LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

AUTOMĀTISKA E-PASTU APSTRĀDES UN ATBILDES SNIEGŠANAS SISTĒMA

KVALIFIKĀCIJAS DARBS

Autors: **Edvards Bukovskis**

Studenta apliecības Nr.: eb18081

Darba vadītājs: Dr. dat. Jānis Zuters

RĪGA 2021

Anotācija

Kvalifikācijas darbā tika izstrādāts Automātiskās E-pastu Atbildes Sniegšanas Sistēmas (turpmāk - AEAASS) projektējums, programatūras specifikācija un daļēja sistēmas programmatūras realizācija. Sistēma un tās koncepti tika veidoti reāla uzņēmuma projekta ietvaros, un šis programmatūras specifikācijas dokukuments ir pamats turpmākai sistēmas programmatūras attīstībai un uzlabošanai nākotnē.

AEAASS ir risinājums reālai darbinieku darba slodzes un šī brīža COVID-19 pandēmijas izraisīto seku problēmām, uzņēmuma interesēs. Sistēmas būtība ir lielo datu izmantošana konkrētu darbību automatizācijai, izmantojot tādus elementus kā: dabiskā valodas apstrāde (NLP) un mašīnapmācība (ML).

Atslēgvārdi: automatizācija, datu apstrāde, lielie dati, mašīnapmācība, dabiskā valodas apstrāde, Python

Automatic E-mail Processing and Response System

Abstract

In the qualification work, the design of the Automatic E-mail Response System (hereinafter - AEAASS), software specification and partial implementation of the system software were developed. The system and its concepts were developed within the framework of a real company project, and this software specification document is the basis for further development and improvement of the system software in the future.

AEAASS is a solution to the real problems of employee workload and the consequences of the current COVID-19 pandemic, in the interests of the company. The essence of the system is the use of big data to automate specific activities using elements such as: natural language processing (NLP) and machine learning (ML).

Keywords: Automation, Data Processing, Big Data, Machine Learning, Natural Language Processing, Python

Saturs

Vārdnīca

NLP	Dabiskā valodas apstrāde (no angļu *Natural Language Processing*)
PPS	Programmatūras prasību specifikācija
PPA	Programmatūras projektējuma apraksts
AEAASS	Automātiska e-pastu apstrādes un atbildes sniegšanas sistēma
ML	Mašīnapmācība (no angļu *Machine Learning*)
Python	Augsta līmeņa interpretējama objektorientētā programmēšanas valoda
SMTP	Vienkāršais pasta pārsūtīšanas protokols
Firewall	Ugunsmūris

1 IEVADS

1.1 Nolūks

Dokuments ir sagatavots kvalifikācijas darba un uzņēmuma projekta ietvaros. Dokumentā tiek aprakstīta AEAASS struktūra un funkcionalitāte, kā arī, dokumetns sevī ietver pasūtītāju gatavā produkta redzējumu un tā daļēju tehnisko izpildījumu, kas turpmāk uzņēmumam būs pamats turpmākai sistēmas attīstībai.

Darbs tika veikts saistībā ar iepriekšējo praksi uzņēmumā SIA "Tet", lai risinātu problēmu uzņēmuma interesēs.

1.2 Darbības sfēra

AEAASS ir paredzēts intergrēt uzņēmuma klientu apkalpošanas un palīdzības nodaļā, lai atvieglotu darba slodzi un virzītos uz kopēju sistēmu automatizēšanu. AEAASS būtu apakšsistēma jau esošai uzņēmuma sistēmai.

1.3 Saistība ar citiem dokumentiem:

Dokumenta PPS noformēšanā ievērotas standarta LVS 68:1996 prasības.

AEAASS PPS ir cieši saistīs ar šīs pašas sistēmas PPA.

Dokumenta PPS un PPA kalpo kā atsauce turpmākiem sistēmas uzlabojumiem un citām šīs sistēmas novirzēm uzņēmumā.

1.4 Pārskats:

Pirmajā daļā ir ievadinformācija un vispārējs apraksts, kas satur dokumenta nolūku, definīciju skaidrojumu, Saistību ar citiem dokumentiem un sistēmas darbības sfēru.

Otrā daļa sniedz pārskatu par sistēmas esošo stāvokli, pasūtītāju, sistēmas perspektīvu un vispārēju sistēmas funkcionalitāti un būtību.

Trešajā daļā tiek aprakstītas funkcionālās prasības un nefunkcionālās prasības

Ceturtajā daļā PPA tiek sīkāk aprakstīta nozīmīgu sistēmas struktūrvienību uzbūve un darbošanās princips.

Piektajā daļā tiek aprakstīta funkciju testēšana.

2 VISPĀRĒJS APRAKSTS

2.1 Esošā stāvokļa apraksts

Ņemot vērā uzņēmuma pieaugošo klientu skaitu, kā arī, pašreizējo situāciju pandēmijas laikā, klientu apkalpošanas un palīdzības nodaļai rodas lielāka noslodze uz esošo darbinieku skaitu. Lai risinātu šo radušos problēmu, rodas nepieciešamība automatizēt atsevišķas, monotonas darbības. Risinājumam ir jābūt pēc iespējas automatizētam, lai mazinātu darbinieku noslogojumu un veicinātu augstāku produktivitāti, tāpēc tiek veidots projekts, kura ietvaros tiek izstrādāta apakšsistema, kas papildinās esošo sistemu.

2.2 Pasūtītājs

Faktiskais sistēmas pasūtītājs ir klientu apkalpošanas un palīdzības nodaļa uzņēmumā.

2.3 Produkta perspektīva

Sistēma pati par sevi ir neatkarīga, tāpēc to ir iespējams pielāgot un modificēt citām vajadzībām. Konkrētajā gadījumā, kad sistēma kalpo kā apakšsistēma jau esošai sistēmai, tai ir iespējams pievienot papildu funkcionalitātes.

2.4 Sistēmas funkcionalitāte

1. Teksta apstrāde
2. Teksta satura atapzīšana
3. Atbildes sniegšana, balstoties uz teksta saturu
4. ML modeļa trenēšana

2.5 Sistēmas lietotāji

Sistēmas netiešie lietotāji ir uzņēmuma klientu apkalpošanas un palīdzības nodaļas darbinieki, jo sistēma darbojas automātiski un tiešas mijiedarbības ar fiziskām personām nav paredzētas.

2.6 Vispārējie ierobežojumi

Sistēmu ir nepieciešams integrēt SMTP serverī, kā arī, nepieciešams interneta savienojums.

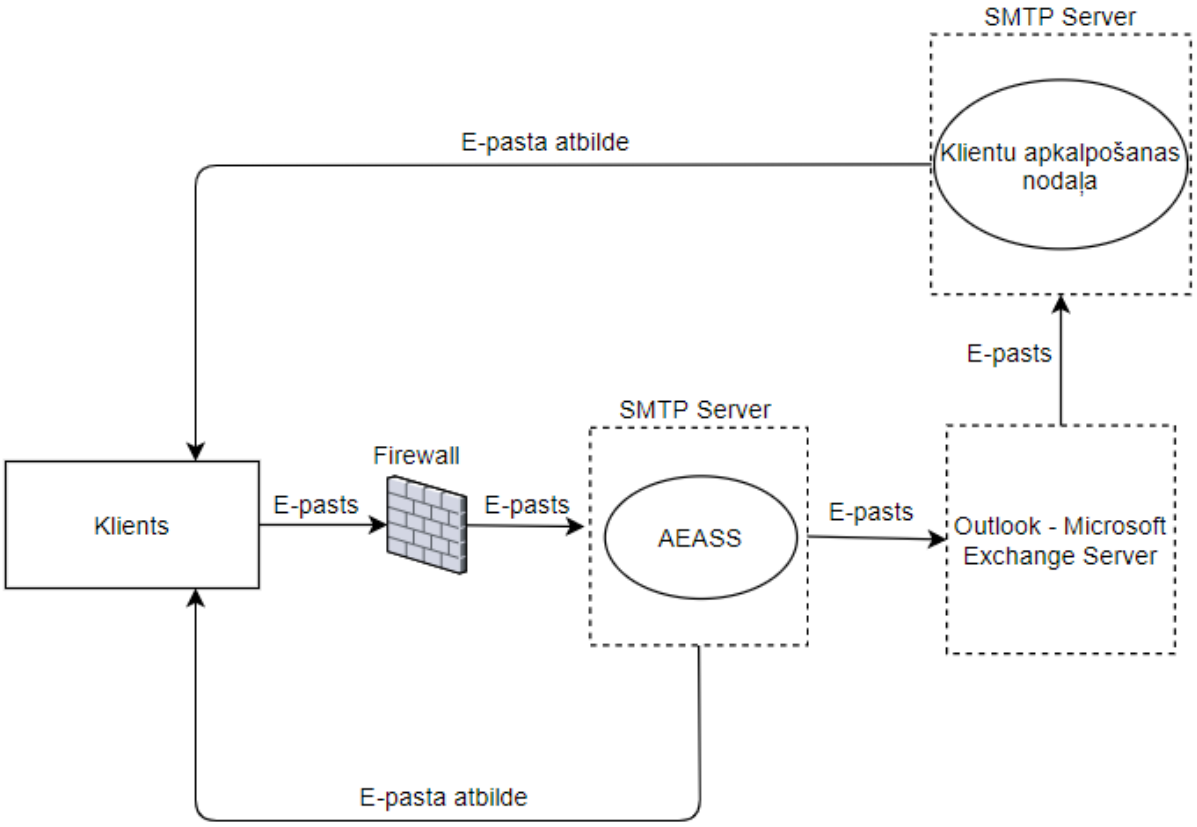
2.7 Pieņēmumi un atkarības

AEAASS izstrādātie pieņēmumi, kuri nepieciešami sistēmas darbībai:

- Sistēma ir citas sistemas apakšsistēma

3 PROGRAMMATŪRAS PRASĪBU SPECIFIKĀCIJA

3.1 Datu plūsma

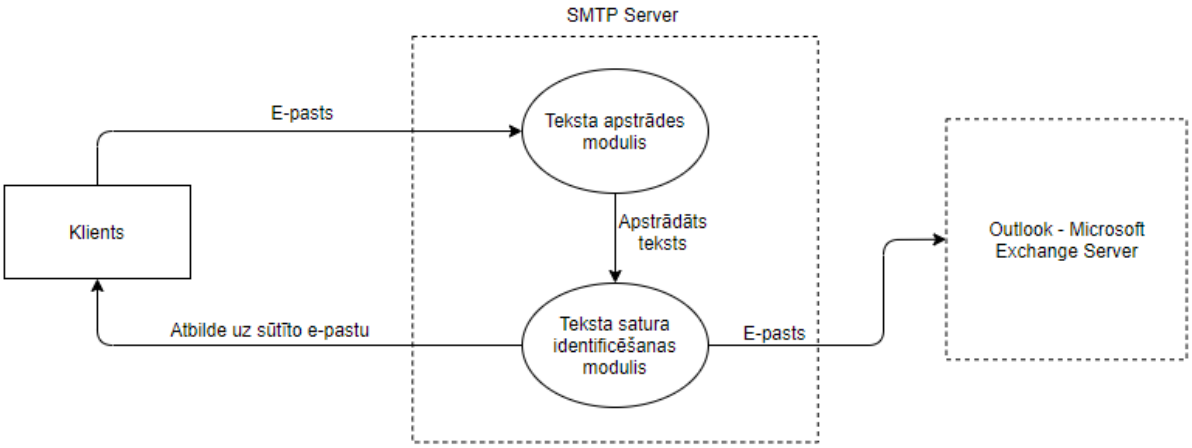


Attēls 3.1 **0. līmeņa datu plūsmas diagramma**

AEASS ir kā starpposms (skat. Attēls 3.1), kas apstrādā klientu e-pastus pirms tie nonāk klientu apkalpošanas un palīdzības nodaļas sistēmā. AEASS tiek integrēts SMPT serverī, kur ar apmācītu ML modeli tiek veikta e-pastu filtrācija. Tālāk tiek realizēts "luksofora" princips – ja e-pasts atbilst AEASS kritērijiem, tad tiek automātiski nosūtīta atbilde atpakaļ klientam, pretējā gadījuma, e-pasts tiek nodots tālāk uz manuālu apstrādi klientu apkalpošanas un palīdzības nodaļai.

3.2 Funkcionālās prasības

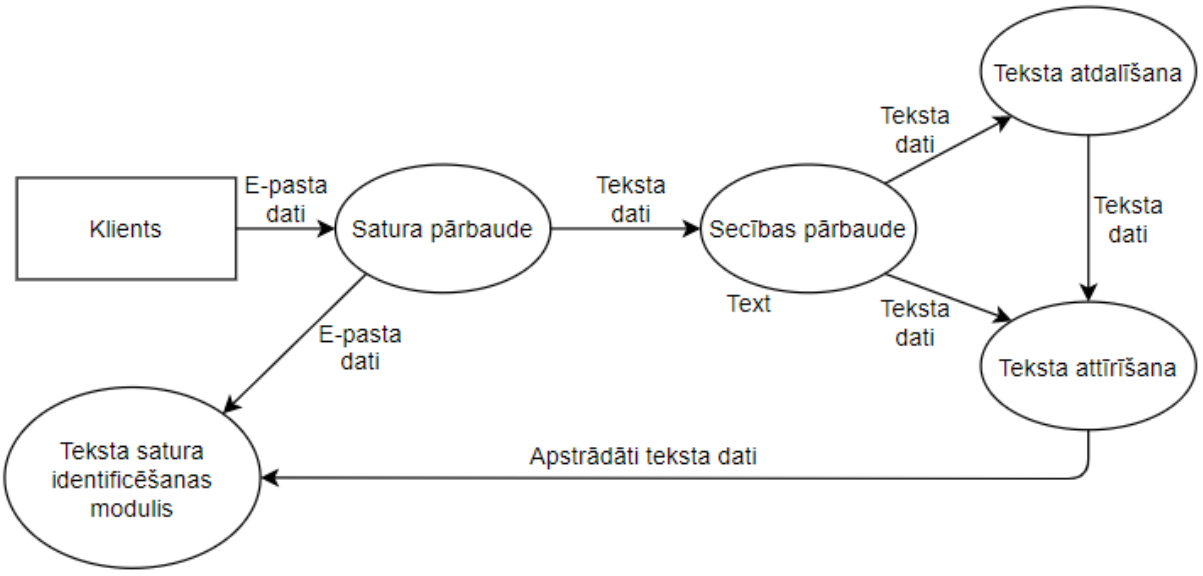
3.2.1 Moduļu sadalījums un apraksts



Attēls 3.2 **1. līmeņa datu plūsmas diagramma**

AEASS satur divus moduļus (skat. Attēls 3.2) – Teksta apstrādes moduli un teksta satura pārbaudes moduli. Teksta apstrādes modulis atbild par ienākošā e-pasta teksta formāta sagatavošanu padošanai ML modelis, kas ir iekļauts teksta satura pārbaudes modulī. Teksta satura identificēšanass modulis atbild par teksta satura būtības pārbaudi, atbilžu sniegšanu, vai e-pasta nosūtīšanu tālākai apstrādei uz klientu apkalpošanas centra sistēmu.

3.2.2 Teksta apstrādes moduļa apraksts



Attēls 3.3 **Teksta apstrādes moduļa datu plūsmas diagramma**

Teksta apstrādes modulis (skat. Attēls 3.3) sastāv no četrām funkcijām – Satura pārbaudes, e-pasta secības pārbaudes, teksta atdalīšanas un teksta attīrīšanas funkcijām. Katras funcijas mērķis šajā modulī ir apstrādāt teksta datus tādā veidā, lai Teksta satura identificēšanas modulis tos spētu atpazīt un pielietot ML modelim.

Tabula 3.1

Satura pārbaudes funkcijas apraksts

Mērķis:
Atlasīt gadījumus, kuros e-pasta saturā tiek sūtīts pielikums, piemēram, dokuments vai attēls
Ievaddati:
SMTP serverī saņemtais e-pasts
Apstrāde:
Tiek pārbaudīts vai e-pasts satur tikai simbolu virkni, vai arī pielikumu/-us
Izvaddati:
Gadījumā, ja e-pasts nesatur pielikumus: • Dati tiek nodoti tālāk secības pārbaudes funkcijai Gadījumā, ja e-pasts satur pielikumu: • Tam tiek piešķirta speciāla iezīme un e-pasts tiek nodots Teksta satura identificēšanas modulim

Tabula 3.2

Secības pārbaudes funkcija

Mērķis:
Atšķirt e-pastus, kurus klients raksta pirmo reizi un e-pastus, kas ir daļa no pastāvošas sarakstes.
Ievaddati:
Teksta dati (simbolu virkne), kurus atgriež funkcija content_check
Apstrāde:
Tiek pārbaudīts vai simbolu virknē ir atrodamas iezīmes, kas liecinātu par to, ka e-pasta teksts nav pirmreizējs, t.i. vai e-pasts nav sastāvdaļa no esošas sarakstes ar klientu apkalpošanas un palīdzības nodaļu.
Izvaddati:
Teksta attīrīšanas vai teksta atdalīšanas funkciju izsaukums

Tabula 3.3

Teksta atdalīšanas funkcijas apraksts

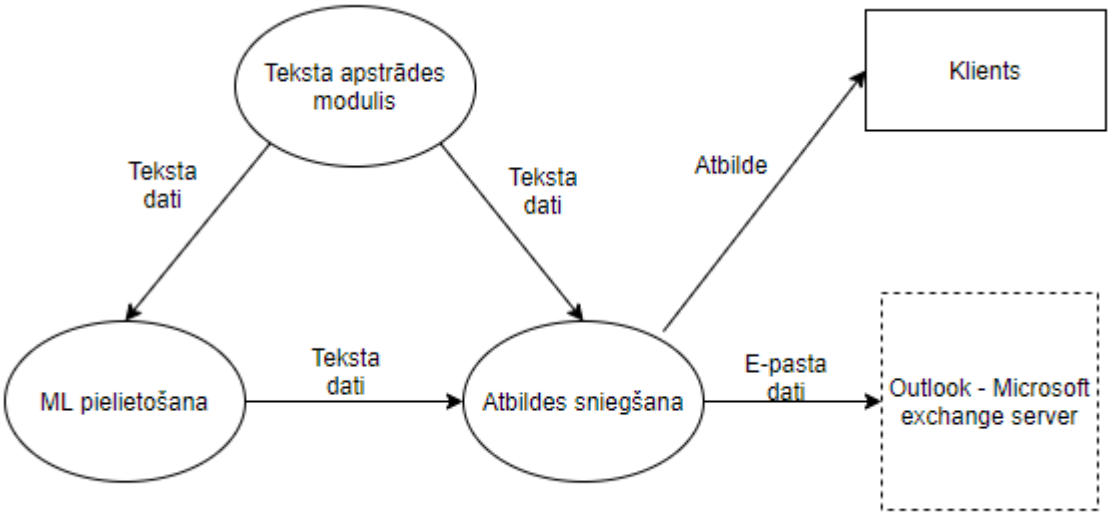
Mērķis:
Atdalīt nevajadzīgās simbolu virknes daļas, gadījumā, ja e-pasts nav pirmreizējs, t.i ja e-pasts ir sastāvdaļa no sarakstes
Ievaddati:
Teksta dati (simbolu virkne), kurus atgriež funkcija content_check
Apstrāde:
E-pastu sarakstes struktūrai ir konkrētas, statiskas iezīmes, pēc šīm iezīmēm tiek atdalīts nepieciešamais teksts.
Izvaddati:
Simbolu virkne, kurā ir konkrēts klienta sūtīts e-pasta jautājums.

Tabula 3.4

Teksta attīrīšanas funkcijas apraksts

Mērķis:
Apstrādāt simbolu virkni, lai to spētu atpazīt un identificēt ML modelis Teksta satura identificēšanas modulī
Ievaddati:
Simbolu virkne no funkcijas split_important vai pirmreizējā simbolu virkne.
Apstrāde:
1. Simbolu virkne tiek attīrīta no nesvarīgas informācijas, kura nemaina teksta galveno domu vai būtību, piemēram, simbolu virkne tiek attīrīta no e-pastu adresēm, biežākajiem stopvārdiem. 2. Simbolu virkne tiek transformēta mazajos burtos un tiek noņemtas garumzīmes. 3. Simbolu virknes, kuras nesatur latīņu alfabēta burtus, tiek speciāli iezīmētas.
Izvaddati:
Apstrādāta un attīrīta simbolu virkne

3.2.3 Teksta satura identificēšanas modulis



Attēls 3.4 **Teksta identificēšanas moduļa datu plūsmas diagramma**

Teksta identificēšanas modulis (skat. Attēls 3.4) sastāv no divām pamatfunkcijām – ML pielietošanas un atbildes sniegšanas. ML pielietošanas funkcija atbild par teksta klasificēšanu, izmantojot apmācītu ML modeli. Pēc attiecīgā klasifikatora, tiek sniegta automātiska e-pasta atbilde klientam vai arī e-pasts tālāk tiek nodots uz klientu apkalpošanas un palīzības centra sistēmu.

Tabula 3.5

ML pielietošanas funkcijas apraksts

Mērķis:
Identificēt un iezīmē tekstu pēc satura būtības
Ievaddati:
Apstrādāta simbolu virkne no teksta apstrādes moduļa
Apstrāde:
Tiek pielietots apmācīts ML modelis, kas identificē un klasificē simbolu virkni
Izvaddati:
Klasificēta simbolu virkne

Tabula 3.6

Atbildes sniegšanas funkcijas apaksts

Mērķis:
Sniegt iepriekš sagatavotu atbildi klientam vai nodot e-pastu tālākai apstrādei uzņēmuma klientu apkalpošanas un palīdzības nodaļas sistēmai.
Ievaddati:
E-pasta dati, klasificēta simbolu virkne.
Apstrāde:
Funkcijai ir noteiktas darbības pie konkrētām iezīmēm, un attiecībā no iezīmes tiek sniegta vai nu atbilde klientam, vai e-pasts tiek nodots tālākai apstrādei.
Izvaddati:
E-pasta atbilde

3.3 Nefunkcionālās prasības

3.3.1 Uzturamība

Programmas kodam ir jābūt labi strukturētam, komentētam un viegli uztveramam, jāievēro labs programmēšanas stils

3.3.2 Lietojamība

Sistēma nav paredzēta tiešai mijiedarbībai ar klientu apkalpošanas un palīdzības nodaļas darbiniekiem, sistēma tiek integrēta jau esošā e-pastu apmaiņas sistēmā kā starpposms. Ja rodas nepieciešamība pie sistēmas koda rediģēšanas vai sistēmas koda modificēšanas citiem nolūkiem uzņēmumā, tad attiecīgi konkrētām personām ir jānodrošina pieeja šādai iespējai.

3.3.3 Veiktspēja

Sistēmai ir jādarbojas ar augstu precizitāti, pretējā gadījumā paredzētais darba slogs netiks atvieglināts, bet papildu noslogots.

3.3.4 Izvietojamība

Tiek paredzēts, ka sistēmu ir viegli modificēt, un sistēmu vai tās daļas ir iespējams viegli pielāgot citiem uzņēmuma nolūkiem.

3.3.5 Drošība

Sistēmas datiem ir jābūt droši glabātiem, lai tiem nav iespējams piekļūt no ārējām sistēmām. Piekļuvi sistēmas datiem, pēc nepieciešamības, drīkst sniegt tikai kvalificētam personālam konkrētiem un skaidriem nolūkiem.

4 PROGRAMMATŪRAS PROJEKTĒJUMA APRAKSTS

4.1 Ievads

4.1.1 Nolūks

AEAASS programmatūras projektējuma apraksts sniedz padziļinātu informāciju par sistēmas elementu uzbūvi, struktūru un darbības principiem. PPA tiek aprakstīta ML modeļa uzbūve, modeļa trenēšana ar datiem un darbības princips, kas sevī ietver lielo datu apstrādi ar NLP metodēm.

4.1.2 Darbības sfēra

AEAASS PPA ir paredzēts padziļinātai informācijai par sistēmas struktūrvienībām. Šis dokuments var tikt izmantots kā pamats turpmākai AEASS attīstībai vai citām tās novirzēm, kas iekļauj AEAASS elementus.

4.1.3 Saistībā ar citiem dokumentiem

PPA ir cieši saistīts ar AEAASS PPS aprakstu. PPA ir izstrādāts pēc standarta “LVS 72:1996 Ieteicamā prakse programatūras projektējuma aprakstīšanai”.[3]

4.2 Moduļu dekompozīcija

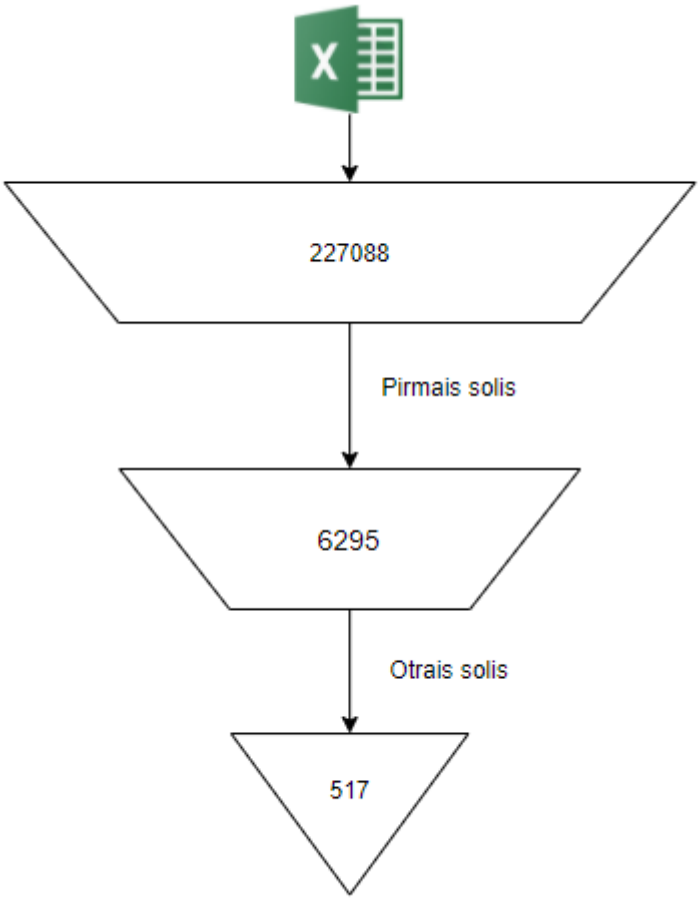
Dekompozīcijā tiek sīkāk aprakstīti sistēmā esošo moduļu funkcijas un elementi. Tabulā tiek parādītas konkrētas sistēmas funkcijas, to apraksts un iekš kura moduļa funkcija tiek izmantota. (skat. Tabula 4.1).

Tabula 4.1

Funkcija	**Apraksts**	**Teksta apstrādes modulis**	**Teksta satura identificēšanas modulis**
Content_check(string)	*Atgriež vērtības True vai False , Pārbauda vai e-pastā ir iekļauts pielikums.*	+	
sequence_check(string)	*Atgriež vērtības True vai False, pārbauda vai e-pasts ir pirmreizējs.*	+	
split_important(string)	*Atgriež simbolu virkni, apstrādā simbolu virkni, atstājot nepieciešamo informāciju.*	+	
text_clean(string)	*Atgriež simbolu virkni, apstrādā un attīra tekstu, sagatavojot to ML modeļa atpazīstamajā formā.*	+	
modelCompile(string)	*Atgriež "vārdnīcu", identificē un klasificē attīrītu simbolu virkni.*		+
send_response(string[])	*Sūta sagatavotu e-pasta atbildi klientam vai nodod e-pastu tālākai apstrādei*		+

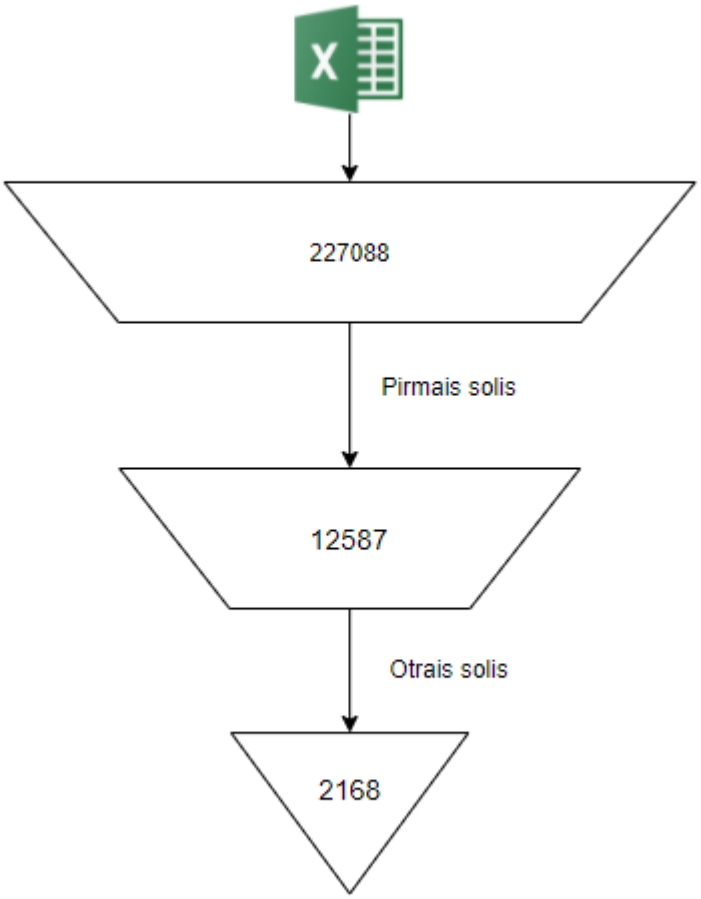
4.3 ML modeļa dekompozīcija

Dekompozīcijā tiek aprakstīti un parādīti veiktie soļi ML modeļa trenēšanā [5]. Dekompozīcijā tiek arī sīkāk aprakstīta teksta apstrādes loģika. Dotajos piemēros tiek aprakstīta klientu identificēšanās iezīmes izveide. Tiek sagatavoti pozitīvie dati (skat. Attēls 4.1) un negatīvie dati (skat. Attēls 4.2). Dati sākotnēji tiek ielasīti Microsoft Excel dokumentā no uzņēmuma datu noliktavas. Datu būtība ir izsūtītie un saņemtie e-pasti divu gadu garumā.



Attēls 4.1. **Pozitīvo datu filtrācijas diagramma**

(skat. Attēls 4.1.) Attēlā tiek parādīts pieejamo datu skaits un tā attiecība pret datu attīrīšanas veiktajiem soļiem pozitīvo datu atlasīšanā.



Attēls 4.2. **Negatīvo datu filtrācijas diagramma**

(skat. Attēls 4.2.) Attēlā tiek parādīts pieejamo datu skaits un tā attiecība pret datu attīrīšanas veiktajiem soļiem negatīvo datu atlasīšanā.

4.3.1 Datu attīrīšanas soļu apraksts

Katras diagrammās (skat. Attēls 4.1, 4.2) Solis sevī iekļauj simbolu virknes attīrīšanas elementus, izmantojot NLP metodes. (sīkāk skat. Tabulas 4.1, 4.2)

Tabula 4.1

Datu attīrīšanas soļu apraksts pozitīviem datiem

Solis	Darbības
Pirmais solis	Būtība: Tiek atlasīti klientu jautājumi uz kuriem, lai sniegtu atbildi, **IR** nepieciešams identificēties. 1. Tiek atlasīti tikai tādi dati (šajā gadījumā e-pasti), kuros **IR** minētas konkrētas frāzes, piemēram: • str.contains(‘Klienta numurs’) • str.contains(‘Personas kods’) Būtiski, šajā solī netiek izmantoti regeksi, lai paātrinātu darbību. 2. Tālāk, tiek atlasīti tikai tādi dati, kuri nāk no privātajiem klientiem, respektīvi, tiek meklētas privātas e-pastu adreses, piemēram: • Gmail.com • Inbox.lv • Yahoo.com 3. Tiek apstrādāti ne-pirmreizēji e-pasti: • Tiek meklētas sakarības starp e-pasta sarakstes elementiem, konkrētajā gadījumā, tiek meklēta simbolu virkne “--Original Message--” , kas atdala klienta pirmreizējo e-pasta tekstu.(skat. tālāk attēlu piemēros 4.3, 4.4, 4.5) • Nepieciešamās teksta daļas tiek atdalītas, šajā gadījumā: o str.split(“--Original Message--”)

Tabulas 4.1 turpinājums

Solis	**Darbības**
Otrais solis	Būtība: Dati tiek apstrādāti pēc NLP[4] metodēm, lai tos izmantotu ML modeļa trenēšanai. 1. Simbolu virknes tiek pārveidotas mazajos burtos 2. Tiek attīrītas tiešsaistes adreses un e-pastu adreses 3. Tiek attīrīti pieklājības vārdi/frāzes un sasveicināšanās vārdi/frāzes (Paldies, labrīt, uz redzēšanos, ar cieņu, u.tml.) 4. Garumzīmes un mīkstinājuma zīmes tiek nomainītas uz attiecīgo burtu bez garumzīmes vai mīkstinājuma zīmes. 5. Tiek attīrīti jebkādi simboli izņemot latīņu burtus (kirilicas simboli, @, (, /, <, u.tml.) 6. Tiek attīrītas nestandarta frāzes, kas tiek automātiski izveidotas atkarībā no ierīces vai e-pastu sistēmas. (‘sent from’, ‘message created on’, u.tml.) 7. Tiek attīrīti stopvārdi (ka ,lai, jo bet, u.tml.)[2] 8. Tiek attīrīti “vientuļie” simboli.

Tabula 4.2

Datu attīrīšanas soļu apraksts negatīviem datiem

Solis	Darbības
Pirmais solis	Būtība: Tiek atlasīti klientu jautājumi uz kuriem, lai sniegtu atbildi, **NAV** nepieciešams identificēties. 1. Tiek atlasīti tikai tādi dati (šajā gadījumā e-pasti), kuros **NAV** minētas konkrētas frāzes, piemēram: • str.contains('Klienta numurs') • str.contains('Personas kods') Būtiski, šajā solī netiek izmantoti regeksi, lai paātrinātu darbību. 2. Tālāk, tiek atlasīti tikai tādi dati, kuri nāk no privātajiem klientiem, respektīvi, tiek meklētas privātas e-pastu adreses, piemēram: • Gmail.com • Inbox.lv • Yahoo.com 3. Tiek apstrādāti ne-pirmreizēji e-pasti: • Tiek meklētas sakarības starp e-pasta sarakstes elementiem, konkrētajā gadījumā, tiek meklēta simbolu virkne "--Original Message--" , kas atdala klienta pirmreizējo e-pasta tekstu.(skat. tālāk attēlu piemēros 4.3, 4.4, 4.5) • Nepieciešamās teksta daļas tiek atdalītas, šajā gadījumā: o str.split("--Original Message--")

Tabulas 4.2 turpinājums

Solis	**Darbības**
Otrais solis	Būtība: Dati tiek apstrādāti pēc NLP metodēm[4], lai tos izmantotu ML modeļa trenēšanai. 1. Simbolu virknes tiek pārveidotas mazajos burtos 2. Tiek attīrītas tiešsaistes adreses un e-pastu adreses 3. Tiek attīrīti sasveicināšanās vārdi/frāzes (Paldies, labrīt, uz redzēšanos, ar cieņu, u.tml.) 4. Garumzīmes un mīkstinājuma zīmes tiek nomainītas uz attiecīgo burtu bez garumzīmes vai mīkstinājuma zīmes. 5. Tiek attīrīti jebkādi simboli izņemot latīņu burtus (kirilicas simboli, @, (, /, <, u.tml.) 6. Tiek attīrītas nestandarta frāzes, kas tiek automātiski izveidotas atkarībā no ierīces vai e-pastu sistēmas. (‘sent from’, ‘message created on’, u.tml.) 7. Tiek attīrīti stopvārdi (ka ,lai, jo bet, u.tml.)[2] 8. Tiek attīrīti “vientuļie” simboli.

4.3.2 Datu attīrīšanas soļu reāls vizualizējums ar piemēru **pozitīviem** datiem

Jūtīga informācija tiek aizstāta ar simbolu 'x'.

Labdien! Lūdzu atsūtiet linku paroles atjaunošanai. On 19.10.2019 18:02, Tet wrote: > > Sveiki! > > Paldies par iesūtīto vēstuli! > > Nodevām atbildīgajiem speciālistiem, pārbaudei. > > Atbildes uz jautājumiem aicinām meklēt arī mūsu mājas lapas Palīdzības > sadaļā - http://palidziba.tet.lv/lat/palidziba_majai/. > Ērtākai ikdienai aicinām ar mums sazināties čatā. > > Tavs Tet > > > > --Original Message-- > From: xxx@gmail.com > Date: 19.10.2019 13:31 > To: tet@tet.lv > Subject: Re: atbloķēt e-pastu [#234449] > > xxx xxx > xxxxx-xxxxx> Viestura laukums , valmiera > On 19 Oct 2019, 13:26 +0300, Tet <tet@tet.lv>, wrote: > > Sveiki! > > Paldies par Jūsu vēstuli! > > Lai pārbaudīt radušos situāciju ar Jūsu e-pastu, lūgums, saistībā ar > personas datu aizsardzību, norādīt : > 1. klienta numuru vai personas kodu > 2. vārds, uzvārds > 3. pakalpojuma adresi. > > Tavs Tet > > > > --Original Message-- > From: xxx@gmail.com > Date: 18.10.2019 21:08 > To: tet@tet.lv > Subject: atbloķēt e-pastu > > > Lūdzu atbloķēt e-pastu xxx@apollo.lv > Visas savas iekārtas esmu iztīrijis no vīrusiem

Attēls 4.3. **Datu attīrīšanas piemērs – oriģināla klienta sarakste**

Attēlā (skat. Attēls 4.3.) ir redzami oriģināli, neapstrādāti e-pastu dati. Šajā gadījumā šī ir sarakste starp klientu un uzņēmuma apkalpošanas un palīdzības nodaļas darbinieku.

Lūdzu atbloķēt e-pastu xxx@apollo.lv > Visas savas iekārtas esmu iztīrijis no vīrusiem

Attēls 4.4. **Datu attīrīšanas piemērs – e-pastu dati pēc pirmā soļa attīrīšanas**

Attēlā (skat. Attēls 4.4.) ir redzami attīrīti e-pasta dati pēc pirmā attīrīšanas soļa darbības principa.

atbloket pastu visas savas iekartas esmu iztirijis virusiem

Attēls 4.5. **Datu attīrīšanas piemērs – e-pastu dati pēc otrā soļa attīrīšanas**

Attēlā (skat. Attēls 4.5.) ir redzami attīrīt e-pasta dati pēc otrā attīrīšanas soļa darbības principiem.

4.3.3 Datu attīrīšanas soļu reāls vizualizējums ar piemēru **negatīviem** datiem.

Jūtīga informācija tiek aizstāta ar simbolu 'x'.

Paldies par informāciju, xxx On Mon, Oct 14, 2019 at 8:08 AM Tet <tet@tet.lv> wrote: > Sveiki! > > > Norādītajā adresē varam piedāvāt tikai 4G internetu. Savukārt internets, > ko nodrošinām ar kabeļiem, nav pieejams. > > Sīkāk par 4G internetu var lasīt šeit: > https://www.tet.lv/internets/pieslegumu-veidi/4g-internets > > > Ja piedāvājums ir saistošs, lūdzam atsūtīt kontakttālruņa numuru, lai > varam vienoties par pakalpojuma pieslēgšanu. > > > Patīkamu dienu vēlot, > Tavs Tet > > > > --Original Message-- > From: xxx@gmail.com > Date: 13.10.2019 16:59 > To: tet@tet.lv > Subject: Internets Ventspili > > > Sveiki ! > > Vēlējos noskaidrot par pakalpojumu pieejamību Ventspili. > Kāda veida interneta savienojumus jūs piedāvājat Ventspilī, Ozolu iela x? > > Ar cieņu, > xxx>

Attēls 4.6. **Datu attīrīšanas piemērs – oriģināla klienta sarakste**

Attēlā (skat. Attēls 4.6.) ir redzami oriģināli, neapstrādāti e-pastu dati. Līdzīgi kā iepriekš šī ir sarakste starp klientu un uzņēmuma apkalpošanas un palīdzības nodaļas darbinieku.

Sveiki ! > > Vēlējos noskaidrot par pakalpojumu pieejamību Ventspili. > Kāda veida interneta savienojumus jūs piedāvājat Ventspilī, Ozolu iela x ? > >

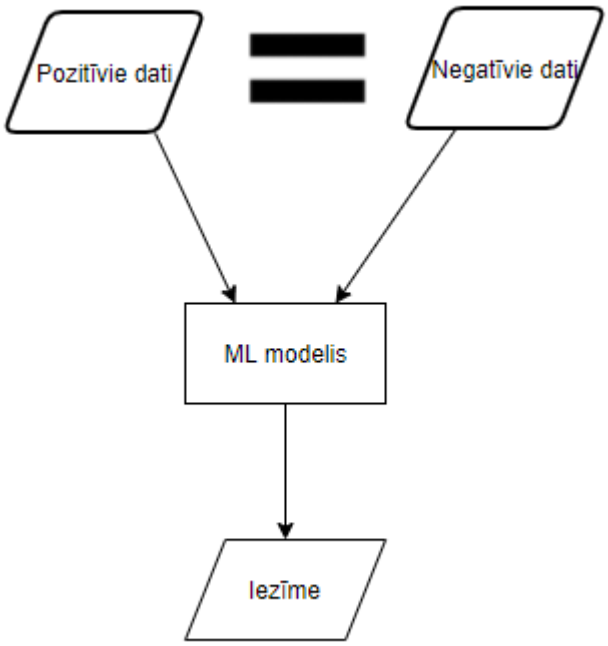
Attēls 4.7. **Datu attīrīšanas piemērs – e-pastu dati pēc pirmā soļa attīrīšanas**

Attēlā (skat. Attēls 4.7.) ir redzami attīrīti e-pasta dati pēc pirmā attīrīšanas soļa darbības principa.

velejos noskaidrot pakalpojumu piejamibu ventspili kada veida interneta savienojumus piedavajat ventspili ozolu iela

Attēls 4.8. **Datu attīrīšanas piemērs – e-pastu dati pēc otrā soļa attīrīšanas**

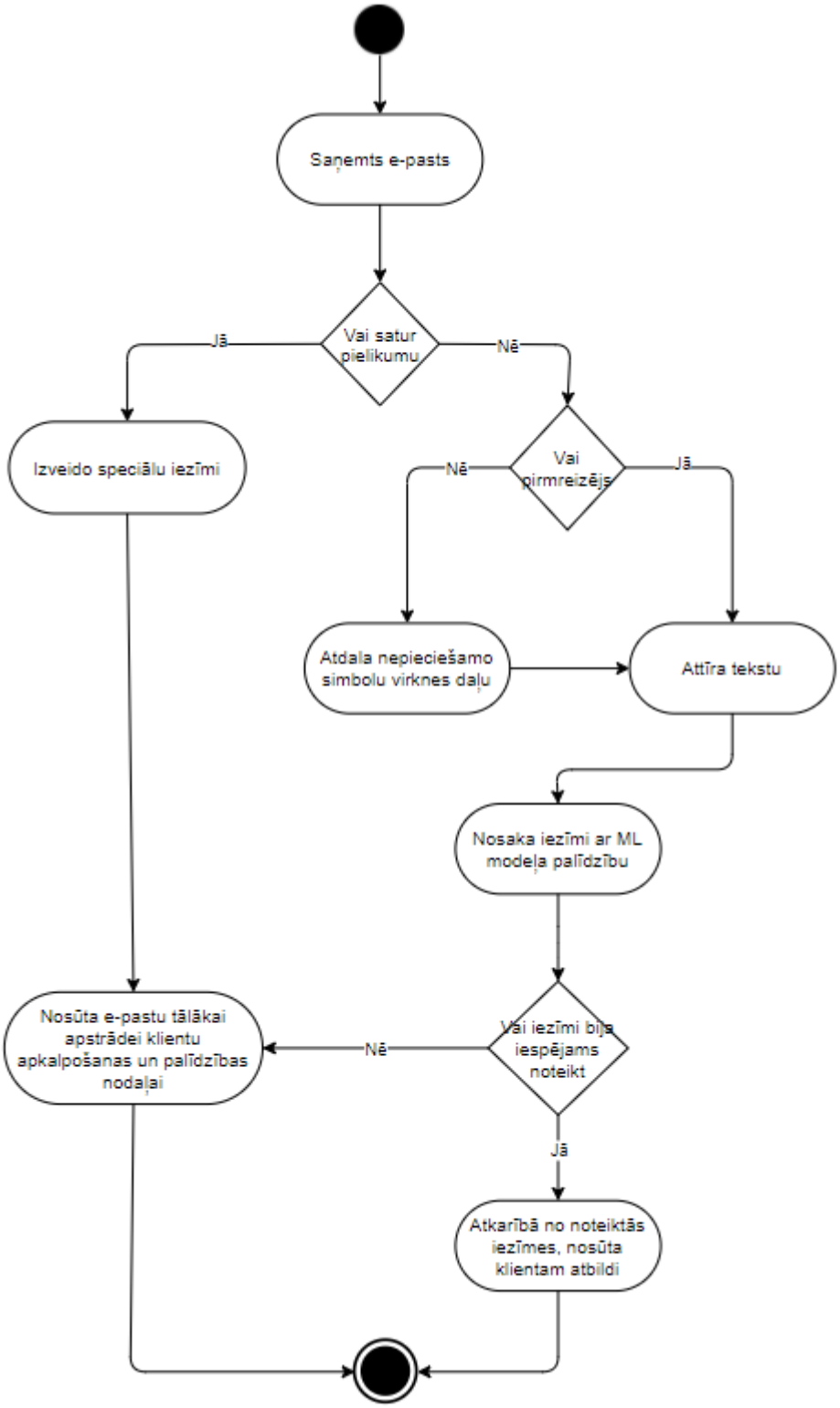
Attēlā (skat. Attēls 4.8.) ir redzami attīrīt e-pasta dati pēc otrā attīrīšanas soļa darbības principiem.



Attēls 4.9. **Triviāls ML modeļa princips**

Attēlā (skat. Attēls 4.9.) Tiek attēlots triviāls princips kā tiek trenēts ML modelis. Modelim tiek padots vienāds skaits pozitīvo datu un negatīvo datu, pēc kā tiek veidota iezīme, kas parāda vai dati pieder pie vienas kopas vai otras.

4.4 Daļējs funkciju projektējums



Attēls 4.10. **AEASS aktivitāšu diagramma**

Diagramma (skat Attēls 4.10) ir parādītas sistēmas galvenās aktivitātes.

5 TESTĒŠANA

AEAASS izstrādes laikā tika veikti testi, lai nodrošinātu sistēmas korektu darbību. Dokumentā tiek aprakstīta sistēmas teksta un e-pastu apstrādes fukncijas.

Tabula 5.1

ID	**Ievaddati**	**Sagaidāmais rezultāts**	**Reālais rezultāts**
isValidEmail	E-pasts tikai ar tekstu	Funkcija atgriež “True”	Izpildās
Apraksts	E-pasts tikai ar pielikumu	Funkcija atgriež “False”	Izpildās
Tiek pārbaudīts e-pasta saturs	E-pasts gan ar pielikumu, gan tekstu	Funkcija atgriež “False”	Izpildās

Tabula 5.2

ID	**Ievaddati**	**Sagaidāmais rezultāts**	**Reālais rezultāts**
IsFirst	E-pasts ar pielikumu	Funkcija atgriež kļūdu	Izpildās
Apraksts	Pirmreizējs e-pasts	Funkcija atgriež “True”	Izpildās
Tiek pārbaudīts vai e-pasts ir pirmreizējs	E-pasts, kas ir daļa no sarakstes ar klientu palīdzības nodaļas operatoru	Funkcija atgriež “False”	Izpildās

Tabula 5.3

ID	**Ievaddati**	**Sagaidāmais rezultāts**	**Reālais rezultāts**
sliceText	Teksts ar sarakstes elementiem	Funkcija atgriež sarakstu ar atrastajiem elementiem, kas liecina par saraksti	Izpildās
Apraksts	Pirmreizējs teksts	Funkcija atgriež tukšu sarakstu	Izpildās
Tiek pārbaudīts vai teksts tiek korekti attīrīts, ja tā ir sarakste			

Tabula 5.4

ID	**Ievaddati**	**Sagaidāmais rezultāts**	**Reālais rezultāts**
cleanText	Simbolu virkne, kas satur tikai dažādus simbolus ($,*,@, u.tml.)	Funkcija atgriež tukšu simbolu virkni	Izpildās
Apraksts	Simbolu virkne, kas satur burtus ar mīkstinajuma un garumzīmēm	Funkcija atgriež simbolu virkni bez mīkstinājuma vai garumzīmēm	Izpildās
Pārbauda vai teksts tiek korekti attīrīts	Simbolu virkne, kas satur ne-latīnu valodas burtus	Funkcija atgriež tukšu simbolu virkni	Izplidās

Tabula 5.5

ID	**Ievaddati**	**Sagaidāmais rezultāts**	**Reālais rezultāts**
isClassified	Attīrīta simbolu virkne	Funkcija atrgriež “True”	Izpildās
Apraksts	Neattīrīta simbolu virkne	Funkcija atgriež “True” (pat ja iezīme nav faktiski korekta)	Izpildās
Pārbauda vai ML modelis korekti piešķir iezīmi	Tukša simbolu virkne	Funkcija atgriež kļūdu	Izplidās

Tabula 5.6

ID	**Ievaddati**	**Sagaidāmais rezultāts**	**Reālais rezultāts**
doRespond	Simbolu virkne ar iezīmi “1”	Funkcija atrgriež “1” iezīmes tekstu	Izpildās
Apraksts	Dimbolu virkne ar iezīmi “0”	Funkcija atgriež “2” iezīmes tekstu	Izpildās
Pārbauda, vai pēc iezīmes tiek noteikts korekts atbildes teksts			

6 PROJEKTA ORGANIZĀCIJA

Projektu autors veica pilnīgi patstāvīgi, projekta ietvaros autoram bija nepieciešams apgūt sekojošas tehniskas prasmes: programmēšanas valodas python apgūšana, NLP principu apgūšana, ML un neironu tīklu pamatprincipu apgūšana.

Šis ir unikāls un pirmreizējs projekts uzņēmumā, kas tiks turpmāk attīstīts ar autora palīdzību. Lai gan uzņēmums strādā pēc SCRUM metodes, autors darbu veica brīvā stilā, neievērojot konkrētas darba metodes, jo skaitījās kā attālināti piesaistītais darba spēks. Autors pats uzņēmās atbildību veikt komunikāciju ar citām uzņēmuma nodaļām, lai saprastu to vajadzības un intereses projekta ietvaros, autors piedalījās projekta saistītajās tikšanās un konferencēs. Kā galvenais konsultants autoram bija uzņēmuma Lielo Datu apstrādes nodaļas grupas vadītājs.

Uzņēmums patur tiesības par informācijas konfidencialitāti, un autors ir aicināts šīs tiesības ievērot. Uzņēmums atļaujas atklāt savu faktisko nosaukumu, bet autors ir aicināts publicēt un demonstrēt tikai atsevišķas sistēmas pirmkoda daļas.

7 KONFIGURĀCIJAS PĀRVALDĪBA

Sistēma tiek iešūta SMTP servera konfigurācijā, programmkoda rakstīšanai tika izmantota atvērtā pirmkoda programmatūra “Jupyter Notebook”. Tika veikta vietēja versijas kontrole, GIT sistēmā.

8 DARBIETILPĪBAS NOVĒRTĒJUMS

Darbietilpība tiek novērtēta balstoties uz “QSM Benchmark Tables Business Systems Implementation Unit (New and Modified IU) Benchmarks” novērtējumu tabulu. Projekts prasīja ļoti daudz plānošanas, par cik autors ko šādu izstrādajis pirmo reizi. Gandrīz viens cilvēkmēnesis tika veltīts projekta plānošanai un komunikācijai. Pirmkoda rindiņu skaits, neskaitot papildus pielāgoto pirmkodu iešūšanai SMTP serverī sastāda 802 koda rindiņas. Jāņem vērā, ka tika izmantota programmēšanas valoda Python un gatavās funkciju bibliotēkas.

Balstoties uz tabulas pirmā kvartīļa informāciju, kas apraksta mazo projektu darbietilpību, ņemot vērā mazāku koda rindiņa skaitu, bet lielāku implementāciju vienību skaitu, tiek secināts, ka projekta izstrādes novērtējums pēc tabulas atbilst 2.5 – 3 personmēnešiem.

9 SECINĀJUMI UN AUTORA KOMENTĀRI

Projekta risinājuma gaitā autors guva unikālas zināšanas gan par uzņēmuma iekšējām darbības struktūrām un principiem, gan par projekta saistītajām tehniskajām lietām. Autors apguva patstāvīgas prasmes projektu izstrādāšanā un problēmu risināšanā. Apgūtās zināšanas ļaus autoram turpināt AEAASS pilnveidošanu un uzturēšanu uzņēmuma interesēs, kā arī ļaus piedalīties līdzīgos izstrādes projektos vai veikt patstāvīgu sistēmas izstrādi.

10 IZMANTOTĀ LITERATŪRA UN AVOTI

1. LVS 68:1996 “Programmatūras prasību specifikācijas ceļvedis”
2. “Intrneta agresivitātes indekss” [Tiešsaiste]. Pieejams: http://barometrs.korpuss.lv/?from=2013-10-13&to=2014-10-13&site=lv&section=stopwords
3. Latvijas Nacionālais standartizācijas un metroloģijas centrs, Latvijas standarts LVS 72:1996, Ieteicamā prakse programmatūras projektējuma aprakstīšanai
4. “Your Guide to Natural Language Processing (NLP)”[Tiešsaiste]. Pieejams: https://towardsdatascience.com/your-guide-to-natural-language-processing-nlp-48ea2511f6e1
5. “LSTM Layer” [Tiešsaiste]. Pieejams: https://keras.io/api/layers/recurrent_layers/lstm/

PIELIKUMI

Teksta apstrādes un ML modeļa trenēšanas pirmkods

```
import csv
import sys
Import os
import re
import enchant
import nltk
import nltk.corpus
from nltk.tokenize import word_tokenize
import numpy as np
import pandas as pd
from pandas import ExcelWriter
from pandas import ExcelFile

pd.options.mode.chained_assignment = None
maxInt = sys.maxsize
maxInt = int(maxInt/10000000000)
csv.field_size_limit(maxInt)

d1 = pd.read_csv(r'E:\email_documents\email_files\10-440-400.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d2 = pd.read_csv(r'E:\email_documents\email_files\11-400-360.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d3 = pd.read_csv(r'E:\email_documents\email_files\12-360-320.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d4 = pd.read_csv(r'E:\email_documents\email_files\13-320-280.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d5 = pd.read_csv(r'E:\email_documents\email_files\14-280-240.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d6 = pd.read_csv(r'E:\email_documents\email_files\15-240-200.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d7 = pd.read_csv(r'E:\email_documents\email_files\16-200-150.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d8 = pd.read_csv(r'E:\email_documents\email_files\17-150-100.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d9 = pd.read_csv(r'E:\email_documents\email_files\18-100-50.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d10 = pd.read_csv(r'E:\email_documents\email_files\19-50-0-END.csv',
sep=';', header=None, engine='python', encoding="utf-8-sig",
error_bad_lines=False, usecols=[17,24])
d11 = pd.read_csv(r'E:\email_documents\email_files\3-720-700.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d12 = pd.read_csv(r'E:\email_documents\email_files\4-700-660.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
```

```
d13 = pd.read_csv(r'E:\email_documents\email_files\5-660-620.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d14 = pd.read_csv(r'E:\email_documents\email_files\6-620-580.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d15 = pd.read_csv(r'E:\email_documents\email_files\7-580-530.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d16 = pd.read_csv(r'E:\email_documents\email_files\8-530-480.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])
d17 = pd.read_csv(r'E:\email_documents\email_files\9-480-440.csv', sep=';',
header=None, engine='python', encoding="utf-8-sig", error_bad_lines=False,
usecols=[17,24])


df=pd.concat([d1,d2,d3,d4,d5,d6,d7,d8,d9,d10,d11,d12,d13,d14,d15,d16,d17])
df.dropna(subset = [17,24], inplace=True)


#Drop invalid rows (in email column - drop invalid emails that do not
contain @)
df = df[df[17].str.contains('@')]
df=df[~df[17].str.contains('tetveikals')]

#reset column and row indexes
df.reset_index(inplace=True, drop=True)
df.columns = range(df.shape[1])


#First part of cleaning
#This part finds tet questions from templates phrases in dataframe

#initialize set of template questions or phrases
template_list = ['klienta numuru vai personas kodu', 'klienta numuru vai >
personas kodu',
'nepieciešami klienta dati', 'personas kodu vai klienta numuru', 'norādīt
klienta numuru', 'personas kodu un/vai klienta numuru',
'klienta numuru vai  personas kodu', 'klienta līguma numuru', 'precizējiet
personas kodu', 'personas kodu un/vai klienta > numuru',
'abonenta personas kodu','precizēt klienta datus, līguma numuru', 'klienta
> līguma numuru', 'klienta līguma  numuru', 'klienta > līguma numuru']
df = df[df[1].str.contains('|'.join(template_list), case =True)]

#Leave only most popular private emails
private_list = ['gmail.com', 'inbox.lv', 'mail.ru', 'bk.ru', 'apollo.lv',
'hotmail.com', 'zb.lv', 'yahoo.com']
df = df[df[1].str.contains('|'.join(private_list), case=True)]

#Select emails only from clients
l = ['tet','lattelecom']
df = df[~df[0].str.contains('|'.join(l), case = True)]
df = df[~df[1].str.contains('Tet E-veikalā')]
df = df[~df[1].str.contains('Tet veikals')]
df = df[~df[1].str.contains('lattelecomveikals')]
df = df[~df[1].str.contains('Lai mēs varētu sameklēt un novirzīt maksājuma
uzdevumā norādīto naudas summu')]
df = df[~df[1].str.contains('nepieciešams atsūtīt ārsta apstiprinātu
izziņu')]
df = df[~df[1].str.contains('nav pielikuma')]
```

```
df.drop_duplicates(subset=0, keep='first', inplace=True)


#Filter out client questions
dfclean = df[1].str.rsplit('--Original Message--', 1).str[1]
dfclean= dfclean.to_frame()

#Prepare for uneccessary data cleaning from questions
dfclean[1] = dfclean
[1].str.replace('>\s>\s>\s>\s>\s>\s>\s>\s>\s>\sLabdien!\s>\s>\s>\s>\s>',' -
-split-- ')
dfclean[1] = dfclean [1].str.replace('\s\s\s\s\s\s\sLabdien,\s\s\s\s\s',' -
-split-- ')
dfclean[1] = dfclean [1].str.replace('Labdien.\s\s\s\s',' --split-- ')
dfclean[1] = dfclean [1].str.replace('\s\s\s\s\s\s\sLabvakar,\s\s\s\s\s','
--split-- ')
dfclean[1] = dfclean [1].str.replace('\s\s\s\sLabdien,\s\s\s',' --split--
')
dfclean[1] = dfclean [1].str.replace('\s\s\s\s\sLabdien,',' --split-- ')
dfclean[1] = dfclean [1].str.replace('\s\s\s\sLabdien!\s\s\s',' --split--
')
dfclean[1] = dfclean [1].str.replace('\s\s\sLabdien.',' --split-- ')
dfclean[1] = dfclean [1].str.replace('\s\s\s\s\s\s\s\s',' --split-- ')
dfclean[1] = dfclean [1].str.replace('\s\s\s\s\s\s\s',' --split-- ')
dfclean[1] = dfclean [1].str.replace('\s\s\s\s\s\s',' --split-- ')
dfclean[1] = dfclean [1].str.replace('Message:',' --split-- ')
dfclean[1] = dfclean [1].str.replace('Paldies\.',' --split-- ')
dfclean[1] = dfclean [1].str.replace('<>',' --split-- ')
dfclean[1] = dfclean [1].str.replace('>\s>\s>',' --split-- ')
dfclean[1] = dfclean [1].str.replace('>\s\s>\s\s>',' --split-- ')
dfclean[1] = dfclean [1].str.replace('>\s\s>\s\s\s>',' --split-- ')
dfclean[1] = dfclean [1].str.replace('>\s\s\s>\s\s\s>',' --split-- ')
dfclean[1] = dfclean [1].str.replace('>\s>\s>\s\s\s\s\s\s\s\s\s\s',' --split-
- ')
dfclean[1] = dfclean [1].str.replace('Ar cieņu.*','')
dfclean[1] = dfclean [1].str.replace('Cieņā.*','')
dfclean[1] = dfclean [1].str.replace('On.*','')
dfclean[1] = dfclean [1].str.replace('.*Jautājums:\s>','')
dfclean[1] = dfclean [1].str.replace('.*Jautājums:','')
dfclean[1] = dfclean [1].str.replace('rēķins nr',' ')
dfclean[1] = dfclean [1].str.replace('klienta nr',' ')
dfclean[1] = dfclean [1].str.replace('--split--/s*--split--','--split--')


#Second part data cleaning
dfclean = dfclean[1].str.split('--split--').str[1]
dfclean = dfclean.to_frame()

final_clean= dfclean
nan_value = float("NaN")
final_clean[1].replace("", nan_value, inplace = True)
final_clean[1].replace(" ", nan_value, inplace = True)
final_clean[1].replace("  ", nan_value, inplace = True)
final_clean[1].replace("   ", nan_value, inplace = True)
final_clean.dropna(subset = [1], inplace = True)

#remove emails
final_clean[1] = final_clean[1].str.replace('\S*@\S*\s?', '')
#remove links
final_clean[1] = final_clean[1].str.replace('http\S+', '')
```

```
final_clean[1] = final_clean[1].str.replace('\w*inbox', '')

#All to lower
final_clean[1] = final_clean[1].str.lower()

#remove all kinds of symbols
final_clean[1] =
final_clean[1].str.replace('ā','a').replace('à','a').replace('ä',
'a').replace('á','a').replace('â', 'a').replace('ã','a').replace('å','a')
final_clean[1] = final_clean[1].str.replace('ē', 'e').replace('é',
'e').replace('è','e').replace('ê','e').replace('ë','e')
final_clean[1] = final_clean[1].str.replace('ś','s')
final_clean[1] = final_clean[1].str.replace('š','s')
final_clean[1] = final_clean[1].str.replace('ī','i').replace('ì',
'i').replace('í', 'i').replace('î','i').replace('ï','i')
final_clean[1] = final_clean[1].str.replace('ū', 'u')
final_clean[1] = final_clean[1].str.replace('č', 'c')
final_clean[1] = final_clean[1].str.replace('ņ', 'n').replace('ñ','n')
final_clean[1] = final_clean[1].str.replace('ķ', 'k')
final_clean[1] = final_clean[1].str.replace('ģ', 'g')
final_clean[1] = final_clean[1].str.replace('Í', 'l')
final_clean[1] = final_clean[1].str.replace('ļ', 'l')
final_clean[1] = final_clean[1].str.replace('ž', 'z')
final_clean[1] = final_clean[1].str.replace('ù', 'u').replace('ú',
'u').replace('ù','ù').replace('û','u').replace('ü','u')
final_clean[1] = final_clean[1].str.replace('😁', '')
final_clean[1] = final_clean[1].str.replace('😩', '')
final_clean[1] = final_clean[1].str.replace('ee', 'e')
final_clean[1] = final_clean[1].str.replace('aa', 'a')
final_clean[1] = final_clean[1].str.replace('ii', 'i')
final_clean[1] = final_clean[1].str.replace('uu', 'u')
final_clean[1] = final_clean[1].str.replace('ee', 'e')
final_clean[1] = final_clean[1].str.replace('sh', 's')
final_clean[1] = final_clean[1].str.replace('lj', 'l')
final_clean[1] = final_clean[1].str.replace(' cee', ' ')
final_clean[1] = final_clean[1].str.replace(' mb ', ' ')
final_clean[1] = final_clean[1].str.replace(' uz ', ' ')
final_clean[1] = final_clean[1].str.replace(' no ', ' ')
final_clean[1] = final_clean[1].str.replace(' ne ', ' ')
final_clean[1] = final_clean[1].str.replace(' ja ', ' ')
final_clean[1] = final_clean[1].str.replace(' ko ', ' ')
final_clean[1] = final_clean[1].str.replace(' ap ', ' ')
final_clean[1] = final_clean[1].str.replace(' ir ', ' ')
final_clean[1] = final_clean[1].str.replace(' un ', ' ')
final_clean[1] = final_clean[1].str.replace(' ka ', ' ')
final_clean[1] = final_clean[1].str.replace(' bet ', ' ')
final_clean[1] = final_clean[1].str.replace(' ar ', ' ')
final_clean[1] = final_clean[1].str.replace(' par ', ' ')
final_clean[1] = final_clean[1].str.replace(' es ', ' ')
final_clean[1] = final_clean[1].str.replace(' nu ', ' ')
final_clean[1] = final_clean[1].str.replace(' tas ', ' ')
final_clean[1] = final_clean[1].str.replace(' man ', ' ')
final_clean[1] = final_clean[1].str.replace(' to ', ' ')
final_clean[1] = final_clean[1].str.replace(' kas ', ' ')
final_clean[1] = final_clean[1].str.replace(' tad ', ' ')
final_clean[1] = final_clean[1].str.replace(' ko ', ' ')
final_clean[1] = final_clean[1].str.replace(' vai ', ' ')
final_clean[1] = final_clean[1].str.replace(' ta ', ' ')
final_clean[1] = final_clean[1].str.replace(' ari ', ' ')
final_clean[1] = final_clean[1].str.replace(' tu ', ' ')
final_clean[1] = final_clean[1].str.replace('gan', ' ')
```

```
final_clean[1] = final_clean[1].str.replace('tikai', ' ')
final_clean[1] = final_clean[1].str.replace('bija', ' ')
final_clean[1] = final_clean[1].str.replace(' lai ', ' ')
final_clean[1] = final_clean[1].str.replace(' tur ', ' ')
final_clean[1] = final_clean[1].str.replace('date', ' ')
final_clean[1] = final_clean[1].str.replace(' mon ', ' ')
final_clean[1] = final_clean[1].str.replace(' tue ', ' ')
final_clean[1] = final_clean[1].str.replace(' thu ', ' ')
final_clean[1] = final_clean[1].str.replace(' wed ', ' ')
final_clean[1] = final_clean[1].str.replace(' fri ', ' ')
final_clean[1] = final_clean[1].str.replace(' sat ', ' ')
final_clean[1] = final_clean[1].str.replace(' sun ', ' ')
final_clean[1] = final_clean[1].str.replace(' jan ', ' ')
final_clean[1] = final_clean[1].str.replace(' feb ', ' ')
final_clean[1] = final_clean[1].str.replace(' mar ', ' ')
final_clean[1] = final_clean[1].str.replace(' apr ', ' ')
final_clean[1] = final_clean[1].str.replace(' may ', ' ')
final_clean[1] = final_clean[1].str.replace(' jun ', ' ')
final_clean[1] = final_clean[1].str.replace(' jul ', ' ')
final_clean[1] = final_clean[1].str.replace(' aug ', ' ')
final_clean[1] = final_clean[1].str.replace(' sep ', ' ')
final_clean[1] = final_clean[1].str.replace(' okt ', ' ')
final_clean[1] = final_clean[1].str.replace(' nov ', ' ')
final_clean[1] = final_clean[1].str.replace(' dec ', ' ')
final_clean[1] = final_clean[1].str.replace('re:', ' ')
final_clean[1] = final_clean[1].str.replace('re ', ' ')
final_clean[1] = final_clean[1].str.replace('fwd:', ' ')
final_clean[1] = final_clean[1].str.replace('fw', ' ')
final_clean[1] = final_clean[1].str.replace('vtv', ' ')
final_clean[1] = final_clean[1].str.replace('p.k.', ' ')
final_clean[1] = final_clean[1].str.replace(' ot ', ' ')
final_clean[1] = final_clean[1].str.replace('vod', ' ')
final_clean[1] = final_clean[1].str.replace(' pl ', ' ')
final_clean[1] = final_clean[1].str.replace('!', ' ')
final_clean[1] = final_clean[1].str.replace(',', ' ')
final_clean[1] = final_clean[1].str.replace('.', ' ')
final_clean[1] = final_clean[1].str.replace('?', ' ')
final_clean[1] = final_clean[1].str.replace('/', ' ')
final_clean[1] = final_clean[1].str.replace('(', ' ')
final_clean[1] = final_clean[1].str.replace(')', ' ')
final_clean[1] = final_clean[1].str.replace('[', ' ')
final_clean[1] = final_clean[1].str.replace(']', ' ')
final_clean[1] = final_clean[1].str.replace('-', ' ')
final_clean[1] = final_clean[1].str.replace('-', ' ')
final_clean[1] = final_clean[1].str.replace('+', ' ')
final_clean[1] = final_clean[1].str.replace('/', ' ')
final_clean[1] = final_clean[1].str.replace('\\', ' ')
final_clean[1] = final_clean[1].str.replace("'", ' ')
final_clean[1] = final_clean[1].str.replace('|', ' ')
final_clean[1] = final_clean[1].str.replace('<', ' ')
final_clean[1] = final_clean[1].str.replace('>', ' ')
final_clean[1] = final_clean[1].str.replace('=', ' ')
final_clean[1] = final_clean[1].str.replace('%', ' ')
final_clean[1] = final_clean[1].str.replace('#', ' ')
final_clean[1] = final_clean[1].str.replace('$', ' ')
final_clean[1] = final_clean[1].str.replace('^', ' ')
final_clean[1] = final_clean[1].str.replace('&', ' ')
final_clean[1] = final_clean[1].str.replace('_', ' ')
final_clean[1] = final_clean[1].str.replace('@', ' ')
final_clean[1] = final_clean[1].str.replace('*', ' ')
final_clean[1] = final_clean[1].str.replace(';', ' ')
```

```
final_clean[1] = final_clean[1].str.replace(':', ' ')
final_clean[1] = final_clean[1].str.replace('"', ' ')
final_clean[1] = final_clean[1].str.replace('€', ' ')
final_clean[1] = final_clean[1].str.replace(' lv', ' ')
final_clean[1] = final_clean[1].str.replace('lv ', ' ')
final_clean[1] = final_clean[1].str.replace(' lv ', ' ')
final_clean[1] = final_clean[1].str.replace('labdien', ' ')
final_clean[1] = final_clean[1].str.replace('labvakar', ' ')
final_clean[1] = final_clean[1].str.replace('sveiki', ' ')
final_clean[1] = final_clean[1].str.replace('ludzu', ' ')
final_clean[1] = final_clean[1].str.replace('paldies', ' ')

#remove more stuff
final_clean[1] = final_clean[1].str.replace('iphone', ' ')
final_clean[1] = final_clean[1].str.replace('twitpd', ' ')
final_clean[1] = final_clean[1].str.replace('from', ' ')
final_clean[1] = final_clean[1].str.replace('reply message', ' ')
final_clean[1] = final_clean[1].str.replace('lattelecom', '  ')
#final_clean[1] = final_clean[1].str.replace('tet', ' ')
final_clean[1] = final_clean[1].str.replace('sent from my iphone', ' ')
final_clean[1] = final_clean[1].str.replace('no subject', ' ')
final_clean[1] = final_clean[1].str.replace('from tet sent friday may pm
to', ' ')
final_clean[1] = final_clean[1].str.replace('tavs bezmaksas pasts inbox', '
')
final_clean[1] = final_clean[1].str.replace('uga records media
wwwugarecordscom', ' ')


#remove non-latin letters
final_clean[1] = final_clean[1].str.replace('[^\x00-\x7F]+', '')

#remove numbers and single chars and whitespaces
final_clean[1] = final_clean[1].str.replace('[0-9]', '')
final_clean[1] = final_clean[1].str.replace(' +', ' ')
#exception case
final_clean[1] = final_clean[1].str.replace('sent mail.*', ' ')
final_clean[1] = final_clean[1].str.replace('sent my.*', ' ')
final_clean[1] = final_clean[1].str.replace('image.*', ' ')
final_clean[1] = final_clean[1].str.replace('ar cienu.+', ' ')
final_clean[1] = final_clean[1].str.replace('reply to mess.+', ' ')
final_clean[1] = final_clean[1].str.replace('sent from.+', ' ')
final_clean[1] = final_clean[1].str.replace('am deply.*', ' ')
final_clean[1] = final_clean[1].str.replace('sincerely.+', ' ')
final_clean[1] = final_clean[1].str.replace('forwarded.+', ' ')
final_clean[1] = final_clean[1].str.replace(' lpojum', ' pakalpojum')
final_clean[1] = final_clean[1].str.replace(' elotu', ' novelotu')
final_clean[1] = final_clean[1].str.replace(' lpojuma', ' pakalpojuma')
final_clean[1] = final_clean[1].str.replace(' lpojumam', ' pakalpojumam')
final_clean[1] = final_clean[1].str.replace(' lpojumi', ' pakalpojumi')
final_clean[1] = final_clean[1].str.replace(' lpojumiem', ' pakalpojumiem')
final_clean[1] = final_clean[1].str.replace(' lpojumus', ' pakalpojumus')
final_clean[1] = final_clean[1].str.replace(' lpojumu', ' pakalpojumu')
final_clean[1] = final_clean[1].str.replace(' lpojums', ' pakalpojums')

#remove template text
final_clean[1] = final_clean[1].str.replace('^jautajums.+jautajums', ' ')


#remove more numbers and whitespaces
final_clean[1] = final_clean[1].str.replace('^\s+|\s+$', '')
```

```
final_clean[1] = final_clean[1].str.replace('\s\s+', '')

#remove www
final_clean[1] = final_clean[1].str.replace('\bwww.*', '')
final_clean[1] = final_clean[1].str.replace('\swww.*', '')

#remove lone characters
final_clean[1] = final_clean[1].str.replace('(^| ).( |$)', ' ')


#remove empty rows (again after cleaning)
final_clean[1].replace('', np.nan, inplace=True) #replace empty strings
witn NaN
final_clean.dropna(subset=[1], inplace=True)
#remove rows where word count is small
count = final_clean[1].str.split().str.len()
~(count == 1)
final_clean = final_clean[~(count == 1)]
final_clean = final_clean[~(count == 2)]
final_clean = final_clean[~(count == 3)]
final_clean = final_clean[~(count == 4)]
final_clean = final_clean[~(count == 5)]


#Set max word count for data
max_size = 25
final_clean[1] =
final_clean[1].str.split(n=max_size).str[:max_size].str.join(' ')


# # ML PART

######### typ2 ##############################################
from sklearn.feature_extraction.text import CountVectorizer
from sklearn import metrics
vect = CountVectorizer()
from tensorflow.keras.preprocessing import sequence
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Embedding
from tensorflow.keras.layers import LSTM
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.preprocessing.text import Tokenizer
from sklearn.metrics import accuracy_score
from tensorflow.keras.preprocessing.sequence import pad_sequences
############################################################
import warnings
warnings.filterwarnings('ignore')

#Load labeled dataframe
df = pd.read_excel(r'E:\email_documents\email\labeled_data.xlsx')
test_df = pd.read_excel(r'E:\email_documents\email\test.xlsx')
df['text'] = df['text'].astype('str')
test_df['text'] = test_df['text'].astype('str')


#Model data
X_train = df['text']
X_test = test_df['text']
y_train = df.label
y_test = test_df.label
tokenize = Tokenizer()
```

```
tokenize.fit_on_texts(X_train.values)

#tokenize data
X_train = tokenize.texts_to_sequences(X_train)
X_test = tokenize.texts_to_sequences(X_test)

#make tokenized data into sequence
max_lenght = max([len(s.split()) for s in df['text']]) #length of input
sequences to Embedding layer
X_train = pad_sequences(X_train, max_lenght)
X_test = pad_sequences(X_test, max_lenght)

#size of the output vectors (bigger num => more params)
EMBEDDING_DIM = 128

#vocabulary size
unknown = len(tokenize.word_index)+1

model = Sequential()
model.add(Embedding(unknown, EMBEDDING_DIM, input_length=max_lenght))
model.add(LSTM(units=10, dropout=0.1, recurrent_dropout=0.1 ))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy', optimizer='adam',
metrics=['accuracy'])
print(model.summary())
model.fit(np.asarray(X_train), np.asarray(y_train), batch_size=64,
epochs=8, verbose=1)

final_pred = model.predict_classes(X_test) #predict_proba
print(accuracy_score(y_test, final_pred))
y_pred_df = pd.DataFrame(final_pred)
y_pred_df.columns = ['prediction']
final_test = pd.concat([test_df,y_pred_df], axis=1)
final_test.to_excel(r'E:\email_documents\email\result_deep_learning.xlsx')

max_length = 25

a=b = ['man ir neapmaksats rekins nakamnedel apmaksasu']

a = tokenize.texts_to_sequences(a)
a = pad_sequences(a, max_length)
print(model.predict_classes(a))
```

Kvalifikācijas darbs „ ***Automātiska e-pastu apstrādes un atbildes sniegšanas sistēma***” izstrādāts Latvijas Universitātes Datorikas fakultātē.

Ar savu parakstu apliecinu, ka darbs izstrādāts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: ***Edvards Bukovskis*** _____ **09.01.2021**.

Rekomendēju darbu aizstāvēšanai
Darba vadītāj**s/a**: ***Prof. Jānis Zuters*** _____ **___.01.2021.**

Recenzents: ***Baltcom SIA, struktūrvienības vadītājs, Lauris Raipulis***

Darbs iesniegts 11.01.2021.
Kvalifikācijas darbu pārbaudījumu komisijas sekretāre: *Darja Solodovņikova*_____

Darbs aizstāvēts kvalifikācijas darbu pārbaudījuma komisijas sēdē
___.01.2021. prot. Nr. _____

Komisijas sekretārs(-e): _____