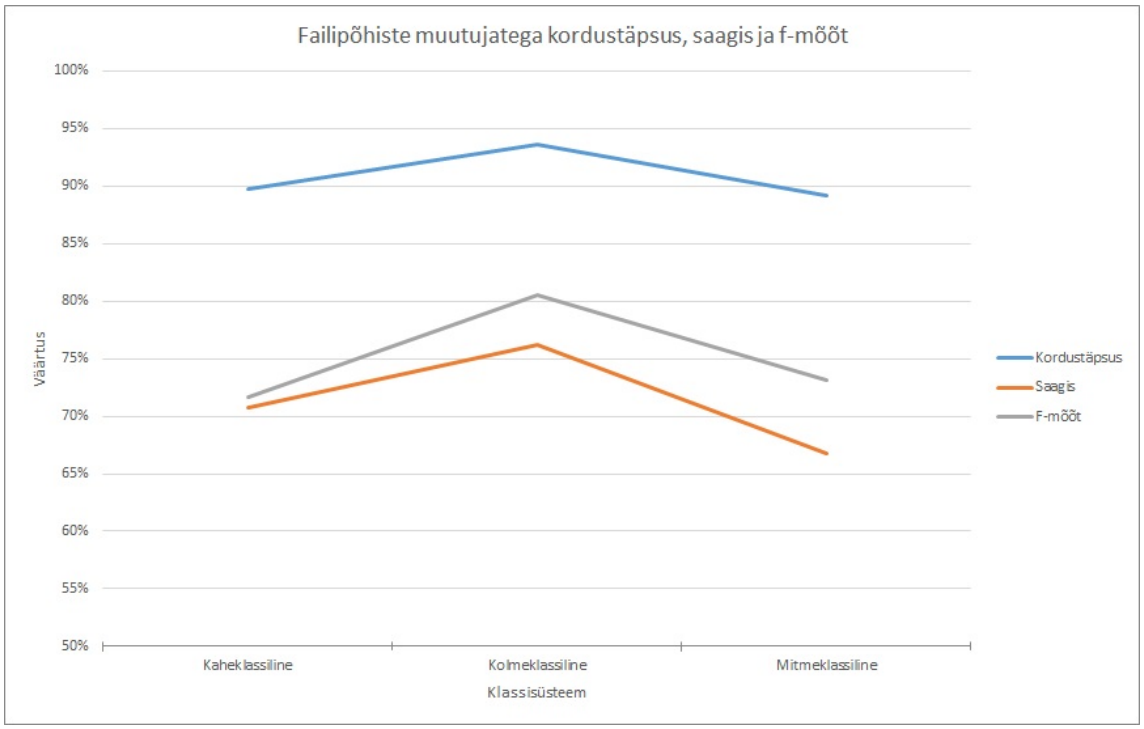
Joonis 6: Lõpptulemusi võrdlev graaf

Hinki ja teiste töös välja toomata jäetud põhjustel on tugivektormasina kordustäpsus, saagis ning f-mõõt kõigis kolmes klassisüsteemis peamiselt alla 0.1, kuid kaheklassiliste andmete puhul on kordustäpsus natuke üle 0.9 [6]. Antud töös saavutati kordustäpsusteks lõikuvate muutujatega ligikaudu 93% ning failipõhistega 91%. Saagised tulid seevastu palju madalamad lõikuvate muutujate valikul, olles keskmiselt 61%. Failipõhiste muutujate valikul oli saagis keskmiselt 71% läheduses. F-mõõdud tulid vastavalt lõikuvate ja failipõhiste muutujatega keskmiselt 66% ja 75%. Kolmeklassiline süsteem sai üldiselt halvimad tulemused lõikuvate muutujatega, kuid parimad failipõhiste muutujatega.

Joonistel 7 ning 8 on välja toodud antud töös saadud kordustäpsuste, saagiste ja f-mõõtude tulemused kõigis kolmes klassisüsteemis ning mõlema muutujate konfiguratsiooniga.



Joonis 7: Lõikuvate muutujatega kordustäpsus, saagis ja f-mõõt



Joonis 8: Failipõhiste muutujatega kordustäpsus, saagis ja f-mõõt

5.4 K-lähima naabri ning otsustuspuu tulemused

Töö lõpus treeniti ka testimise mõttes k-lähima naabri ning otsustuspuu mudelid erinevate lõikuvate muutujatega, sest võrreldavas töös on kasutatud ka lähima naabri sarnast mudelit ning otsutuspuu esines parima mudeli valimises. Muutujate arvuks valiti N, kus N=1...21. Kokkuvõttev info on kajastatud Tabelites 8 ja 9. Iga muutuja arvu valikul treeniti vastava klassisüsteemi igat 15 faili ning nende tulemuste keskmine valiti selle muutuja arvu valiku esindajaks.

Otsustuspuu treenimiste keskmisteks täpsusteks saadi 63%, 58% ning 25% vastavalt kaheklassiliste, kolmeklassiliste ja mitmeklassiliste andmetega. Parimateks tulemusteks saavutati vastavalt 70%, 59% ning 45% ning kasutati vastavates süsteemides 2, 19 ja 21 muutujat.

Tabel 8: Otsustuspuu tulemused lõikuvate muutujatega

Klassisüsteem	Keskmine	Parim	Parima muutujate arv
Kaheklassiline	63%	70%	2
Kolmeklassiline	58%	59%	19
Mitmeklassiline	25%	45%	21

Kaheklassiliste andmetega saavutas k-lähima naabri mudel keskmiselt 64% täpsuse, kolmeklassiliste andmete puhul 60% ning mitmeklassilises süsteemis saadi keskmiseks täpsuseks 27%. Kaheklassiliste andmete suurim täpsus oli 68% ning kasutas 2 muutujat. Kolmeklassiliste andmete parim saavutas 19 muutujaga 62% täpsuse. Mitmeklassilises süsteemis kasutas parim tulemus 21 muutujat, saades täpsuseks 49%.

Nii kaheklassiliste kui ka kolmeklassiliste andmete tulemused ei ületanud tulemused tugivektormasina täpsusi, kuid mitmeklassiliste andmete puhul saavutas nii k-lähima naabri kui ka otsustuspuu mudel parema tulemuse. Võrreldava töö lähima naabri täpsused on keskmiselt 63%, 64% ning 24% vastavalt kaheklassilises, kolmeklassilises ja mitmeklassilises süsteemis [6]. Võrreldes Hinki ja teiste lähima naabri tulemustega, saavutasid nii kaheklassiline kui ka kolmeklassiline süsteem

k-lähima naabri mudeliga hinnanguliselt sarnase tulemuse, kuid nagu ka tugivektormasinaga, ületas antud töö mitmeklassilise süsteemi tulemus võrreldava töö tulemuse peaaegu 20% jagu.

Tabel 9: K-lähima naabri tulemused lõikuvate muutujatega

Klassisüsteem	Keskmine	Parim	Parima muutujate arv
Kaheklassiline	64%	68%	2
Kolmeklassiline	60%	62%	19
Mitmeklassiline	27%	49%	21

5.5 Arendussuunad

Andmete klassifitseerimisel on kaks peamist tegurit, millega peab arvestama - muutujate ning klassifikaatori valik. Selle tõttu on saadud tulemusi võimalik kahel viisil parendada. Esimeseks viisiks on muutujate valiku muutmine ning teiseks on teistsuguse klassifikaatori valimine.

Kuigi antud lõputöös toimus klassifikaatorite sisenditeks minevate muutujate korrapärane valik Fisheri väärtuste põhjal, tulid vastavad väärtused soovitust väiksemad. Muutujate valiku puhul oleks kindlasti võimalik kasutada teistsuguseid valikumeetodeid kui Fisheri väärtuste põhjal.

Antud töös kasutati erinevatesse klassidesse kuuluvate andmete klassifitseerimiseks paari lihtsamat ja klassikalisemat klassifikaatorit. Hinki ja teiste töös kasutati veel Ripperi klassifikaatorit ning sellega koos ka AdaBoosti (*Adaptive Boosting*) meta-algoritmi, mis saavutasid võrreldavas töös kõigis kolmes klassisüsteemis parimad tulemused [6].

6 Kokkuvõte

Antud lõputöö eesmärgiks oli klassifikaatorite abil küberrünnakute tuvastamine, treenides klassifikaatorid testandmete abil. Teise taseme eesmärgiks oli võimalusel treenida klassifikaator, mis kasutaks kas vähem muutujaid või oleks täpsem kui Hinki ja teiste sama mudeli klassifikaator [6].

Klassifikaatorite sisenditeks valitavate muutujate paremaks otsustamiseks arvutati iga muutuja Fisheri väärtused, valides suuremate väärtustega muutujad sisenditeks. Lõpliku täpsuse arvutamine ning klassifikaatori valideerimine toimus k-korda ristvalideerimise abil.

Esmalt kasutati viit erinevat klassifikaatorit - k-lähima naabri, tugivektormasina, otsustuspuu, logistilise regressiooni ning lineaarse diskriminandi analüüsi mudelit. Analüüsi käigus leiti, et antud andmete jaoks sobib kõige paremini tugivektormasina mudel, kuid lõpus treeniti samade muutujate valiku sätetega ka k-lähima naabri ning otsustuspuu mudeleid. Muutujate valiku suhtes otsustati kasutada parimaid failipõhiseid ning lõikuvaid muutujaid.

Parimaks tulemuseks saadi tugivektormasina klassifikaator, mis kolmeklassilise süsteemi andmeid kasutades tuvastab ja eristab rünnakuid mitte-rünnakutest edukalt 72% täpsusega. Võrreldava tööga on muutujate arv tunduvalt väiksem, antud töös 21 ning teises töös 40. Samas saavutati mitmeklassilisi andmeid kasutades võrreldava tööga palju täpsem klassifikaator - antud töös 29% ning võrreldavas töös 11% [6]. Tulemused tulid kas samaväärsed või isegi paremad, kui võrreldavas töös, sest muutujute valimeid optimiseeriti ning selle tõttu on ka saadud klassifikaatorite keerukus madalam.

Töö käigus omandas autor uusi kogemusi progammeerimises üldiselt ning ka esmaseid teadmisi ja kogemusi masinõppe ja klassifikaatorite valdkonnas.

Kasutatud kirjandus

[1] "Industrial Control System." [Võrgumaterjal]. Available: https://www.trendmicro.com/vinfo/us/security/definition/industrial-control-system [Kasutatud 24.04.2019].

[2] "The state of industrial cybersecurity 2018," 2018. [Võrgumaterjal]. Available: https://ics.kaspersky.com/media/2018-Kaspersky-ICS-Whitepaper.pdf. [Kasutatud 05.04.2019].

[3] D. McMillen, "Security attacks on industrial control systems," October 2015. [Võrgumaterjal]. Available: https://www.ibm.com/downloads/cas/4PE7DQ3G. [Kasutatud 05.04.2019].

[4] "Dell Security annual threat report 2015," 2015. [Võrgumaterjal]. Available: http://proconics.co.za/wp-content/uploads/2017/10/2425.pdf. [Kasutatud 06.04.2019].

[5] D. McMillen, "Attacks Targeting Industrial Control Systems (ICS) Up 110 Percent," December 2016. [Võrgumaterjal]. Available: https://securityintelligence.com/attacks-targeting-industrial-control-systems-ics-up-110-percent/ [Kasutatud 09.04.2019].

[6] R. C. Borges Hink, J. M. Beaver, M. A. Buckner, T. Morris, U. Adhikari, and S. Pan, "Machine learning for power system disturbance and cyber-attack discrimination," in *2014 7th International Symposium on Resilient Control Systems (ISRCS)*, pp. 1–8, Aug 2014.

[7] R. C. Borges Hink, J. M. Beaver, T. Morris, U. Adhikari, and S. Pan, "Industrial Control System (ICS) Cyber Attack Datasets," October 2015. [Võrgumaterjal].

Available: https://sites.google.com/a/uah.edu/tommy-morris-uah/ics-data-sets. [Kasutatud 09.04.2019].

[8] R. C. Borges Hink, J. M. Beaver, T. Morris, U. Adhikari, and S. Pan, "Power System Attack Datasets - Mississippi State University and Oak Ridge National Laboratory - 4/15/2014," April 2014. [Võrgumaterjal]. Available: http://www.ece.uah.edu/~thm0009/icsdatasets/PowerSystem_Dataset_README.pdf [Kasutatud 24.04.2019].

[9] "Snort FAQ." [Võrgumaterjal] URL: https://www.snort.org/faq/what-can-i-do-with-snort. [Kasutatud 13.09.2019].

[10] C. C. Aggarwal, *Data Mining: The Textbook*. Springer, 2015.

[11] J. Brownlee, "K-Nearest Neighbors for Machine Learning," Aprill 2016. [Võrgumaterjal]. Available: https://machinelearningmastery.com/k-nearest-neighbors-for-machine-learning/. [Kasutatud 02.04.2019].

[12] "1.4 Support Vector Machines," 2018. [Võrgumaterjal]. Available: https://scikit-learn.org/stable/modules/svm.html. [Kasutatud 02.04.2019].

[13] "1.10 ecision Trees," 2018. [Võrgumaterjal]. Available: https://scikit-learn.org/stable/modules/tree.html. [Kasutatud 28.03.2019].

[14] S. Joglekar, "Logistic Regression (for dummies)," August 2015. [Võrgumaterjal]. Available: https://codesachin.wordpress.com/2015/08/16/logistic-regression-for-dummies/. [Kasutatud 01.04.2019].

[15] J. Brownlee, "Linear Discriminant Analysis for Machine Learning," Aprill 2016. [Võrgumaterjal]. Available: https://machinelearningmastery.com/linear-discriminant-analysis-for-machine-learning/. [Kasutatud 02.04.2019].

[16] "sklearn.model_selection.train_test_split." [Võrgumaterjal]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html. [Kasutatud 13.05.2019].

[17] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning with Applications in R*. Springer, 2013.