Rīgas 64. Vidusskola

**Kāda Konkrēta YouTube Video Komentāru Sadaļas Sentimenta Nolasīšana: *VADER* un *RoBERTa* Modeļu Salīdzinājums**

Zinātniski pētnieciskais darbs: Datorzinātņu un datorlingvistikas sadaļā

**Darba autors:**

Rīgas 64. vidusskolas 12. klases skolnieks

Kristers Laganovskis

**Darba vadītājs:**

Rīgas 64. vidusskolas programmēšanas skolotājs

Edvards Bukovskis

Rīga 2023

**Anotācija**

Zinātniski pētniecisko darbu: “Sentimenta novērtēšana YouTube komentāru sadaļai” Izstrādājis autors Rīgas 64. vidusskolas 12.DIT klases skolēns – Kristers Laganovskis. Darba Vadītājs Rīgas 64. vidusskolas programmēšanas skolotājs - Edvards Bukovskis.

Zinātniski pētnieciskais darbs tika veikts ar mērķi, izpētīt, kā ar dabiskās valodas apstrādi (NLP) iegūst sentimenta analīzi. Kādas ir būtiskākās atšķirības starp *VADER* un *RoBERTa* modeļiem. Kā, tiek veikta efektīva datu ievākšana. Kā, tiek apstrādāti dati. Kā arī, cik precīzi ir iespējams novērtēt komentāru sadaļu balsoties uz mākslīgo intelektu un programmēšanas palīdzību. Zinātniski pētnieciskajā darbā balstoties uz analizēt un aplūkoto, tika izveidots pārlūka paplašinājums, ar kura palīdzību iespējams analizēt sentimentu kādam konkrētam YouTube video komentāru sadaļai.

Teorētiskajā daļa tika detalizēti izpētīti dabiskās valodas apstrādes (NLP) pamati, teksta sentimenta analizēšanai. Tās pielietošana sociālajā un ētiskajā kontekstā, kā arī tās nozīme mūsdienu un vēsturiskajā kontekstā. Tika apskatīts, kas ir *VADER* modelis un kas ir *RoBERTa* modelis un to atšķirības. Kā, tiek veikta efektīva datu ievākšana un tā principi. Kā arī datu apstrādes principi, lai nodrošinātu pēc iespējas precīzāku sentimenta novērtējumu. Kā arī tika apsvērti sentimenta analīzes ierobežojumi.

Praktiskā daļā tika izstrādāts pārlūka paplašinājums, kas veidots uz VEDER un *RoBERTa* modeļu bāzes, šis paplašinājums sniedz iespēju vizualizēt sentimentu statistiski par kāda konkrēta YouTube video komentāru sadaļu, sniedzot vērtīgu perspektīvu par to, kāda ir publikas reakcija uz konkrēto saturu video. Paplašinājums nodrošina iespēju efektīvi filtrēt komentārus pēc to sentimenta, padarot analīzes procesu vēl efektīvāku. Taču šāda veida analīzei kļūdas ir tik lielas, ka visticamāk tam ir neliela praktiska pielietojuma vērtība. Iespējamie metodes uzlabojumi ietver nevēlamo reklāmu komentāru filtrēšanu, sarkasma filtrēšānu un “emoji” sentimenta iekļaušanu.

Atslēgas Vārdi: Datu ievākšana, datu apstrāde, Komentāri, sentimenta analīze, pārlūka paplašinājums, dabiskās valodas apstrāde (NLP), mašīnmācīšanās, *VADER*, *RoBERTa*.

**Abstract**

Scientific Research Work: "Sentiment Analysis of YouTube Comment Section" Developed by the author, a student of the 12th-grade DIT class at Riga Secondary School No. 64 – Kristers Laganovskis. Supervisor: Edvards Bukovskis, programming teacher at Riga Secondary School No. 64.

The scientific research work was conducted to explore how sentiment analysis is obtained through Natural Language Processing (NLP). It delves into the significant differences between the *VADER* and *RoBERTa* models, as well as how to efficiently gather data and process it. Additionally, it evaluates the precision of assessing the comment section using artificial intelligence and programming assistance. Based on the analysis and examination of the scientific research work, a browser extension was developed to analyze the sentiment of specific YouTube video comment sections.

In the theoretical part, the fundamentals of Natural Language Processing (NLP) for text sentiment analysis were thoroughly studied, including its application in social and ethical contexts, as well as its significance in modern and historical contexts. The *VADER* model and the *RoBERTa* model, along with their differences, were examined. The effective principles of data collection and processing were also discussed to ensure the most accurate sentiment assessment possible. The limitations of sentiment analysis were also considered.

In the practical part, a browser extension was developed based on the *VADER* and *RoBERTa* models. This extension provides the opportunity to statistically visualize the sentiment of a specific YouTube video comment section, offering valuable insight into the audience's reaction to the specific content of the video. The extension allows for the efficient filtering of comments by their sentiment, making the analysis process even more effective. However, the errors in this type of analysis are likely so significant that it has little practical application value. Possible method improvements include filtering out unwanted advertising comments and incorporating “emoji” sentiment.

Keywords: Data collection, data processing, comments, sentiment analysis, browser extension, Natural Language Processing (NLP), machine learning, *VADER*, *RoBERTa*.

**Saturs**

**Anotācija**…......................................................................................................................................2

**Abstract**...........................................................................................................................................3

**Ievads**…………………………………………………………………………………………………………………………………..…..5

**1. Literatūras apskats**…………………………………………………………………..…………………..……………………….6

1.1 Dabiskās Valodas Apstrādes (NLP) Pamati………………………………………………………………….………..6

1.2 Sentimenta nolasīšanas principi (NLP)…………………………………………………………………………………7

1.2.1 Datu iegūšana……………………………………………………………………………………………………………7

1.2.2 Datu apstrāde……………………………………………………………………………………………………………7

1.2.2.1 Datu apstrādes metodes……………………………………………………………………………….8,9

1.2.3 Sentimenta nolasīšana………………………………………………………………………………………….9,10

1.3 *VADER* un *RoBERTa* modeļu salīdzināšana………………………………………………………………………….10

1.4 Datu ievākšana no YouTube video komentāru sadaļas……………………………………………………11,12

**2. Praktiskā daļa**……………………………………………………………………………………………………………………….13

2.1 VADER un RoBERTa modeļu salīdzināšana praktiski…………………………………………………………….13

2.2 Datu kopas izvēle………………………………………………………………………………………………………………..14

2.3 VADER modeļa testēšana………………………………………………………………………………………………..….14

**Secinājumi**………………………………………………………………………………………………………………….……………..0

**Izmantotie informācijas avoti**…………………………………………………………………………………………………….0

**Pielikumi**……………………………………………………………………………………………………………………………………0

**Ievads**

Mūsdienās informācijas plūsma ir pārāk bagātīga, un dažkārt tas var radīt problēmu gan saturu patērētājam, gan to veidotājiem. Būtisks šķērslis ir tas, ka cilvēki bieži vien vērš pārāk lielu uzticību satura veidotājiem, kas, kā rāda pieredze, var nebūt vienmēr objektīvi. Šāda pieeja rada risku, ka satura veidotājs varētu apslēpt vai pat manipulēt ar informāciju, ietekmējot patērētājus. Tādēļ ir ārkārtīgi svarīgi uzzināt, kā citi cilvēki novērtē konkrētā video saturu. Īpaši šo var novērot interneta platformā "YouTube", kur satura veidotāji bieži veicina savu subjektīvo viedokli, ko patērētāji bieži uztver kā objektīvu faktisko informāciju. Tomēr tas var radīt lielas problēmas, jo ne vienmēr šim veidotājam būs pareiza nostāja. Tāpēc ir svarīgi uzzināt, ko citi cilvēki domā par konkrēto viedokli, vai saturu, un veikt pašam savus secinājumus. Veikt secinājumus, pārskatot pāris komentārus, ir ātri un ērti, bet, kad to ir simtiem vai tūkstošiem, šāda analīze prasa ievērojami vairāk laika. Tāpēc sentimenta analīze ir lietderīga, jo tā ļauj strukturēt komentāru sadaļu nostājās - pozitīvas, neitrālas vai negatīvas.

Šajā pētījumā tiks apskatīts, kāda YouTube video komentāru sadaļa, izmantojot valodas apstrādes (NLP) modeļus, spēj nolasīt komentāru noskaņojumu, izmantojot *VADER* un *RoBERTa* modeļus. Tiks analizētas metodes, kas tiek izmantotas, lai iegūtu nepieciešamos datus analīzei, kā arī kā šie dati ir jāapstrādā, lai sasniegtu veiksmīgu sentimenta novērtējumu. Turklāt tiks izvērtēta mākslīgā intelekta spēja precīzi noteikt komentāru noskaņojumu. Papildus tam, tiks izveidots paplašinājums, kas ļaus ikvienam analizēt komentārus un tos filtrēt pēc noteiktām kategorijām. Tas būs noderīgs ne tikai satura patērētājiem, bet arī veidotājiem, ļaujot tiem ietaupīt laiku, filtrējot komentārus.

**Mērķis:** Izpētīt dabiskās valodas (NLP) principus kā arī salīdzināt VADER un RoBERTa modeļus un pēc teorijas izpētes izveidot paplašinājumu, kas ļauj analizēt komentāru sadaļu kādam konkrētam *YouTube* video.

**Hipotēze:** Autoram izdosies veiksmīgi salīdzināt *VADER* un *RoBERTa* modeļus kā arī analizēt un interpretēt YouTube video komentārus, izmantojot (NLP) modeļus un programmēšanu.

**Darba uzdevumi:**

1. Izpētīt (NLP) modeli un kā to pielietot sentimenta novērtēšanai.
2. Salīdzināt VADER un RoBERTa sentimenta modeļus.
3. Izpētīt, kā iegūt nepieciešamos datus no YouTube video komentāru sadaļas.
4. Izpētīt, kā apstrādāt datus efektīvi, lai tie sniegtu precīzāku novērtējumu.
5. Izstrādāt paplašinājumu, kas ļauj ikvienam analizēt un filtrēt komentārus pēc noteiktām kategorijām.
6. Veikt secinājumus

**Izmantotās darba metodes:** Literatūras apskats, lai varētu izpētīt, Dabiskās valodas apstrādes (NLP). Salīdzināšanas metode, lai varētu salīdzināt *VADER* un *RoBERTa* (NLP) modeļus. Sentimenta nolasīšanas metode, lai interpretētu iegūtos datus. Statistikas un analīzes metodes. Atvērtā pirmkoda pielietošana, *Python* *(3.8)* programmēšanas valodas pielietošana radošās daļas veikšanai.

**Darba struktūra:** Darbs sastāv no x nodaļām, x apakšnodaļām, secinājumiem, izmantoto literatūras avotu saraksta un x pielikumiem. Darbā ir x attēli.

**Literatūras apskats**

* 1. **Dabiskās Valodas Apstrādes (NLP) Pamati**

Dabiskās Valodas Apstrāde (angliski: natural language processing, turpmāk — NLP) ir mākslīgā intelekta joma, kas nodarbojas ar datoru spēju saprast, analizēt un interpretēt cilvēku valodu. Tās sākotnējie pētījumi sākās jau 1950. un 1960. gadu mijā, kad tika veidoti pirmie mēģinājumi izveidot datorprogrammas, kas spētu analizēt un interpretēt tekstu. Šajā laikā tika izveidots pirmās valodas analīzes algoritms, kas varēja identificēt vārdus, frāzes un sintaktiskos elementus. [1] Kopš tā laika NLP ir piedzīvojis ievērojamu attīstību un progresu, pateicoties tehnoloģiju un algoritmu uzlabojumiem. Ar NLP mūsdienu pasaulē mēs saskaramies teju katru dienu, jo to pielieto plaši ikdienas ērtībās ko ikkatrs lietojam. To izmanto balss vadības GPS sistēmas, digitālajos asistentos, runas-uz-tekstu dikcijas programmās, tulkošanas programmās, klientu apkalpošanas “čatbotos” un citās patērētāju ērtībās. [2]

Tomēr NLP arvien lielāku lomu ieņem uzņēmējdarbības risinājumos, kas palīdz optimizēt uzņēmējdarbības darbības, palielina darbinieku produktivitāti un vienkāršo būtiskus uzņēmējdarbības procesus. [2] Arvien biežāk NLP tiek izmantota arī medicīnas jomā, lai analizētu medicīnisko dokumentāciju un izgūtu noderīgu informāciju ārstiem un pētniekiem. [3]

Sentimenta nolasīšana, jeb analīze ir viena no svarīgākajām NLP apakšnozarēm. Tā ļauj noteikt teksta vai runas emocionālo noskaņu, palīdzot izsekot un analizēt viedokļus un emocijas. Sentimenta nolasīšana piedāvā plašas pielietošanas iespējas, tostarp sociālajos medijos, uzņēmējdarbībā, mārketingā, pētniecībā un citviet. [4]

Sentimenta analīze tiek veikta, izmantojot dator algoritmus, kas analizē vārdus, frāzes, un teikumus, lai noteiktu to emocionālo saturu. Šie algoritmi izmanto vārdnīcas un emociju leksikonus, lai klasificētu vārdus un frāzes polaritātes: pozitīvs, negatīvs vai neitrāls. Tas ļauj kopēji novērtēt teksta noskaņu un viedokļus. [4]

Sentimenta analīze ir īpaši noderīga, lai novērtētu un klasificētu lielus teksta apjomus, piemēram, sociālo mediju ierakstus, produktu atsauksmes vai klientu aptaujas. Tā palīdz uzņēmumiem iegūt ieskatu par to, kā viņu produkti vai pakalpojumi tiek uztverti tirgū, un sniedz iespēju uzlabot darbību, balstoties uz klientu atsauksmēm un viedokļiem. [5]

NLP ir kļuvusi par neatņemamu mūsdienu tehnoloģiju daļu, kas iesaistīta gan ikdienas patērētājus, gan uzņēmējdarbības risinājumus. Tā piedāvā iespēju saprast un analizēt cilvēku valodu, atvieglojot daudzas ikdienas darbības un uzlabojot uzņēmējdarbības procesus. Svarīga apakšnozare - sentimenta nolasīšana - kas palīdz novērtēt teksta emocionālo saturu un piedāvā plašas pielietošanas iespējas dažādās nozarēs. Tas veicina uzņēmumu izaugsmi, sniedzot iespēju labot un pielāgot piedāvātos produktus un pakalpojumus, balstoties uz klientu atsauksmēm un viedokļiem.

**1.2 Sentimenta nolasīšanas principi**

Kā jau iepriekš tika minēts, sentimenta nolasīšana ir viena no svarīgākajām NLP apakšnozarēm. Tāpēc šajā sadaļā tiks dziļāk apskatītas sentimenta analīzes pamatprincipi un tehnikas. Sentimenta nolasīšanas mērķis ir saprast un klasificēt tekstu vai runu, lai noteiktu tā emocionālo noskaņojumu. Sentimentu var klasificēt pēc polaritātes – pozitīvi, neitrāli un negatīvi, vai pēc vajadzības vēl mazākos iedalījumos. Sentimenta nolasīšanai, ir jāveic trīs būtiski soļi. Pirmais solis ir iegūt datus, otrais ir apstrādāt datus un trešais ir veikt datu analīzi jeb sentimenta nolasīšanu. [6]

**1.2.1 Datu ievākšana**

Šis ir pirmais solis sentimenta nolasīšanā. Tas ietver iegūto datu kopumu, kuram tiks veikta sentimenta analīze. Šos datus var iegūt no ziņām, komentāriem, sociālo mediju ierakstiem, klientu atsauksmes, klientu apkalpošanas centra datiem, darbinieku mijiedarbības datiem un citu veidu datiem, lai veiktu sentimenta nolasīšanu no tiem. [7] Datu ievākšanu var veikt ar dažādām metodēm. Viena no populārākajiem veidiem ir lietot *API (Lietojumprogrammas saskarne)*, ko nodrošina sociālo mediju platformas, kas ļauj savākt, datus straumējot. Piemēram: *Twitter* *API*, lai iegūtu ziņas ar noteiktām “heštegu” atsauksmēm, *News API*, lai izgūtu ziņas pēc kategorijām no dažādiem ziņu izdevējiem. Otrs populārs veids ir izmanot *web scraping (rasmošana)*, kas pārmeklē tīmekļa datus un savāc norādīto informāciju. Piemēram var izmantot *Python* *(3.8)* bibliotēku *BeautifulSoup*, lai iegūtu jebkādu informāciju no tīmekļa, piemēram, ziņu raksti vai komentāri no blogiem, analizējot *HTML* tagus. Datus var ievākt arī izmantojot tīmekļa pārlūka paplašinājumus, ar kura palīdzību lietotājs var izgūt informāciju no jeb kuras publiskas tīmekļa vietnes un tad eksportējot datus vēlamā faila formātā. Piemēram: *Webscraper.io*. Kā arī datus var, ievākt no esošiem datu krājumiem, kas jau ir apstrādāti un ir gatavi lietošanai. Piemēram: *Rotten Tomatoes*, *IMDB* filmu apskats, *Yelp*, *Amazon* produkta apskats, *Twitter* tvīti un citas. [8]



1. attēls: Rasmošanas pamatprincipa shēma [28])

****

1. attēls: API pamatprincipa shēma [30]

**1.2.2 Datu apstrāde**

Datu apstrāde: Otrais svarīgais solis sentimenta noteikšanā ir iegūto datu apstrāde. Datu apstrāde ir kritisks solis sentimenta nolasīšanā, jo labi apstrādāti dati, veido pamatu efektīvai sentimenta klasifikācijas modeļa izveidei. Teksta apstrāde ir process, kurā tiek iepriekš iegūtie dati attīrīti un apstrādāti. Teksta jeb datu apstrādē ietilpst procesi kā - pieturzīmju izņemšana, bieži sastopamu vārdu noņemšana *(stop word removal),* visu tekstu pārveidojot, mazajos burtos *(lowercasing)*, lemmatizācija, tokenizācija *(tokenization)*, noliegumu apstrāde, intensitāšu apstrāde, “*emojie”* apstrāde, vektorizācija *(vectorization)* un retu vai zemu frekvenci vārdu apstrāde. [9] Datu zinātniekiem ir teiciens *“garbage in, garbage out”*, kas tulkojas kā *“atkritumi iekšā, atkritumi ārā.”* Šī teiciena nozīme ir uzsvērt to, ka datu apstrāde ir ļoti svarīgs process sentimenta noteikšanā, jo gala rezultāts būs atkarīgs no tā cik kvalitatīvi dati tiks padoti analīzes modelim. [10]

**1.2.2.1 Datu apstrādes metodes**

Kā, jau autors minēja iepriekš ir daudz un dažādas metodes, kā apstrādāt tekstu jeb datus. Taču, katra metode pilda noteiktu funkciju datu apstrādes procesā un ne vienmēr būs būtiski izmantot visas metodes kopā. Tāpēc tagad tiks izskatītas, pāris galvenās un nozīmīgākās metodes atsevišķi.

1. **Metode visa teksta pārveidošana mazajos burtos** ***(lowercasing)***: Viss teksts tiek pārveidots mazajos burtos, lai nodrošinātu vienmērību un novērstu iespējamu vārdu dublēšanos ar atšķirīgiem lielajiem un mazajiem burtiem. Piemēram: vārdi “Labs” un “labs” tiks uzskatīti par vienu un to pašu vārdu. [9] Šo metodi izmanto, lai ietaupītu resursus, jo tas samazina vārdu skaitu, kas ir jāapstrādā. Kā arī strādāt, ar viena formāta vārdiem ir vieglāk, tas palīdz uzlabot precizitāti un vienkāršo algoritma darbu. Šo metodi izmanto, ja ir daudz vārdu, kas dublējas, taču nav vienā formātā un šo vārdu nozīme nemainās, kad tiek izmantoti lielie burti, piemēram, sociālajos mēdijos. Šo metodi var veikt ar *Python* *(3.8)* programmēšanas valodu.[11]
2. **Metode tokenizācija *(tokenization)*:** Viss teksts tiek sadalīts individuālos vārdos, kurus sauc par *tokeniem*, šis ir būtisks process datu apstrādē, jo tas ļauj standartizēt ievades datus modelī, padarot tos vieglāk apstrādājamus, lai analizētu teksta sentimentu. Jo bez tokenizācijas *(tokenization)* metodes sentimenta noteikšanas modelim būtu jāstrādā ar visu tekstu kā vienu virkni ar burtiem, kas prasītu lielus resursus un padarītu datus sarežģīti modelējamus. Šo metodi var veikt ar *Python (3.8)* bibliotēkas *nltk* palīdzību. [12] Piemēram: Teikums no ievāktajiem datiem “Šodien ir saulaina diena.”, tiktu pārveidots pēc tokenizācijas *(tokenization)* uz “[‘Šodien’, ‘ir’, ‘saulaina’, ‘diena’, ‘.’]”, taču, ja nebūtu piemērota toeknizācija *(tokenization)* un tiktu padoti ievades dati modelim bez tās, tad teksts izskatītos šādi - “Šodienirsaulinadiena.” Šo metodi izmanto vienmēr, kad veic datu apstrādi sentimenta noteikšanai.
3. **Metode atbrīvošanās no visām pieturzīmēm:** No visa teksta tiek izņemtas ārā visas pieturzīmes un tagi. Tas iekļauj – komatus, punktus, izsaukuma zīmes un citas zīmes. Pieturzīmes nenes lielu informācijas vērtību sentimenta nolasīšanas modelim un tās var tikt noņemtas, lai samazinātu tekstā troksni. [11]
4. **Metode bieži sastopamu vārdu noņemšana *(stop word removal)*:** Bieži sastopami vārdi, piemēram, "un", "bet", "ir", utt. Tiek noņemti, jo tie nenes lielu nozīmi, lai noteiktu sentimentu un pat var negatīvi ietekmēt sentimenta rezultātu. Šo metodi izmanto vienmēr, kad veic datu apstrādi sentimenta noteikšanai. [11]
5. **Metode lemmatizācija un pamatvārda noteikšana *(Lemmatization and Stemming)*:** Lemmatizācija attiecas uz vārda pamatformas noteikšanu (parasti tas ir vārds, ko var atrast vārdnīcā). Piemēram, vārda 'ābols' lema joprojām būtu 'ābols', bet vārda 'ir' lema būtu 'būt'. Lemmatizācija ir pamats katrai sentimenta nolasīšanas procesam. Tā ņem vērā dažādās esošās valodas struktūras, un katrā valodā lematizācija var būt atšķirīga. Pamatvārda noteikšana, jeb *stemming* ir process, kurā vārdi tiek samazināti līdz to pamatformai, atgriežot tos vispārīgāko veidā.

Piemēram, vārda "mājas" pamatvārds būtu "māj", un vārda "gāju" pamatvārds būtu "gāj". Pamatvārda noteikšana ir vienkāršāka un mazāk resursu prasīga nekā lematizācija, bet var būt mazāk precīza. Galvenais atšķirības punkts ir tas, ka lematizācija cenšas saglabāt vārda nozīmi, samazinot to līdz pamatformai, savukārt pamatvārda noteikšana vienkārši mēģina atgriezt vārdu vispārīgākajā formā. [14]

A close up of a graph

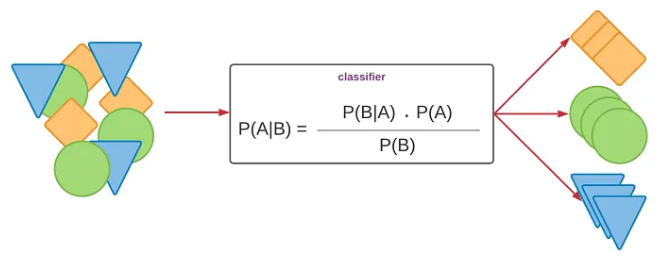
Description automatically generated

1. attēls: Pamatvārda noteikšanas piemērs un lemmatizācijas piemērs shematiski [29])
2. **Metode negāciju apstrāde un intensifikatori *(handaling negations and intemsifiers)*:** Noliegumu apstrāde ir metode, kas nosaka nolieguma ietekmes apjomu uz viedokļa vārdiem, ko ietekmē noliegums. Noliegumi ir vārdi kā 'nav', 'ne', 'nekad', utt., kas ietekmē citu vārdu noskaņu frāzē. Noliegums var pilnībā mainīt citu vārdu noskaņu vienvirziena teikumā. Ir trīs noliegumu kategorijas: sintaktiskie, samazinātāji un morfoloģiskie noliegumi. [15] Intensifikatori *(intemsifiers),* piemēram “ļoti” vai “ārkārtīgi”, maina vārda nozīmi. Intensifikatori *(intemsifiers)* ir apstākļa vārdi, kas pastiprina citu izteicienu nozīmi. Par pastiprinātājiem parasti tiek lietoti tādi vārdi kā absolūti, pilnīgi, ārkārtīgi, ļoti. Sentimenta analīzē noliegumu un intensifikatoru apstrāde ir būtiska, jo tā var mainīt teikuma noskaņu. [16]

Šis ir tās metodes, kuras ir jāņem vērā, apstrādājot iegūtos datus. Lai gan, protams, ir daudzas citas metodes, kas īsumā tika pieminētas iepriekš, šīm izceltajām metodēm ir būtiska nozīme, lai nodrošinātu sentimenta nolasīšanu precīzāk un efektīvāk. Atkarībā no konkrētā uzdevuma prasībām optimālu rezultātu sasniegšanai var izmantot šo metožu kombināciju. Svarīgi atcerēties, ka metodes izvēlei ir jāsaskan ar konkrēto sentimenta noteikšanas mērķi.

**1.2.3 Sentimenta nolasīšana**

Sentimenta nolasīšana ļauj noteikt emocionālo noskaņu, sajūtu, vai tā būtu pozitīva, negatīva vai neitrāla. Lai veiktu sentimenta noteikšanu ar mašīnmācīšanās palīdzību, tiek izmantoti dažādi algoritmi un tehniskie risinājum, piemēram, *Navie Bayes* algoritms. Iepriekš apstrādātie dati, tiek padoti algoritmiem, kas analizē datus un veic sentimenta nolasīšanu. Šie algoritmi darbojas, apstrādājot tekstu, un analizējot tā sastāvdaļas, lai saprastu, kāda ir tā emocionālā noskaņa. Katrā algoritmā ir savas stiprās un vājās puses. [16.] Kopumā ir trīs veida pieejas kā noteikt sentimentu: **noteikumiem balstīta *(rule-based)*, automātiskā *(automatic)* un hibrīds *(hybrid)****.* [17]



1. attēls: *Navie Bayes* klasifikatora algoritms, to pielieto ne tikai sentimenta noteikšanā, bet gan arī citur. Klasificēt datus var arī izmantojot dator mācīšanās ceļu, taču tā precizitāte pārsvarā ir tikai par 0,05% lielākā nekā izmantojot *Navie Bayes* varbūtības algoritmu, kas ir daudz ātrāks par mašīnmācīšanos, lai iegūtu tos pašus rezultātus. [31])

**Noteikumiem balstīta pieeja *(rule-based)*:** Šis ir praktisks veids, kā analizēt tekstu bez nepieciešamības apmācīt vai izmantot mašīnmācīšanās modeļus. Šī pieejas rezultātā tiek izstrādāti noteikumi, pēc kuriem teksts tiek iezīmēts kā pozitīvs/negatīvs/neitrāls. Šie noteikumi ir pazīstami arī kā leksikoni. Tāpēc šo pieeju sauc par leksikonu balstītu pieeju. Plaši izmantoti noteikumu balstīti risinājumi ir *TextBlob* modelis, *VADER* modelis, kā arī *SentiWordNet* modelis. [18]

Automātiskā pieeja *(automatic)*: Atšķirībā no noteikumu balstītās metodes automātiskā metode nenosaka manuāli izstrādātus noteikumus, bet gan izmanto mašīnmācīšanās tehnikas. Emocionālās analīzes uzdevums parasti tiek modelēts kā klasifikācijas problēma. Klasifikators saņem tekstu un nosaka kategoriju, piemēram, pozitīvu, negatīvu vai neitrālu. Piemēram, RoBERTa ir viens no populārākajiem modeļiem, kas izmanto automātisko pieeju. Kā arī polārākie algoritmi automātiskajai pieejai ir *Navie Bayes*, *Support Vector Machines (SVM)* un citi [19]

**Hibrīds *(hybrid)* pieeja:** Hibrīda pieeja apvieno abu pieeju veidu, gan noteikumu balstītu, gan automātisko tehniku, integrējot tos vienā pieejā. Viena no nozīmīgākajām priekšrocībām izmantojot šo pieeju, ir tas, ka tā spēj sniegt precīzākus rezultātus. [19]

Sentimenta precizitāti mēra ar dažādām metodēm kā piemēram: *F1-Score, Confusion Matrix, recall* un citām metodēm. Pieejamie sentimenta noteikšanas veidi sniedz dažādas iespējas, kā veikt sentimenta nolasīšanu. Piemērotāko pieeju var izvēlēties pēc projekta nosacījumiem un datu kvalitātes.

**1.3 *VADER* un *RoBERTa* modeļu salīdzināšana**

NLP Sentimenta nolasīšanas nozarē ir divi ievērojami modeļi – *VADER* un *RoBERTa*. *VADER* jeb *(Valence Aware Dictionary for Sentiment Reasoning)* ir sentimenta lasīšanas modelis kā arī algoritms, taču algoritms nav publiski nekur minēts. Modelis ir balstīts uz noteikumu pieeju *(rule-based approach)*. To ieviesa pētnieki C. J. Hutto un Ēriks Gilberts savā darbā - *"VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text"* tulkojumā *"VADER: Taupīgs noteikumu balstīts modelis sociālo mediju tekstu sentimenta analīzei",* kas tika publicēts 2014. gadā. Modelis ir daļa no *Python (3.8)* bibliotēkas *nltk.sentiment* un tika speciāli radīts, lai strādātu ar sociālo tīklu tekstiem un tās īpašībām piemēram, *“emoji”*, slengu un neformālās valodas lietojumu. *VADER* iedala sentimentu polaritātes – negatīvs, pozitīvs un neitrāls. *VADER* darbības principi ietver vārdu kopu izmantošanu un to sentimenta vērtību analīzi, lai aprēķinātu kopējo teksta sentimentu. Turklāt modelis ņem vērā specifiskas īpašības, piemēram, negācijas un intensitāti, lai precīzi interpretētu tekstu. Tas ļauj *VADER* modelim veiksmīgi tikt galā ar sarežģītākiem teikumiem, saglabājot precizitāti. [25]

RoBERTa *(Robustly Optimized BERT Pre-Training Approach)* 2018. gadā *Google* izstrādāja atvērtā koda mašīnmācīšanās NLP modeli – *BERT (Bidirectional Encoder Representations from Transformers)*, taču šim modelim bija pāris ierobežojumi, tāpēc 2019. gadā *Facebook* komanda izstrādāja modificētu *BERT* modeli ar nosaukumu – *RoBERTa*. Šim modelim mērķis ir uzlabot oriģinālo *BERT* modeli, paplašinot modeli tā, lai labāk izmantotu transformatoru arhitektūru. *RoBERTa* izmanto automātisko pieeju apmācībā *(automatic approach).* Tādējādi tiek radīta labāk izteikta un noturīgāka valodas reprezentācija. Ir pierādīts, ka ar *RoBERTa* modeli ir iespējams sasniegt izcilus rezultātus dažādos valodas apstrādes uzdevumos. Šis modelis tika trenēts, izmantojot plašu teksta datu kopu no vairākām valodām, kas ļauj tai saprast un ģenerēt tekstu dažādās valodās. *RoBERTa* ir viena no vadošajām NLP modeļu arhitektūrām, kas ir panākusi lielus sasniegumus dažādos valodas apstrādes uzdevumos. [26]

Salīdzinot *VADER* un *RoBERTa* var novērot, ka abiem modeļiem ir dažādas pieejas sentimenta nolasīšanai. *VADER* izmanto tradicionālu leksikona vērtējuma pieeju, kas balstīta uz noteikumiem *(rule-based approach*), un tas ir efektīvs, īpaši, strādājot ar sociālo tīklu tekstu un neformālu valodu. Savukārt *RoBERTa* ir jaunāka un uzlabota pieeja, kas balstās uz transformatoru arhitektūru. Tas nodrošina labāk izteiktu un noturīgāku valodas reprezentāciju, kas sniedz precīzākus rezultātus dažādos valodas apstrādes uzdevumos. Turklāt *RoBERTa* modelis ir apmācīts ar plašu teksta datu kopu no vairākām valodām, ļaujot tai saprast tekstu dažādās valodās. Tā kā *RoBERTa* ir jaunāka un pilnveidotāka modeļa versija, tam var būt priekšrocības, it īpaši, strādājot ar sarežģītiem un daudzveidīgiem valodas apstrādes uzdevumiem. [27]

**1.4 Datu ievākšana no YouTube video komentāru sadaļas**

Datu iegūšana no kāda konkrēta *YouTube* video komentāru sadaļas ir svarīga, jo tā ļauj iegūt informāciju par sabiedrisko viedokli un reakcijām uz konkrētu video saturu. Tas var būt noderīgi, lai izprastu, kā cilvēki uztver, vai reaģē uz noteiktu tēmu vai notikumu, kā arī lai analizētu populārus viedokļus un tendences, kas izplatās sociālajā vidē. Pašlaik *YouTube* platformu lieto vairāk nekā 2 miljardi lietotāju katru mēnesi. Tāpēc *YouTube* video komentāros esošo informāciju var izmantot, lai veidotu mašīnmācīšanās modeļus, veiktu lietotāju sentimenta nolasīšanu, radītu lielas datu kopas un veiktu datu analīzes. Tas piedāvā plašas iespējas gan akadēmiskajos pētījumos, gan komerciālajā jomā. No juridiskās perspektīvas datu ievākšana, piemēram, komentāri un citi atribūti, ir pilnīgi likumīgs process, ja tiek izmantots *YouTube data API*. [20]

Vispopulārākais veids, kā iegūt datus no kāda konkrēta, video ir, izmantojot *YouTube data API v3*, ko piedāvā *Google*. *API* ir saīsinājums no angļu valodas *"Application Programming Interface"* jeb *"Pielietojuma programmēšanas saskarne"*. Tas ir programmēšanas protokols vai interfeiss, kas ļauj vienai programmai (vai daļai programmatūras) komunicēt ar citu programmu vai komponenti. API nosaka, kādā veidā programmatūras komponentes vai sistēmas var mijiedarboties un kādas darbības tās var veikt. [21] YouTube arī ir savs API, kuru var izmantot par velti un ar kura palīdzību ir iespējams iegūt dažādu informāciju no konkrēta *YouTube* video komentāru sadaļas. Kā, piemēram, ir iespējams iegūt komentārus, komentāru *“patīk”* balsu skaitu, komentāru atbildes, komentētāju lietotāj vārdus un citus datus par komentāru sadaļu. [22] Šo *API* var, izmanot ar *Python (3.8)* un tās piedāvātajām bibliotēkām, un datus izvadīt *JSON* failā.

Taču ir arī citas metodes, piemēram, rasmošana *(web scraping)*. Rasmošana ir process, kurā programmētājs izmanto programmatūru, lai automātiski iegūtu informāciju no interneta lapām. Tas ietver HTML kodu analīzi un datu iegūšanu no tīmekļa vietnēm. Rasmošana var būt noderīga gadījumos, kad API nav pieejams vai neattiecas uz konkrēto informāciju, ko vēlaties iegūt. [23] Tomēr ir svarīgi atzīmēt, ka rasmošana var pārkāpt YouTube lietošanas noteikumus specifiskos gadījumos. Ja tiek apskatīta konkrēta YouTube video komentāru sadaļa, tad nav optimāli izmantot rasmošanas metodi, jo datu iegūšana var, prasīs ilgu laiku. Tas prasa ilgu laiku tāpēc, ka šai programmai ir jāapstrādā katrs komentārs atsevišķi. Taču, ja izvēlas lietot šo metodi, tad šo var pilnveidot ar *Python (3.8)* bibliotēku *selenium* un tad datus ievākt *JSON* failā. [24]

1. **Praktiskā daļa**

Autors praktisko daļu sadalīja divās daļās. Vienā daļā autors salīdzina un pielieto VADER un RoBERTa modeļus, taču otrajā daļa autors izveido *Google Chrome* paplašinājumu, kas ļauj klasificēt kāda konkrēta *YouTube* video komentāru sadaļu trīs kategorijās – pozitīvi, neitrāli un negatīvi.

* 1. **VADER un RoBERTa modeļu salīdzināšana praktiski**

Kā jau iepriekš teorijas daļā autors apskatīja divus sentimenta nolasīšanas modeļus VADER un RoBERTa. Praktiskajā daļā autors veica eksperimentus, kas ļāva salīdzināt šos modeļus pēc noteiktām mērsistēmām - ***accuracy, recall, precision, F1-Score un confusion Matrix.*** Abus modeļus autors salīdzināja ar trīs dažādu apjomu datu kopām - 10 komentāru, 100 komentāru un 3534 komentāru datu kopu.

Lai salīdzinātu VADER un RoBERTa modeļus ir svarīgi izprast, ar kādām mērsistēmām autors veicis mērījumus, kā arī izprast, ko katra mērsistēma dara jeb mēra. Precizitāte jeb ***accuracy***tā nosaukums jau izsaka mērsistēmas būtību. Precizitāte mēra modeļa pareizo prognožu procentuālo daļu. To aprēķina, dalot pareizo prognožu skaitu ar kopējo modeļa veikto prognožu skaitu. Tā formula: Precizitāte = (pareizo prognožu skaits / kopējais prognožu skaits) \* 100. Piemērs: Ja modelis pareizi, paredz 80 no 100 paraugiem, tā precizitāte ir 80%. [32]

Atgriezeniskā izsaukšana jeb ***recall***,kas pazīstama arī kā jutīgums, mēra patieso pozitīvo prognožu īpatsvaru starp visiem faktiskajiem pozitīvajiem paraugiem datu kopā. To aprēķina, dalot patieso pozitīvo rezultātu skaitu ar patieso pozitīvo un nepatieso negatīvo rezultātu summu. Tā formula: Jūtīgums = patiesie pozitīvie rādītāji / (patiesie pozitīvie rādītāji + viltus negatīvie rādītāji). Piemēram: Ja teikums ir, pozitīvs un to modelis nolasa kā pozitīvu, tad to uzskata par patiesi pozitīvu teikumu, taču, ja modelis to nolasa par negatīvu, tad tas ir viltus negatīvs teikums. [32]

Precīzums jeb ***precision*** izklausās līdzīgi taču būtība mazliet mainās. Precizīzums fokusējas uz pozitīvajām prognozēm, ko veicis modelis. Tā ir attiecība starp patiesi pozitīvajām prognozēm un kopējo pozitīvo prognožu skaitu. Tā formula: Precīzums = patiesie pozitīvie rezultāti / (patiesie pozitīvie rezultāti + viltus pozitīvie rezultāti). Piemēram, ka mēs analizējam sentimentu par filmām un modelis prognozē, ka 20 no 30 apskatītajām filmām ir pozitīvs noskaņojums. No šiem 20, modelis precīzi identificē 15 filmas ar pozitīvu noskaņojumu. [32]

F1-rādītājs jeb ***F1-Score*** ir vidējais balancs starp precizitāti (precision) un jūtību (recall) un to izmanto, lai līdzsvarotu šīs divas mērsistēmas. Šī mērsistēma tiek izmantota situācijās, kad ir nepieciešams novērtēt modeļa veiktspēju, ņemot vērā gan precizitātes, gan jūtības nozīmi. Tā formula: F1-rādītājs = 2 \* ((precizitāte \* atsaukšana) / (precizitāte + atsaukšana)).Piemēram, ja modelim ir augsta precizitāte (***precision***), bet zems jūtīgums (***recall***), tas nozīmē, ka tas rada mazāk kļūdaini pozitīvu rezultātu, bet izlaiž daudz patiesi pozitīvu rezultātu. Turpretī modelis ar augsta jūtīguma (***recall***) spēju, bet zemu precizitāti (***precision***) rada vairāk kļūdaini pozitīvu rezultātu, bet uzrāda vairāk patiesi pozitīvu rezultātu. Šādos gadījumos F-1 rādītājs var palīdzēt noteikt, kurš modelis ir labāks. [32]

Apjukuma matrica jeb ***confusion Matrix*** ļauj vizualizēt rezultātus. Tā ir tabula, kurā norādīts modeļa prognozēto patieso pozitīvo, patieso negatīvo, viltus pozitīvo un viltus negatīvo rezultātu skaits. Sajukuma matrica palīdz novērtēt, cik veiksmīgi modelis atšķir divas vai vairāk klases. [32]

* 1. **Datu kopas izvēle**

Lai testētu jebkāda veida modeli, ir jābūt datiem ar ko to testēt. Autors izvēlējās testēt modeli ar īsa teksta datiem, kas līdzinās komentāriem, jo autors darba gaitā veidos, *Google Chrome* paplašinājumu, kas ļaus klasificēt kāda konkrēta *YouTube* video komentāru sadaļu. Pamatojoties uz to, autors izvēlējās datu kopu no pētījuma "*Twitter Sentiment Analysis Datasets: A Valuable Resource for Social Media Research*". Šī datu kopā ir iekļauti 3534 komentāri, un katram no tiem ir piešķirts savs noskaņojums. [33]

A screenshot of a computer

Description automatically generated

1. attēls: Daļa no datu kopas: “*Twitter Sentiment Analysis Datasets: A Valuable Resource for Social Media Research*”, kur var novērot datu kopas struktūru)
   1. **VADER modeļa testēšana**

Autors visu praktisko daļu veica *Visual Studio* *Code* vidē, kā arī modeļu testēšanai izmantoja *Python (3.8)* programmēšanas valodu. Kā arī pirmkods ir pieejams pielikumā ar nosaukumu “VADER-testēšana”. VADER testēšana tiks veikta šādos soļos: datu kopas atvēršana, datu attīrīšana, sentimenta nolasīšana, mērījumu veikšana un rezultātu parādīšana. (skatīt 6. attēlā)

A diagram of a data flow

Description automatically generated

1. Attēls: VADER modeļa testēšanas shēma)

Pirmais solis - datu kopas atvēršana: (skatīt 7. attēlā)

A black screen with white text

Description automatically generated

1. Attēls: Datu kopas atvēršana, *Python (3.8)*)

Septītajā attēlā var novērot, kā autors atver datu kopu ar *“csv.reader()”* funkciju, kas ir viena no “csv” bibliotēkas funkcijām, tad autors izmanto *“next”* funkciju, lai izlaistu pirmo datu kopas rindu. Tas ir nozīmīgi, jo modelim nevajag padot nekādu lieku informāciju. Pēc tā tiek definēts *“rows”*, kas pārtaisa katru rindu vienā lielā sarakstā.

Otrais solis – datu kopas attīrīšana: (skatīt 8. attēlā)

A screen shot of a computer code

Description automatically generated

1. Attēls: Datu kopas attīrīšana, *Python (3.8)*)

Astotajā attēlā var novērot *“clean()”* metodi, kurā tiek attīrīti dati ar *“re.sub()”* metodi, kura ir viena no *“re”* bibliotēkas metodēm. Datu kopa tiek attīrīta septiņos dažādos veidos: *“'http\S+'”* attīra visus datus no linkiem, *“'[^a-zA-Z\s]'”* atbrīvojas no visiem simboliem, cipariem un burtiem, kas nav angļu valodas alfabētā, *“'@\w+'”* tiek noņemti lietotāj vārdi, kuriem sākumā ir simbols *“@”*, *'<href+'* attīra no vārda *“<href”* datu kopā, *“'(\s+)'”* atbrīvojas no liekajām atstarpēm, *“'\[[^]]\*\]'”* atbrīvojas no teksta, kas atrodas starp kvadrāta iekavām. Tad visbeidzot tiek viss teksts samazināts uz mazajiem burtiem ar “comment.lower()” funkciju. Kā var ievērot autors neizmantoja dažas metodes datu apstrādei kā piemēram – tokenizāciju (tokenization), negāciju un intensifikātoru metodes, jo VADER izmanto iepriekš izveidotu leksikonu vārdnīcu, lai veiktu sentimenta analīzi. Tas neiesaista mācīšanos no teksta struktūras, kā to dara mašīnmācīšanās modeļi kā piemēram RoBERTa. Tāpēc arī tokenzācija būtu lieks process, jo to izmanto modeļiem, kas analizēt tekstu ar mašīnmācīšanās palīdzību. Taču negācijas un intensifikātori ir jau iestrādāti modeļa leksikonu vārdnīcā. Tad visbeidzot viss teksts tiek samazināts uz mazajiem burtiem.

Trešais solis – sentimenta noteikšana attīrītajiem datiem: (skatīt 9. attēlā)



1. Attēls: sentimenta nolasīšana attīrītajiem datiem, *Python (3.8)*)

Devītajā attēlā ir novērojams, ka autors izveido divus sarakstus *“actual\_label”* jeb “īstais sentiments” un *“predicted\_lable”* jeb “pareģotais sentiments”. Pēc tam tiek izveidots “*for”* cikls, ar kura palīdzību iziet cauri katrai datu kopas rindai, iegūstot komentārus un to faktiskos apzīmējumus jeb īsto sentimentu. Tad tiek palaista datu attīrīšanas metode *“clean”*, kura attīra katru rindu jeb komentāru. Pēc teksta datu attīrīšanas tas aprēķina sentimenta rezultātu, izmantojot *“SentimentIntensityAnalyzer”* jeb VADER modeli, kas ir no “*nltk”* bibliotēkas. Pamatojoties uz šo rezultātu, komentāri tiek iedalīti pozitīvos, negatīvos vai neitrālos komentāros. VADER modeļa rezultātu nosaka no polaritātes skalas, kur vērtības tiek izteiktas no -1 līdz 1. 1 norāda uz ļoti pozitīvu sentimentu, -1 norāda uz ļoti negatīvu sentimentu, un 0 norāda uz neitrālu sentimentu. Ja rezultāts ir lielāks par 0.05, tad komentārs tiek iedalīts kā "pozitīvs", taču, ja mazāks par -0.05, tad tas tiek iedalīts kā "negatīvs" un ja rezultāts ir starp -0.05 un 0.05, tad komentārs tiek iedalīts kā "neitrāls". Tad visbeidzot pareģotais un īstais sentiments tiek saglabāts attiecīgajos sarakstos.

Ceturtais solis – mērījumu veikšana un to rezultāti:

A screen shot of a computer code

Description automatically generated

1. Attēls: mērījumu veikšana - VADER, *Python (3.8)*)

Desmitajā attēlā autors veica mērījumus ar pieciem mērījumu veidiem - “*accuracy”* jeb precizitāti, *“precision”* jeb precīzumu, *“recall”* jeb jūtīgums, *“F1-score”* jeb F1 rezultāts un visbeidzot ar vispārīgo novērtējumu.

* 1. **RoBERTa modeļa testēšana**

RoBERTa modeļa testēšana ir līdzīga VADER modeļa testēšanai, taču RoBERTa nav balstīts uz leksikoniem, bet gan uz mašīnmācīšanos, tāpēc būs būtiskas atšķirības datu sagatavošanā un programmas izstrādē, tāpēc autors iesaka izpētīt pievienoto pirmkodu pielikumā “RoBERTa-testēšana”. RoBERTa Testēšana tiks veikta šādos soļos: datu kopas atvēršana, datu attīrīšana, datu sagatavošana modelim, sentimenta nolasīšana, mērījumu veikšana un rezultātu parādīšana. (skatīt 11. attēlā)

A diagram of a product

Description automatically generated

1. Attēls: RoBERTa modeļa testēšanas shēma)

Pirmais un otrais - datu kopas atvēršana un datu attīrīšana ir identiski VADER modeļa testēšanai. Trešais solis – datu sagatavošana modelim: Atšķirībā no VADER modeļa, kur nevajadzēja veikt tokenzāciju jeb *tokenization* šeit gan tā būs būtiska, jo tas palielina modeļa precizitāti un tas ir formāts, kā RoBERTa modelis saprot padotos datus. (skatīt 12. attēlā)



1. Attēls: Attīrīto datu tokenizācija. *“AutoTokenizer.from\_pretrained()”* metode no *“Hugging Face Transformers”* bibliotēku, *Python (3.8)*)

Ceturtais solis - sentimenta nolasīšana: Šis solis atšķiras būtiski no VADER modeļa sentimenta nolasīšanas, ja VADER modelim to varēja izdarīt ar vienu rindu kodu, tad šeit tas aizņems vairākas. Sākumā ir svarīgi atgādināt, ka tiek izmantots jau iepriekš uztrenēts modelis, jo resursu un laika dēļ nebija objektīvi trenēt savu mašīnmācīšanās modeli. Autors izvēlējās veikt testēšanu uz *Twitter* komentāru datu kopu, tāpēc autors izvēlējās piemklēt attiecīgi sagatavotu iepriekš uztrenētu RoBERTa modeli- *“cardiffnlp/twitter-roberta-base-sentiment”*.

A screen shot of a computer program

Description automatically generated

1. Attēls: Kā tiek definēts uztrenētais RoBERTa modelis “cardiffnlp/twitter-roberta-base-sentiment” , *Python (3.8)*)

Trīspadsmitajā attēlā var novērot, kā autors ir inicializējis iepriekš uztrenēto modeli, kā arī tokenizatoru, ielādējis sentimenta marķējumus *“LABEL\_MAPPING\_LINK”* un izveidojis konstruktoru *“\_\_init\_\_(self)”.*

A computer screen shot of a program code

Description automatically generated

1. Attēls: Izveidota metode *“analyze\_sentiment(self, text)”*, *Python (3.8)*)

Četrpadsmitajā attēla var novērot, kā tiek nolasīts sentiments no katras datu kopas rindas jeb komentāra. RoBERTa modelis savus rezultātus arī pasniedz polaritātes skalā, taču katram komentāram tiek piemērotas 3 polaritāšu vērtējumi – pozitīvo, neitrālo un negatīvo. Piemēram, teikumam *"Laiks ir jautrs, bet diena ir grūta."* Šādam teikumam būs trīs vērtējumi – pozitīvs: 0.5, neitrāls: 0.3 un negatīvs: 0.2. Augtākais vērtējumus ir teikuma sentiments, šajā gadījuma teikums tiktu novērtēts “pozitīvs” ar 0.5 pozitivitātes vērtējumu. Tad kods atgriež atpakaļ *“top\_label”* jeb augstāko vērtējumu. Kas pēc tam tiek sadalīts sarakstos - pozitīvi, neitrāli, negatīvi.

Piektais solis – mērījumu veikšana un to rezultāti:



1. Attēls: mērījumu veikšana - RoBERTa, *Python (3.8)*)

RoBERTa modelim tika veikti tie paši pieci mērījumi, kuri tika veikti VADER modelim - precizitāte, precīzums, jūtīgums, F1 rezultāts un vispārīgais novērtējumus.

**Izmantotā literatūra un citi avoti**

1. Peter EH Smee and Linda Smee. Neuro-Linguistic Programming, The Key To Accelerated Learning. 9. lappa - NLP explained. (2002) (Skatīts 24.10.2023)

2. IBM - What is natural language processing (NLP)? Pieejams: https://www.ibm.com/topics/natural-language-processing (Skatīts 24.10.2023)

3. foreseemed - NLP negation. Pieejams: https://www.foreseemed.com/natural-language-processing-in-healthcare#:~:text=NLP%20negation%20in%20healthcare%20is,have%20a%20condition%20or%20symptom. (Skatīts 24.11.2023)

4. techtarget - How does sentiment analysis work? Pieejams: https://www.techtarget.com/searchbusinessanalytics/definition/opinion-mining-sentiment-mining (Skatīts 07.11.2023)

5. ScienceDirect - A review on sentiment analysis from social media platforms. Pieejams: https://www.sciencedirect.com/science/article/pii/S0957417423003639 (Skatīts 07.11.2023)

6. voiceform - How to Do Sentiment Analysis in 4 Steps (With Examples). Pieejams: https://www.voiceform.com/blog-posts/sentiment-analysis (Skatīts 07.11.2023)

7. repustate - What Are The Sources Of Gathering Sentiment Analysis Data? Pieejams: https://www.repustate.com/blog/sentiment-analysis-data-source/ (Skatīts 07.11.2023)

8. medium - Ways to collect data for sentiment analysis. Pieejams: https://medium.com/analytics-vidhya/data-collection-and-annotation-measures-for-sentiment-analysis-767da1dd4272 (Skatīts 07.11.2023)

9. dataconomy - Role of data preprocessing in sentiment analysis. Pieejams: https://dataconomy.com/2023/07/28/data-preprocessing-steps-requirements/ (Skatīts 11.11.2023)

10. monkeylearn - Your Guide to Data Cleaning & The Benefits of Clean Data. Pieejams: https://monkeylearn.com/data-cleaning/ (Skatīts 11.11.2023)

11. monkeylearn - 8 Effective Data Cleaning Techniques for Better Data. Pieejams: https://monkeylearn.com/blog/data-cleaning-techniques/ (Skatīts 11.11.2023)

12. arize - Tokenization: Unleashing The Power of Words. Pieejams: https://arize.com/blog-course/tokenization/#:~:text=It%20involves%20breaking%20down%20a,entity%20recognition%2C%20and%20text%20classification. (Skatīts 11.11.2023)

13. nlpcloud - Kas ir žetonizācija? Pieejams: https://nlpcloud.com/lv/nlp-tokenization-api.html (Skatīts 11.11.2023)

14. nlp.stanford.edu - Stemming and lemmatization. Pieejams: https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html (Skatīts 11.11.2023)

15. analyticsindiamag - What is negation handling? Pieejams: https://analyticsindiamag.com/when-to-use-negation-handling-in-sentiment-analysis/#:~:text=Negation%20handling%20is%20a%20method,vicinity%20or%20scope%20of%20negation. (Skatīts 11.11.2023)

16. monkeylearn - Sentiment Analysis & Machine Learning. Pieejams: https://monkeylearn.com/blog/sentiment-analysis-machine-learning/ (Skatīts 11.11.2023)

17. monkeylearn - How Does Sentiment Analysis Work? Pieejams: https://monkeylearn.com/sentiment-analysis/ (Skatīts 11.11.2023)

18. analyticsvidhya - Rule-Based Sentiment Analysis in Python. Pieejams: https://www.analyticsvidhya.com/blog/2021/06/rule-based-sentiment-analysis-in-python/ (Skatīts 11.11.2023)

19. monkeylearn - Rule-based Approaches. Pieejams: https://monkeylearn.com/sentiment-analysis/#:~:text=Automatic%20Approaches,positive%2C%20negative%2C%20or%20neutral (Skatīts 11.11.2023)

20. octoparse - Scrape YouTube Comments for Sentiment Analysis. Pieejams: https://www.octoparse.com/blog/youtube-comment-scraper (Skatīts 11.11.2023)

21. mulesoft - What is an API?. Pieejams: https://www.mulesoft.com/resources/api/what-is-an-api (Skatīts 17.11.2023)

22. developer.google.com - The following table defines the properties that appear in this resource: Pieejams: https://developers.google.com/youtube/v3/docs/comments#:~:text=The%20snippet%20object%20contains%20basic%20details%20about%20the%20comment.&text=The%20display%20name%20of%20the%20user%20who%20posted%20the%20comment.&text=The%20URL%20for%20the%20avatar%20of%20the%20user%20who%20posted%20the%20comment.&text=The%20URL%20of%20the%20comment%20author's%20YouTube%20channel%2C%20if%20available. (Skatīts 17.11.2023)

23. zyte- What Is Web Scraping? Pieejams: https://www.zyte.com/learn/what-is-web-scraping/ (Skatīts 17.11.2023)

24. Quora - Is scraping YouTube legal? Pieejams: https://www.quora.com/Is-scraping-YouTube-legal (Skatīts 17.11.2023)

25. GitHub - VADER-Sentiment-Analysis. Pieejams: https://github.com/cjhutto/vaderSentiment (Skatīts 25.11.2023)

26. comet - RoBERTa: A Modified BERT Model for NLP. Pieejams: https://www.comet.com/site/blog/roberta-a-modified-bert-model-for-nlp/#:~:text=An%20open%2Dsource%20machine%20learning,Facebook%20in%20the%20year%202019. (Skatīts 25.11.2023)

27. Github - vader-vs-roberta. Pieejams: https://github.com/topics/vader-vs-roberta (Skatīts 25.11.2023)

28. ReserchGate - [attēls] Pieejams: https://www.researchgate.net/figure/Overview-of-web-scraping-system\_fig2\_347999311 (Skatīts 04.12.2023)

29. Linkedin - [attēls] Pieejams: https://www.linkedin.com/pulse/stemming-lemmatization-ashik-kumar (Skatīts 11.12.2023)

30. CodeCademy - What is REST? Pieejams: https://www.codecademy.com/article/what-is-rest (Skatīts 11.12.2023)

31. analyticsvidhya - Building Naive Bayes Classifier from Scratch to Perform Sentiment Analysis. Pieejams: <https://www.analyticsvidhya.com/blog/2022/03/building-naive-bayes-classifier-from-scratch-to-perform-sentiment-analysis/> (Skatīts 12.12.2023)

32. Medium- Accuracy, Precision, Recall, F-1 Score, Confusion Matrix, and AUC-ROC. Pieejams: https://medium.com/@riteshgupta.ai/accuracy-precision-recall-f-1-score-confusion-matrix-and-auc-roc-1471e9269b7d (skatīts: 05.01.2024)

33. Gigasheet- Twitter Sentiment Analysis Datasets: A Valuable Resource for Social Media Research. Pieejams: <https://www.gigasheet.com/sample-data/sentiment-analysis-dataset> (skatīts: 06.01.2024)

34. VADER-testēšana

35. RoBERTa-testēšana