Summer 7-2020

# KLASIFIKIMI I LAJMEVE ME ANE TE TEXT MINING

Myhedin Zika

Programi për Shkenca Kompjuterike dhe Inxhinieri

# KLASIFIKIMI I LAJMEVE ME ANE TE TEXT MINING

Shkalla Bachelor

Myhedin Zika

Korrik / 2020
Prishtinë

Programi për Shkenca Kompjuterike dhe Inxhinieri

Punim Diplome
Viti akademik 2014 – 2015

Myhedin Zika

# KLASIFIKIMI I LAJMEVE ME ANE TE TEXT MINING

Mentori: Dr. Bertan Karahoda

Korrik / 2020

Ky punim është përpiluar dhe dorëzuar në përmbushjen e kërkesave të pjesshme për Shkallën Bachelor

# ABSTRAKT

Në këtë punim trajtohet problemi se cili algoritëm funksionon më së miri në klasifikimin e tekstit shqip në kategori të caktuara. Arsyeja kryesore pse kam zgjedhur këte temë është për shkak se klasifikimi i tekstit në gjuhën shqipe mund të jetë pak me i komplikuar për shkak të gjuhës sonë. Rëndesia e ketij punimi është se mund të implementohen këto algoritme në krijimin e platformave e ndryshme qe implementojne keto algoritme, mirepo me fokus kryesisht me permbajtje në gjuhën shqipe.

Për zgjedhjen e ketij problemi kemi marrur dy algoritme për krahasim, njëri prej tyre Naive Bayes dhe tjetri SVM. Kemi grumbulluar të dhëna për trajnim dhe testim të modeleve, ku në bazë të statistikave të dy algoritmet kane pasur një saktësi të madhe, mirepo për klasifikim të tekstit në gjuhën shqipe më mirë funksionon algoritmi i Naive Bayes.

Rezultatet e ketij punimi tregojne qe mund te krijohet një platformë ku mund të mblidhen lajme të ndryshme në gjuhen shqipe dhe me anë të ketyre algoritmeve të klasifikohen në kategoritë që i përkasin.

## MIRENJOHJE/FALENDERIME

Kjo arritje nuk do të ishte e mundur pa familjen time të cilet gjate gjithë kohës më ofrojne përkrahjen e tyre ne synimet e mia.

Për këshillimin e vazhdueshëm, inkurajimin e dhënë dhe rolin që patën në formësimin tim intelektual e profesional, falënderoj gjithë stafin e kolegjit UBT, në veçanti mentorin e temës, profesorin Bertan Karahoda.

Gjithashtu, dua t'i falënderoj të gjithë kolegët studentë të fakultetit për përkrahjen dhe kujtimet e mira nga studimet.

Falënderoj të gjithë kolegët e punës të kompanisë Gjirafa, që më ndihmuan me ide lidhur me aplikacionin.

## TABELA E PËRMBAJTJES

# LISTA E FIGURAVE

## LISTA E TABELAVE

## FJALORI I TERMAVE

NLP – Natural Language Processing

MT – Machine Translation

ML – Machine Learning

SVM – Support Vector Machine

HTTP – Hypertext Transfer Protocol

# 1. HYRJE

Me zhvillimin e shpejtë të teknologjise dhe rritjes shumë të shpejtë të informacionit online, klasifikimi i tekstit në kategori është bërë njëra nga teknikat kryesore per trajtimin dhe organizimin e të dhënave. Klasifikimi i teksit ndryshe i njohur si kategorizim i tekstit i përket më së shumti fushës së text mining. Text mining është nxjerrja e informacioneve më të rendësishme nga një tekst i dhënë, duke gjetur modele nga teksti i dhënë dhe na jep rezultat sa më të saktë që mundet.

Klasifikimi i tekstit është njëra prej detyrave më të rendesishme dhe njëlloj të zakonshme në supervised machine learning. Teknikat e kategorizimit te tekstit përdoren për të klasifikuar informacione të ndryshme në kategori të paracaktuara, për të gjetur informacione interesante qe gjenden ne web, si dhe per te ndihmuar perdoruesin për të kerkuar rreth ndonje temë specifike që i intereson. Klasifikimi i teksit mund të bëhet në dy mënyra: në klasifikim manual dhe klasifikim automatik. Ne klasifikim manual, rezultatet mund të jenë shumë te sakta mirepo merr shume kohë dhe kushtron shtrenjte. Në vend se të mbeshtetemi në klasifikim manual, klasifikimi automatik mësohet të beje klasifikime në bazë të vezhgimeve të mëhershme. Duke perdorur të dhëna paraprake si të dhëna trajnimi, nje algoritëm i machine learning mund të mesoje dallimet e ndryshme ne mes pjeseve te tekstit dhe te jap nje rezultat duke u bazuar ne te dhenat e trajnimit. Automatizimi i procesit të klasifikimit të tekstit është tepër i rendesishem në menyrë që të jemi të aftë të klasifikojmë një grumbull të dhënash për një kohë sa më të shpejtë. Duke marrur parasysh se sa shumë informacione gjenden online, automatizimi i klasifikimit të teksit gjen perdorim të gjëre në fusha të ndryshme si për shembull në permbledhje të informacionit. Disa nga algoritmet me të njohura të machine learning për klasifikime të teksit perfshijnë Naive Bayes algoritmet të cilat bazohen në teoremën e Bayes-it, Support Vector Machines algoritmet të prezantuara nga V. Vapnik dhe algoritmet e deep learning .

Ne në ketë punim do të trajtojmë saktësinë e algoritmit të Naive Bayes dhe algoritmin e Support Vector Machines për klasifikim të tekstit në gjuhen shqipe ku do të marrim rreth 1140 lajme në gjuhën shqipe dhe do ti klasifikojmë ato në kategori të ndryshme nëpermjet

ketyre dy algoritmeve. Një pjesë të lajmeve i ndajmë për trajnim të modelit, ndërkaq një pjese tjeter të lajmeve i ndajmë për matjen e saktësise së dy algoritmeve të perdorura.

Implementimi i algoritmit të Naive Bayes është bëre duke përdorur gjuhen programuese Java, ku në ketë gjuhe e përdorim një librari Open Source Apache Open NLP e cila ka shumë metoda të gatshme për klasifikim të tekstit. Ndërkaq implementimi i algoritmit të Support Vector Machines është bëre duke përdorur gjuhën programuese C# dhe librarine Open Source libsvm për klasifikim.

Në kapitullin e parë shqyrtojmë se çka është Machine learning, si ndahen llojet e algoritmeve, avantazhet dhe disavantazhet e Naive Bayes dhe avantazhet e Support Vector Machines.

Në kapitullin e katërt do të shqyrtojme pjesen e kodimit ku i implementojmë algoritmet, dhe i ruajmë të dhënat e predikimeve në një file.

# 2. SHQYRTIMI I LITERATURES

## 2.1 Machine learning

Machine learning (ML) është studimi i algoritmeve të kompjuterit që përmisohen automatikisht në bazë të përvojes [1]. Pra ML është shkenca ku kompjuteret mësojnë dhe veprojnë në menyrë njerëzore duke e ushqyer më të dhëna. Pra kompjuterit i ipen të dhëna për procesim, sistemi meson nga këto të dhëna dhe me anë te kësaj përvoje kompjuteri është i aftë të marr vendime logjike bazuar në të dhënat që i ka marrur më herët.

Machine learning i ka disa lloje të algoritmeve për mësim, mirëpo dy nga më kryesoret janë

1. Supervised learning (Mësimi i mbikëqyrur)
2. Unsupervised learning (Mësimi i pa-mbikëqyrur)

Supervised learning është mësimi i mbikëqyrur i makinës për të mësuar një funksion që hartëzon një hyrje në një dalje të bazuar në çifte shembull hyrje-dalje [2].

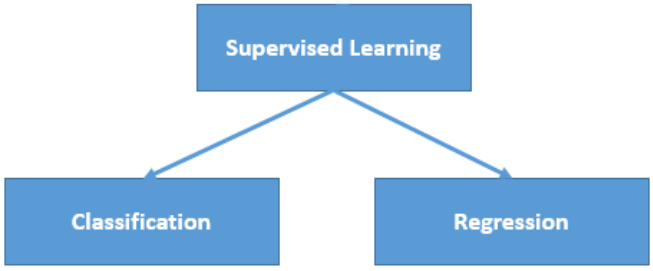



Figura 1. Llojet e supervised learning

Klasifikimi është njeri prej tipeve të supervised learning në të cilën programi kompjuterik mëson nga të dhënat e dhëna, dhe më pas e perdor ketë mësim për të klasifikuar vëzhgime të reja.

Ky grup i të dhënave thjeshtë mund të jenë dy klasa. Shembull mund të jetë klasifikimi i emailit ne rast se eshte spam ose nuk eshte spam.

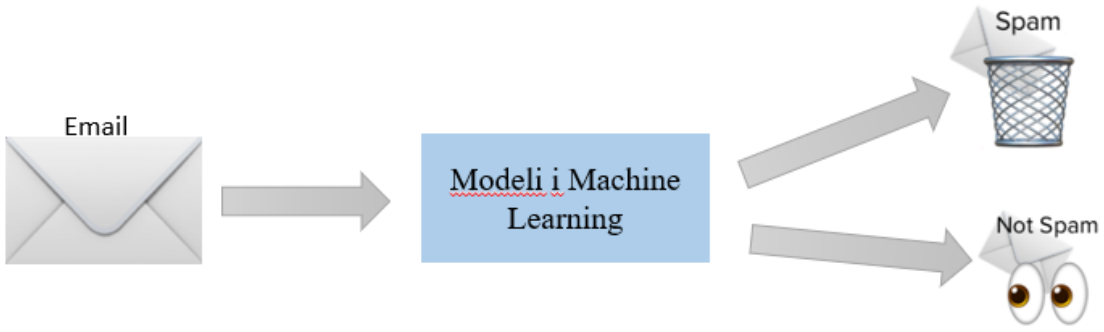



Figura 2. Shembull i supervised learning me klasifikim

Disa shembuj praktikë të klasifikimit janë: njohja e të folurit, identifikimi biometrik, klasifikimi i dokumenteve, njohja e shkrimit të dorës etj.

Regression në supervised learning është marrëdhënia midis dy ose më shumë ndryshoreve ku një ndryshim në nje ndryshore ndikon në ndryshimin e ndryshores tjeter.

Dy karakteristika të rëndësishme të regresionit janë se përgjigjet që mund të priten nga modeli janë gjithnjë sasiore në natyrë. Gjithashtu, modeli mund të krijohet vetëm duke marrë parasysh të dhënat e kaluara.

Një algoritëm popullor që kryen regresion është algoritmi i regresionit linear. Si shembull mund ta marrim çmimin e një shtepie, çmimi i saj varet nga hapesira e shtepise.

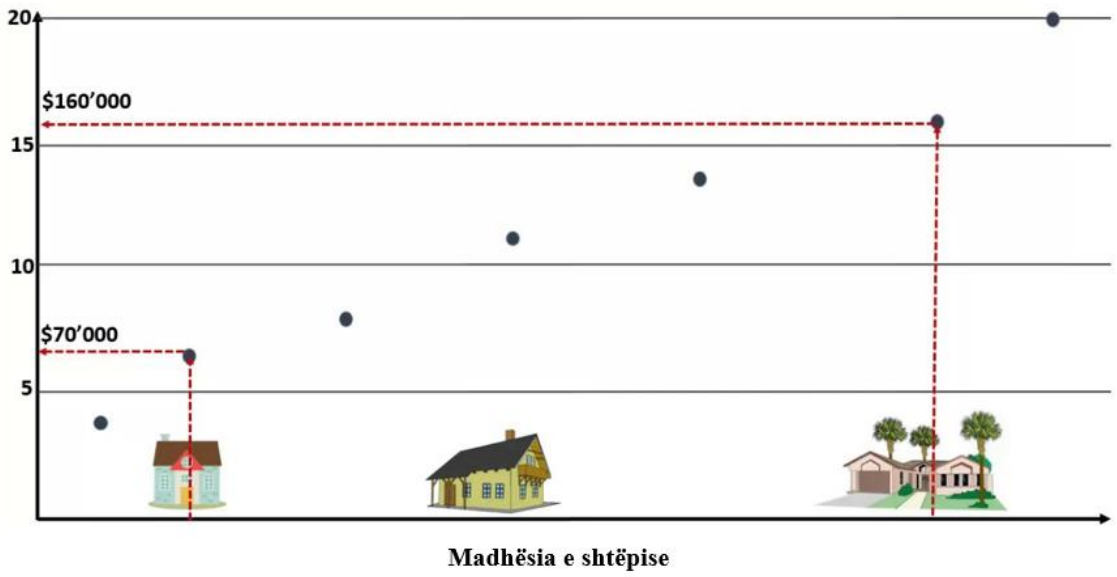



Figura 3. Shembull i supervised learning me regression

Unsupervised learning është një lloj mësimi i makinerisë që kërkon modele të pazbuluara më parë në një grup të dhenash pa etiketa paraprake dhe me një minimum të mbikqyrjes njerëzore.

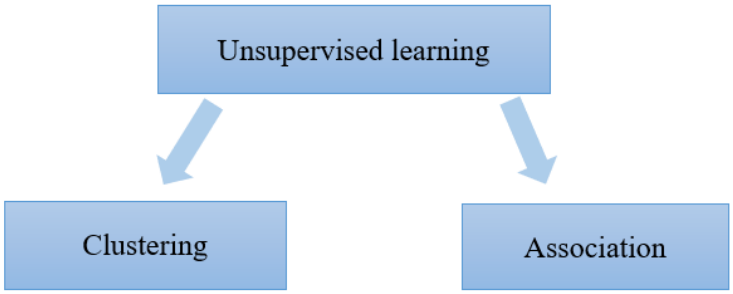



Figura 4. Llojet e unsupervised learning

Clustering mund të konsiderohet si problemi më i rendësishem në unsupervised learning, për shkak se duhet të përballet me problemin që të gjej një strukturë në një koleksion të të dhënave që nuk janë të organizuara. Më thjeshte definicioni i clustering mund të jetë procesi i organizimit të objekteve në grupe, anëtaret e të cilit janë të ngjajshëm në ndonjë menyrë.

Kështu që clusteri është një koleksion objektesh të cilat janë të ngjajshme mes tyre, dhe janë të ndryshme me objektet që i përkasin cluster-ave tjerë.


Figura 5. Shembull i clustering tek unsupervised learning

Association na ndihmon të krijojme ngjajshmëri të objekteve brenda ndonjë koleksioni të të dhëna-ve. Kjo teknikë është shumë e rëndesishme për të gjetur ngjajshmëri ndërmjet variabla-ve në databaza të mëdha.

| **Klienti** | **Gjerat e blera** |
|---|---|
| **1** | {Bukë, qumësht} |
| **2** | {Bukë, Alkohol, Pampersa, Vezë} |
| **3** | {Qumësht, Pampersa, Alkohol, Coca cola} |
| **4** | {Bukë, Qumësht, Pampersa, Alkohol} |
| **5** | {Bukë, Qumësht, Pampersa, Coca cola} |
| **...** | ... |

Figura 6. Shembull i unsupervised learning association

Nga kjo tabele, ne mund të kuptojmë që gjerat qe po shiten më se shumti në një market janë alkoholi dhe pampersat. Keshtu që ne mund ta nxjerrim si rregull të association qe edhe klienti i rradhes ka gjasa shumë të mëdha për të blere alkohol dhe pampersa.

Fushat ku më së shumti gjen përdorim unsupervised learning është në analizim të shitjeve të gjësendeve të ndryshme të ndonjë marketi, semantic clustering, optimizim të dërgimit të porosive, identifikimin e vendeve ku ka gjasa më se shumti të ndodh ndonjë aksident etj.

## 2.2 Natural Language Processing (NLP)

Natural language processing (NLP) është një fushë në machine learning e cila ka aftësi për të kuptuar, analizuar, manipuluar dhe potencialisht të gjeneroje gjuhë. Më saktësisht, NLP është nënfushe e linguistikës, shkencave kompjuterike, inxherinierisë së informacionit dhe intelegjences artificiale që mirret me ndërveprimet midis kompjuterëve dhe gjuhëve njerëzore, në veçanti se si të programojnë kompjuteret për të përpunuar dhe analizuar sasi të mëdha të të dhënave të gjuhës natyrore.

NLP ka filluar që disa dekada, thuhet që ka filluar që nga vitet 1950, mirepo mund të gjinden punime edhe nga kohet më te hershme, si rast konkret mund të mirret Weather dhe Booth të cilët kanë filluar njërin prej projekteve më të hershme të Machine Translation në vitin 1946 në përkthim të kompjuterit duke u bazuar në ekspertizen për të thyer kodet e armikut gjatë luftes së dytë botërore. Kjo ndikoi që më vonë të inspirohen shumë projekte tjera që mirren me përkthim të kompjuterit. Poashtu në vitin 1954 është bëre një eksperiment i quajtur Georgetown experiment i cili ka perfshirë përkthimin automatik të më shumë se 60 fjalëve ruse në gjuhën angleze. Autorët atehere kanë deklaruar që brenda tre deri pesë vjeteve përkthimi i makines do të jete problem i zgjidhur, mirëpo progresi real ishte shumë më i ngadalshëm për shkak të reduktimit të shpenzimeve për përkthim të makines [3].

NLP është një nëndegë e shkencave kompjuterike dhe inteligjencës artificale e cila mirret me bashkeveprimin ndërmjet kompjuterëve dhe gjuhëve njerëzore. Përdoret për të aplikuar algoritme makinore në tekst dhe fjalë.

Në jeten e perditshme, ne mund të hasim në shumë fusha të cilat e përdorin NLP, disa nga fushat më të njohura janë:

- Machine Translation (Përkthim i makinës) - Secili prej nesh e kupton se nevojitet shumë kohë për të perkthyer manualisht një tekst prej nje gjuhe në një gjuhë tjetër. Mirepo kur nje kompjuter e bën ketë, atëhere e kuptojme qe jemi duke e parë njerën prej aplikimeve më të mira të NLP e cila është "Machine" Translation. Ideja prapa Machine Translation është e thjeshtë, që të krijohen algoritme kompjuterike të cilat na mundesojnë përkthimin automatik të ndonjë teksti pa ndonjë nderhyrje njerëzore. Aplikacioni më i famshëm që perdor machine translation është Google Translate. Google Translate është e bazuar në SMT - Statistical Machine Translation. Google

Translate nuk i përkthen fjalët për fjalë, mirepo grumbullon sa më shumë të dhëna që është e mundur, dhe pastaj gjen ndonjë ngjajshmëri në gjuhë për ta perkthyer sa më sakte. Sipas Google, çdo ditë Google Translate përdoret rreth 500 milion here me rreth 100 miliardë fjale te perkthyera çdo ditë [4].

- Speech recognition (Njohja e zërit) - Speech recognition është aftësia e makinës apo një programi për të identifikuar fjalët dhe shprehjet që fliten dhe konvertimi i ketyre fjaleve në nje format që mund të kuptohet nga makina. Sot speech recognition përdoret në një numer të madh produktesh, si shembull mund të marrim voice assistances ku ndër to më të njohurat janë:
    a. Cortana i cili është nje assistent virtual i ndërtuar nga Microsoft për Windows 10, Windows 10 Mobile, Windows Phone 8.1 etj. Rreth 800 milionë njerëz kane pasur mundësi për të përdorur Cortanen në vitin 2019, nderkaq numri i pyetjeve që i është bere Cortanes që nga lansimi i saj është më shumë se 18 miliard [5].
    b. Siri eshte poashtu nje assistent virtual, mirepo është i zhvilluar nga kompania Apple. Siri është asistent virtual i integruar në shumicën e pajisjeve te zhvilluara nga Apple, posaçerisht ne iPhone-s, Apple TV-s dhe Airpods. Sipas nje raporti nga Apple, deri ne vitin 2018 me shume se 500 milione kliente e perdorin Siri-n [6].
    c. Alexa është një assistent virtual i zhvilluar nga Amazon, i cili fillimisht është perdorur në pajisjen Amazon Echo smart speakers. Në janar te vitit 2019, ekipi i Amazon-it ka treguar që kanë shitur më shumë se 100 milionë pajisje të Alexa-s [7].
- Text classification (Klasifikim i tekstit) - Text classification është procesi i percaktimit të kategorise në një ndonjë tekst nga njëra prej kategorive të paradefinuara. Text classifications përdoren për të organizuar, strukturuar dhe kategorizuar shumicën e të dhënave. Shembull konkret mund ta marrim ndarjen e dokumenteve në disa kategori, na vjen një dokument i ri dhe është e nevojshme për të caktuar se cilës kategori i perket ky dokument i ri. Duke perdorur NLP, klasifikuesit e tekstit mund të analizojne automatikisht tekstin dhe i caktojnë një kategori që i

pershtatet më së shumti në bazë të përmbajtjes që ka ai tekst. Njëri prej produkteve më të famshme ne bote i cili perdor text classification është Gmail-i. Gmail është shërbim i postës elektronike që është zhvilluar nga kompania Google. Gmail e perdor NLP për të detektuar email-at spam, me anë të NLP Google i dërgon këto emaila në një folder të veçantë SPAM. Me ndihmën e NLP, Google i dërgon rreth 100 milion emaila në SPAM çdo ditë [8].

Perveq ketyre fushave, NLP gjen aplikim edhe në fusha tjera si chatbots, predikim të shkrimit, njohjen e fjalëve etj.

Funksionet themelore të NLP për klasifikim të tekstit janë:

1. Sentence Tokenization
2. Word Tokenization
3. Stemming
4. Stop Words
5. Regex
6. Bag Of Words
7. TF – IDF

**Sentence Tokenization** – është procesi i ndarjes së tekstit në fjali. Në bazë te hulumtuesve, rreth 86% e artikujve përfshijnë fjaline e rendesishme ne nje ose dy fjali te para [9].

Për të demonstruar Sentence Tokenization po e marrim një paragraf si shembull:

```
var text = "Kompania Samsung ka prezantuar telefonin me ekran palosës dhe atë dy ditë para se zyrtarisht pritet të prezantohet. Bëhet fjalë për Galaxy Z Flip, që besohet se ka ekran palosës prej 6.7 inç dhe ka një xham tejet të hollë dhe që u zbulua gjatë ceremonisë së Academy Awards, transmeton Telegrafi. Reklama tregon telefonin që është i disponueshëm me ngjyrë të zezë dhe purpurtë, duke realizuar video thirrje në formatin gjysmë të hapur.";
```

Me anë të Sentence Tokenization, ky paragraf shndërrohet në nje varg të fjalive:

| |
|---|
| Kompania Samsung ka prezantuar telefonin me ekran palosës dhe atë dy ditë para se zyrtarisht pritet të prezantohet. |
| Bëhet fjalë për Galaxy Z Flip, që besohet se ka ekran palosës prej 6.7 inç dhe ka një xham tejet të hollë dhe që u zbulua gjatë ceremonisë së Academy Awards, transmeton Telegrafi. |
| Reklama tregon telefonin që është i disponueshëm me ngjyrë të zezë dhe purpurtë, duke realizuar video thirrje në formatin gjysmë të hapur. |

Tabela 1. NLP Shembull i Sentence Tokenization

**Word Tokenization –** është procesi i ndarjes së një teksti të madh në fjalë. Tokenizimi na mundëson të krijojmë nje strukturë në nje tekst te pastrukturuar më heret.
Shembull i Word Tokenization mund të jetë kjo fjali:
var text = "Kompania Samsung ka prezantuar telefonin me ekran palosës dhe atë dy ditë para se zyrtarisht pritet të prezantohet."
['Kompania', 'Samsung', 'ka', 'prezantuar', 'telefonin', 'me', 'ekran', 'palosës', 'dhe', 'atë', 'dy', 'ditë', 'para', 'se', 'zyrtarisht', 'pritet', 'të', 'prezantohet', '.']
**Stemming –** është procesi i zvogelimit të një fjalë në formen e tyre bazë(rrënje) duke i larguar parashtesat dhe prapashtesat që gjinden në ate fjalë.

| Fjala | Forma baze | Shtesa |
|---|---|---|
| Punoj | Pun | oj |
| Rishqyrto | shqyrto | Ri |
| Përkrah | krah | për |

Tabela 2. NLP Shembull i Stemming

**Stop words –** në gjuhën kompjuterike, stop words janë fjalët të cilat filtrohen paraprakisht ose pas perpunimit të të dhenave të gjuhës natyrore [10]. Stop words janë fjalët që përdoren më së shumti në një gjuhë. Këto fjalë nuk ndikojnë në klasifikim për shkak se gjinden kudo.

I pari i cili e ka perdorur këte shprehje për keto fjale ka qene Hans Peter Luhn, njëri prej pionerve në text mining [11].

Shembuj të stop words në gjuhën shqipe janë:

[‘unë’, ‘ti’, ’ai’, ‘ajo’, ’ne’, ’ju’, ’ata’, ‘ato’, ‘kush’, ‘cili’, ‘cila’, ‘veten’, ‘vetveten’, ‘te tijat’, ‘im’, ‘e’, ‘tu’, ’i’, ‘saj’, ‘dikush’, ‘asnje’, ‘kam’, ‘jam’, ‘eshte’, ‘ishte’]

**Regular Expressions** – është njëra prej veglave më të rëndesishme në gjuhët programuese. Regular Expressions mund të thirren edhe me një emër tjetër të quajtur regex. Regex është një set i karaktereve, ose një patern i cili përdoret për shembull për të gjetur fjalë te caktuara brenda ndonjë teksti, për te gjetur të gjitha fjalët që fillojnë me shkronjë të madhe, apo për te gjetur nje numër telefoni brenda një dokumenti etj.

Shembull i regex mund ta marrim kete tekst ku ne me ane te regex-it deshirojmë ti gjejmë emailat që janë në ketë tekst:

Regex patern = ‘\w+@\w+.\w{3}’;

Teksti: Ky është vetëm një shembull i regex, ku me anë të ketij regexi ne do ti gjejme emailat qe jane shkruar ne këtë tekst. Për më shumë informata na kontaktoni në emailin myhedinzika@gmail.com apo ne emailin mz33396@ubt-uni.net.

Rezultatet: Ky regex do te shfaq 2 rezultate e ato jane “myhedinzika@gmail.com”, dhe mz3 3396@ubt-uni.net.

**Bag-of-words model** – është një mënyre për të nxjerrë tiparet nga teksti për tu perdorur ne modelim me algoritmet e Machine Learning. Bag of words modeli zakonisht perdoren ne m etodat e klasifikimit te dokumenteve ku frekuenca e seciles fjale përdoret si tipar per te trajn uar nje klasifikues. Një reference e mehereshme e Bag of words modelit mund te gjendet te k artikulli i Zellig Harris ne vitin 1954 “Distributional Structure” [12].

**Term Frequency – Inverse Document Frequency (TF-IDF) -** është një peshë e cila zako nisht perdoret në marrje të informacionit dhe text mining. Kjo peshë është një masë statistik ore e perdorur për të vleresuar se sa është e rëndesishme një fjalë për një dokument në një k oleksion ose korpus. Rëndesia rritet proporcionalisht me numrin e herëve që një fjalë shfaqe t në dokument, mirëpo kompensohet nga shpeshtësia e fjalës në korpus. TF-IDF shpesh për

doret nga motorët e kerkimit si një mjet qendror në shënimin dhe renditjen e rëndesise së nj ë dokumenti duke marrë një pyetje të perdoruesit. Sot 83% e sistemeve rekomanduese të ba zuara në tekst të librarive digjitale e përdorin TF-IDF [13].

### 2.3 Naive Bayes

Algoritmi i Naive Bayes është një algoritëm probabilistik i cili zakonisht perdoret per problemet që kanë lidhje me klasifikim. Naive Bayes është i thjeshtë, intuitiv dhe zakonisht performon tepër mirë në raste te shumta. Per shembull, filterat për spam qe i perdorin sherbimet e emaila-ve janë te ndertuar ne Naive Bayes. Funksionon në bazë të probabilitetit të kushtëzuar. Probabiliteti i kushtëzuar është probabiliteti qe diçka të ndodhë, duke pasur parasysh qe diçka tjeter tashmë ka ndodhur. Duke përdorur probabilitetin e kushtëzuar, ne mund te llogarisim probabilitetin e një ngjarje duke përdorur njohuritë e saj paraprake.

$$P(A|B) = \frac{P(B|A) * P(A)}{P(B)}$$

Figura 7. Formula e Teoremes se Bayes

ku:

- P(A) është probabiliteti i hipotezes A të jetë e vertetë. Kjo njihet si probabiliteti paraprak.
- P(B) është probabiliteti i provës(pavarësisht hipotezës)
- P(B|A) është mundësia e provës duke pasur parasysh që hipoteza është e vertetë.
- P(A|B) është probabiliteti i hipotezës duke pasur parasysh se provat janë aty

Tipet e klasifikuesve Naive Bayes jane:

- Multinomial Naive Bayes – Ky tip më së shumti përdoret për probleme të klasifikimit të dokumenteve, domethenë nëse një dokument i përket kategorisë së sportit, politikës, teknologjisë etj. Karakteristikat / parashikuesit e përdorur nga klasifikuesi janë frekuenca e fjalëve të pranishme në dokument.
- Bernoulli Naive Bayes – Ky tip është i ngjajshëm me Multinomial Naive Bayes, mirëpo predikuesit janë vlera boolean. Parametrat që përdorim për të parashikuar

ndryshoren e klases marrin vetëm vlera po ose jo, per shembull nëse një fjalë ndodhet në tekst ose jo.

- Gaussian Naive Bayes – Kur parashikuesit marrin një vlerë të vazhdueshme dhe nuk janë diskrete, supozojmë se keto vlera janë marrë nga një shperndarje gaussiane.

E marrim nje shembull te Naive Bayes ku i marrim disa të dhëna trajnimi për mot si dhe një vlerë "Luaj"(duke treguar per mundesitë per te luajtur). Tani neve na duhet të klasifikojmë në bazë të motit se a do të luajnë lojtarët apo jo. I vendosim keto te dhena ne disa tabela.

| Moti | Luaj |
|---|---|
| Me diell | Jo |
| Me re | Po |
| Me shi | Po |
| Me diell | Po |
| Me diell | Po |
| Me re | Po |
| Me shi | Jo |
| Me shi | Jo |
| Me diell | Po |
| Me shi | Po |
| Me diell | Jo |
| Me re | Po |
| Me re | Po |
| Me shi | Jo |

Tabela 3. Shembull Naive Bayes – Të dhënat për trajnim të modelit

| Moti | Jo | Po |
|---|---|---|
| Me re | | 4 |
| Me shi | 3 | 2 |
| Me diell | 2 | 3 |
| Totali | 5 | 9 |

Tabela 4. Shembull Naive Bayes – Frekuencat e të dhenave

| Moti | Jo | Po | | |
|---|---|---|---|---|
| Me re | | 4 | = 4/14 | 0.29 |
| Me shi | 3 | 2 | = 5/14 | 0.36 |
| Me diell | 2 | 3 | = 5/14 | 0.36 |
| Te gjitha | 5 | 9 | | |
| | = 5/14 | = 9/14 | | |
| | 0.36 | 0.64 | | |

Tabela 5. Shembull Naive Bayes – Gjasat e të dhenave

E marrim shembullin në rastin kur moti është me diell:

P(Po | Me diell) = P ( Me Diell | Po) * P(Po) / P( Me Diell)

Këtu kemi P(Me diell | Po) = 3/9 = 0.33, P(Me diell) = 5/14 = 0.36, P(Po) = 9/14 = 0.64

Tani, P (Po | Me Diell) = 0.33 * 0.64 / 0.36 = 0.60 e cila ka probabilitet më të lartë

Disa nga aplikimet e algoritmit të Naive Bayes janë:

- Parashikime në kohë reale: Naive Bayes është një klasifikues i etur për të mësuar dhe është i shpejtë. Prandaj mund të përdoret për të bëre parashikime në kohë reale.
- Parashikimi me shumë klasa: Ky algoritëm është gjithashtu shumë i njohur për veçorinë e parashikimit të shumë klasave. Ketu mund të parashikojmë probabilitetin e klasave të shumta të ndryshores së synuar.
- Text classification(klasifikim i tekstit) / Spam filtering (filtrimi i spam) / Sentiment Analysis (Analiza e ndjenjës) : Klasifikuesit Naive Bayes janë të përdorur më së shumti në klasifikimin e tekstit(për shkak të rezultatit më të mire në problemet me shumë klasa dhe rregullin e pavaresise) kanë shkallë më të lartë suksesi krahasuar me algoritmet tjera. Si rezultat, perdoret gjerësisht në filtrimin e spam-it dhe sentiment analysis(në analizën e rrjeteve sociale për të identifikuar ndjenjat pozitive dhe negative të klientit).
- Sistemi i rekomandimeve: Klasifkuesi Naive Bayes dhe filtrimi bashkëpunues ndertojnë së bashku një sistem rekomandimi që përdor teknikat e mësimit të makinave dhe minerave të të dhenave për të filtruar informacionin e padukshëm dhe të parashikojne nëse një përdorues do të donte ndonjë material të caktuar apo jo.

### 2.3.1 Avantazhet dhe disavantazhet e Naive Bayes

Naive Bayes algoritmi ka shumë avantazhe mirepo më kryesoret e saj janë këto:

- Eshtë i thjeshte dhe i lehte për tu implementuar
- Eshtë i shpejte dhe mund të përdoret për të bere predikime në kohë reale
- Nuk i nevojiten shumë të dhëna për trajnimin e modelit
- Eshtë i matshëm
- Mund të bejë parashikime të mundshme
- Mund të trajtojë të dhëna të vazhdueshme dhe diskrete.
- Mund të funksionojë lehtësisht edhe nese ka vlera që mungojnë
- Eshtë lehtë për tu përditesuar me të dhëna të reja
- Më i pershtatshëm në problemet e klasifikimit të tekstit

Sikurse çdo algoritëm tjeter qe nuk mund të jetë perfekt, edhe Naive Bayes ka disavantazhet e saj, e ato janë:

- Supozimi se veçoritë janë të pavarura e cila është rralle herë e vërtete në aplikimet e jetës reale.
- Mungesa e të dhënave.
- Shanset për humbje të saktësisë.
- Frekuenca zero, për shembull në rast se kategoria e ndonjë ndryshore kategorike nuk shihet në grupin e të dhënave për trajnim, atëhere modeli i jep një probabilitet zero asaj kategorie dhe atëhere nuk mund të bëhet një parashikim.

## 2.4 Support Vector Machines

Support Vector Machine(SVM) është një algoritëm i mbikëqyrur mësimi, i aftë për të kryer klasifikimin, regresionin dhe madje edhe zbulimin më të larget. SVM është një klasifikues i shpejte dhe i besueshëm që performon shumë mire edhe me të dhëna të kufizuara.

Algoritmi origjinal i SVM është shpikur nga Vladimir N. Vapnik dhe Alexey Ya. Chervonenkis në vitin 1963, nderkaq në vitin 1992 Bernhard E. Boser, Isabelle M. Guyon dhe Vladimir N. Vapnik sygjeruan një mënyre për të krijuar klasifikues jolinear duke aplikuar mashtrimin e kernelit në hiperplanet me marzhë maksimale [14].

SVM është popullarizuar kryesisht për shkak të suksesit që ka pasur në njohjen e shifrave të shkruara me dorë.

SVM-s përdoren për të zgjidhur probleme të ndryshme në jetën reale si:

- SVM-s janë të dobishem në kategorizimin e tekstit dhe hipertekstit, pasi aplikimi i tyre mund të zvogëloje ndjeshëm nevojen për raste trajnimi të etiketuara në të dy parametrat standard induktiv dhe transduktiv [15].
- Duke perdorur SVM mund të behet klasifikimi i imazheve [16].
- Klasifikimi i të dhënave satelitore sikur të dhenat SAR duke përdorur SVM të mbikëqyrur [17].
- Me anë të SVM mund të lexohen karakteret e shkruara me dorë [18].
- Algoritmi i SVM është aplikuar gjerësisht në shkencat biologjike dhe shkencat e tjera. Ato janë perdorur për te klasifikuar proteinat me deri ne 90% te komponimeve te klasifikuara saktë. Testet e permutacionit bazuar në peshat e SVM janë sygjeruar si një mekanizëm për interpretimin e modeleve SVM [19].

Qëllimi i SVM algoritmit është krijimi i vijës më të mirë ose kufirit të vendimeve që mund të veçojnë hapesirën n-dimensionale në klasa, në menyrë që të mund ta vendosim lehtësisht pikën e re të të dhënave në kategorine e saktë në të ardhmen. Ky kufi i vendimit më të mirë quhet hiperplan(hyperplane).

SVM zgjedh pikat / vektoret ekstreme që ndihmojnë në krijimin e hiperplanes. Këto raste ekstreme quhen si vektorë mbështetes(Support vectors) dhe keshtu algoritmi cilësohet si SVM(Machine vektori mbështetës).

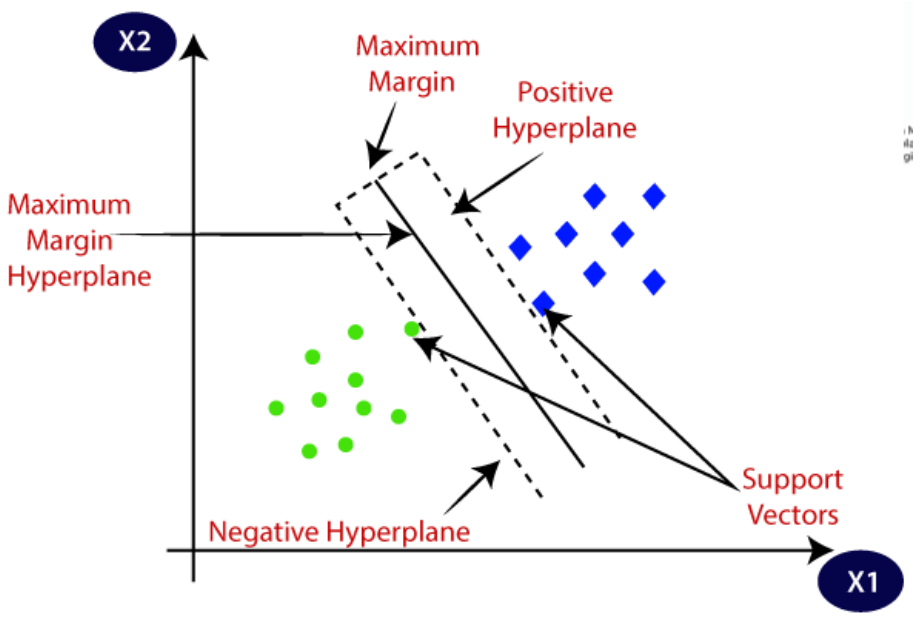



Figura 8. Shembull i SVM Hyperplane

**Shembull**: Supozojmë se shohim një mace të çuditshme që ka edhe disa tipare të qenve, kështu që nëse duam një model që mund të identifikojë me saktësi nëse është mace apo qen, kështu që një model i tillë mund të krijohet duke përdorur algoritmin SVM. Ne së pari do ta trajnojmë modelin tonë me shumë imazhe të maceve dhe qenve, në mënyrë që ai të mësojë për tipare të ndryshme të maceve dhe qenve, dhe pastaj ta testojmë atë me këtë krijesë të çuditshme. Pra, pasi vektori mbështetës krijon një kufi vendimi midis këtyre dy të dhënave (mace dhe qen) dhe zgjidhni raste ekstreme (vektorët mbështetës), do të shohë rastin ekstrem të maceve dhe qenve. Në bazë të vektorëve mbështetës, do ta klasifikojë atë si një mace. Konsideroni diagramin e mëposhtëm:

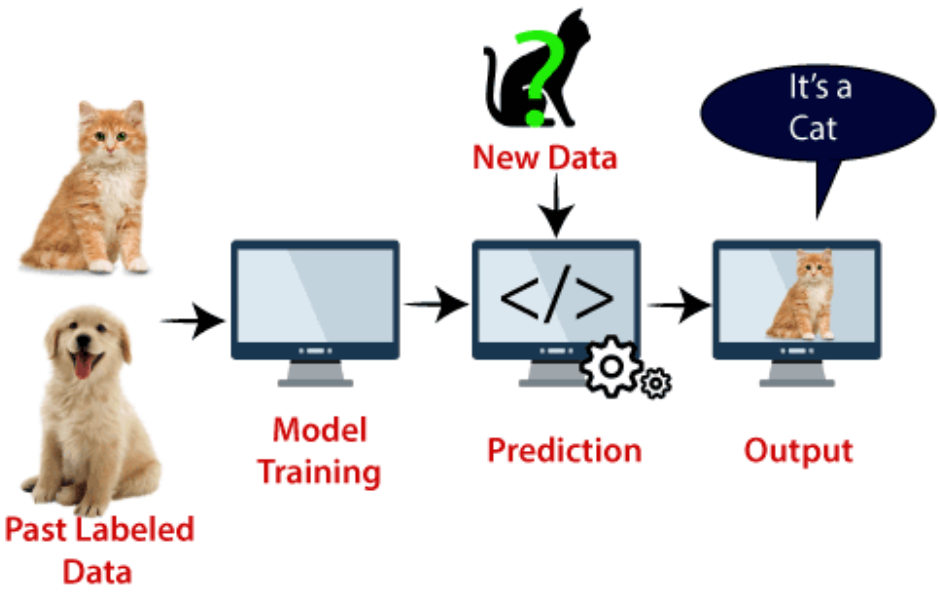



Figura 9. Shembull i modelit trajnues SVM

SVM mund të jetë i dy llojeve:

- Linear SVM: Linear SVM përdoret për të dhëna linearisht të ndara, që do të thotë nëse një e dhënë mund të klasifikohet në dy klasa duke përdorur një linjë të vetme të drejtë, atëherë të dhënat e tilla cilësohen si të dhëna lineare të ndara, dhe klasifikuesi përdoret si klasifikues linear SVM.
- SVM jo-lineare: SVM jo-lineare përdoret për të dhëna jo lineare, që do të thotë nëse një databazë nuk mund të klasifikohet duke përdorur një vijë të drejtë, atëherë të dhënat e tilla cilësohen si të dhëna jo lineare dhe klasifikuesi i përdorur quhet jo-Klasifikuesi linear SVM.

### 2.4.1 Avantazhet dhe disavantazhet e SVM

Disa nga avantazhet e SVM jane:

- Mund te zgjidhe probleme me dimensione te larte, domethene hapesira te medha me tipare
- Mund te zgjidh probleme te ML me Solve machine learning problems me mostra te vogla
- Aftesia per te trajtuar bashkeveprimet jo-lineare te tipareve

- Asnje problem lokal minimal (krahasuar me algoritmet siq jane rrjetet nervore)
- Nuk ka nevoje per tu mbeshtetur ne te gjitha te dhenat
- Aftesi e forte e pergjithesimit

Disa nga disavantazhet e SVM jane:

- Kur kampioni i vëzhgimit është i madh, efikasiteti nuk është shumë i lartë
- Nuk ka zgjidhje universale për problemet jolineare, dhe nganjëherë është e vështirë të gjesh një funksion të përshtatshëm të kernelit
- Fuqia shpjeguese e hartës me dimensione të lartë të funksioneve të kernelit nuk është e fortë, veçanërisht funksionet e bazës radiale
- SVM konvencional mbeshtesin vetem dy klasifkime
- Te ndjeshme ndaj te dhenave qe mungojne

# 3. DEKLARIMI I PROBLEMIT

Në gjuhë të ndryshme posaçerisht në atë angleze përdoren jashtëzakonisht klasifikatorë të ndryshëm për të automatizuar fusha te ndryshme, ndër klasifikuesit që përdoren më së shumti janë ato të tekstit meqenese shumë të dhëna gjenden online të cilat ndihmojne shumë në kursim të kohës së njeriut. Në web-in shqiptar ekzistojnë shumë të dhëna mirëpo fatkeqesisht keto të dhëna nuk klasifikohen me anë të ndonjë algoritmi të machine learning, kjo kryesisht për shkak se gramatika e gjuhës shqipe është e komplikuar dhe kërkohet dedikim nga gjuhëtaret shqiptarë për të ndihmuar në klasifikim të tekstit, për të larguar fjalët që nuk kanë ndonjë ndikim në klasifikim të tekstit. Në kohën e sotme, mund të argumentohet se të dhënat për shkak të depërtimit dhe njohurive që mund të nxjerren prej tij janë potencialisht më të vlefshme sesa çdo gjë tjeter. Synimi i kësaj teme është që të përdoren dy prej algoritmeve më të njohura të klasifikimit, ai i Naive Bayes dhe ai i SVM për klasifikimin e tekstit shqip në një kategori të caktuar, ku do të analizohet saktesia e secilit algoritm, dhe cili mund të hyjë në punë më shumë në të ardhmen në ndonjë projekt ku mund të klasifikohen të dhëna të pastruktuara në gjuhën shqipe.

# 4. METODOLOGJIA

Metodologjia qe jemi duke perdorur në këte temë bazohet në librari qe janë open source dhe që mund të përdoren nga kushdo falas. Ne në këte temë jemi duke perdorur gjuhët programuese C# dhe Java. Për të bërë të mundshme testimin e efikasitetit të dy algoritmeve Naive Bayes dhe SVM neve na duhen të dhëna përkatesisht lajme për trajnim të modelit dhe lajme për testimin e efikasitetit të algoritmeve. Për këte shkak me anë të gjuhës programuese C# ne në fillim do të shkarkojmë një numër lajmesh nga ndonjë portal i caktuar, po i shkarkojmë keto lajme për shkak se po ti shkruanim manualisht do të merrte kohë jashtëzakonisht e gjate. Para se të fillojmë të i shkarkojmë lajmet, ne duhet ti definojmë kategoritë manualisht për të cilat ne do të shkarkojmë lajmet. Për secilën kategori të lajmeve ne marrim një numër të njejtë të lajmeve në mënyre që modeli i stërvitur të jetë sa më i saktë.Në rast se numri i artikujve për kategori është i ndryshëm, modeli i stervitur do të jete goxha i pasakte për ndonjë kategori të caktuar.

Për klasifikimin e lajmeve me algoritëm të Naive Bayes do të përdoret libraria open source në gjuhen programuese Java që quhet Apache Open NLP e cila ka funksione te shumëllojshme të cilat na ndihmojnë në procesimin e gjuhës. Ndërkaq për klasifikimin e lajmeve me SVM do të përdorim librarine LibSVM e cila është shkruar në gjuhën C# dhe na ndihmon ti kategorizojmë lajmet.

# 5. IMPLEMENTIMI

## 5.1 Përshkrimi i aplikacionit

Me anë të ketij aplikacioni synojmë që të kuptojmë se cili algoritëm për klasifikim e bën klasifikimin e lajmeve më saktë në gjuhën shqipe, në menyrë që në të ardhmen ky krahasim i ketyre dy algoritmeve të mund të na ndihmojë në mbledhjen e të gjitha lajmeve në gjuhën shqipe nga të gjitha portalet, domethene do te jete nje lloj platforme ku do te kete lajme te shumta per secilen kategori. Në të ardhmen mund të perdoret njëri prej ketyre algoritmeve, ku mund të i dergojme njoftime lexues-ve për ndonje kategori te caktuar që ai apo ajo është i interesuar.

## 5.2 Algoritmet e perdorura

Algoritmet e përdorura në këtë punim janë:

- Naive Bayes Algoritmi dhe
- Support Vector Machine(SVM)

Këto dy algoritme janë përdorur në testimin e klasifikimit të të dhënave testuese që ne i kemi grumbulluar.

## 5.3 Hapat e aplikacionit

Hapat e aplikacionit janë këto:

- Përcaktimi i kategorive të lajmeve

Para se të fillojmë të bëjme shkarkimin e lajmeve, ne duhet të percaktojme manualisht se cilat kategori do ti ketë klasifikuesi yne për lajme, meqenese jemi duke perdorur Supervised learning. Kemi vendosur qe ti kemi tete kategori ne klasifikues, keto kategori janë: Lajme, Sport, Politike, Ekonomi, Teknologji, Shendetesi, Magazine dhe Auto.

- Shkarkimi i lajmeve

Për të pasur të mundur të testojme saktesinë e algoritmeve, neve na duhen të dhena lajmesh, prandaj duhet të kemi një numër të caktuar lajmesh të cilat do të i shkarkojmë nga ndonjë portal lajmesh, ku një pjesë të tyre do ta përdorim për trajnimin e modeleve, dhe një pjesë tjeter për testimin e saktesisë së algoritmeve. Si fillim, kemi vendosur që të i shkarkojmë rreth 140 lajme për secilën kategori, ku prej tyre 100 prej lajmeve të kategorisë do të perdoren

për trajnimin e modelit, nderkaq 40 prej tyre do të perdoren për analizimin e saktesisë së modeli-t të trajnuar. Eshtë tepër e rendesishme që numri i lajmeve për secilën kategori gjatë trajnimit të modelit të jetë i njejte, përndryshe mund të ndikojë në humbjen e saktesisë së algoritmit. Meqenëse na duhet një numer i konsiderueshëm i lajmeve për trajnim dhe testim, kemi vendosur që të i shkarkojmë këto të dhena nga një portal shqiptar, në këte rast Telegrafi.com meqënese Telegrafi.com është njëri prej portaleve që i ka lajmet e klasifikuara në kategori shumë sakte prandaj e kam zgjedhur këte portal në mënyre që të minimizohen gjasat që të dhënat për trajnim dhe testim të jenë gabim.

```
public Dictionary<string, List<string>> GetClassifierCategories()
{
    var categories = new Dictionary<string, List<string>>();

    categories.Add("Lajme", new List<string>());
    categories.Add("Sport", new List<string>());
    categories.Add("Politike", new List<string>());
    categories.Add("Ekonomi", new List<string>());
    categories.Add("Teknologji", new List<string>());
    categories.Add("Shendetesi", new List<string>());
    categories.Add("Magazina", new List<string>());
    categories.Add("Auto", new List<string>());
    foreach (var newsCategory in categories)
    {
        for (var i = 1; i < 8; i++)
            categories[newsCategory.Key].Add($"https://telegrafi.com/categor
y/{newsCategory.Key}/page/{i}");
    }
    return categories;
}
```

Figura 10. Pjesë e kodit për definimin e kategorive

Në kodin e mësiperm, ne e kemi deklaruar një metodë e cila na kthen kategorite e klasifikuesit. Për të marrur kategoritë, e kemi definuar një Dictionary e cila në gjuhën programuese C# është një lloj fjalori ku ketu i shtojmë vlerat manualisht me kod. Dictionary është një koleksion i çelësave dhe vlerave në gjuhen programuese C#. Kjo metode kthen një

fjalor të kategorive, ku secila kategori permban edhe një listë të fjalëve, fjalët ne kete rast janë link-at ku ne do të bejmë kërkesa për të shkarkuar HTML-n e faqeve që i kemi percaktuar për ti marrur.

Për të bëre të mundshme shkarkimin e lajmeve me kod, neve na duhet një librari e cila na mundeson manipulimin me përmbajtjen e HTML që na kthehet nga faqja pasi të kemi berë kerkesën në web. Librarinë të cilen kemi vendosur ta përdorim është Html AgilityPack e cila është një HTML parser e shkruajtur ne C# për të shkruar/lexuar DOM(document object model), kjo librari deri më tani është shkarkuar më shumë se 29.75 milionë herë.

Për organizim sa më të mire të aplikacionit, kemi krijuar tre klasa ndihmëse në menyrë që kodi të jetë sa më i organizuar dhe i kuptueshëm nga të tjeret.

Klasat ndihmëse qe i kemi krijuar janë:

CrawlerHelper – këte klasë ndihmëse kryesisht e kemi krijuar për shkarkimin e lajmeve, të gjitha metodat që kanë të bëjne me shkarkim dhe manipulim të permbajtes së lajmit i kemi vendosur këtu .

NaiveBayesHelper – këtë klasë ndihmese e kemi krijuar per të gjitha metodat qe kanë te bejnë me perpunimin e te dhenave ne format qe kerkohet nga ana e algoritmit Naive Bayes. Ketu i deklarojmë kategorite sipas formatit të Naive Bayes, poashtu ketu i kemi dy metodat te cilat na ndihmojnë në pergatitjen e të dhenave për trajnim të modelit dhe testim te modelit.

SupportVectorMachineHelper – edhe kjo klase ndihmëse është krijuar për të ndihmuar në përpunimin e të dhënave në formatin që kerkohet nga ana e algoritmit SVM.

```
var crawlerHelper = new CrawlerHelper(_articleLinks, _categories);
```

Figura 11. Pjesë e kodit për inicializim të helperit

Si fillim, e inicializojmë klasën ndihmëse CrawlerHelper ku në konstruktor të saj i dërgojmë dy parametra, njëri prej parametrave është _articleLinks që është një dictionary që në fillim është i zbrazët, dhe parametrin tjetër _categories që është dictionary që përmban kategoritë e klasifikuesit.

Hapi i rradhes është thirrja e metodes StartCrawlingNewsAsync() e cila gjendet ne klasen CrawlerHelper. Brenda kesaj metode fillohet të iterohet për secilën kategori që është definuar

më larte, ku poashtu brenda secilës kategori iterohet edhe për vlerat që i përmban ajo kategori. Secila vlerë e kategorise përmban një link ku me anë të atij linku ne bëjme një kërkese me anë të HttpClient e cila perdoret për të derguar apo pranuar pergjigje në web. Në rastin tonë ne jemi duke bere kerkese per te pranuar nje pergjigje per linkun tonë, webi pastaj na kthen një përgjigje që përmban HTML. Këte HTML që ne e pranojmë e vendosim në një HtmlDocument() që i perket librarise HtmlAgilityPack, dhe pastaj me anë të metodave të librarise ne marrim një listë që i përmbush kushtet që ne i kemi caktuar. Kjo listë na kthen rreth 20 informata, ku pastaj ne fillojme që të iterojme në keto informata për te marre linkun e artikullit për secilen informate ku ne përseri bëjme kerkese në web për të shkarkuar permbajten e secilit artikull.

```
public async Task StartCrawlingNewsAsync()
{
    foreach (var categoryInformation in _categories)
    {
        foreach (var categoryInfo in categoryInformation.Value)
        {
            var httpClient = new HttpClient();
            var html = httpClient.GetStringAsync(categoryInfo).Result;
            var htmlDocument = new HtmlDocument();
            htmlDocument.LoadHtml(html);

            var categoryNews = htmlDocument.DocumentNode.Descendants("div").
Where(node => node.GetAttributeValue("class", "").Contains("col-md-24 cat-
box")).ToList();

            foreach (var category in categoryNews)
            {
                var articleUrl = GetArticleUrl(category.OuterHtml, "href=\""
, "/\"");
                if (!_articleLinks.ContainsKey(categoryInformation.Key))
                    _articleLinks.Add(categoryInformation.Key, new List<News
Model> { new NewsModel { Category = categoryInformation.Key, Link = articleU
rl, Content = await GetArticleContentAsync(articleUrl) } });
                else
                    _articleLinks[categoryInformation.Key].Add(new NewsModel
 { Category = categoryInformation.Key, Link = articleUrl, Content = await Ge
tArticleContentAsync(articleUrl) });
```

```
            }
        }
    }
}
```

Figura 12. Pjese e kodit te shkarkimit te lajmeve

```
public static Dictionary<string, List<NewsModel>> _articleLinks = new Dictio
nary<string, List<NewsModel>>();
```

Figura 13. Pjese e kodit që permban informatat e artikujve

Të gjithë artikujt e shkarkuar janë ruajtur në një dictionary statike që gjendet në klasen CrawlerHelper. Ky dictionary përmban një listë të modeleve të lajmeve që ne ketë rast ne e kemi definuar si NewsModel.

```
public class NewsModel
{
    public string Category { get; set; }
    public string Link { get; set; }
    public string Content { get; set; }
}
```

Figura 14. Pjese e kodit qe permban informatat e lajmit

Modeli NewsModel përmban informatat baze që mund ti ketë një artikull si kategoria, linku si dhe permbajtjen e atij artikulli.

Pasi të ketë perfunduar i gjithë procesi i shkarkimit të lajmeve, meqënese të dhenat e lajmeve i kemi te ruajtura vetëm në dictionary gjatë ekzekutimit te aplikacionit, këto të dhëna do ti ruajmë edhe me një text file, ku e perdorim metodën WriteCrawledNewsInATextFile ku ketu në fillim iteron për lajmet e secilës kategori dhe i shkruajme ato në filen CrawledNews.txt.

```
public void WriteCrawledNewsInATextFile(Dictionary<string, List<NewsModel>>
_articleLinks)
{
    using (StreamWriter writetext = new StreamWriter("CrawledNews.txt"))
```

```
    {
        foreach(var category in _articleLinks)
        {
            foreach(var articleInfo in category.Value)
            {
                writetext.WriteLine($"{articleInfo.Category} {articleInfo.Co
ntent}");
                writetext.Flush();
            }
        }
    }
}
```

Figura 15. Pjese e kodit që i shkruan lajmet në një tekst file

### 5.4 Procesimi i te dhenave

Pasi të jenë shkruar lajmet në tekst file, hapi i radhes që duhet të bëhet është procesimi i të dhënave. Gjëja e parë që duhet të bëjme është që të i gjejmë fjalët që përseriten shpesh në lajmet e shkarkuara. Në metoden e meposhtme, ne po i lexojmë te gjitha lajmet që i kemi shkarkuar një nga një dhe fillojme për secilën fjalë që gjendet në nje lajm të llogarisim se sa here është përseritur ajo fjalë. Në fund të metodës ne i kthejmë vetem fjalet që janë përseritur më shume se 75 herë sepse konsiderojmë se ai numër është i pershtatshëm për të gjetur fjalët që përseriten në 1140 lajme që ne i kemi shkarkuar.

```
public Dictionary<string, int> GetStopWordsFromCrawledNews()
{
    var newsData = File.ReadAllText(@"C:\Users\Myhedin Zika\source\repos\New
sCrawler\NewsCrawler\bin\Debug\CrawledNews.txt");
    string[] lines = newsData.Split(new[] { Environment.NewLine }, StringSpl
itOptions.RemoveEmptyEntries);

    var commonWordsDictionary = new Dictionary<string, int>();

    foreach (var line in lines)
    {
        string[] news = line.Split(' ');
        if (news.Length > 0)
```

```
        {
            var stringDictionary = new Dictionary<string, int>();
            var category = news[0];
            for (var i = 1; i < news.Length; i++)
            {
                if(news[i] != "")
                {
                    if (!stringDictionary.ContainsKey(news[i].ToLower()))
                        stringDictionary.Add(news[i].ToLower(), 1);
                    else
                        stringDictionary[news[i].ToLower()] += 1;
                }
            }

            foreach (var comm in stringDictionary)
            {
                if (!commonWordsDictionary.ContainsKey(comm.Key))
                    commonWordsDictionary.Add(comm.Key, comm.Value);
                else
                {
                    commonWordsDictionary[comm.Key] += comm.Value;
                }
            }
        }
    }
    return commonWordsDictionary.Where(x => x.Value > 75).ToDictionary(x=> x
.Key, y => y.Value);
}
```

Figura 16. Pjese e kodit per gjetjen e fjaleve qe perseriten me se shumti

Pasi ti kemi gjetur fjalët që përseriten më së shumti, ato fjalë i shkruajme në një file të quajtur CommonWordsFromNews.txt.

```
në
ka
se
e
```

```
saj
dhe
ende
nuk
është
të
mos
nga
sipas
duke
u
```

Figura 17. Disa nga fjalët që përseriten shpesh në një lajm dhe që nuk ndikojnë në klasifikim të lajmit

Përveq kësaj ne kemi edhe fjalë kyçe për secilen kategori e cila na ndihmon në klasifikim më të saktë të lajmeve. Disa nga këto fjale kyçe prej të dhenave që ne kemi mbledhur janë:
Për shembull po i listojmë fjalët kyçe për disa nga kategoritë:
Kategoria politikë fjalët kyçe ketu janë kryetari, kryeministri, formimi, qeveria, partia, presidenti, albin kurti, zgjedhje, detyrë etj.
Kategoria sport fjalë kyçe këtu janë: klubi, shenuar, ndeshjet, ligës, lojtarëet, gola, trajneri, barcelona, real madrid etj.
Kategoria ekonomi fjalë kyçe këtu janë biznese, rënie, muaji, ekonomik, masa, euro, pako, ekonomik etj.
Kategoria teknologji fjalë kyçe këtu janë samsung, galaxy, hauwei, telefoni, facebook, apple, google, përdoruesit, ekran etj.

### 5.5 Transformimi i të dhënave në format të Naive Bayes

Para se te fillojme të bëjmë trajnimin e modelit te Naive Bayes, po e shpegojme nje shembull se si duhet te jene te dhenat e organizuara ne formatin qe e kerkon algoritmi i Naive Bayes. Të dhenat në format të Naive Bayes gjithmonë në fillim të tekstit shkruhet kategoria e tekstit, mandej shkruhet pjesa e tjeter e tekstit më saktesisht permbajtja e tekstit.
E marrim një shembull:

**Lajme** Ankuesja femër në Pejë ka raportuar se vajza e saj ka shkuar në shkollë në mëngjes dhe ende nuk është kthyer. E njëjta deklaron se asnjëherë nuk ka ndodh qe viktima të mos kthehet nga shkolla. Sipas policisë, rasti është duke u hetuar. Po ashtu, një vajzë nga Vushtrria është larguar nga shtëpia dhe nuk është kthyer më. Rasti e ka raportor në Polici, babai i vajzës. Ndërkohë, njësia përkatëse policore është duke e hetuar rastin.

Siç po e vereni, në fillim e kemi shkruajtur kategorinë e cila në ketë rast është Lajme, pas kategorise e kemi shkruar edhe permbajtjen e lajmit. Mirepo siç mund të verehet, ka fjalë që janë duke u përseritur mirëpo që nuk kanë ndonjë ndikim në klasifikim të lajmit, prandaj do ti largojmë ato fjalë me anë të metodës së meposhtme e cila bën procesimin e të dhënave.

```
public Dictionary<string, List<string>> PreProcessData()
{
    var newsData = File.ReadAllText(@"C:\Users\Myhedin Zika\source\repos\New
sCrawler\NewsCrawler\bin\Debug\CrawledNews.txt");
    string[] news = newsData.Split(new[] { Environment.NewLine }, StringSpli
tOptions.RemoveEmptyEntries);

    var commonWordsData = File.ReadAllText(@"C:\Users\Myhedin Zika\source\re
pos\NewsCrawler\NewsCrawler\bin\Debug\CommonWordsFromNews.txt");
    string[] commonWords = commonWordsData.Split(new[] { Environment.NewLine
 }, StringSplitOptions.RemoveEmptyEntries);

    var commonWordsDictionary = new Dictionary<string, string>();
    foreach (var commonWord in commonWords)
    {
        if (!commonWordsDictionary.ContainsKey(commonWord))
            commonWordsDictionary.Add(commonWord, commonWord);
    }

    var proccessedData = RemoveStopWords(news, commonWordsDictionary);

    return proccessedData;
}
```

Figura 18. Pjesa e kodit ku procesohen të dhënat

Me anë të kësaj metode ne e bëjmë procesimin e lajmeve, këto lajme ne i shkruajmë në një text file ProccessedNews.txt. Po i shkruajmë komplet lajmet e procesuara këtu pa i ndarë ato në lajmet për trajnim dhe testim te modelit për shkak se keto lajme te procesuara do ti

përdorim më vone edhe për transformimin e të dhënave në formatin siç e kërkon algoritmi SVM.

Pas procesimit të të dhenave, shembulli i mëparshem do të transformohet kështu:

**Lajme** ankuesja femër pejë raportuar vajza shkuar shkollë mëngjes kthyer. e njëjta deklaron asnjëherë ndodh qe viktima kthehet shkolla. sipas policisë, rasti hetuar. po ashtu, vajzë vushtrria larguar shtëpia kthyer më. rasti raportor polici, babai vajzës. ndërkohë, njësia përkatëse policore hetuar rastin.

```
public void WriteProccessedData(Dictionary<string, List<string>> proccessedD
ata)
{
    using (StreamWriter writetext = new StreamWriter("ProccesedData.txt"))
    {
        foreach (var proccessedNews in proccessedData)
        {
            foreach(var news in proccessedNews.Value)
            {
                writetext.WriteLine($"{proccessedNews.Key} {news}");
                writetext.Flush();
            }
        }
    }
}
```

Figura 19. Pjesa e kodit ku shkruhen të dhënat e procesuara ne nje text file

```
public void WriteTrainingDataset(Dictionary<string, List<string>> processed
NewsDic)
{
    using (StreamWriter writetext = new StreamWriter("NewsTrainingDataset.tx
t"))
    {
        foreach (var proccessedNews in proccessedNewsDic)
        {
            for (var i = 0; i < 100; i++)
            {
                writetext.WriteLine($"{proccessedNews.Key} {proccessedNews.V
alue[i]}");
                writetext.Flush();
```

```
            }
        }
    }
}
```

Figura 20. Pjesa e kodit ku shkruhen të dhënat për trajnim të modelit

Në metoden e mësiperme po i shkruajmë lajmet e procesuara, ku po i shkruajmë 100 lajmet e para për secilën kategori ku këto lajme do të përdoren për trajnim të modelit.

```
public void WriteTestingDataSet(Dictionary<string, List<string>> proccessedN
ewsDic)
{
    using (StreamWriter writetext = new StreamWriter("NewsTestingDataset.txt
"))
    {
        foreach (var proccessedNews in proccessedNewsDic)
        {
            for (var i = 100; i < 140; i++)
            {
                writetext.WriteLine($"{proccessedNews.Key} {proccessedNews.V
alue[i]}");
                writetext.Flush();
            }

        }
    }
}
```

Figura 21. Pjesa e kodit ku shkruhen të dhënat për testim të modelit

Nderkaq ketu po i marrim prej secilës kategori 40 lajme të cilat do të përdoren për testimin e saktesisë së algoritmit te Naive Bayes.

### 5.6 Transformi i te dhenave ne format te SVM

Transformimi i te dhenave në format të SVM-s është pak më ndryshe nga Naive Bayes. Në menyre që të bëhet trajnimi i modelit për SVM, në fillim duhet ti pergatisim të dhenat ku

duhet ti etiketojmë të dhënat, të gjenerojmë një fjalor, dhe në fund të gjenerojmë një matricë document-term. Ketu të dhenat duhet të transformohen nga teksti në numër për shkak se ne atë formë funksionon algoritmi i SVM-së.

Për këte së pari duhet të gjenerohet fjalori, i cili është shkruar më poshtë. Secila fjale që është unike shkruhet në këte fjalor. Secila fjalë gjendet në një rresht të caktuar.

```
public void PrepareVocabulary()
{
    var newsData = File.ReadAllText(@"C:\Users\Myhedin Zika\source\repos\New
sCrawler\NewsCrawler\bin\Debug\ProccesedData.txt");
    string[] lines = newsData.Split(new[] { Environment.NewLine }, StringSpl
itOptions.RemoveEmptyEntries);

    var stringDictionary = new Dictionary<string, int>();
    foreach (var line in lines)
    {
        string[] news = line.Split(' ');
        if (news.Length > 0)
        {
            var category = news[0];
            for (var i = 1; i < news.Length; i++)
            {
                if (!stringDictionary.ContainsKey(news[i].ToLower()))
                    stringDictionary.Add(news[i].ToLower(), 1);
                else
                    stringDictionary[news[i].ToLower()] += 1;
            }
        }
    }

    using (StreamWriter writetext = new StreamWriter("SvmVocabulary.txt"))
    {
        foreach (var item in stringDictionary.OrderBy(x => x.Key))
        {
            writetext.WriteLine(item.Key);
            writetext.Flush();
        }
    }
}
```

Figura 22. Pjesa e kodit ku gjenerohet fjalori

Po e marrim shembullin e njejtë si tek Naive Bayes, në menyrë që të tregojmë dallimin në mes te Naive Bayes dhe SVM.

Shembull:

Lajme ankuesja femër pejë raportuar vajza shkuar shkollë mëngjes kthyer. e njëjta deklaron asnjëherë ndodh qe viktima kthehet shkolla. sipas policisë, rasti hetuar. po ashtu, vajzë vushtrria larguar shtëpia kthyer më. rasti raportor polici, babai vajzës. ndërkohë, njësia përkatëse policore hetuar rastin

Në format të SVM-s lajmi i mësiperm transformohet në këtë menyrë

1 5211:1 12544:1 28865:1 32925:1 40886:1 35956:1 35888:1 23854:1 20584:1 11148:1 26761:1 9284:1 5884:1 26035:1 32334:1 41826:1 20545:1 35886:1 36968:1 30688:1 32960:2 15649:1 30632:1 5835:1 40891:1 42390:1 21288:1 36508:1 20582:1 23602:1 32921:1 30681:1 6272:1 40895:1 25732:1 26860:1 29550:1 30695:1 15647:1 32964:1

Ku në këte rast numri 1 tregon që i përket kategorisë lajme. Ndërkaq numri 5211 tregon rreshtin ku gjendet fjala ndërsa :1 tregon që kjo fjale eshte perseritur vetëm nje herë brenda atij lajmi.

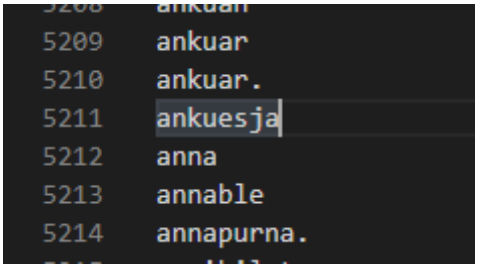



Figura 23. Pjesë e fjalorit të gjeneruar për SVM

Në këte mënyrë fillojmë ti transformojmë të gjitha të dhënat për trajnim dhe testim të modelit në format që nevojitet për SVM.

```
public void PreProcessTestingDataset(Dictionary<string, int> vocabularyDicti
onary, Dictionary<string, int> _svmCategories)
{
    var newsData = File.ReadAllText(@"C:\Users\Myhedin Zika\source\repos\New
sCrawler\NewsCrawler\bin\Debug\NewsTestingDataset.txt");
```

```
    string[] lines = newsData.Split(new[] { Environment.NewLine }, StringSpl
itOptions.RemoveEmptyEntries);

    List<string> transformedText = new List<string>();

    foreach (var line in lines)
    {
        string[] news = line.Split(' ');
        if (news.Length > 0)
        {
            var category = news[0];
            string transformedTextLine = _svmCategories[category].ToString()
 + ",  ";
            var stringDictionary = new Dictionary<string, int>();
            for (var i = 1; i < news.Length; i++)
            {

                if (!stringDictionary.ContainsKey(vocabularyDictionary[news[
i]].ToString()))
                    stringDictionary.Add(vocabularyDictionary[news[i]].ToStr
ing(), 1);
                else
                    stringDictionary[vocabularyDictionary[news[i]].ToString(
)] += 1;
            }

            foreach (var item in stringDictionary)
            {
                transformedTextLine += " " + item.Key + ":" + item.Value;
            }
            transformedText.Add(transformedTextLine);
        }
    }

    WriteSVMDatasetToCsv(transformedText, "SVMTransformTestingDataset.csv");
}
```

Figura 24. Transformimi i të dhënave në format të SVM

```
private void WriteSVMDatasetToCsv(List<string> transformedText, string fileN
ame)
{
```

```
    File.WriteAllLines(fileName, transformedText.Select(x => string.Join(","
, x)));
}
```

Figura 25. Ruajtja e të dhënave të SVM në një file CSV

Në kodin e mesipërm të gjitha fjalite po i shkruajme ne nje file CSV qe do te mund te perdoren nga ana e klasifikatorit te SVM.

### 5.7 Trajnimi dhe perdorimi i modelit me Naive Bayes

Për trajnim dhe testim të algoritmit Naive Bayes, do të përdorim një librari open source Apache Open NLP e cila na ndihmon me klasifikimin e lajmeve me SVM. Kjo librari është shkruar në gjuhën programuese Java prandaj kodin për trajnim dhe testim të algoritmit Naive Bayes po e shkruajmë në gjuhen Java.

Në fillim i importojmë librarite që na duhen për përdorimin e algoritmit Naive Bayes

```
import java.io.BufferedOutputStream;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileReader;
import java.io.IOException;
import java.util.HashMap;
import java.util.Map;
import opennlp.tools.doccat.DoccatFactory;
import opennlp.tools.doccat.DoccatModel;
import opennlp.tools.doccat.DocumentCategorizer;
import opennlp.tools.doccat.DocumentCategorizerME;
import opennlp.tools.doccat.DocumentSampleStream;
import opennlp.tools.ml.AbstractTrainer;
import opennlp.tools.ml.naivebayes.NaiveBayesTrainer;
import opennlp.tools.util.InputStreamFactory;
import opennlp.tools.util.MarkableFileInputStreamFactory;
import opennlp.tools.util.ObjectStream;
import opennlp.tools.util.PlainTextByLineStream;
import opennlp.tools.util.TrainingParameters;
```

Figura 26. Importimi i librarive në Java

Këtu i kemi importuar librarite që do ti përdorim, ku theks te veçante kanë klasat që gjenden në librarinë opennlp.

```
            InputStreamFactory dataIn = new MarkableFileInputStreamFactory(n
ew File("C:\\Users\\Myhedin Zika\\Documents\\NetBeansProjects\\AlbanianNewsC
lasssifier\\Data\\" + "NewsTrainingDataset.train"));

            ObjectStream lineStream = new PlainTextByLineStream(dataIn, "UTF
-8");
            ObjectStream sampleStream = new DocumentSampleStream(lineStream)
;

            // define the training parameters
            TrainingParameters params = new TrainingParameters();
            params.put(TrainingParameters.ITERATIONS_PARAM, 100 + "");
            params.put(TrainingParameters.CUTOFF_PARAM, 0 + "");
            params.put(AbstractTrainer.ALGORITHM_PARAM, NaiveBayesTrainer.NA
IVE_BAYES_VALUE);

            // create a model from training data
            DoccatModel model = DocumentCategorizerME.train("sq", sampleStre
am, params, new DoccatFactory());

            // save the model to local
            BufferedOutputStream modelOut = new BufferedOutputStream(new Fil
eOutputStream("C:\\Users\\Myhedin Zika\\Documents\\NetBeansProjects\\Albania
nNewsClasssifier\\Data\\" + "model-classifier-naive-bayes.bin"));
            model.serialize(modelOut);

            DocumentCategorizer doccat = new DocumentCategorizerME(model); /
/the real NLP Model
```

Figura 27. Trajnimi i modelit me Naive Bayes

Në kodin e mësiperm, po i lexojmë të dhënat nga file NewsTrainingDataset.train ku këtu gjenden lajmet që do ti përdorim për të trajnuar modelin.

Pastaj i definojmë parametrat për trajnim ku aty i caktojmë disa tipare ku caktojme në një prej parametrave që dëshirojmë të ushtrojmë modelin me algoritmin e Naive Bayes.

Pastaj e krijojmë modelin me të dhenat për trajnim, ku në fund e krijojmë një DocumentCategorizer nga modeli i trajnuar i cili do të përdoret nga ne për testimin e klasifikuesit.

```
 BufferedReader br = new BufferedReader(new FileReader("C:\\Users\\Myhedin Z
ika\\Documents\\NetBeansProjects\\AlbanianNewsClasssifier\\Data\\" + "NewsTe
stingDataset.txt"));
            String line;

            Map<String,Map<String, Integer>> predictedCategories = new HashM
ap<>();

            while ((line = br.readLine()) != null) {
                // process the line.
                String newsInfo[] = line.split(" ", 2);

                if(!predictedCategories.containsKey(newsInfo[0]))
                    predictedCategories.put(newsInfo[0], new HashMap<>());

                String[] docWords = newsInfo[1].replaceAll("[^A-Za-
z]", " ").split(" ");
                double[] aProbs = doccat.categorize(docWords);
                Category = doccat.getBestCategory(aProbs);

                if(!predictedCategories.get(newsInfo[0]).containsKey(Categor
y)){
                    predictedCategories.get(newsInfo[0]).put(Category, 0);
                }

                Integer data = predictedCategories.get(newsInfo[0]).get(Cate
gory);
                predictedCategories.get(newsInfo[0]).put(Category, data+1);

            }

            for(String i : predictedCategories.keySet()){
                System.out.println(predictedCategories.get(i));
            }
```

Figura 28. Testimi i modelit me Naive Bayes

Në mënyre që të mos e testojmë secilin lajm manualisht për kategorinë e saj, po i lexojmë të dhënat për testim të modelit nga një file, ku pastaj me anë të një iterimi për secilin rresht po e testojme klasifikuesin. E kemi deklaruar një HashMap ku në fillim ruhet kategoria, nderkaq

pastaj për kategorine e caktuar shtohet kategoria e cila është predikuar nga klasifikuesi dhe sa herë e ka predikuar algoritmi atë kategori.

### 5.8 Trajnimi dhe perdorimi i modelit me SVM

Për trajnim të modelit me SVM do të përdorim librarine libsvm e cila është shkruar në gjuhën programuese C#. Se pari e lexojme filen-n e të dhënave për trajnim që është shkruar në formatin .csv

```
const string trainingDatasetFilePath = @"C:\Users\Myhedin Zika\source\repos\
NewsCrawler\NewsCrawler\bin\Debug\SVMTransformTrainingDataset.csv";
var dataTable = DataTable.New.ReadCsv(trainingDatasetFilePath);
List<string> contentRows = dataTable.Rows.Select(row => row["Content"]).ToLi
st();

var vocabulary = contentRows.SelectMany(GetWords).Distinct().OrderBy(word =>
 word).ToList();

var categoryRows = dataTable.Rows.Select(row => double.Parse(row["Category"]
)).ToArray();
```

Figura 29. Pjese e kodit për leximin e të dhënave për trajnim me SVM

Pastaj këte file e shnderrojmë në datatable ku i marrim veçmas përmbajtjen e lajmeve, dhe veçmas përmbajtjen e kategorive.

```
var problemBuilder = new TextClassificationProblemBuilder();
var problem = problemBuilder.CreateProblem(contentRows, categoryRows, vocabu
lary.ToList());
```

```
const int C = 1;
var model = new C_SVC(problem, KernelHelper.LinearKernel(), C);

var accuracy = model.GetCrossValidationAccuracy(10);
Console.Clear();
Console.WriteLine("Accuracy of the model is {0:P}", accuracy);
model.Export(string.Format(@"C:\Users\Myhedin Zika\source\repos\AlbanianNews
ClassifyingSVM\AlbanianNewsClassifyingSVM\bin\Debug\model_{0}_accuracy.model
", accuracy));
```

Figura 28: Pjese e kodit për trajnimin e modelit me SVM

Këtu e inicializojmë një TextCassifcationProblemBuilder() ku nga këtu e thirrim metodën CreateProblem e cila i dërgon si parametra përmbajtjen e lajmeve, kategoritë dhe fjalorin e gjeneruar.

```
public class TextClassificationProblemBuilder
{
    public svm_problem CreateProblem(IEnumerable<string> x, double[] y, IRea
dOnlyList<string> vocabulary)
    {
        return new svm_problem
        {
            y = y,
            x = x.Select(xVector => CreateNode(xVector, vocabulary)).ToArray
(),
            l = y.Length
        };
    }

    public static svm_node[] CreateNode(string x, IReadOnlyList<string> voca
bulary)
    {
        var node = new List<svm_node>(vocabulary.Count);

        string[] words = x.Split(new[] { ' ', '\t' }, StringSplitOptions.Rem
oveEmptyEntries);

        for (int i = 0; i < vocabulary.Count; i++)
```

```
        {
            int occurenceCount = words.Count(s => String.Equals(s, vocabular
y[i], StringComparison.OrdinalIgnoreCase));
            if (occurenceCount == 0)
                continue;

            node.Add(new svm_node
            {
                index = i + 1,
                value = occurenceCount
            });
        }

        return node.ToArray();
    }
}
```

Figura 30. Pjese e kodit për krijimin e problemeve te klasifikimit

Në kodin e mesipërm kthehet nje svm_problem i cili gjenerohet nga libraria libsvm. Poashtu në këte klase e kemi metodën CreateNode e cila përdoret për të krijuar një varg të node-ve që perdoren për klasifikimin e nje lajmi.

Pastaj e krijojmë një dictionary e cila përmban kategorite të cilat mund të predikohen, dhe pastaj e marrim file-n për testim, dhe me anë të një iterimi e testojmë klasifikuesin.

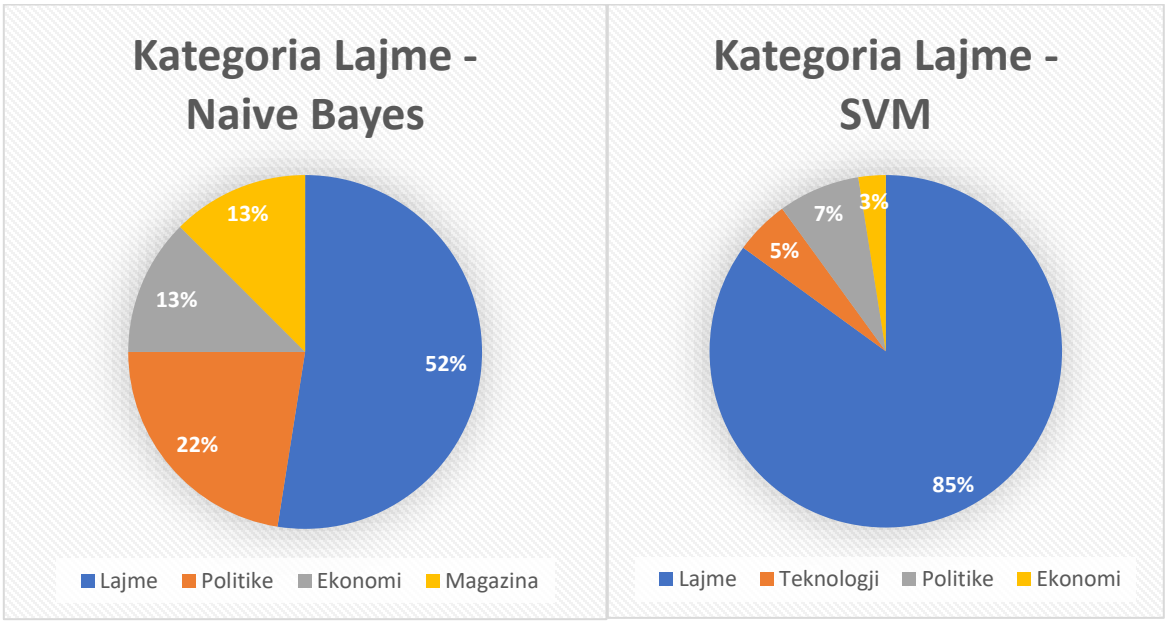
```
_predictionDictionary = new Dictionary<int, string> { { 1, "Lajme" }, { 2, "
Sport" }, { 3,"Politike"}, { 4, "Ekonomi"}, {5,"Teknologji" },{6, "Shendetes
i" },{7 , "Magazina" },{8, "Auto" } };


const string testDatasetFilePath = @"C:\Users\Myhedin Zika\source\repos\News
Crawler\NewsCrawler\bin\Debug\SVMTransformTestingDataset.csv";
var testDatasetDatatable = DataTable.New.ReadCsv(testDatasetFilePath);
var testData = testDatasetDatatable.Rows.GroupBy(r => r.Values[0]).
                    ToDictionary(g => g.Key,
                        g => g.Select(r => r.Values[1])).ToList();

var predictedCategories = new Dictionary<string, Dictionary<string, int>>();
```

```
foreach (var datasetData in testData)
{

    if (!predictedCategories.ContainsKey(datasetData.Key))
        predictedCategories.Add(datasetData.Key, new Dictionary<string, int>
());
    foreach (var item in datasetData.Value)
    {
        var newX = TextClassificationProblemBuilder.CreateNode(item, vocabul
ary);

        var predictedY = model.Predict(newX);

        if (!predictedCategories[datasetData.Key].ContainsKey(_predictionDic
tionary[(int)predictedY]))
            predictedCategories[datasetData.Key].Add(_predictionDictionary[(
int)predictedY], 0);
        predictedCategories[datasetData.Key][_predictionDictionary[(int)pred
ictedY]] += 1;
    }
}
```

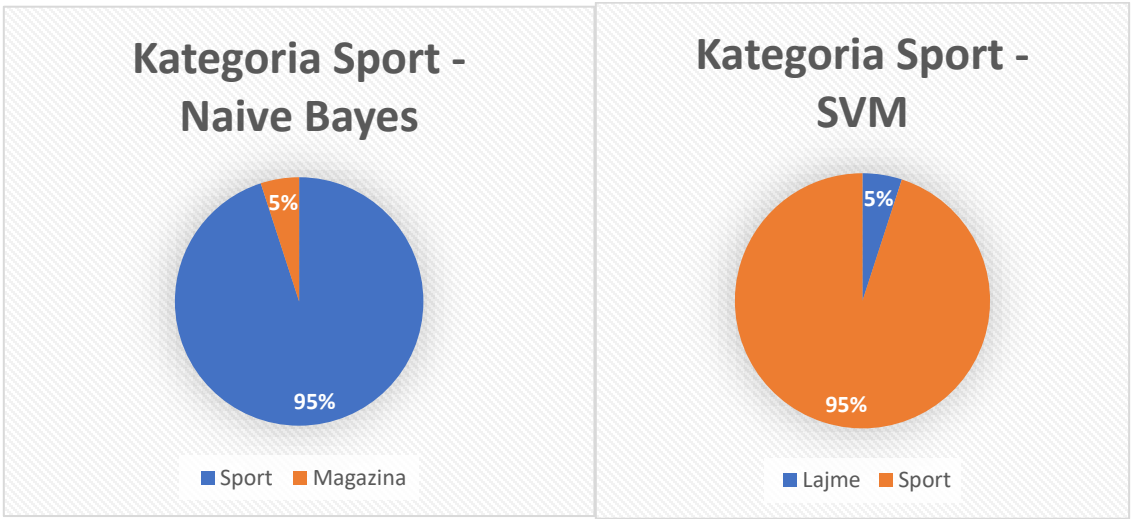
Figura 31. Pjese e kodit për predikimin e kategorise se lajmeve me anë të SVM

Në kodin e mësiperm nëpermjet metodes model.Predict(newX) na kthehet rezultati I kategorisë së predikuar nga modeli e cila është numër, mirepo ne pastaj këte numer e konvertojmë në tekst. Të gjitha rezultatet e predikuara i kemi ruajtur në një dictionary në menyrë që të mos i klasifikojmë lajmet një nga një e cila është një punë manuale që merr kohë.

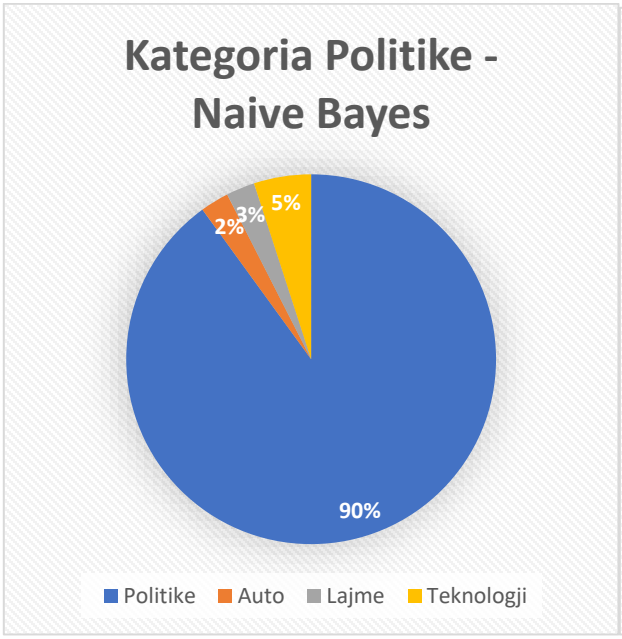
# 6. REZULTATET

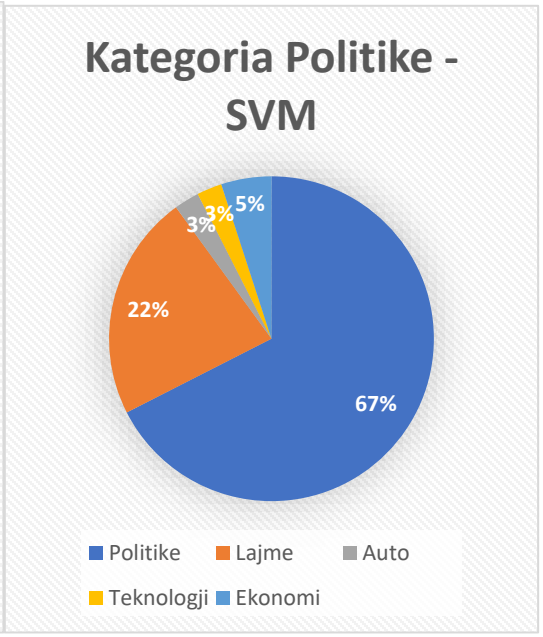



Sipas algoritmit të Naive Bayes nga të dhënat për testim për kategorinë lajme algoritmi ka qenë 52% i sakte. Ndërkaq me të dhënat e njëjta mirepo në formatin e SVM, algoritmi i SVM ka qenë i sakte 85%, pra një dallim prej 33%.

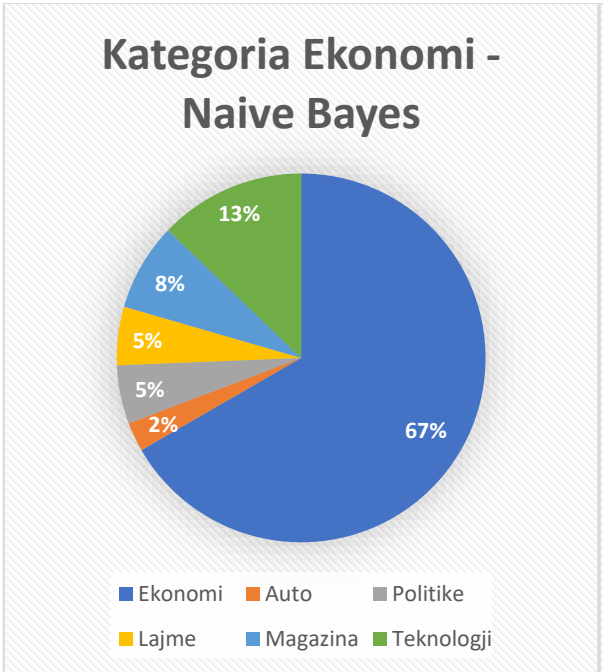


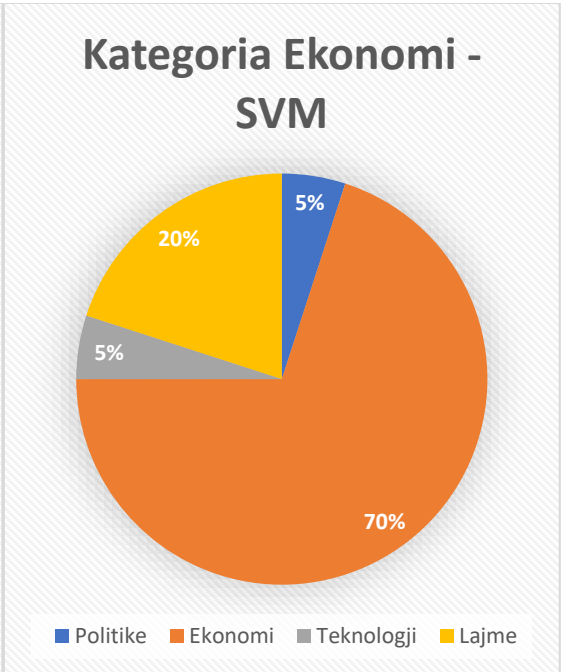

Në kategorinë Sport, algoritmi i Naive Bayes ka qenë i sakte 95%. Njejtë poashtu edhe për algoritmin SVM. Dallimi i vetëm që është verejtur është që tek 5% tjetër i predikuar, secili algoritm ka predikuar kategori të ndryshme njëra nga tjetra.



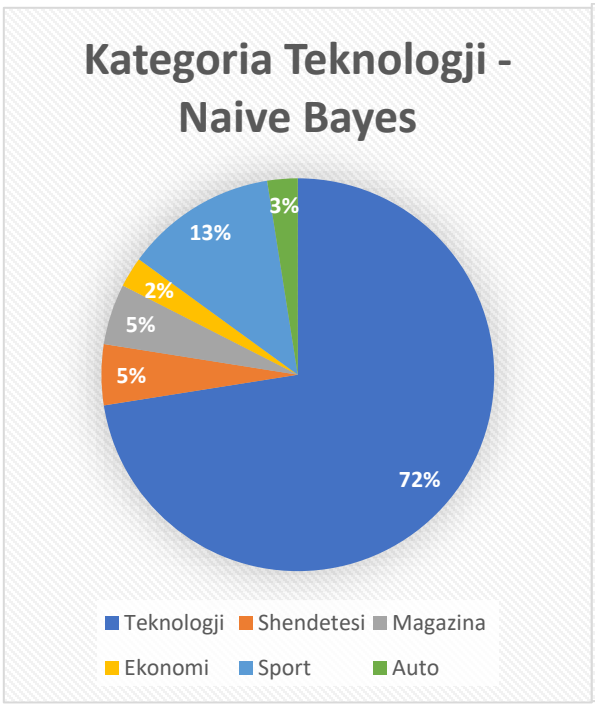



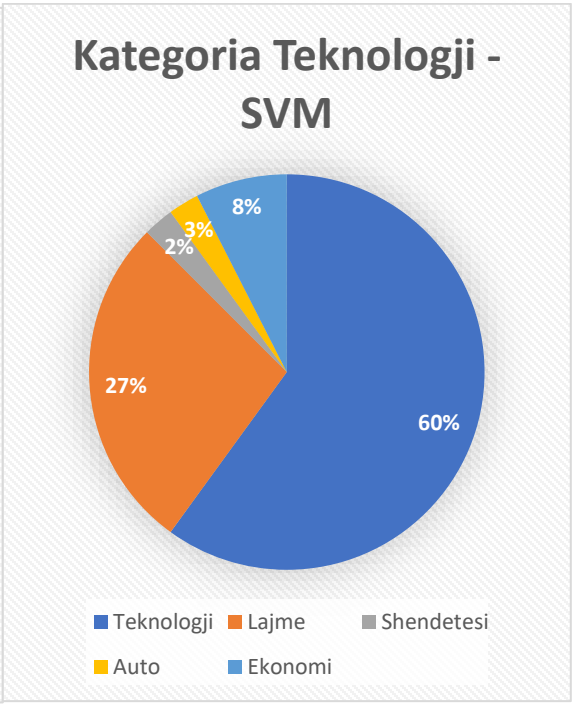

Në kategorinë Politikë, algoritmi i Naive Bayes ka qenë i saktë 90% në predikim të kategorisë. Nderkaq algoritmi i SVM ka qenë i sakte vetëm 67%.



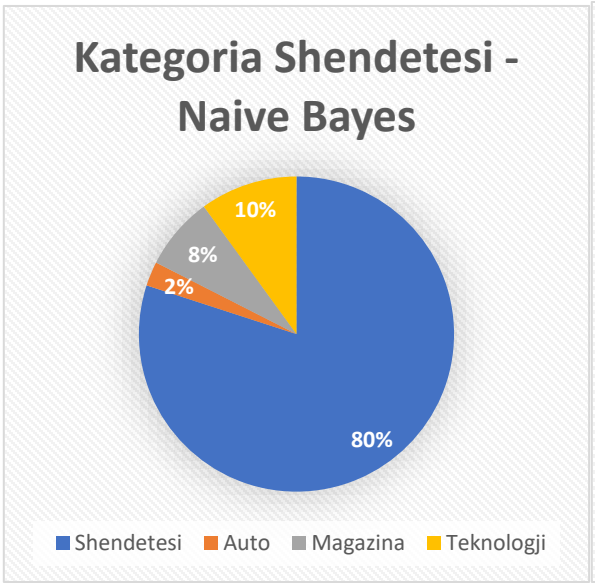



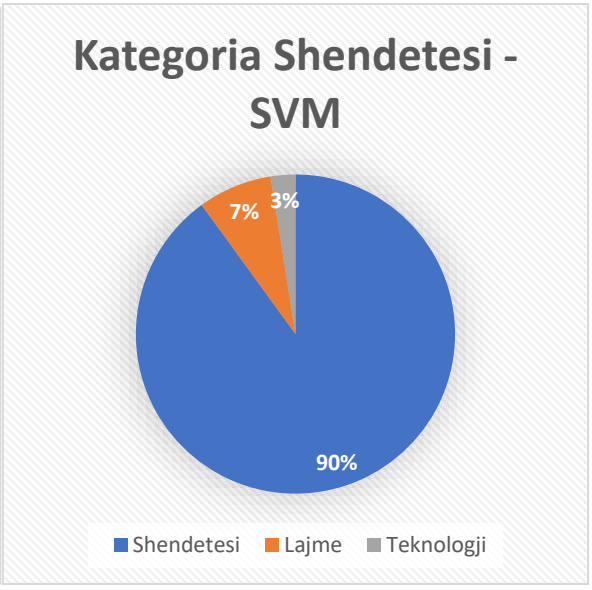

Tek kategoria Ekonomi, algoritmi i Naive Bayes ka qenë i sakte rreth 67%, vlen të theksohet se në ketë kategori janë predikuar edhe 5 kategori tjera nga ana e Naive Bayes të cilat kanë përqindje shumë të afert njëra me tjetrën. Nderkaq algoritmi i SVM ka qene i saktë 70%.



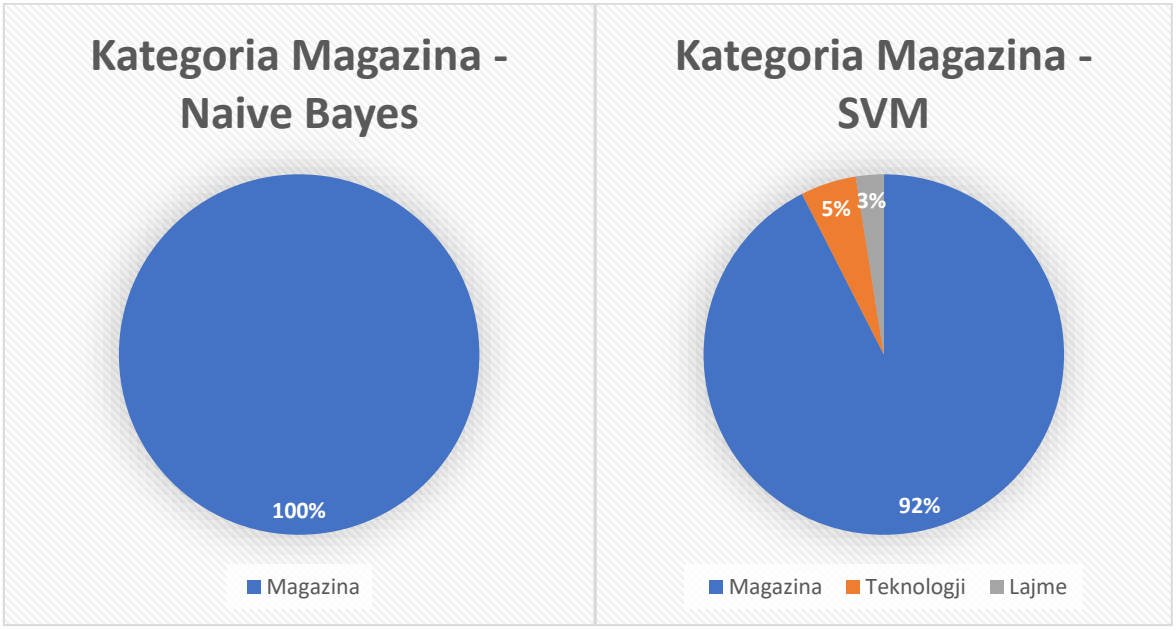




Tek kategoria Teknologjia, algoritmi i Naive Bayes ka pasur një saktësi prej 72%, ndërkaq algoritmi i SVM-s ka pasur një saktësi prej 60%. Tek algoritmi i SVM-së, kategoria që eshte predikuar më së shumti pas Teknologjise ka qenë kategoria Lajme me 27% që është një vlere goxha e madhe.

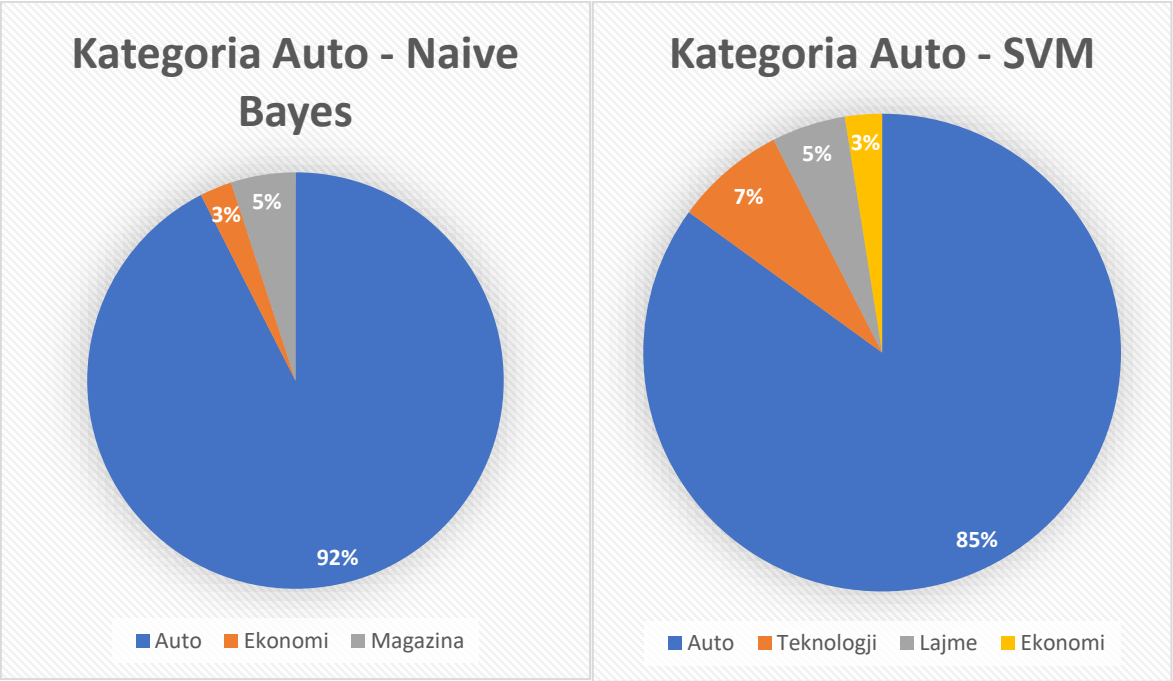


Tek kategoria Shëndetesi, algoritmi i Naive Bayes ka pasur një saktësi prej 80%,  ndërkaq ai i SVM-së një saktësi prej 90%.




Tek kategoria Magazina, algoritmi i Naive Bayes ka pasur një saktesi prej 100%, nderkaq algoritmi i SVM-së ka pasur një saktësi prej 92%.




Tek kategoria Auto, algoritmi i Naive Bayes ka pasur një saktësi prej 92%, ndërkaq ai i SVM-së një saktësi prej 85%.

| Kategoria | Lajme | Sport | Politikë | Ekonomi | Teknologji | Shendetesi | Magazina | Auto |
|---|---|---|---|---|---|---|---|---|
| Lajme | 21 | 0 | 9 | 5 | 0 | 0 | 5 | 0 |
| Sport | 0 | 38 | 0 | 0 | 0 | 0 | 2 | 0 |
| Politike | 1 | 0 | 36 | 0 | 2 | 0 | 0 | 1 |
| Ekonomi | 2 | 0 | 2 | 26 | 5 | 0 | 3 | 1 |
| Teknologji | 0 | 5 | 0 | 1 | 29 | 2 | 2 | 1 |
| Shendetesi | 0 | 0 | 0 | 0 | 4 | 32 | 3 | 1 |
| Magazina | 0 | 0 | 0 | 0 | 0 | 0 | 40 | 0 |
| Auto | 0 | 0 | 0 | 1 | 0 | 0 | 2 | 37 |

Tabela 6. Rezultatet e predikuara per secilen kategori me anë të algoritmit Naive Bayes

| Kategoria | Lajme | Sport | Politikë | Ekonomi | Teknologji | Shendetesi | Magazina | Auto |
|---|---|---|---|---|---|---|---|---|
| Lajme | 34 | 0 | 3 | 1 | 2 | 0 | 0 | 0 |
| Sport | 2 | 38 | 0 | 0 | 0 | 0 | 0 | 0 |
| Politike | 9 | 0 | 27 | 2 | 1 | 0 | 0 | 1 |
| Ekonomi | 8 | 0 | 2 | 28 | 2 | 0 | 0 | 0 |
| Teknologji | 11 | 0 | 0 | 3 | 24 | 1 | 0 | 1 |
| Shendetesi | 3 | 0 | 0 | 0 | 1 | 36 | 0 | 0 |
| Magazina | 1 | 0 | 0 | 0 | 2 | 0 | 37 | 0 |
| Auto | 2 | 0 | 0 | 1 | 3 | 0 | 0 | 34 |

Tabela 7. Rezultatet e predikuara per secilen kategori me anë të algoritmit SVM

Në tabelat e mësiperme për secilin algoritëm i kemi shfaqur rezultatet e parashikimit te kategorive, në rreshta i kemi vendosur secilen kategori me të dhenat per trajnim, nderkaq në kolona jane rezultatet e predikuara per secilen kategori.

# 7. DISKUTIME DHE PERFUNDIME

Në bazë të rezultateve, e kemi verejtur që në shumicën e rasteve algoritmi i Naive Bayes dhe ai i SVM-së kanë qenë goxha të saktë, dhe të afërt njëri me tjetrin, ka pasur disa dallime në disa raste mirëpo kryesisht të dy algoritmet kanë shfaqur saktesi të madhe. Një ndër arsyet kryesore pse ato ndonjehere nuk kanë qenë të sakte ka qene se te dhenat per trajnim te modelit nuk kane qene te shumta.

Gjate ketij hulumtimi e kemi verejtur qe algoritmi i Naive Bayes ne rastet kur nuk e qellon sakte kategorine, % e kategorive tjera te predikuara jane te peraferta njera me tjetren. Nderkaq ne rastin e algoritmit te SVM-se, ne baze te testimeve e kemi verejtur qe zakonisht kur kategoria nuk eshte predikuar sakte, kategoria qe eshte predikuar e dyta me se shumti pas kategorise se sakte ka pasur gjithmone nje % me te larte ne krahasim me kategorite tjera qe jane predikuar gabimisht. Mirepo perseri vlen te ceket se te dy algoritmet kane nje saktesi goxha te madhe, dhe mund te implementohen ne klasifikim te tekstit ne gjuhen shqipe. Keto algoritme mund te perdoren ne situata reale dhe te funksionojne goxha mire. Sipas meje algoritmi i Naive Bayes ne rastin tone ka funksionuar me mire sesa SVM. Sipas mendimit tim algoritmi i Naive Bayes ia vlen me se shumti te implementohet per shkak se eshte me lehte i menaxhueshem dhe me lehte te perdoret, per dallim nga algoritmi i SVM-se qe duhet gjithmone te behet transformimi i te dhenave ne numra dhe qe eshte proces qe merr kohe te implementohet.

Qfare mund të bëhet shtesë në ketë projekt është që të fillojmë të bëjme shkarkimin e lajmeve nga më shumë portale, dhe të krijohet një platformë ku mblidhen këto lajme dhe klasifikohen me anë të algoritmit keto lajme automatikisht në platformë. Mirepo çfarë duhet të kemi parasysh është se pas një kohe të caktuar duhet të rishikohet klasifikuesi i lajmeve për arsye se në rast se lejohet të vazhdoje pa mbikëqyrje ky klasifikues mund të humbë saktesine pas një kohe të caktuar për shkak të lajmeve të reja që dalin çdo dite. Kjo mund të behet në mënyre automatike, mirepo preferohet edhe të mbikqyret nga dikush në mënyre që saktësia e klasifikuesit te jete më e madhe.

# 8. BIBLOGRAFIA


[1] Tom Mitchell, Machine Learning, McGraw Hill, 1997

[2] Stuart J. Russell dhe Peter Norvig, Artificial Intelligence: A Modern Approach. 3rd edition, Prentice Hall, 2010

[3] John Hutchins, The history of machine translation in a nutshell, 2005.[Online] E disponueshme: http://www.hutchinsweb.me.uk/Nutshell-2005.pdf

[4] Barak Turovsky, "Ten years of Google Translate", 2016. .[Online] E disponueshme tek https://www.blog.google/products/translate/ten-years-of-google-translate/

[5] "Microsoft by the Numbers", 2020 .[Online] E disponueshme tek https://news.microsoft.com/bythenumbers/en/

[6] Statistikat e Siri, 2018. [Online] e disponueshme tek https://www.businessinsider.com/apple-says-siri-has-500-million-users-2018-1

[7] Al-Heeti, Abrar "Amazon has sold more than 100 million Alexa devices", 2019. [Online] – e disponueshme tek https://www.cnet.com/news/amazon-has-sold-more-than-100-million-alexa-devices/

[8] Google spam 100milion, 2019. [Online] https://www.theverge.com/2019/2/6/18213453/gmail-tensorflow-machine-learning-spam-100-million

[9] Sentence tokenization. [Online] E disponueshme tek https://medium.com/@makcedward/nlp-pipeline-sentence-tokenization-part-6-86ed55b185e6

[10] A. Rajaraman dhe J.Leskovec dhe J. Ullman, "Data Mining", 2011. [Online] E disponueshme tek http://i.stanford.edu/~ullman/mmds/ch1.pdf

[11] H.P.Luhn, Keyword-in-Context Index for Technical Literature (KWIC Index) American Documentation, Wiley Periodicals, Inc., A Wiley Company, 1960

[12] Zellig Harris, Distributional Structure, 1954, pp 288-295

[13] Breitinger, Corinna; Gipp, Bela; Langer, Stefan, "Research-paper recommender systems: a literature survey", 2015.[Online] E disponueshme tek https://kops.uni-konstanz.de/handle/123456789/32348

[14] Boser, Bernhard E.; Guyon, Isabelle M.; Vapnik, Vladimir N, "A training algorithm for optimal margin classifiers", 1992. [Online] E disponueshme tek http://www.clopinet.com/isabelle/Papers/colt92.ps.Z

[15] Joachims Thorsten , "Text categorization with Support Vector Machines: Learning with many relevant features". Machine Learning: ECML-98. Lecture Notes in Computer Science. Springer, 1998, pp 137-142.

[16] Barghout, Lauren. "Spatial-Taxon Information Granules as Used in Iterative Fuzzy-Decision-Making for Image Segmentation". Granular Computing and Decision-Making. Springer International Publishing, 2015, pp 285–318.

[17] A. Maity, "Supervised Classification of RADARSAT-2 Polarimetric Data for Different Land Features", 2016

[18] DeCoste, Dennis "Training Invariant Support Vector Machines" (PDF). Machine Learning, 2002, pp 161-190

[19] Gaonkar, Bilwaj; Davatzikos, Christos; "Analytic estimation of statistical significance maps for support vector machine based multi-variate image analysis and classification", 2013