ERABAKIAK HARTZEKO EUSKARRI SISTEMAK

TALDE LANA: Multilayer Perceptron

SPAM Detector

MIKEL GOJENOLA, GALDER RODRÍGUEZ ETA JON TOMÁS

2023-ko martxoaren 31

Indizea

1 Atazak eta arduren banaketa

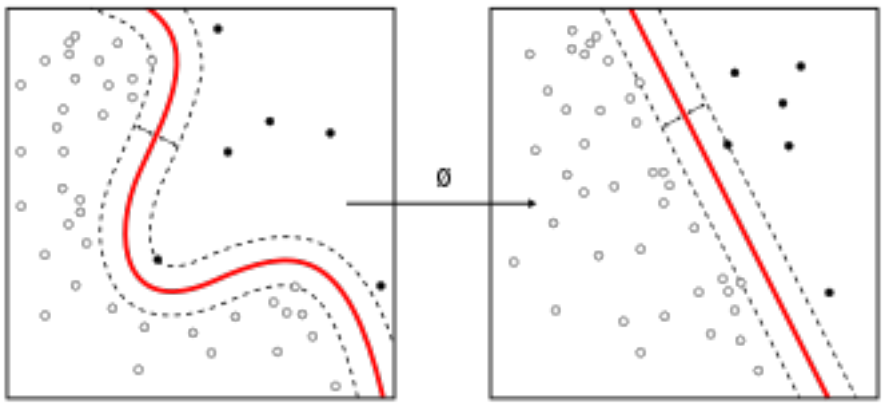
Proiektua gauzatzeko, hiru taldekideak izan gara, eta bakoitzak ataza desberdinak programatu eta egin izan ditu.
Egindako lana, nolabait laburtuta, horrela izan da:

- Mikel: Parametroen ekorketa, eredua eta estimatutako kalitatea.
  - Ataza honetan, Multilayer Perceptron-etik paramatro egokienak ateratzea eta ereduak lortzen duen kalitatea lortzea du helburu.
- Galder: Bag of Words(StringToWordVector eta attributeSelection filtroa erabili) eta Sailkapena.
  - Train eta Test atributuak bateragarriak izatea.
- Jon: Ateratako datuak AttributeSelection filtrotik datu egokien hautapena egite du helburu.

2 Esparru teorikoa

2.1 Dokumentazioa eta sintesia

Lehen aipatzen den bezala, lana egiteko Multilayer Perceptron teknika erabili egin dugu. Artificial Neural Network motakoa da, Machine Learning eta data Mining serieen barruan aurkitzen dena.



Irudia 1: Perceptron irudia.

Teknika hau hainbat geruzaz osatuta dago, zeinek sare neuronal artifizial bat sortzen duen(RNA edo SNA). Honi esker, teknika honek linealki bateragarriak ez diren programa edo problemak ebazteko gai da. Horri Perceptron deitzen zaio.

Azken honek bi egoera desberdinetan egon daiteke:

- Guztiz konektatuta : Irteera bakoitzak (I geruzakoak) hurrengo geruzako(I+1) neurona guztietako sarrera da.

- Lokalki konektatuta : I geruzako neurona bakoitza hurrengo geruzako(I+1) neuronamultzo baten sarrera da.

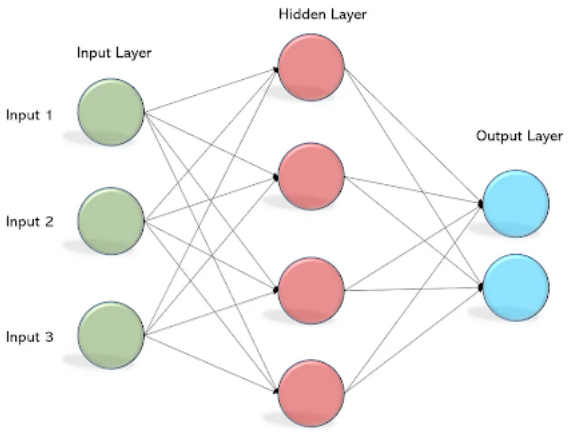
Motak:

- Sarrera geruza : Sareari sarrera ematen dioten neurona multzoz osatuta dagoen geruza

- Geruza izkutua : Atzeko geruzetatik sarrera egindako neurona multzoz osatutako sarea.
- Irteera geruza : Irteera balioekin bat egien duen neurona multzoen geruza.

Algoritmoa era egokian funtzionatzeko, elementuen transferentzia funtzioak deribagarriak izan behar dira algoritmoa era egokian exekutatzeko.

Halaber, MultilayerPerceptron mugaketa sorta oso garrantzitsuak ditu, hurrengoak esanguratsuenak izanik:



Irudia 2: Multilayer eredua (Sare neuronala).

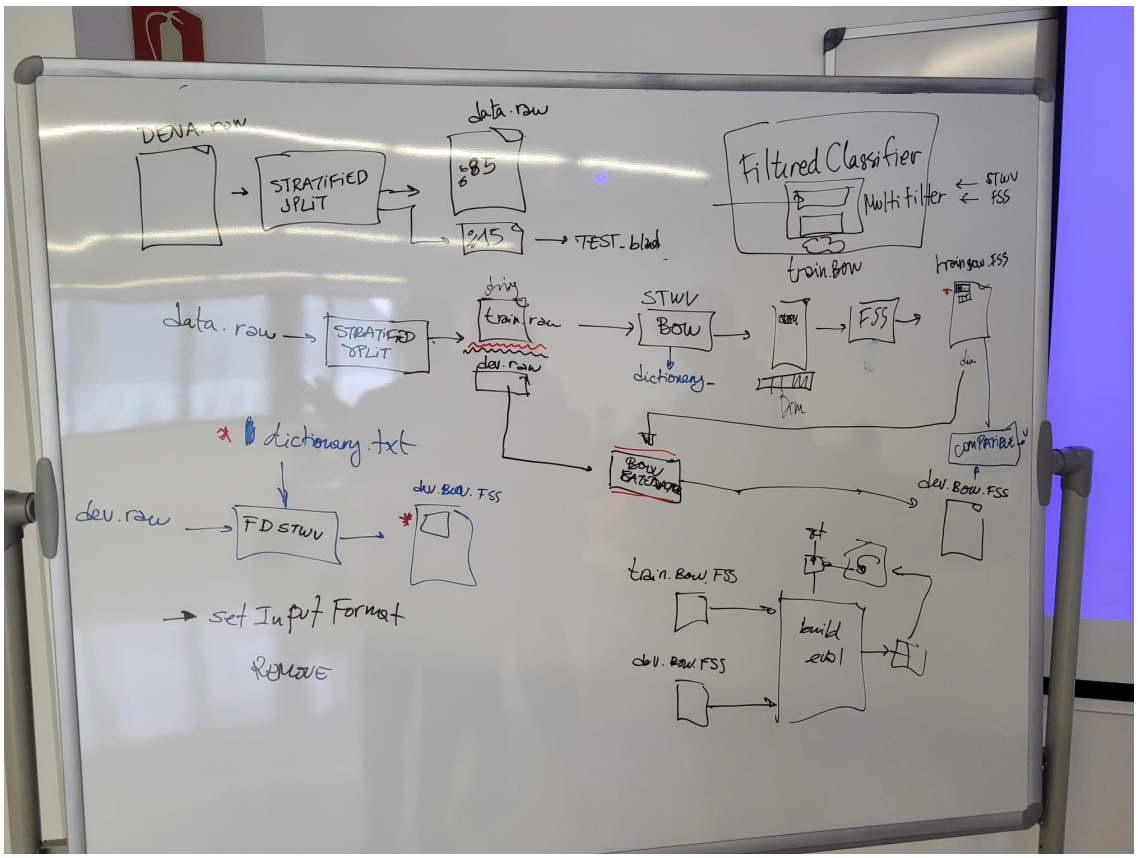
- Sarea era desegokian entrenatzen bada, edo beharrezko ez bada entrenatzen, ateratako emaitzak desegokiak edo lausoak izan daitezke.

- Minimo lokalen agerpena entrenamenduaren eragina kaltetu ahl du. Entrenamendu maila minimo batera ailegatzean, hor gelditzen da, adostutako konbergentzia mailara ailegatu gabe.

- MultilayerPerceptron RNA sarea, n sarrera neurona, m geruza iskutuko neurona eta irteera neurona batekin

2.2 Diseinua eta atazen banaketa

Diseinuari dagokionez, lehen aipatu dugun bezala, Multilayer Perceptron sarea inplementatzeko erabiliko da. Nahiz eta klasean lehen bertsioko diseinua eginda egonda, egunak pasa ahala, diseinuaren eskema eta lanaren birplanteamendua egin behar izan dugu kanpo faktoreengatik. Horregaitik, nahiz eta atazen banaketa berdin mantendu, diseinua hurrengo eran egin dugu:



Irudia 3: Egindako diseinua klasean irakaslearekin.

3 Esparru esperimentala

Ondoren emandako datuekin egin den prozesua azalduko da:

3.1 Datuak

Proiektua gauzatzeko erabili ditugun datuak Spam.txt fitxategia izan da. Spam fitxategi honek,eta proiektuaren helburua azkenean, textu multzo batzuetan Spam-a detektatzeko aplikazioa bat sortzea izan zen.

Proiektua egiteko, hiru datu sorta edo multzo erabili ditugu, Multilayer Perceptron sarea inplementatzeko eta ataza era egokian gauzatzeko. Erabilitako multzoak hurrengoak dira

- train.arff
- test.arff
- dev.arff

Egitura honek, esan dugun moduan, hiru zatitan banatu egin ditugu, eta bakoitzak, funtzionatzeko hurrengo filtroak erabili ditugu:

- StringToWordVector(STWV)
- AttributeSelection(AT)

Ikusi dugun bezala, filtro hauei esker, datuak era kualitatibo eta kunatitatiboan analizatzeko aukera ematen digu. Adibidez AS filtroari esker, lehen filtroa pasatu duten atributuen egokitzapena eta aukeraketa era egokian egiteko kapaza izango da.

Berdin gertatuko lirateke STWV filtroarekin, beharrezko atributuak hartzen ditu, eta baldintzak betetetzne baditu hurrengo egoerara pasatzeko, ongi egingo da programa, inolako oztoporekin.

Jakiteko datu sorta bakoitzak duen atributu kopurua, hurrengo taula erabili dugu, RAW DATA izenekoa, datuak aurreprozesatu baino lehen, hau da, programan sartu baino lehen, kopurua adierazteko. Taula hurrengoa da:

	Raw Data		
	Train	Dev	Test
Nominal Atributes	2	2	2
Numeric Atributes			
String Atributes	3538	771	766
Boolean Atributes			
Atributes Total	3540	773	768
Instances +	1036	224	240
Instances -	2584	552	536
Instances Total	3620	776	776

Irudia 4: Datuak aurreprozesatu baino lehen.

3.2 Aurre-prozezamendua

Datuen aurre-prozezamendua egiteko bi filtro erabili ditugu.
Lehenik Weka-ko "StringToWordVector"(String-etan gordetako hitzak hitzen lista izango den hiztegian eta honen maiztasunean goretzen dituen) filtroa erabili dugu emandako informazioa modu erabilgarri batean banatzeko hiztegi baten hitzez eta haien maiztasunarekin, honen bidez informazioaren atal bat galtzen dugu baina gakotz hitz-en errepikapen kopurua oso argi lortzen dugu. Hau oso garrantzitsua izango da SPAM-en detekziorako, hain zuzen ere mezu hauetak hitz erakargarriak erabili ohi dira jasotzen dituen pertsonak sinisten badu klik egiteko.

Honen ostean 'AttributeSelection' (Jasotako atributuen selekzioa egingo duena emandako parametroekiko hauen garrantzia kontutan izanda) filtroaren erabilera nahitaezkoa da, honek hiztegiaren luzeera murriztuko duelako, esanguratsuak ez diren hitzak ezabatuz. Honen erabilerarako, gure kasuan iragarri nahi izango den datuarekiko, (klasearekiko,) erlazio estuagoa duten algoritmoak mantenduko du, eta erlazio ahul edo erlaziorik ez duten algoritmoak ezabatuko dira.

Hemen ikus daiteke gidoian proposatukoariketetan erabili den taularen egikarapena gure datu sortaren informazioarekin beteta:

Tauletan ikus daitekeen bezala, "AttributeSelection"filtroa aplikatu eta gero datu sorta guztiek atributu nominal kopuru bera izango dute, hiztegi bera erabili beharko delako datuak bateragarriak izateko.

		Pre-processed data: BOW and FSS	
	Train	Dev	Test
Nominal Atributes	991	991	991
Numeric Atributes	1001	1001	1001
String Atributes			
Boolean Atributes			
Atributes Total	1992	1992	1992
Instances +	1036	224	240
Instances -	2584	552	536
Instances Total	3620	776	776

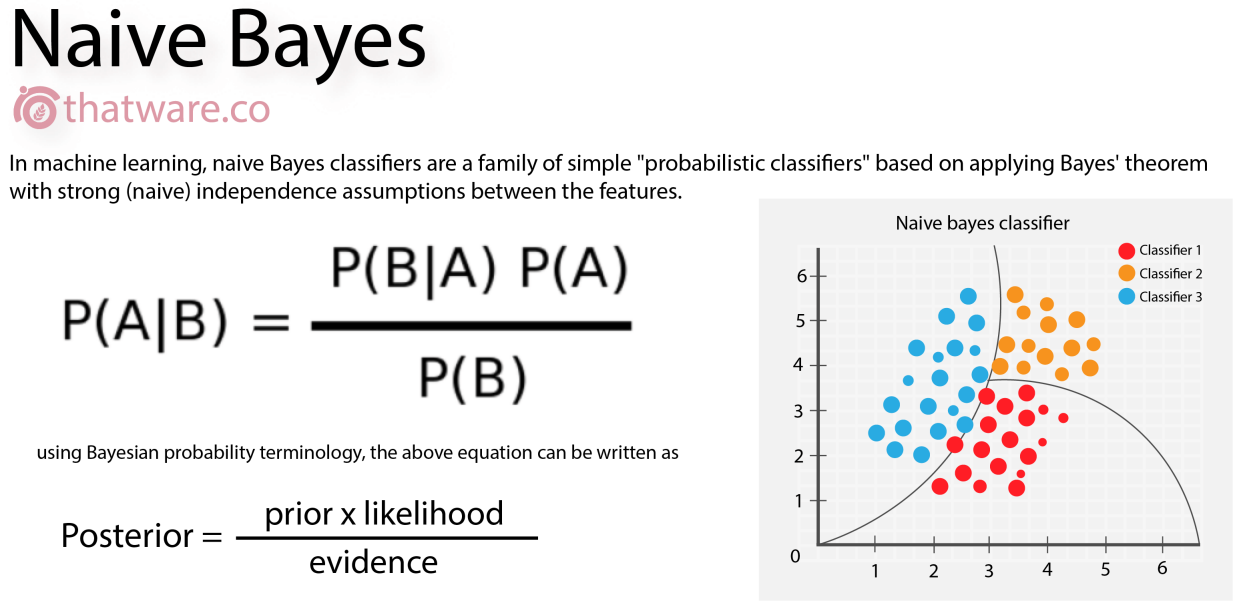
Irudia 5: Datuak filtratu eta gero.

Logikoa denez, emandako datu sorta txikituz joango da filtroak aplikatu ahala, maneiatzeko errezagoa izango da STWV (StringToWordVector) erabiltzen denean, datu total kantitatea hainbat ez aldatuz, baina errezago erabiliko dira.
Azkenik, "AttributeSelection-en bidez FSS artxiboak lortzen direnean, atributu kantitatea asko murrizten da, esanguratsuak ez direnak ezabatzen baitira.

3.3 Emaitza esperimentalak

Gure lanaren baseline modura, probak egiteko, "Naive bayes"klasifikatzailea erabili dugu. Gure klasifikatzaile finala "Multilayer perceptron"izanda, perzeptroi simple baten erabilera zuzenagoa izango litzateke, familia bereko eredu simpleago bat baitdelako, baina momentuan ez genekien aukera hori zegoenik ezta hobeagoa zela.

5- fold crossValidation egitura ere erabili egin dugu, algoritmo honek datuen aukeraketa eta prozesamendua baimentzen du, eta datu guztien balidazio partzial eta totala baieztatzen du erabilitako datuak filtroa pasatzen duenean.



Irudia 6: naiveBayes adibidea.

3.4 Emaitzen diskusioa

Behin datu finalak edo datu definitiboak programatik hartuta, baieztatu behar dugu datuak aprpopsak direla egin nahi dugunarekin. Kasu honetan, Multilayer Perceptron erabili dugunez eta data sorta asko erabili ditugunez atazak zehazteko eta aurrera eramateko, hurrengoa baiezta dezakegu:

```
// cross-validation 5-fold
Evaluation evalKFCV = new Evaluation(dataMini);
evalKFCV.crossValidateModel(model, dataMini, numFolds: 5, new Random( seed: 1));

pw.println("-------------------- CROSS-VALIDATION --------------------");
pw.println(evalKFCV.toSummaryString());
pw.println(evalKFCV.toClassDetailsString());
pw.println(evalKFCV.toMatrixString());
pw.close();
```

Irudia 7: 5-Fold Cross Validation adibidea.

- Lehen tauletan ikusitakoarekin, ikus dezakegu egindako lana eta ateratako datuak bat etortzen direla. Datuak behin aurreprozesatuta, era egokian kargatzen dira eta behar bezala funtzionatzen du guztia

3.5 Exekutagarrien adibidea

Hainbat exekutagarri sortu ditugu gure proiektuan, zortzi hain zuzen ere, eta hurrengo moduan eta ordenean exekutatu beharko dira programa behar bezala funtzionatzeko:

- Lehen atala: Aurreprozesamendua
  - rawDatuakKargatu.jar: Bi parametro
    - p1: Datuak dauden direktorioaren path-a
    - p2: Datuak arff formatuan gordetzeko path-a
  - DatuakZatitu.jar: 4 parametro
    - p1: Datuak arff formatuan dauden fitxategiaren path-a
    - p2: train gordetzeko path-a
    - p3: dev gordetzeko path-a
    - p4: test gordetzeko path-a
  - DatuakFiltratu.jar: 2 parametro
    - p1: Filtratu nahi den arff fitxategiaren path-a
    - p2: Datu filtratuak gordetzeko path-a
  - BateragarriEgin.jar: 3 parametro
    - p1: train.arff fitxategiaren path-a
    - p2: Bateragarri bilakatu nahi den fitxategiaren path-a
    - p3: Datu sorta bateragarria gordetzeko path-a
- Bigarren atala: Eredu optimoa lortu
  - BaselineLortu.jar: 3 parametro
    - p1: train.arff fitxategiaren path-a
    - p2: Eredua gordetzeko .model fitzategiaren path-a
    - p3: Irteerako datuak idazteko testu fitxategiaren path-a
  - ParametroEkorketa.jar: 3 parametro
    - p1: train.arff fitxategiaren path-a
    - p2: Learning rate parametroaren balio optimoa gordetzeko fitxategiaren path-a
    - p3: Hidden Layers parametroaren balio optimoa gordetzeko fitxategiaren path-a
  - EreduOptimoaLortu.jar: 3 parametro
    - p1: train.arff fitxategiaren path-a
    - p2: Eredu optimoa gordetzeko .model fitxategiaren path-a
    - p3: Hidden Layers parametroaren balio optimoa gordetzeko fitxategiaren path-a

- Hirugarren atala: Sailkapena
  - Sailkapena.jar: 3 parametro
    - p1: testBlind.arff fitxategiaren path-a
    - p2: Erabiliko den ereduaren .model fitxategiaren path-a
    - p3: Emaitzak idazteko testu fitxategiaren path-a

.jar fitxategiak exekutatzeko, terminala ireki eta "java -jar rawDatuakKargatu.jar parametro1 parametro2 parametro3"komandoa idatziz exekutatuko da, beharrezko parametroen balioak jar fitxategiaren ondoren zehaztuz.

Ondoren datuen sailkapena egiteko hasieratik egin beharrezko pausuak azalduko ditugu:

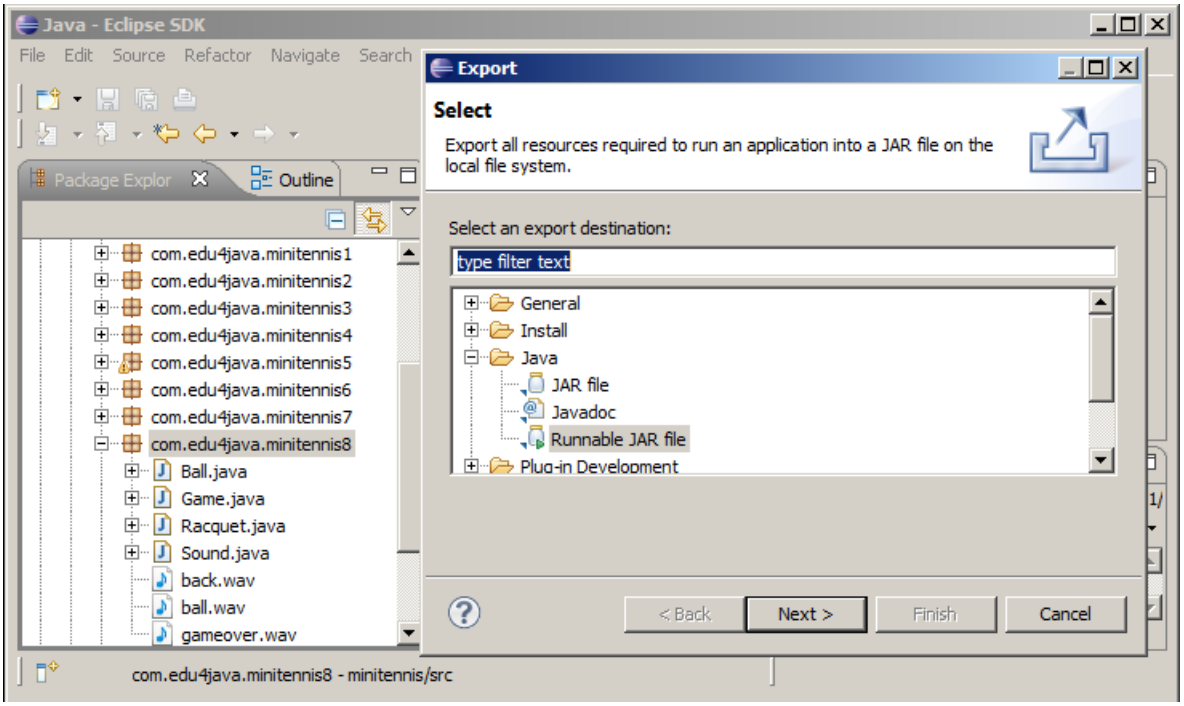
Lehenik eta behin rawDatuakKargatu.jar erabili behar da, datuak arff formatura pasatzeko, eta segituan DatuakZatitu.jar datuak hiru datu sortetan banatzeko: train, dev eta test. Behin datuak irakurtzeko moduan eta banatuta ditugula, train eta dev filtratu behar ditugu DatuakFiltratu.jar erabiliz, hau da, DatuakFiltratu.jar bi aldiz exekutatu behar da. Hori egin ondoren, dev eta train ez dira bateragarriak izango, horretarako, BateragarriEgin.jar exekutatu beharko da, train lehen parametro bezala eta dev bigarren parametro bezala emanda. Hau orain edo beranduago egin daiteke, baina test datu sorta ere filtratu eta bateragarri bilakatu behar da, dev-rekin jarraitu den prozesu bera erabiliz.

Behin datuak filtratuta daudela eta bateragarriak direla, baseline ereduaren kalitatea lortu dezakegu. Horretarako BaselineLortu.jar deitu beharko da. Sailkapena egin baino lehen geratzen den azken pausua eredu optimoa lortzea da, eta horretarako, lehenik parametro ekorketa egin beharko dugu. Parametro ekorketa egiteko ParametroEkorketa.jar deitu eta horrek sortzen dituen 2 fitxategietatik eredua lortzeko parametroak lortu ditzakegu. EreduOptimoaLortu.jar erabiliz eredua lortu eta gordeko dugu.

Azkenik, Sailkapena.jar exekutatuz, amaierako estatistikak lortuko ditugu fitxategi batean.

4 Ondorioak eta etorkizuneko lana

- Behin egindako atazaren funtzio eta ezaugarrien deskribapen orokorra, ondorioekin pasatuko gara.
- Bigarren puntuan esan dugun bezala, Multilayer Perceptron sarea erabili dugu lana burutzeko. Sare honek hainbat geruzaz osatuta dago, zeinek sare neuronal artifizial bat sortzen duen(RNA edo SNA) eta linealki bateragarriak ez diren ataza edo programak exekutatzeko gai da.
- Lanari dagokionez, sendotasunak eta ahuleziak izan ditu, hurrengo eran laburtuta:



Irudia 8: .jar baten sorkuntzaren adibidea.

  - Sendotasunak : MultilayerPerceptron sareak era egonkorran exekutatzeko gai da emandako datuekin, eta algoritmo berriak inplementatzeko aukera ezinhobea izan da Machine Learning-i buruz ikasteko.
  - Ahultasunak : MultilayerPerceptron sareak, entrenamendu falta badu, aterako dituen emaitzak ez esanguratsua izatea posible izango da. Lanaren diseinua ez da guk nahi izan genuen bezain modularra izan, beraz nahiko paketatuta dago zati batzuetan. Exekuzioa ez da behar den bezain azkarra eta eraginkorra.

- Ondorio nagusiei begira, esan dezakegu gauzarik garrantzitsuena sare honen inplementazioa izan dela, bai algoritmoan, bai kodea exekutatzerakoan. Machine Learning gaiari buruz ikastea ahalbidetu digu lan honek, eta etorkizunerako lagungarria izango del uste dugu.

- Amaitzeko, lanaren hobeketari buruzko hausnarpena egitea nahiko komenigarria da. Lehenik eta behin, nahiz eta sarearen inplementazioa nahiko lagungarria izanda gure etorkizuenarako, egia da aplikatutako teknikaren eredua ez dela baliogarria izango testu meatzaritzako antzeko atazetan, inplementatutako algoritmoa sinpleegia delako mundu errealerako. Hori esanda, proiektua hobetzeko hainbat ataza ditu, hala nola:

  - Diseinuaren modularitate eskasa
  - Lan atazen banaketa beranduegi egin da
  - Planifikazioa lehen momentutik ez da egin,

- Halaber, lana datuen tratamenduei buruz ikasteko oso baliogarria izan da, eta etorkizunerako esperientzia hartzeko baliogarria ere.

5 Bibliografia

Ondoren aipatzen dira proiekturako erabili ditugun iturriak, bai informazioa biltzeko, komandoen funtzionamendua ulertzeko edo erabilitako metodoen logika ezagutzeko:

- Multilayer Perceptron-en informazioa 1.
- Multilayer Perceptron-en informazioa 2.
- Hirugarrem esteka.

Honetaz aparte, erabilitako argazkien iturria ere azalduko da hemen:

- Lehenengo irudia.
- Bigarren irudia.
- Zortzigarren irudia.
- Seigarren irudia.

6 Balorazio subjektiboa

- Dokumentazioari itxiera aproposa emateko, proiektuari buruzko balorazio edo iradokizun subjektiboak egingo ditugu, hurrengorako hobetzeko eta akats berdinak ez errepikatzeko.
- Helburuei dagokionez, esan dezakegu ez direla guzti-guztiak bete. Proiektua hasi baino lehen, gure intentzioa gidoian agertzen den pausu guzti-guztiak gauzatzea izan da, baina denbora aurrera joan ahala eta ikusita proiektuaren lan karga handiegia zela, eskatutako algoritmoa eta sarea bakarrik inplementatzea erabaki genuen.
- Konpondu eta egikaritu ditugun ataza guztiak egiteko, talde lana ezinbestekoa izan da, bakoitzak bere zatia ongi eginda eta taldekideen arteko komunikazioa sustatzeko. Behin bakoitzaren zatia amaituta, guztiok elkartu gara zatiak lotzeko eta dokumentazio finala gauzatzeko.
- Amaitzeko, lanean interes handiena sortu diguna Machine Learning eta Data Mining gaiak izan dira. Nahiz eta era egokian edo "profesionalean" ez inplementatu, esan dezakegu egindakoa nahiko interesgarria izan dela, gaur egun teknologiekin zerikusia duen gairik garrantzitsu bati buruz trebatzeko eta ikasteko.