Universidad del País Vasco
Euskal Herriko Unibertsitatea
ZIENTZIA ETA TEKNOLOGIA FAKULTATEA
FACULTAD DE CIENCIA Y TECNOLOGÍA

Gradu amaierako lana
Ingeniaritza Elektronikoko gradua

Estreinaldi zinematografikoen arrakastaren iragarpena ikasketa automatikoa erabiliz

Egilea:
Oier López Larrambe
Zuzendaria:
Raquel Justo Blanco

© 2023, Oier López Larrambe

Leioa, 2023ko uztailaren 25a

Aurkibidea

Irudien aurkibidea

Taulen aurkibidea

1. Sarrera eta helburua

Adimen artifizialean egindako etengabeko ikerketa eta garapenari esker, gero eta sistema sofistikatuagoak eta autonomoagoak sortzea lortu da. Aurrerapen horiek hobekuntza garrantzitsuak ekarri dituzte arlo ezberdinetan, eta industria zinematografikoa ez da salbuespen bat. Izan ere, arlo honetan, estreinaldi zinematografikoen iragarpena gai kritikoa eta interes handikoa da bai inbertitzaileentzat bai ekoizleentzat. Iragarpen lanetarako garatu diren teknikak datu-meatzaritzan eta ikasketa gainbegiratuan izan dira besteak beste, eta oso erabilgarriak direla erakutsi dute film batek merkatuan lortuko duen etekinari buruzko informazio zehatza lortzeko [2].

Gauzak horrela, estreinaldi zinematografikoen arrakasta iragartzeko gai den eredu bat egiteko asmoarekin, lan honetan ikasketa automatikoko eredu ezberdinak aztertuko dira. Adimen artifizialaren gorakadarekin, estreinatuko diren pelikulen arrakasta nolakoa izango den aurreikusten duten makina nahiko zehatzak sortuak daude ([2]); baina, hauek, gehien bat, IMDb (Internet Movie Database), Rotten Tomatoes, AllMovie, TMDb (The Movie Database) eta antzeko datu-base handietan oinarritzen dira. Izan ere, tamaina handiko datu-base hauek pelikula eta telesail gehienen aktoreen-zerrendak, zuzendariak, generoak, erreseina kopuru handiak eta abar dituzte, eta hainbeste daturekin, eredu onak egin daitezke. Datu horiek hainbat sare sozialetako iritziekin konbinatuz, pelikulen diru-sarrerak nahiko zehazki iragar ditzakete. Hala ere, bai erabilitako pelikulen datuak bai sare-sozialetatik lortutako datuak herrialde ezberdinetakoak dira. Herrialde bakoitzak bere kultura, tradizio, pentsatzeko modu edota ohiturak dauzka, eta ezaugarri hauek iritzietan islatzen dira. Horregatik, lan honen helburua ez da jada egin diren ereduak hobetzen edo gainditzen dituen eredu bat eraikitzea, baizik eta ingurune soziokultural konkretu batean, bertako pelikula eta iritziekin, pelikula ikusi baino lehen jendeak helarazitako iritziak edo sentimenduak erabiltzea bertako estreinaldi zinematografikoen arrakasta iragartzeko. Kasu honetan, Espainiako pelikula eta iritziak erabiliko dira.

Horretarako, hainbat neurona-sare artifizial sortuko dira, ondoren eraginkorrena zein den baloratzeko. Hauen entrenamendua eta testatzea aurrera eramateko, bi datu multzo erabiliko dira. Batetik, Gizarte eta Komunikazio Zientzien Fakultateak helarazitako pelikulen datu-taula bat erabiliko da, zazpi mila pelikula ingururen estreinu datuak eta bakoitzak izandako diru-sarrerak azaltzen dituena. Bestetik, jendearen aurretiko iritzia lortzeko, Twitter sare soziala erabiliko da, bertan jendeak pelikula bakoitzari buruz estreinaldia baino lehen idatzitako *tweet*-ak datu bezala erabiltzeko.

Laburbilduz, lan honen helburua Espainiako estreinaldi zinematografikoen arrakasta aurresateko eredu ezberdinak eraikitzea eta alderatzea da, sentimenduen analisian oinarrituz eta input bezala estreinaldia baino lehenagoko iritziak erabiliz.

2. Ikasketa Automatikoa

Ikasketa Automatikoa, Machine Learning bezala ere ezaguna, adimen artifizialaren arlo bat da, eta honela defini daiteke: datuetan patroiak antzemateko gai den prozesu automatizatua. Hortaz, ikasketa automatikoari esker, jarraitu behar den urrats bakoitza esplizituki programatu beharrean, makinek datuetan patroiak identifikatzeko eta haien kabuz erabakiak hartzeko gai dira [3].
Atal honetan, ikasketa automatiko motak aztertuko dira, eta adimen artifizialak mugarik ez dituela dirudien arren, honek dituen erronkak ikusiko dira.

2.1. Ikasketa automatiko motak

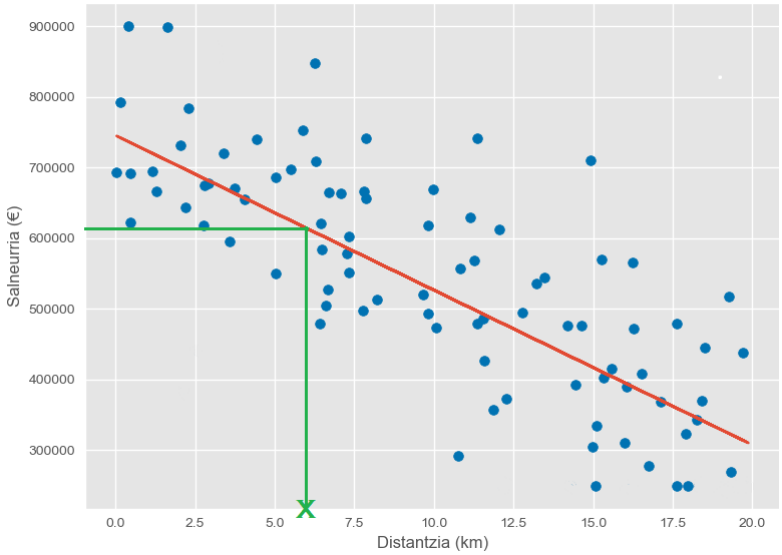
Ikasketa automatikoaren arloan, ikuspegi desberdinak daude problema bati aurre egiterakoan. Ikuspegi horiek sistemek entrenamenduan jasotzen duten gainbegiratze-kopuruaren eta -motaren arabera sailka daitezke: gainbegiratutako ikasketa, gainbegiratu gabeko ikasketa, erdigainbegiratutako ikasketa eta errefortzu bidezko ikasketa. Bat edo bestea aukeratzeko, heldu nahi zaion problemaren izaera eta eskuragarri dauden datuak kontuan hartu behar dira, besteak beste [4].

Ikasketa mota bakoitza ondo ulertzeko, datuen egitura nolakoa den jakitea beharrezkoa da. Datu-base bat osatzen duten datu bakoitzari lagin (edo *sample*) esaten zaio, eta lagin bakoitza deskribatzen duten propietateei ezaugarri (edo *feature*) deritze. Halaber, lagin bakoitzak iragarri nahi den parametro edo desiratzen den soluzio bat eduki ohi du, eta etiketa (edo *label*) bezala ezagutzen da [5].

2.1.1. Gainbegiratutako ikasketa

Gainbegiratutako ikasketan, algoritmoa entrenatzeko erabiliko diren datuek soluzioei dagozkien etiketak dituzte. Hots, halabeharrez datu-basearen laginek etiketak izan behar dituzte. Gainbegiratutako ikasketa klasifikazio zereginetan eta balio numerikoen iragarpenetan asko erabiltzen da. Iragarpen-zeregin horiek erregresio deritze eta adibide ugari behar dira sistema entrenatzeko [4].

Erregresio baten adibide bat ondokoa izan liteke: Demagun Bilboko etxebizitzen salneurria iragarri nahi dela. Horretarako, algoritmoa Bilboko etxebizitza anitzen datuekin entrenatu beharko litzateke. Lagin bakoitzean ezaugarria etxebizitzaren eta Bilboko hirigunearen arteko distantzia izan liteke, eta etiketa etxebizitzaren salneurria. Hortaz, erregresio bat egingo balitz, 2.1 irudiko eredu bezalako bat lortuko litzateke.



2.1 Irudia: Erregresio lineal baten adibidea.

Irudian, puntu urdinak erabilitako datuak izango lirateke eta gorriz dagoen marra zuzena entrenatutako eredua izango litzateke, hots, erregresioa. Behin erregresioa egin dela, berdez ikus daiteke 6 km-tara dagoen etxebizitza baten salneurria jakin nahi ezkero, ereduak 600.000€ ingurukoa dela esango duela.

Kasu honetan distantzia eta salneurria gutxi gora behera linealki proportzionalak direnez, nahikoa izan da erregresio lineal bat egitearekin. Hala ere, izan daiteke ezaugarriak eta etiketak linealki proportzionalak ez izatea, eta ondorioz, lagin horiekin aritzeko,erregresio lineala ez litzateke algoritmorik egokiena izango. Kasu horietan, beste algoritmo ezagun batzuk erabil daitezke, hala nola:

- Erregresio Ez-lineala
- SVMR (Support Vector Machines Regression)
- Erabaki Zuhaitzak
- Neurona-sare artifizialak

Klasifikazio lanetarako, aldiz, beste algoritmo ezberdin batzuk erabiltzen dira; baina, beste ikuspegi bat emanez, iragarpen numerikoetan erabiltzen diren zenbait algoritmo ere erabil daitezke. Spam mezuen detekzioa, adibidez, gainbegiratutako ikasketan oinarritutako klasifikazio lan bat da. Algoritmoak etiketatutako spam mezu eta mezu arruntetatik ikasten du eta, ondoren, mezu berriak spam edo arruntak diren detektatzeko gai da, hots, mezuak spam edo arrunt bezala klasifikatzen ditu. Hauek dira klasifikazio lanetarako erabil daitezkeen batzuk:

- Erregresio Logistikoa
- k-Nearest Neighbors
- SVMs (Support Vector Machines)
- Erabaki Zuhaitzak

- Neurona-sare artifizialak

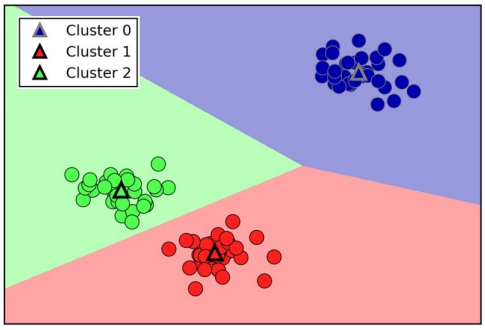
Lan honetan klasifikazio lanak egin nahi dira, eta bai datuen bai problemaren konplexuatsuna direla medio, neurona-sareak eraikiko dira. Aipatzekoa da neurona-sare guztiak ez daudela gainbegiratutako ikasketan oinarrituak. Autoenkoder[1] baten arkitektura, adibidez, gainbegiratu gabea izan daiteke.

2.1.2. Gainbegiratu gabeko ikasketa

Gainbegiratu gabeko ikasketan, sistemari ematen zaizkion datuak etiketarik gabekoak dira, hots, datu-laginetan soilik ezaugarriak daude. Beste modu batera esanda, sistemak irakaslerik gabe ikasten saiatzen da.
Ikasketa mota hau hiru aplikazio nabarmen ditu: Clustering-a, anomalia-detekzioa eta dimentsio-murrizketa [4].

✦ **Clustering:**
Clustering-a datu-basea *cluster* izeneko talde ezberdinetan banatzean datza. Helburua datuak banantzea da, non *cluster* batean dauden datu-puntuak oso antzekoak diren baina beste *cluster* batean daudenekin ezberdinak diren. Klasifikazio algoritmoetan bezala, clustering-algoritmoak datu-puntu bakoitzari zenbaki bat esleitzen diote, bakoitzari *cluster* bat egokitzeko.

k-means clustering-a gehien erabiltzen den clustering-algoritmoetako bat da. Datuen zonalde jakin batzuetan adierazgarrienak diren *cluster*-zentruak bilatzen saiatzen da. 2.2 irudian ikus daiteke *cluster* bakoitzeko zentrua triangelu batekin adierazita dagoela, datu-puntuak zirkuluak direla eta *cluster* bakoitza kolore bat duela. Algoritmoak bi urrats jarraitzen ditu: lehenik, datu-puntu bakoitza hurbilen duen *cluster*-zentrura esleitzen du, eta, bigarrenik, *cluster*-zentru bakoitzak esleituta dituen puntuen posizioaren batezbestekoa eginez, zentruaren kokapena berriz eguneratzen da. Irudian ere *cluster* bakoitzaren mugak ageri dira, eta beraz, zonalde urdinean 'erortzen' den edozein puntu *Cluster 0* multzoaren parte izango da [5].



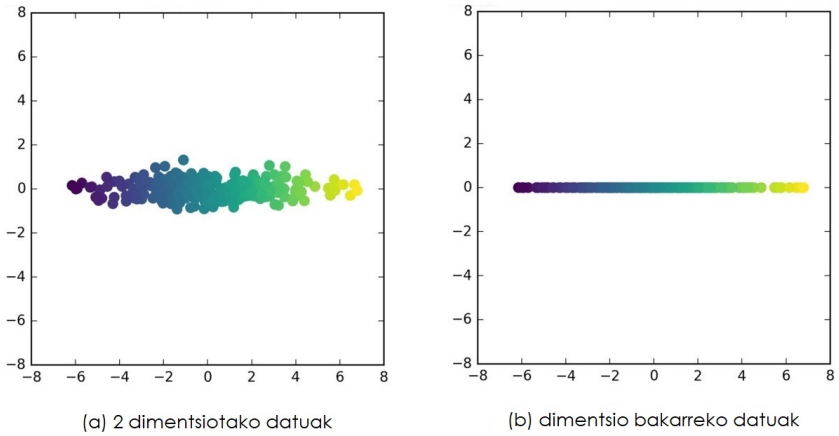
2.2 Irudia: k-means bidezko clustering analisia [5].

✦ **Anomalia-detekzioa:**
Ideia nagusia arruntak edo ohikoak diren datuak antzematen ikastea da, eta ondoren, ikasitakoa ezohikoak diren kasuak antzemateko erabili, adibidez, produkziokate batean produktu akastunak detektatzeko. Azaldutako clustering-a anomaliak

[1]Gainbegiratu gabeko neurona-sare artifizial bat da. Datuak modu eraginkorrean konprimitzen eta kodetzen ikasten du, eta, ondoren, datuak berreraikitzen ikasten du [6].

detektatzeko ere balio du: *cluster* guztietatik aldentzen den edozein lagin ziurrenik anomalia bat izango da [4].

✦ **Dimentsio-murrizketa:**
Helburua datuak informazio askorik galdu gabe sinplifikatzea da. Horretarako korrelazio[2] handia duten datuen zenbait ezaugarri bateratzen dira. Adibidez, auto batek egindako kilometro kopurua autoaren adinarekin erlazionaturik egongo da, hots, korrelazio handia izango dute. Hortaz, dimentsio-murrizketarako algoritmoak bi ezaugarri horiek 'narriadura' adierazten duen ezaugarri bakar batean bateratuko ditu. 2.3 irudian argi ikusten da hasierako datuak (ezkerrean) bi ezaugarri izatetik bat izatera igaro dela (eskuinean), informazio nabarmenik galdu gabe [4].



2.3 Irudia: Dimentsio-murrizketa [5].

2.1.3. Erdigainbegiratutako ikasketa

Erdigainbegiratutako ikasketa aurreko bi ikasketa moten arteko nahasketa da, partzialki etiketatuako datak erabiltzen baitira. Ikasketa hau datu mota bat (argazkiak, testuak, etab.) multzotan klasifikatzeko erabil daiteke. Adibidez, Google Photos argazki-biltegiak erdigainbegiratutako ikasketa erabiltzen du pertsonen aurpegiak antzemateko. Clutering-algoritmoak erabiliz, argazki ezberdinetatik berdinak diren aurpegiak antzematen ditu (aurpegi bakoitza *cluster* bat izango litzateke) eta ondoren, erabiltzaileak aurpegi ezberdin bakoitzari, hau da, pertsona bakoitzari, etiketa bat jartzen dio (pertsonaren izena). Modu honetan, aurpegi guztiak etiketaturik geratzen dira eta erraza izango da orain pertsona bati dagozkion argazkiak bilatzea [4].

2.1.4. Errefortzu bidezko ikasketa

Errefortzu bidezko ikasketa oso ezberdina da ikusitako ikasketa motekin alderatuta. Ikasketa-sistemak, *agente* deiturikoa, ingurunea azter dezake eta ekintzak bere kabuz gauzatu ditzake, trukean sariak edo isunak (sari negatiboak) jasoz. Modu honetan, bere kabuz, zeregin bat aurrera eramateko estrategiarik hoberena zein den ikasten du, denboran zehar ahalik eta sari gehien lortzeko asmoz. Robot askok errefortzu bidezko ikasketarako algoritmoak erabilten dituzte ibiltzen ikasteko. Halaber, AlphaGo programak ikasketa mota

[2]Korrelazioa bi aldagai linealki zenbateraino erlazionatuak dauden erakusten duen parametro bat da, 1 eta -1 balioen artekoa [7].

hau erabili zuen Go jokoaren munduko irabazlea menderatzeko. Sistemak milioika Go-partida aztertu zituen, eta egoera ezberdinen aurrean, estrategiarik hoberenak burutzen ikasi zuen.

2.1.5. Beste sailkapen batzuk

Aipatzekoa da ikasketa motak, sistemak entrenamenduan jasotzen duten gainbegiratze-motaren arabera sailkatzeaz gain, beste alderdi batzuen arabera sailkatu daitezkela. Adibidez, denbora errealean edo ez ikasten duen arabera sailka liteke: funtzionamenduan dagoen sistema batek modu sekuentzial batean datu-korronte berri batetik ikasi ahal badu, *online*-ikasketa daitzen zaio, eta, aldiz, sistema deskonektatu behar bada datu berriekin entrenatu ahal izateko, eta behin entrenatu dela, berriz funtzionamenduan jarri behar bada, *batch*-ikasketa izango litzeteke.

Sailkatzeko beste modu bat ereduetan oinarritutako ikasketa eta adibideetan oinarritutako ikasketa izango litzateke. Lehenengoan, adibideetatik eredu bat eraikitzen da, ondoren iragarpenak egiteko. Bigarrenean, aldiz, iragarpenak antzekotasunaren arabera egiten dira.

2.2. Entrenamendu eta testatze faseak

Nahiz eta adimen artifizialeko eredu mota ezberdinak egon, guztien garapenerako entrenamendua ezinbestekoa da, eta datuek etiketak izatekotan, testatzea ere beharbeharrezkoa da. Prozesu hauek burutzeko, datu-basea banatu egiten da, eta honi esker, eredua bat eraiki eta ondoren ebaluatu daiteke.

2.2.1. Entrenamendua

Adimen artifizialeko eredu baten entrenamendua datuak erabiliz ereduari zeregin bat burutzen irakasteko prozesua da. Datu horiek entrenamendu-datuak bezala ezagutzen dira, eta datu-basearen proportzio handi bat izan ohi da (%80 ingurukoa). Prozesu honetan, datuak etiketaturik badaude, ereduari datuen ezaugarriak aurkezten zaizkio eta horietako bakoitzari dagokion soluzio edo etiketari esker, ereduak zeregin jakin bat burutzen ikasten du.

Entrenamenduan zehar, optimizazio algoritmo bati esker ereduaren parametroak iteratiboki aldatzen joaten dira. Algoritmo horrek ereduak emandako soluzioen eta datuen etiketen arteko diferentzia minimizatzea bilatzen du. Hau sakonago azalduko da 2.4.2 atalean.

Entrenamendua-datuen multzo txiki bat ereduaren balioztatzerako erabiltzen da. Datumultzo hori ereduaren hiperparametroak doitzeko eta suerta daitezkeen zenbait arazo ekiditzeko erabiltzen da. Orokorrean, datu-basearen %10 erabiltzen da.

2.2.2. Testatzea

Testatzearen helburu nagusia ereduak ikasitakoa datu-multzo berri baten aurrean modu egoki batean aplikatzeko gai dela frogatzea da. Datu-multzo hori datu-basearen proportzio txiki bat izan ohi da (%20 ingurukoa); zehazki, entrenamendurako erabili ez diren

datuak.

Prozesu honi esker, ereduaren zehaztasuna, sentsibilitatea edo egonkortasuna bezalako parametroak neur daitezke, 2.6 atalean azaltzen den moduan. Hortaz, testatzeari esker, eredua ebaluatzeko egokiak diren metrika ezberdinak aplika daitezke. Hauen arabera, ereduan aldaketak egitea planteatu daiteke, hala nola, zenbait parametro berriz egokitu edo ereduaren arkitektura birplanteatu.

Halaber, testatzeari esker, ereduaren entrenamenduan zehar 2.3.2 atalean aurkeztuko diren *overfitting* eta *underfitting* fenomenoak gertatu diren azter daiteke.

2.3. Ikasketa automatikoaren erronkak

Adimen artifizialeko eredu bat sortzeko bi betebehar nagusi daude: ikasketa-algoritmo bat aukeratzea eta datu-base batekin entrenatzea. Hala ere, betebehar hauek burutzean, emaitza desegokiak lor daitezke, bai datu-base desegoki bat erabiltzeagatik bai algoritmo desegoki bat aukeratzeagatik.

2.3.1. Datu-basearen arazoak

Izan daiteke ikasketa-algoritmo oso on bat aukeratu izan arren, gogobetetzen ez duten emaitzak lortzea. Hau datu-basean arazoren bat dagoelako gertatu ohi da.

- **Datu-kopuru eskasa**
  Ikasketa automatikoko algoritmoaren arabera, datu-kopurua txikia izan daiteke. Orokorrean, algoritmoa konplexua bada, datu-kopuru handiekin entrenatu behar da. Datu-eskasia egotekotan, algoritmoa ez da behar bezala ereduztatuko.

- **Adierazgarria ez den datu-basea**
  Izan daiteke datu-basea ehuneko ehunean adierazgarria ez izatea. Demagun 2.1 irudiko erregresioa egiteko Bilboko itsasadarraren alde batean dauden etxebizitzak soilik kontuan hartu direla. Egindako eredua beste itsasadarrean dauden etxebizitzekin frogatzean, lortutako emaitzak, seguraski, ez dira errealitatean dauden salneurrien antzekoak izango. Horregatik, garrantzitsua da erabiliko den datu-basea ondo ezagutzea eta errealitatearen adierazgarri ona den aztertzea.

- **Kalitate txikiko datu-basea**
  Datu-basea erroreak dituzten eta ezohikoak diren laginez osatuak baldin badaude, algoritmoak ez du bere betebeharra ondo burutuko. Orokorrean, datu-basea garbitzen eta zuzentzen denbora emateak merezi duen esfortzua da.

- **Ezaugarri hutsalak**
  Lortu nahi den betebeharrerako informazio handirik eskaintzen ez duten ezaugarriak eredua okertu dezakete. Garrantzitsua da datu-baseak esanguratsuak diren ezaugarriak izatea bai hutsalak diren ezaugarririk ez edukitzea. Hortaz, emaitza onak lortzeko aproposa da ezaugarri erabilgarrienak hautatzea, zenbait ezaugarri konbinatzea erabilgarriagoa den beste bat lortzeko (eta aurretik aipatu den bezala, dimentsioak murrizteko) eta behar ezkero, datu gehiago biltzea ezaugarri berriak sortzeko. Zenbait lagin ezaugarri hutsalak baldin badituzte, lagin horiek ezabatu edo gainontzeko laginen batezbestekoa eginez ezaugarri horietan informazioa sartu beharko da. [4].

2.3.2. Algoritmo-aukeraketaren arazoak

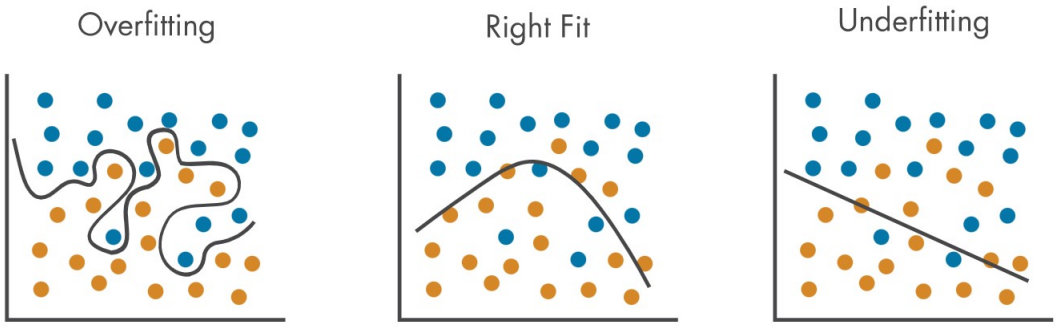
Esan bezala, datu-basea garrantzitsua den moduan, ikasketa-algoritmoa ere berebiziko garrantzia du, gaizki aukeratuz gero, zenbait arazo suerta baitaitezke.

- **Overfitting**
  Eredua entrenamenduan zehar datuetara gehiegi ajustatzean ematen den fenomenoa da. Entrenamenduan emaitza onak lortzen dira, baina testatzean ez ditu emaitza onak ematen.

  Bilboko etxebizitzen salneurrien problemarekin jarraituz, demagun eredu konplexu batek hirigunetik 20 kilometrotara dagoen luxuzko etxebizitza bat milioi bat euro balio duela ikasten duela. Eredua entrenatu ostean, beste norabide batean 20 kilometrotara dagoen herri baten etxe baten salneurria iragartzea eskatzen zaionean, milioi baten ingurukoa dela esango du, errealitatean 300.000€-koa izan arren. Argi eta garbi datuak gainorokortzen ari ditu, eta horri *overfitting* deritzo.

  Hau neurona-sareak bezalako eredu konplexuetan gertatu ohi da. Hauek patroi hautemanezinak antzeman dezakete, eta datu-basea nahiko zarata badu edo oso txikia bada, izan daiteke ereduak zarata horretan patroiak aurkitzea. 2.4 irudian, *overfitting*-aren kasuan, eredua datu-basean dagoen zarata kontuan hartzen ari dela beha daiteke.

  Hortaz, eredua asko egokitzen denean, entrenamendu-datuetan dagoen zarata patroi bezala interpretatzen du eta, ondorioz, *overfitting*-a gertatzen da. Egon daitezkeen konponbideak eredua sinplifikatzea (parametro-kopurua txikituz, adibidez), datu gehiago lortzea edo datu-basearen zarata txikitzea dira.



2.4 Irudia: *Overfitting*, puntu optimoa eta *underffiting*-aren adibideak [1].

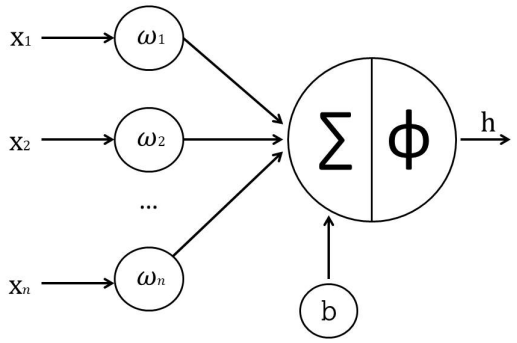
- **Underfitting**
  Heldu nahi zaion problemarekin alderatuta, simpleegia den eredu bat aukeratzeagatik sortzen da, hots, eredua datuen azpian dauden patroiak edo estrukturak antzemateko sinpleegia denean gertatu ohi da. Beraz, *Underfitting*-a *Overfitting*-aren kontrakoa da. Datuetara ondo ajustatzen ez denez, jada entrenamenduan zehar eredua gaizki dabil, eta ondorioz, testatzean ere.

2.4 irudian ikus daiteke erregresio lineal sinple bat ez dela nahikoa datuen konplexutasuna antzemateko. Arazo hau konpontzeko aukerak konplexuagoa den beste eredu bat aukeratzea (parametro gehiagorekin) eta ezaugarri gehiago dituzten datuak erabiltzea dira [4].

2.4. Neurona-sareak

Neurona-sare artifizialak giza garunaren funtzionamenduan oinarritutako eredu konputazional mota bat dira. Datu-base handietatik ikasteko eta patroi konplexuak antzemateko gaitasuna dute, eta horregatik oso erabilgarriak dira klasifikazio- eta iragarpen-lanak egiteko, lengoaia naturala eta irudiak prozesatzeko, eta abar.

Neurona-sareak neurona izeneko prozesaketa-unitatez osatuak daude. Unitate logiko hauek geruzatan multzokaturik daude eta haien artean konexioak dauzkate. Neuronek seinale bat jasotzen dute, eta zenbait kalkulu eta prozesaketa egin ondoren, irteera bat sortzen dute. Neuronen arteko konexioei esker, sarrerak sarean zehar aldaketak jasaten dituzte, eta azkenean, sareak emaitza bat sortzen du. Emaitza horren arabera, entrenamendu fasean neuronen parametroak aldatzen dira, emaitza egokiago bat lortu nahian.



2.5 Irudia: Neurona artifizial orokor baten diagrama.

Neurona batek hainbat parametro kontuan hartzen ditu, 2.5 irudian ikus daitekeen moduan. Batetik, sarrera-seinaleak daude (X_i), balio numerikoak direnak. Seinale horiek neuronaren sarrera-konexioen bitartez neuronara iristen dira, eta konexio bakoitzari pisu bat (ω_i) dagokio. Pisua zenbakizko parametro bat da, eta honen arabera, konexio horrek neuronaren irteeran izango duen eragina zehazten da: geroz eta pisua handiagoa izan, orduan eta eragin handiagoa izango du konexio horren sarrerak bukaerako emaitzan. Neurona-sareak lortutako emaitzaren arabera, entrenamenduan zehar, pisu hauek automatikoki aldatzen joango dira emaitza egokiago batera hurbiltzeko. Neuronek, orokorrean, hainbat seinale aldi berean jasotzen dituzte, bat sarrera-konexio bakoitzeko. Ondoren, seinale eta bakoitzari dagokion konexio bakoitzaren pisuen biderkaduren arteko batuketa egiten da, 2.1 ekuazioan bezala, eta *bias* izeneko termino independente bat (b) gehitzen zaio. Azkenik, batura horri aktibazio-funtzio bat (ϕ) aplikatzen zaio. Funtzio honen arabera, neurona aktibatu edo desaktibatu egingo da, sarreraren arabera neurona batzuk edo beste batzuk aktibatzen baitira. Aurretik aipatutako *bias* terminoari esker, neuronak aktibatzeko eta desaktibatzeko duen sentsibilitatea doitzen da. *Bias* terminoa neurona

bakoitzerako ezberdina izan daitekeen arren, aktibazio-funtzioa berdina da geruza berdineko neurona guztientzat [4].

$$z = \omega_1 \cdot x_1 + \omega_2 \cdot x_2 + \ldots + \omega_n \cdot x_n \tag{2.1}$$

Gauzak horrela, 2.2 ekuazioan neuronaren irteera-seinalea azaltzen da, non $\mathbf{x}^T$ eta $\boldsymbol{\omega}$ seinaleen bektorea eta konexioen pisuen bektoreak diren, hurrenez hurren.

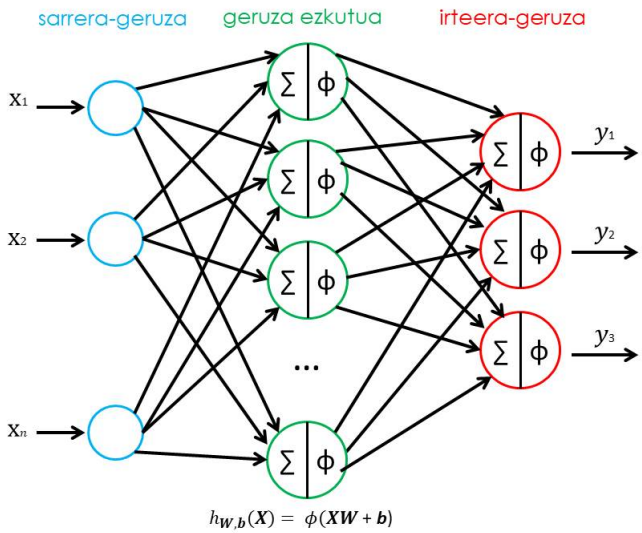
$$h(\mathbf{x}) = \phi(\mathbf{x}^T \cdot \boldsymbol{\omega} + b) \tag{2.2}$$

Aurretik aipatu den bezala, neurona-sare bat hainbat neurona-geruzataz osatua dago. 2.6 irudian neurona-sare baten egitura orokorra azaltzen da, eta ikus daitekeen bezala, hiru geruza mota nagusi ditu: sarrera-geruza, geruza ezkutuak eta irteera-geruza [8].

- **Sarrera-geruza:** Sarearen lehenengo geruza da, eta bertan, jatorrizko datuak sartuko lirateke. Orokorrean, geruza honetan, neuronen irteera sarrerako terminoen berdina da, hots, sarrera-geruzak soilik zubi bezala egiten du datuen eta ezkutuko geruzen artean. Datuen arabera, neurona gehiago edo gutxiago beharko dira sarrera-geruzan; neurona bat behar da datuaren osagai bakoitzeko.
  Adibidez, Bilboko etxebizitzen probleman, hiriguneraino distantzia zehazteko iparreranzko, hegoalderanzko, ekialderanzko eta mendebalderanzko distantzia biltzen dituen bektore bat erabiltzen bada, sarrera-geruzan lau neurona erabili beharko dira, datu edo bektore bakoitzak 4 osagai baititu.

- **Geruza ezkutuak:** Sarrera-geruzaren ondoren datoz. Bertan sarreraren prozesaketa guztia ematen da, eta sarearen konplexutasuna eta datu-motaren arabera geruza ezkutu gehiago edo gutxiago behar dira.

- **Irteera-sarea:** Geruza-ezkutuen ostean dator, eta hau da emaitzak bueltatzen dituena. Egin nahi den analisiaren arabera, neurona kopurua ezberdina da. 2.6 irudian irteera-geruzak hiru neurona ditu, eta ondorioz, neurona-sarea egiten ari den analisimota sailkapenekoa izango da; hiru neuronaz osatuta dagoenez, sarrerako datuak hiru klase ezberdinetan sailkatuko dira.



2.6 Irudia: Neurona-sare artifizial orokor baten diagrama.

Irudian ikus dezakegu erabiltzen diren datuak n osagaiez osatuta daudela. Sarrera-geruzari esker datuak geruza ezkutura iristen dira eta bertan, konexio bakoitzaren pisuarekin, neurona bakoitzaren *bias* terminoarekin eta aktibazio-funtzioarekin $h_{\mathbf{W},\mathbf{b}}(\mathbf{X}) = \phi(\mathbf{XW} + \mathbf{b})$ irteera lortzen da, non $\mathbf{X}$ sarrera-parametroen matrizea (lerro bat instantzia bakoitzeko eta zutabe bat datuen osagai bakoitzeko), $\mathbf{W}$ konexio bakoitzaren pisua (lerro bat sarrerako konexio bakoitzeko eta zutabe bat neurona bakoitzeko), $\boldsymbol{b}$ *bias* terminoen bektorea eta ϕ aktibazio-funtzioa diren. Behin irteera kalkulatuta, irteera-geruzara doa, eta bertan azken prozesaketa egiten da ondoren 3 irteerak sortzeko.

2.4.1. Neurona-sare motak

Esan bezala, datuen eta ereduak bete behar duen lanaren konplexutasunaren arabera, geruza ezkutu gehiago edo gutxiago ipintzen dira. Geruza kopurua neurona-sare artifizialak sailkatzeko modu bat da. Geruza gutxi baditu, azaleko neurona-sarea izango litzateke, eta geruza asko baditu, aldiz, neurona-sare sakona.
Hala ere, sarearen arkitekturaren arabera, hiru neurona-sare mota nagusi desberdintzen dira: *feedforward* neurona-sareak, konboluzio neurona-sareak eta neurona-sare errekurrenteak.

- ***Feedforward* Neurona-sareak:** Informazioa norabide bakar batean igarotzen da sarean zehar, hain zuzen ere, sarrera-geruzatik irteera-geruzarantz. Gainera, geruza bakoitza hurrengoarekin guztiz kontektaturik dago. *Feedforward* sareak klasifikazio, erregresio eta patroi-errekonozimendurako erabiltzen dira.

- **Neurona-sare Errekurrenteak:** *Feedforward* sareak bezalakoak dira, baina konexioen berrelikadurak dituzte. Berrelikadurei esker, memoria daukate eta datu-sekuentziak prozesa ditzakete. Ondorioz, neurona bakoitzak bi sarrera dauzka: sarrera-konexioa (atzetik dauden neuronen irteerak sartzen direneko konexioa) eta berrelikadura-konexioa (neuronak aurreko iterazioan eman duen irteera sartzen deneko konexioa). Sekuentzia-prozesaketan oso erabilgarriak dira, hala nola, ahotsa antzemateko, itzulpen automatikorako, testuen sorkuntzarako, etab.

- **Konboluzio Neurona-sareak:** Irudiak bezalako bi dimentsiotako datuekin (matrize itxurako datuekin) erabiltzen diren sareak dira. Konboluzio-geruzak eta multzokatze-geruzak dauzkate, lehenengoak datuen eta zenbait parametroren arteko konboluzioa egiteko eta horrela datuetatik ezaugarriak lortzeko, eta bigarrenak dimentsionalitatea murrizteko. Konputazio bidezko ikusmenerako, argazki-klasifikaziorako, objektuen antzematerako eta segmentazio semantikorako erabiltzen dira, besteak beste.

Hala ere, azkenaldian arkitektura berriak sortu egin dira, esate baterako, transformerrak, 2.5 atalean agertuko direnak.

2.4.2. Optimizazio-algoritmoak

Adimen artifizialeko ereduak eraikitzeak ikaragarrizko konplexutasun konputazionala dakar eta ondorioz, denbora asko behar da hauek entrenatzeko eta erabiltzeko. Hala ere, arazo horri aurre egiteko algoritmo ezberdinak sortu egin dira, eta neurona-sare artifizial batean optimizazio-algoritmo egoki bat aukeratzea garrantzi handikoa da.

Neurona-sare bat entrenatzeko, kostu-funtzio bat definitu behar da datu jakin batzuekin sareak egiten dituen iragarpenen eta datu horien etiketen arteko aldea neurtzeko.

Kostu-funtzio horri optimizazio-algoritmo hauek aplikatzen zaizkio, helburua sarearen entrenamenduan pisu-parametro sorta zehatz bat lortzea baita, zeinetarako sareareak iragarpen zehatzak egiten dituen. Modu sinple batera esanda, helburua parametroak doituz kostu-funtzio hori minimizatzea da. Gainera, hainbat hiperparametro izan ohi dituzte, eta garrantzitsuenetako bat *learning rate* (η) izenekoa da [9].

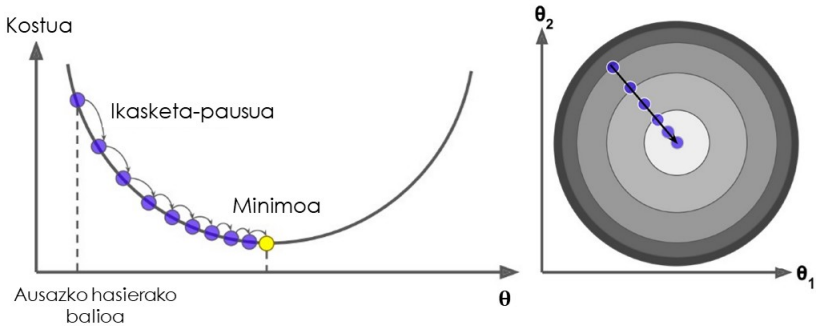
2.4.2.1. Gradiente-jaitsiera (*Gradient Descend*)

Optimizazio-algoritmo oso orokorra den arren, gainontzeko algoritmoen oinarria da. Gradiente-jaitsieraren ideia nagusia ondokoa da: kostu-funtzio bat minimizatzeko, iteratiboki parametroak doitzea. Algoritmo honek kostu-funtzioaren gradiente lokala neurtzen du θ parametroaren bektorearekiko (θ bektorea aurretik aipatutako pisu-sorta da), eta beheranzko gradientearen norabidean doa (ikusi 2.7 irudia). Behin gradientea zero denean, minimora iritsi da. θ parametroaren hasieratzea ausaz egiten da, eta ondoren, iterazio bakoitzean pauso txiki bat hartuz, hau da, orokorrean parametroaren baliotik gertu dagoen beste balio bat hartuz, kostu-funtzioa txikitzen den norabidean joango da.

Algoritmo hau 2.3 ekuazioan adierazten da. θ parametroa bere buruari kostu-funtzioaren ($J(\theta)$) gradientea kenduz eguneratzen da. Baina, ikus daiteke gradientea η parametroak biderkatzen duela. Parametro hau lehen aipatutako *learning rate*-a da, eta pauso bakoitzaren tamaina zehazten du.

$$\theta \leftarrow \theta - \eta \nabla_\theta J(\theta) \tag{2.3}$$

Oso garrantzitsua da η egoki bat aukeratzea. Pauso oso txikiak hartzen badira ($\eta \sim 0$), algoritmoak iterazio asko igaro beharko ditu konbergitu arte, eta horrek denbora asko beharko du. Aldiz, pauso oso handiak hartzen badira ($\eta \sim 1$), iterazio bakoitzean kurbaren beste haraneraino jauzi egin dezake eta izan daiteke aurreko puntuan baino puntu altuago batean amaitzea. Honek, algoritmoa dibergitzea eragingo luke.



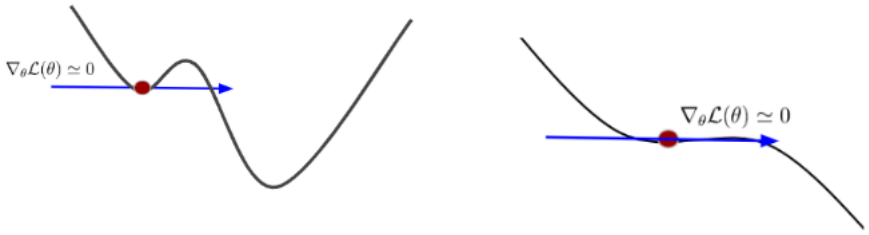
2.7 Irudia: Gradiente-jaitsiera [4].

Gradiente-jaitsiera modu ezberdinetan inplementa daiteke:

- **Batch Gradient Descend (BGD):** iterazio bakoitzean datu guztiak erabiltzen ditu gradientea kalkulatzeko. Horrek algoritmoa moteltzen du baina nahiko egonkorra da.

- **Stochastic Gradien Descent (SGD):** iterazio bakoitzean ausaz datu bakarra erabiltzen du gradientea kalkulatzeko. Honek algoritmoa azkartzen du eta datu-base handiekin erabil daiteke, iterazio bakoitzean memorian soilik datu bakarra egongo baita. Hala ere, ez da BGD bezain egonkorra.
- **Mini-batch Gradient Descend (MBGD):** Aurreko bien konbinaketa bat da. Iterazio bakoitzean ausaz datu multzo bat hartzen du gradientea kalkulatzeko. Modu honetan, BGD eta SGD-aren abantailak ditu.

Tamalez, kostu-funtzio guztiak ez dira 2.3 irudia bezalako kurba perfektu bat. Izan daiteke kostu-funtzio batek hainbat minimo lokal[3] izatea, eta minimo absolutuan[4] konbergitu ordez, lokal batean konbergitzea. Zeladura puntuak[5] ere arazo bat dira, bertan kostu-funtzioa konstantea baita eta ondorioz gradiantea nulua (ikusi 2.8 irudia).



2.8 Irudia: Zeladura puntua (eskuinean) eta minimo lokala (ezkerrean) [9].

Arazo hauek gainditzeko, gradiente-jaitsieraren aldaerak diren hainbat algoritmo erabiltzen dira, haien artean, Momentum optimizazioa, Nesterov Azelerazio Gradientea, AdaGrad, RMSProp eta Adam optimizazioa.

2.4.2.2. Momentum optimizazioa

Pilota bat malda handiko aldapa batean ipintzean, gutxinaka-gutxinaka abiadura handiagoa hartzen joango da muturreko abiadurara[6] heldu arte. Ideia hau optimizazio-algoritmo honen oinarria da. 2.3 ekuazioan ikus daiteke gradiente-jaitsierak ematen dituen pausoak η eta kostu-funtzioaren gradienteak zehazten dituztela; eta ondorioz, gradiente lokala txikia bada, emango den pausoa txikia izango da. Momentum optimizazioan ez da hau gertatzen. Algoritmo honek, θ-ren eguneraketan momentuaren termino bat txertatuz, aurreko gradienteak kontuan hartzen ditu, hurrengo ekuazioan ikus daitekeen bezala.

$$\begin{aligned} &1. \quad m \leftarrow \beta - \eta \nabla_\theta J(\theta) \\ &2. \quad \theta \leftarrow \theta + m \end{aligned} \tag{2.4}$$

Iterazio bakoitzean, η eta gradiente lokalaren arteko biderketa θ bektoreari kendu beharrean, $\mathbf{m}$ momentu bektoreari kentzen zaio, eta θ soilik bere buruari momentu bektorea batuz eguneratzen da. Oraingoan, β parametro berria daukagu. Parametro honi

[3]Minimo lokal batean, funtzioak balio txikiena du hurbileko puntuekin alderatuta. Baina, izan daiteke puntu hori funtzioko baliorik txikiena ez izatea.

[4]Funtzioko baliorik txikiena duen puntua da.

[5]Funtzio baten lehen deribatu partzialak zero diren baina funtzioaren maximo edo minimoa ez den puntua da.

[6]Muturreko abiadura objektu batek fluido batetik (orokorrean airea) erori ahala lortzen duen abiadura maximoa da.

esker, θ-ren aurreko balioak kontuan hartzen dira.

Demagun momentuaren hasierako balioa m_0 dela eta $G_i = \eta \nabla_\theta J(\theta_i)$. Ordunan, hurrengo iterazioetan:

$$\begin{aligned}
&\textbf{i=1:} \quad m_1 = \beta m_0 - G_1 \\
&\textbf{i=2:} \quad m_2 = \beta m_1 - G_2 = \beta^2 m_0 - \beta G_1 - G_2 \\
&\textbf{i=3:} \quad m_3 = \beta m_2 - G_3 = \beta^3 m_0 - \beta^2 G_1 - \beta G_2 - G_3
\end{aligned}$$

β parametroaren balioa 0 (*marruskadura* handia) eta 1 (*marruskadura* txikia) tartean egonik, hasierako gradienteak kontuan hartzen diren arren, geroz eta iterazio gehiago egin, orduan eta gutxiago hartuko dira kontuan ($\beta^3 < \beta^2 < \beta$).

Egiazta daiteke gradientea konstante mantentzen baldin bada eta $\beta = 0{,}9$ (orokorrean balio hau erabiltzen da) aukeratuz, momentum optimizazioa gradiente-jaitsiera baino 10 aldiz azkarrago joango dela (zeinua kontuan hartu gabe):

$$m_i = \beta m_{i-1} - \eta \cancel{\nabla_\theta J(\theta_i)} \Rightarrow m_i = m_0 = m$$
$$m = \beta m - \eta \nabla_\theta J(\theta_i) \Rightarrow m = -\frac{1}{1-\beta} \eta \nabla_\theta J(\theta_i)$$

Eta $\beta = 0{,}9$ ordezkatuz:

$$\boxed{\text{m = -10} \cdot \eta \nabla_\theta J(\theta_i)}$$

Honek malda gorakor batetik errezago ihes egitea ahalbidetzen du.

2.4.2.3. Nesterov Azelerazio Gradientea (NAG)

Momentum optimizazioaren aldaera bat da (ikusi 2.5 ekuazioa). Algoritmo honen oinarria ondokoa da: kostu-funtzioaren gradientea posizio lokalean neurtu beharrean, apur bat momentuaren norabiderantz mugitu eta bertan neurtzen du.

$$\begin{aligned}
&1. \quad m \leftarrow \beta - \eta \nabla_\theta J(\theta + \beta m) \\
&2. \quad \theta \leftarrow \theta + m
\end{aligned} \tag{2.5}$$

Aldaera hau ondo dabil orokorrean momentu bektorea puntu optimoaren norabiderantz jotzen duelako, eta ondorioz, egokiagoa izango da gradientea norabide horretan pixka bat urrunago aplikatzea puntu lokalean aplikatzea baino [9].

2.4.2.4. AdaGrad

Adaptative Gradient-etik dator, hau da, egokitzen den gradientea, eta oso erabilgarria da dimentsio ugariko kostu-funtzioetarako. 2.6 ekuazioan ikus daiteke oraingoan momentu bektorea erabili beharrean, s bektorea erabiltzen dela.
Demagun kostu-funtzioa bi dimentsiotako espazioan dagoela, zeinetan lehenengo dimentsioan zehar kostu-funtzioa malda txikikoa den eta bigarren dimentsioan zehar malda handikoa den. Iterazio bakoitzean, lehendabizi, θ_i pisuen gradienteen karratua s_i bektorean metatzen da, eta modu honetan, lehenengo dimentsioan s_1 geroz eta handiagoa

egingo da iterazio bakoitzean, bigarren dimentsioan s_2 geroz eta txikiagoa egiten den bitartean. Bigarrenik, pisuak eguneratzen dira, baina oraingoan gradientea s_i bektorearen eta ϵ terminoaren (orokorrean $\epsilon = 10^{-10}$ izan ohi da, zerorengatik zatitzea saihesteko) arteko batuketaren karratuarekin zatitzen da. Lehenengo dimentsioan (kostu-funtzioak malda txikikoa deneko dimentsioan), zatitzailea oso txikia denez, zatiduran zenbaki handi bat lortuko dugu, eta bigarren dimentsioan, aldiz, kontrakoa. Esan daiteke gradientea eskalatu egiten dela dimentsio bakoitzeko pisuak ratio berdinarekin eguneratzeko.
Modu honetan lortzen da gradiente handiko zonaldeetan *learning rate*-a txikia izatea eta gradiente txikiko zonaldeetan, aldiz, handia izatea. Beraz, modu eraginkor eta azkar batean gradientea nulua den zonalde batera iritsiko da.

$$\begin{aligned} &1. \quad s \leftarrow s + (\nabla_\theta J(\theta))^2 \\ &2. \quad \theta \leftarrow \theta - \frac{\eta}{\sqrt{s+\epsilon}} \nabla_\theta J(\theta) \end{aligned} \tag{2.6}$$

Hala ere, optimizazio-algoritmo hau soilik zeregin errazetarako erabiltzen da eta ez da guztiz egokia neurona-sareak entrenatzeko. Kasu askotan η parametroa gehiegi txikitzen da eta algoritmoa puntu optimora iritsi baino lehen gelditu ohi da, $\frac{\eta}{\sqrt{s+\epsilon}}$ eta gradientearen arteko biderketa anulatzen baita.

2.4.2.5. RMSProp

Algoritmo honek AdaGrad algoritmoaren arazoa konpontzen du ekuazioan $\beta - 1$ parametroa sartuz (ikusi 2.7 ekuazioa). Momentum optimizazioan ikusi den bezala, β parametroari esker, egindako azkeneko iterazioek pisu handiagoa hartzen dute; baina kasu honetan, $\beta - 1$ parametroa daukagu, eta ondorioz, azkeneko iterazioetako gradienteak kontuan hartzen diren arren, gradiente lokala da pisu gehien hartzen duena. Modu honetan, *learning rate*-a ez da hainbeste txikitzen, eta algoritmoa puntu optimora iritsi baino lehen gelditzea saihesten da.

$$\begin{aligned} &1. \quad s \leftarrow \beta s + (1-\beta)(\nabla_\theta J(\theta))^2 \\ &2. \quad \theta \leftarrow \theta - \frac{\eta}{\sqrt{s+\epsilon}} \nabla_\theta J(\theta) \end{aligned} \tag{2.7}$$

Algoritmo hau, problema errezegietan izan ezik, gehienetan emaitza onak ematen ditu eta oso erabilia izan zen Adam optimizazioa agertu zen arte.

2.4.2.6. Adam optimizazioa

Adaptative moment estimation-etik dator (egokitzen den momentuaren estimazioa). Momentum optimizazioa eta RMSProp konbinatzen dituen algoritmoa da, hots, soilik iteratu berri diren gradienteak eta hauen karratua kontuan hartzen ditu.

$$\begin{aligned} &1. \quad m \leftarrow \beta_1 m - (1-\beta_1)\nabla_\theta J(\theta) \\ &2. \quad s \leftarrow \beta_2 s + (1-\beta_2)(\nabla_\theta J(\theta))^2 \\ &3. \quad m \leftarrow \frac{m}{1-\beta_1^i} \\ &4. \quad s \leftarrow \frac{s}{1-\beta_2^i} \\ &5. \quad \theta \leftarrow \theta + \frac{\eta}{\sqrt{s+\epsilon}} \hat{m} \end{aligned} \tag{2.8}$$

2.8 ekuazioan 1. pausoak Momentum optimizazioaren itxura dauka, eta esan bezala, $1-\beta_1$ parametroak gradientea biderkatzean, gradiente lokalari aurreko gradienteei baino pisu handiagoa ematen zaio. 2. pausoak, aldiz, RMSProp-en antzekoa da, eta berriz ere, gradiente lokalaren karratua da kontuan gehien hartzen den terminoa. 3. eta 4. pausoetan, m eta s-ren zuzenketa gertatzen da, hots, bi huen balioak doitu egiten dira aurreko irterazioen eragina leunduz. Azkeneko pausoan, pisuak eguneratzen dira.

Adam optimizazioa, ikasketa sakoneko arloan, gaur egungo algoritmoritmoetatik eraginkorrenetariko bat dela frogatu egin da. Izan ere, atal honetan zehar ikusi den bezala, azkenaldian proposatu diren algoritmo guztien ukituak ditu, bakoitzak dituen abantailak bereganatuz.

Lan honetan, Adam optimizazioa erabili da, nahi egin izan den atazarako hobekien jorratzen duen algoritmoa baita.

2.4.3. Aktibazio-funtzioak

Aktibazio-funtzioak garrantzi handia dauka neurona-sare artifizial baten funtzionamenduan. Izan ere, aktibazio-funtzioaren arabera lortutako emaitzak nabarmenki alda daitezke. Lehen aipatu den bezala, aktibazio-funtzioari esker, neurona, beharrezkoa denean, kitzikatu egiten da. Modu honetan, sarean ez-linealtasuna lortzen da, eta konplexuak diren patroiak hauteman daitezke. Datuen eta neurona-sareak egin behar duen lanaren arabera aktibazio-funtzio bat edo beste bat aukeratu beharko da [10].

2.4.3.1. Sigmoide

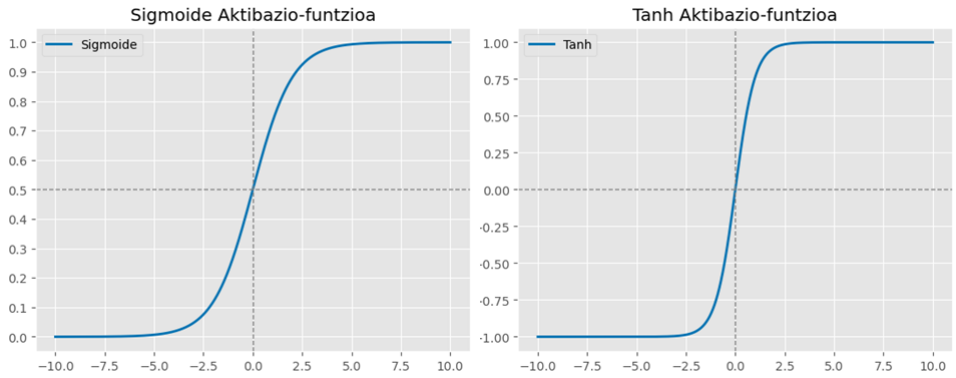
Sigmoide aktibazio-funtzioa 2.9 irudiko forma matematikoa du. Ikus daitekeen bezala, funtzio honek, sarrera edozein dela ere, bere irteeran 1 eta 0 tartean dagoen balio bat sortzen du. Sarrera zenbaki positibo oso handia bada, funtzioaren irteeran 1 lortuko da, eta moduluz oso handia den zenbaki negatibo baten aurrean, aldiz, funtzioak 0 itzuliko du. Aktibazio-funtzio honen irteerak neuronaren kitzikapen-maiztasunaren berri ematen du: kitzikatzen ez bada 0 itzultzen du eta aldiro kitzikaturik badago 1 itzultzen du.

$$\phi(x) = \frac{1}{1 + e^{-z}} \tag{2.9}$$

Hala ere, sigmoide aktibazio-funtzioa azkenaldian ospea galtzen joan da bi arrazoi nagusirengatik:

1. **Gradientea desagertu:** Aktibazio-funtzio hau bere limiteetan saturatzen da, eta ondorioz, muga horietan dauden gradienteak anulatzen dira. Izan ere, sigmoide aktibazio-funtzioaren deribatua $\phi'(z) = \phi(z) \cdot (1 - \phi(z))$ da, eta z_0 funtzioaren mugetan dagoen sarrera bada, $\phi'(z_0) = 0$ izango da. Gradientea funtzio honen irteeragatik biderkatzen da amaierako funtzioa lortzeko, baina neuronaren irteera nulua denez, sarean zehar ez da seinalerik hedatuko. Horregatik ere, hasierako pisuak ezin dira oso altuak izan, bestela neuronak asko saturatuko direlako.

2. **Irteerak zeron zentratu gabe:** Funtzioaren irteerak beti positiboak dira zero balioan zentraturik ez daudelako. Gradiente-jaitsieran zehar, pisuen gradientea atzeranzko hedapenean negatiboa edo positiboa izan daiteke, baina aktibazio-funtzioaren irteera beti positiboa denez, pisuen eguneraketa batzuk norantza batean egingo dira eta beste batzuk kontrako norantzan, optimizazioa zailduz [10].



2.9 Irudia: Sigmoide eta Tanh aktibazio funtzioak.

2.4.3.2. Tanh

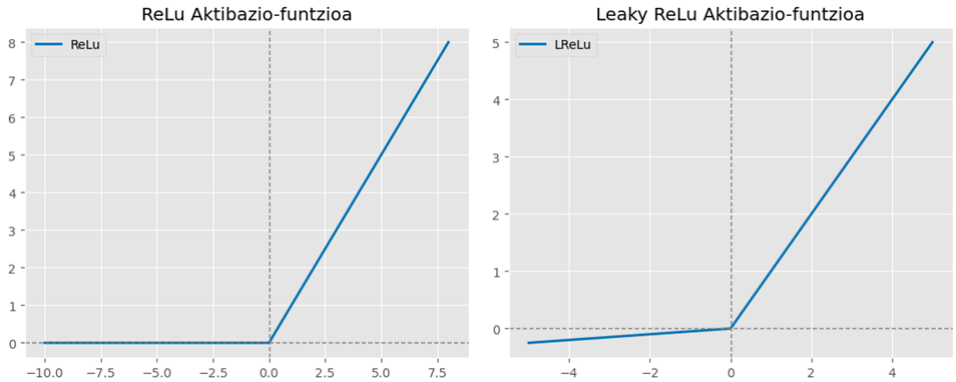
Tangente hiperboliko aktibazio-funtzioa sigmoide funtzio desplazatu bat dela esan daiteke. Funtzio hau 0 balioan zentraturik dago eta oraingoan, irteerak [-1,1] tartean egongo dira, non moduluz handiak diren zenbakiak -1 eta 1 balioak hartuko dituzten. Hortaz, irteerak ere zeron zentraturik egongo dira eta sigmoide aktibazio-funtzioaren arazoetako bat konpontzen da. Horregatik, aproposagoa da geruza ezkutuetan funtzio hau erabiltzea [10].

2.4.3.3. ReLu

Relu (*Rectified Linear Unit*) 2.10 ekuazioan bezala definitzen da, eta ikus daiteke soilik balio negatiboetarako zeron saturatzen dela, hots, gradienteen balio positibo handien aurrean ez da saturatzen (ikusi 2.10 irudia).

$$\phi(x) = max(0, x) \tag{2.10}$$

Aktibazio-funtzio hau, aurretik azaldutakoekin alderatuta, hainbat abantaila dauzka: Batetik, SGD-aren konbergentzia nabarmenki azeleratzen du (horren arrazoia balio handietarako saturatzen ez dela uste da). Bestetik, errazagoa da konplexuak diren neuronetan inplementatzen. Hala ere, desabantaila bat dauka: ReLu erabiltzen duten neuronak ahulak dira eta entrenamenduan zehar 'hil' daitezke. Neuronan etengabe gradiente handiak sartzen badira, pisuak modu jakin batean eguneratzeagatik neurona desaktibatzea eta berriro ezin aktibatzea eragin dezake. Honen ondorioz, neurona horretatik igarotzen diren gradienteak beti anulatuko dira. Hala ere, hau arazo bat izan ohi da gehien bat *learning rate* parametroa sobera handia denean [10].



2.10 Irudia: ReLu eta Leaky ReLu aktibazio funtzioak.

2.4.3.4. Leaky ReLu

ReLu neuronen desaktibazio itzulezinaren arazoa konpontzeko saiakera bat da. Gradientearen balio negatiboetarako aktibazio-funtzioa 0 izan beharrean, malda positibo leun bat (0,01 ingurukoa) ipintzen da, hots, funtzioak 2.11 ekuazioa kalkulatzen du, non α balio txikiko konstante bat den. Askotan emaitza onak ematen dituen arren, ez dira beti guztiz egokiak [10].

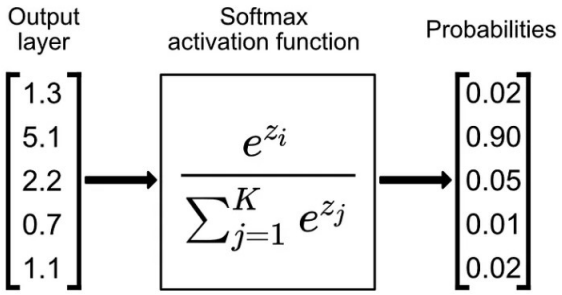
$$\phi(x) = max(\alpha \cdot x, x) \tag{2.11}$$

2.4.3.5. Softmax

Softmax aktibazio-funtzioa geruza ezkutuetan erabili beharrean, irteera-geruzan erabiltzen da. Sarrera-balio multzo bat banaketa probabilistiko batean bihurtzen du, eta horregatik oso funtzio erabilgarria da klasifikazio problemetarako; bereziki klase anitzeko klasifikazio problemetarako. Hortaz, softmax funtzioak emandako puntuaketen edo probabilitateen batuketak 1 emango du. Iragarritako klasea sarrera-multzotik probabilitate-balio handiena lortu duen elementuak zehazten du.

$$\phi(x) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{2.12}$$

2.12 ekuazioan funtzio honen adierazpen matematikoa adierazten da. Sarrera-multzoaren elementu bakoitzaren esponentziala batu egiten da, eta elementu bakoitzaren probabilitatea lortzeko bakoitzaren esponentziala egindako batuketagatik zatitzen da. 2.11 irudian honen adibide bat ikus daiteke [11].



2.11 Irudia: Softmax aktibazio-funtzioaren adibidea [11].

2.5. Lengoaia Naturalaren Prozesaketa

Lengoaia Naturalaren Prozesaketa (NLP bere ingelesezko siglengatik) azken urteetan hazkunde gehien jasan duen adimen artifizialaren alderdietako bat izan da. Ikasketa automatikoaren adar hau ordenagailuei testuak eta ahozko hitzak ulertzeko ahalmena ematearen arduraduna da. NLP-ak linguistika konputazionala[7] eta eredu estatistikoak konbinatzen ditu, eta testu edota ahozko datuetaz baliatuz, ordenagailuek gizakion lengoaia prozesatzea eta esanahai osoa 'ulertzea' ahalbidetzen du. Esanahiaz gain, hizlari edo idazlearen asmoa eta sentimendua ere antzematea posiblea da.

Gaur egun NLP-ak aplikazio ugari dauzka, hala nola, desiratzen ez diren mezuen detekzioa, hizkuntza-itzulpen automatikoa, agente birtualak eta chatbot-ak, testuen laburpen automatikoak eta sare sozialetako iritzien analisia. Lan honetan NLP-a azken aplikazio

[7] Giza hizkuntzaren arauetan oinarritutako ereduztapena.

honetarako erabiliko da, eta horregatik, jarraian, analisi hori lortzeko teknologia azalduko da [12].

2.5.1. BERT

BERT (ingelesetik Bidirectional Encoder Representations from Transformers) Googlek 2018an sortutako testua prozesatzen duen lengoaia-eredu bat da. Transformerretan oinarritzen da eta gaur egun erabilgarri dagoen aurreentrenatutako lengoaia-eredu ezagun eta eraginkorrenetako bat da. Transformer bat neurona-sarez osatutako eredu bat da, aditasun mekanismo bati esker, esaldi baten hitzen arteko erlazioak aztertzeko gai dena [13].

BERT sortu zen arte, lengoaia prozesatzeko eredurik erabilienak (Word2Vec, FastText eta OpenAI GPT 1, besteak beste) hitz bakoitzaren esanahian oinarritzen ziren gehien bat, eta zenbait hitzen arteko erlazioak kontutan hartzen zituzten arren, esaldiak soilik ezkerretik eskuinera edo eskuinetik ezkerrera prozesatzen zituzten. Horrek testu bat bere osotasunean ulertzeko gaitasuna mugatzen zuen. Adibidez, *'etxe ondoko bankuan dirua atera nuen'* esaldiaren aurrean, Word2Vec bezalako ereduek hitz bakoitzaren esnahiaren arabehera bektore bat itzultzen zuten, *'bankua'* diru-bankua edo esertzeko bankua desberdindu gabe. Aldiz, esaldia norantza batean prozesatzen zutenek, *'bankua' 'etxe ondoko'* hitzen arabera prozesatzen zuten, baina *'dirua atera nuen'* kontutan hartu gabe [14].

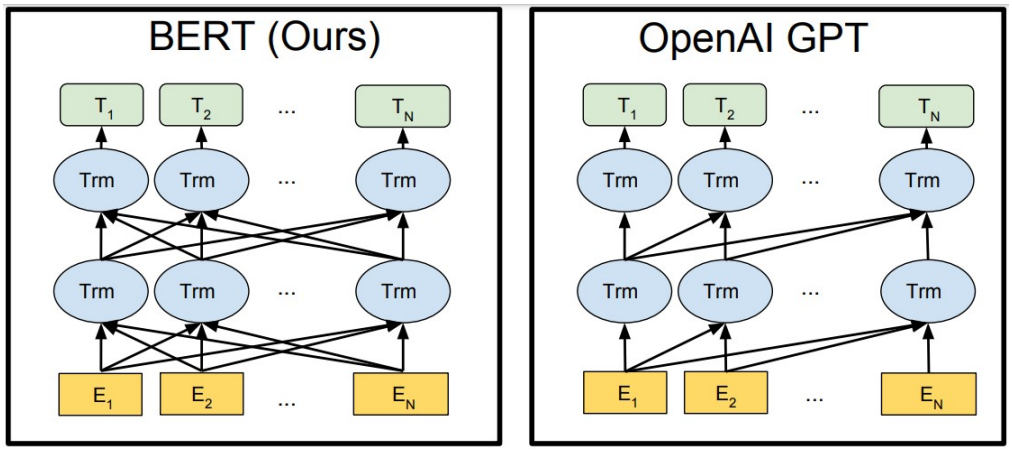
Testuinguruaren ulermenaren muga horiekin bukatzeko, BERT bezalako eredu bat behar zen. Eredu hau hitzen esanahiaz gain, hitz bakoitzaren testuingurua kontuan hartzen du, hots, bai eskuinean bai ezkerrean dauden hitzak aintzat hartzen ditu, testua nabarmenki 'ulertzea' ahalbidetuz. Aurreko adibidean, *'bankuan'* hitza aurreko zein atzeko hitzen arabera prozesatuko luke [14].

Hortaz, BERT-ek esaldiak bi noranzkoetan prozesatu ditzake, eta hau lortzeko, zeregin ezberdinak betetzeko aurrentrenatu zen. Ereduari input bezala esaldi bat ematen zitzaion, zeinaren hitz bakoitza *token* bat zen. Ausaz, bi token maskaratzen ziren, eta ereduaren lana maskaratutako token horietan hasiera batean zeuden hitzak igartzea zen. Modu honetan, ereduak token horien inguruko hitzetan arreta jartzen ikasi zuen, hots, testuingurua bi norantzetan kontuan hartzen ikasi zuen. Horrela entrenatutako ereduei, ingelesez, Masked Language Model (MLM) deritze. Halaber, input bezala berriz ere esaldi bat emanda, hurrengo esaldia iragartzeko zeregina betetzeko entrenatu zuten.

Aditasun mekanismo horiekin eta ingelesezko korpus handiekin aurrentrenatzeari esker BERT transformerra eraiki zen, eta input bezala esaldi bat hartzean, esaldiaren semantika eta testuingurua errepresentatzen duen bektore numeriko bat lortzea ahalbidetzen du. Esan bezala, ingelesezko korpusekin aurrentrenatu zen, eta hortaz, soilik ingelesezko esaldiak prozesatu ditzake. Hala ere, BERT sortu zenetik, teknologia horretan oinarritzen diren transformer asko atera dira, hala nola, RoBERTa, hainbat hizkuntzatan aurrentrenatutako eredua, haien artean, gaztelera. RoBERTa lan honetan erabili den transformerra da.

Aipatutakoa 2.12 irudian garbi ikusten da. Horiz, esaldiaren *token* bakoitza adierazten da; urdinez, *token* horiek neurona-sarean zehar jasandako eraldaketak eta berdez, ereduak

itzultzen duen bektore numerikoa. OpenAI GPT 1-ren kasuan ikus daiteke esaldia ezkerretik eskuinera prozesatzen dela, non azkeneko *token*-a ezkerraldean dituen *token*-ak aintzat hartzen dituen, baina ezkerreko *token*-ak ez diren eskuinean dituztenekin lotzen. BERT-en kasuan, berriz, ikus daiteke esaldiaren *token* guztiak neurona guztietara doaztela, hots, prozesaketa ez da norantza bakarrean gertatzen, baizik eta bi norantzetan.



2.12 Irudia: BERT eta OpenAI GPT 1 ereduen alderaketa [14].

2.5.2. RoBERTa

RoBERTa (ingelesetik Robustly Optimized BERT pre-training Approach) BERT transformerrean oinarritzen da, baina desberdintasun batzuk dauzka. Batetik, eredua luzaroago eta korpus handiagoekin aurrentrenatu zen. Gainera, korpus hauek hizkuntza ezberdinetakoak ziren (gaztelera, ingelera, frantsesa eta alemaniera, besteak beste). Bestetik, aurrentrenamendutik hurrengo esaldiaren iragartzeko zeregina baztertu zuten eta maskaratze patroian aldaketak egin zituzten [15].

Lan honetan RoBERTa erabiltzea erabaki da gazteleraz idatzitako esaldiekin lan egingo delako, eta ondorioz, helburuak lortzeko transformerrik egokiena delako. Eredu honek aurrentrenatutako edozein hizkuntzako esaldi bat input bezala jasotzean *embedding* deritzon 768-ko taminako (768 zenbakiz osatutako) bektore numerikoa itzultzen du.

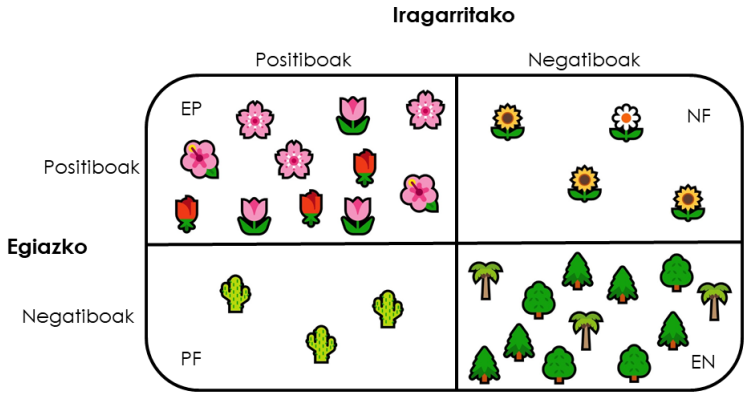
2.6. Ebaluazio metrikak

Edozein ikasketa automatikoko algoritmoa ebaluatzea funtsezkoa da eraikitako eredua erabiltzeko prest dagoen edo hobekuntzak behar dituen jakiteko. Orokorrean, datu-base eta algoritmo egokiak erabiliz, *zehaztasuna* metrikak emaitza egokiak agertzen ditu, baina, ez da nahikoa eredua benetan ondo jarduten duela egiaztatzeko. Horrenbestez, informazio gehiago ematen duten beste metrika batzuk erabiltzen dira [16].

Metrika garrantzitsuenak azaldu aurretik, hainbat kontzeptu azaltzea funtsezkoa da. Klasifikazio lanetan, algoritmoek gehienetan datuak ondo klasifikatzen dituzten arren, zenbaitetan datu batzuk gaizki klasifikatzen dituzte, hau da, datu batzuk ez dituzte dagozkien klasera esleitzen. Demagun klasifikatzaile bitar batek klase konkretu bat detektatu nahi duela. Klasifikatzaileak klase horri esleitutako eta benetan klase horri dagozkion laginak, *egiazko positiboak* (EP, ingelesez TP) deitzen zaie. Aldiz, klase horri esleitutako

baina beste klaseari dagozkien datuei *positibo faltsuak* (PF, ingelesez FP) deritze. Detektatu nahi den klaseari esleitu beharrean, beste klase bati esleitu zaizkien datuak eta benetan klase horri dagozkienak, *egiazko negatiboak* (EN, ingelesez TN) esaten zaie, eta klase horri esleitutakoak baina errealitatean detektatu nahi den klaseari dagozkionak, *negatibo faltsuak* (NF, ingelesez FN) [4].

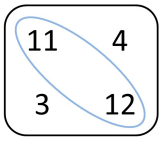
Adibide bezala 2.13 irudia hartuz, loreak detektatzen dituen eredu batek, 30 landaretik (15 lore eta 15 zuhaitz edo kaktus) 11 lore, lore bezala hauteman ditu, 3 zuhaitz, lore bezala detektatu ditu eta gainontzekoak (4 lore eta 12 zuhaitz) loreak ez direla esan du. Ondorioz, egiazko 11 positibo, 3 positibo faltsu, 4 negatibo faltsu eta egiazko 12 negatibo egon dira.



2.13 Irudia: Klasifikatzaile bitar baten egiazko positibo eta negatiboak eta positibo eta negatibo faltsuak.

2.6.1. Konfusio Matrizea

Bere izenak dioen bezala, matrize bat da, ereduaren jokabidea deskribatzen duena. Loreen adibidera itzuliz, 2.14 irudia eredu horren konfusio-matrizea izango litzateke. Diagonal nagusiko balioen batezbestekoa eginez, konfusio-matrizerako zehaztasuna kalkula daiteke (ikusi 2.13 ekuazioa). Honi *accuracy* deritzo.



2.14 Irudia: Konfusio matrizea.

$$Accuracy = \frac{EP + EN}{lagin\text{-}kopurua} = \frac{11 + 12}{30} = 0{,}767 \tag{2.13}$$

2.6.2. Klasifikazio-zehaztasuna (*precision*)

Klasifikazio-zehaztasuna ereduak duen zehaztasuna adierazten du. Asmatutako iragarpen-kopuruaren eta guztira dauden lagin-kopuruaren arteko ratioa da (ikusi 2.14 ekuazioa). Ingelesez, *precision* bezala ezagutzen da.

$$Precision = \frac{Asmatutako\ iragarpenak}{Egindako\ iragarpen\ guztiak} \tag{2.14}$$

Baina aurretik aipatutako terminoak erabiliz, berdina da esatea *precision*-a egiazko positiboen eta positibo guztien arteko batuketaren zatiketa dela. Aurreko adibidea berriro hartuz, *precision*-ak ondoko galderari erantzuten dio: haumentan diren lore guztietatik, zenbat dira benetan loreak?

$$Precision = \frac{EP}{EP + PF} = \frac{11}{11 + 3} = 0{,}786 \tag{2.15}$$

Metrika hau erabilgarria da klase bakoitzeko lagin-kopurua berdina edo oso antzekoa bada. Demagun klase bati dagozkion laginak datu-basearen %98 dela, eta beste klaseari %2. Ereduak soilik lehenengo klaseko laginak hautematearekin %98-ko zehaztasuna lortuko luke, baina bi klaseak antzemateko gai izatea da helburua.

2.6.3. Sentsibilitatea (*recall*)

Sentsibilitatea, 2.16 ekuazioan agertzen den bezala, egiazko positiboen eta lagin esanguratsuen kopuruaren arteko zatiketa da (positibo bezala klasifikatu beharko liratekeen lagin kopurua). Ingelesez, *recall* bezala ezagutzen da.

$$Recall = \frac{EP}{EP + NF} = \frac{11}{11 + 4} = 0{,}733 \tag{2.16}$$

Recall metrikak hurrengo galdera erantzuten du: lagin esanguratsu guztien artean, zenbat identifikatu ditu ongi ereduak?

2.6.4. F1 *score*

Metrika hau zehaztasunaren eta sentsibilitatearen arteko batez besteko harmonikoa da (ikusi 2.17 ekuazioa), bi hauen arteko oreka bilatzen baitu. Ereduak zenbateko zehaztasuna (zenbat lagin ondo klasifikatzen dituen) eta mardultasuna (zenbat hutsegite egin dituen) duen adierazten du. Zehaztasun handia eta sentsibilitate txikia duen eredu bat oso zehatza izango da, baina lagin kopuru handi bat ihes egingo zaizkio. F1-ek [0, 1] tarteko puntuaketa bat ematen du; geroz eta handiagoa izan, hobe jokatuko du ereduak.

$$F_1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} = \frac{2}{\frac{1}{0{,}767} + \frac{1}{0{,}733}} = 0{,}75 \tag{2.17}$$

3. Ataza eta Corpusa

Lan honen helburua Espainia mailan estreinaldi zinematografikoek izango duten arrakasta aurresatea da. Hau aurrera eramateko, gainbegiratutako ikasketa erabiliko da, hain zuzen ere, neurona-sare artifizialak. Hortaz, aurretik aipatu den moduan, gainbegiratutako ikasketarako etiketak dituzten datuak beharko dira.

Erabiliko den datu-basea 6945 lagin ditu, eta lagin bakoitzak bost atributu dauzka: pelikularen izenburua, pelikularen nazionalitatea, pelikularen generoa, estreinaldi-data, ikusle kopurua eta lortutako diru-sarrerak. Azken atributu honen arabera, pelikula bakoitzak izandako arrakasta aztertuko da (diru-sarrera eta arrakasta zuzenki proportzionalak baitira) eta pelikula bakoitzari bere arrakasta kuantifikatzen duen etiketa bat esleituko zaio. Ondoren, Twitter-en pelikula bakoitzari buruz egin diren iruzkinak (bere estreinaldia baino lehen) bilatuko dira, ondoren prozesatzeko eta errepresentazio bektorial bat lortzeko (embedding-a). Bektore eta etiketa horiek eredua sortzea ahalbidetuko dute.

Hau guztia lortzeko, lehenengo datu-basea aztertu, iritzien corpusa sortu eta datuak prozesatu beharko dira.

3.1. Datu-basearen analisia

Datu-basearen egitura 3.1 irudian ikus daiteke. Esan bezala, 6945 film daude eta bakoitzak sei ezaugarri ditu.

	TITULO	NACIONALIDAD	ESPECTADORES	RECAUDACION	FECHA_ESTRENO	GENERO
0	ANIMALOPOLIS	ESTADOS UNIDOS	5482	24239.18	02/01/2010	DOCUMENTAL
1	SALIDAS	ESTADOS UNIDOS	2274	8005.50	04/01/2010	PORNOGRÁFICA
2	CAPITALISMO: UNA HISTORIA DE AMOR	ESTADOS UNIDOS	148	541.00	08/01/2010	DOCUMENTAL
3	TENIENTE CORRUPTO	ESTADOS UNIDOS	222	785.00	08/01/2010	THRILLER
4	LICENCIADO EN TOCOLOGIA	ALEMANIA	2345	8604.50	11/01/2010	PORNOGRÁFICA
...	...	...	...	...	...	...
6942	NO ODIARÁS	ITALIA	59	340.50	24/09/2021	DRAMA
6943	NO RESPIRES 2	ESTADOS UNIDOS	5887	35827.69	24/09/2021	THRILLER
6944	PREPARATIVOS PARA ESTAR JUNTOS UN PERIODO DE T...	HUNGRÍA	487	3117.49	24/09/2021	DRAMA ROMÁNTICA
6945	RESPECT	ESTADOS UNIDOS	558	4041.46	24/09/2021	BIOGRÁFICA

3.1 Irudia: Datu-basearen egitura.

Datu-basearekin egin den lehenengo prozedura analisi orokor bat izan da. Ikusi da 85 nazionalitate ezberdinetako pelikulak daudela (Estatu Batuak, Italia, Frantzia, Espainia, etab.) eta osotara 67 genero daudela (abentura, dokumentala, animazioa, komedia, etab.). Estreinaldi-datari dagokionez, 2010 urtetik 2021 urterako pelikulak daude.

Ikusi da lagin batzuk zenbait ezaugarrietan ez dutela informaziorik, eta lagin horiek ezabatzea erabaki da. Halaber, genero ezberdinen artean pornografikoa agertzen dela behatu da, eta Twitter-en jendeak pelikula pornografikoen estreinaldiei buruz gutxi hitz egiten dutela egiaztatu ondoren, genero horri dagozkien film guztiak ezabatzea erabaki da, seguraski zarata sortuko baitute. Aldaketa hauekin, datu-basea 6517 filmekin geratu da.

Azkenik, zenbait animazio-pelikulen izenburuetan estudioaren izena agertzen dela behatu da, hala nola, 'pixar' edo 'dreamworks'. Gainera, beste film batzuetan gaztelerazko izenburua agertzeaz gain, jatorrizko izenburua parentesi tartean agertzen da. Erabaki da bai jatorrizko izenburua bai estudioen izena izenburuetatik kentzea, ondoren, corpusaren bilaketan, tweet gutxiago lortzea eragin baitezakete.

Lagin hauekin, korrelazio analisi bat egin nahi izan da aldagai ezberdinek diru-sarrerarekin (hots, 'RECAUDACIÓN' ezaugarriarekin) duten erlazioak ikusteko. Agian, korrelazio handia duten ezaugarriak jendearen iritziarekin konbinatu daiteke eta eredu eraginkorrago bat sortu. Korrelazioa ikusleekin (aldagai jarraitu bat dena), generoarekin (aldagai ordinala) eta nazionalitatearekin (aldagai ordinala) ikusi nahi izan da, eta horretarako, genero eta nazionalitate bakoitza ezaugarri independenteetan transformatu dira. Hortaz, pelikula bat Alemaniakoa bada, 'ALEMANIA' ezaugarrian '1' zenbakia izango du, eta gainontzeko nazionalitate-ezaugarrietan '0'. Generoekin berdina gertatuko da: abenturetako pelikula guztiak '1' balioa izango dute 'AVENTURA' ezaugarrian, eta '0' gainontzeko genero-ezaugarrietan. Hauek dira lortutako korrelazio-balio esanguratsuenak:

- **Ikusleak/Diru-sarrerak:** Pearson[8] korrelazioa erabili da. 0.996-ko korrelazioa erakutsi du, eta zentzuzkoa da, film batera geroz eta ikusle gehiago joan, orduan eta diru gehiago lortzen baita.
- **Generoak/Diru-sarrerak:** Spearman[9] korrelazioa erabili da. Korrelazio gehien duen generoa abentura izan da, baina soilik 0.127 balioarekin. Dokumentala, aldiz, korrelazio negatiboena erakutsi duen generoa izan da, -0.370 balioarekin. Honek esan nahi du, abenturetako pelikulek diru-sarrera handienak lortzen dituztela gainerako generoekin alderatuta, eta dokumentalak, aldiz, txikienak. Hala ere, korrelazio balio hauek ez dira oso handiak eta, hortaz, ez dira kontuan hartzeko modukoak.
- **Nazionalitateak/Diru-sarrerak:** Spearman korrelazioa erabili da. Korrelazio gehien erakutsi duen nazionalitatea Estatu Batuak izan da, 0.338-rekin. Aldiz, korrelazio negatiboena Espainiak dauka, -0.283-rekin. Hala ere, ez da bidezkoa Espainia analisian sartzea, Espainian estreinatzen diren atzerriko filmek haien herrialdeetan arrakasta izan dutelako eta, ondorioz, Espainian ere arrakasta izango dutelako. Gainontzeko nazionalitateak zero inguruko korrelazioa dute. Hala ere, esan bezala, balio hauek ez dira oso handiak eta, hortaz, ezin dira kontuan hartzeko modukoak.

Beraz, badirudi nazionalitate eta generoek ez dutela erlazio sendorik diru-sarrerekin, baina antza denez, dokumentalak arrakasta gutxien duten filmak dira eta estatu batuetako pelikulak diru-sarrera gehien lortzen dituztenak dira.

Behin datu-basea aztertu eta garbitu dela, lagin bakoitzari etiketa bat ipiniko zaio. Horretarako, diru-sarreran oinarrituz, pelikulak lau klasetan sailkatu dira, eta ondorioz, datu-baseari 'CLASE' izeneko ezaugarria gehitu zaio. Diru-sarreraren arabera, lagin bakoitzaren klasearen aldagaian 1, 2, 3 edo 4 zenbakia izango du.

- **1. Klasea:** 20 milioi eurotik gorako diru-sarrerak dituzten pelikulak.

[8]Pearson korrelazioa bi aldagai jarraitu ebaluatzeko erabiltzen da [7].
[9]Spearman korrelazioa aldagai jarraitu bat eta ordinal bat ebaluatzeko erabiltzen da [7].

- **2. Klasea:** 1 eta 20 milioi euro arteko diru-sarrerak dituzten pelikulak.
- **3. Klasea:** 10 mila eta milioi 1 euro arteko diru-sarrerak dituzten pelikulak.
- **4. Klasea:** 10 mila eurotik beherako diru-sarrerak dituzten pelikulak.

Banaketa hau egitean, klase bakoitzak dauzkan pelikula kopurua 3.1 taulan beha daiteke.

1. Klasea	2. Klasea	3. Klasea	4. Klasea
25	1009	2888	2595

3.1 Taula: Klase bakoitzeko film kopurua.

3.2. Corpusaren lorpena eta prozesaketa

Pelikua bakoitzetik gaztelaniadun jendeak estreinaldia baino lehen izandako iritzia lortzeko, Twitter sare soziala erabili da, eta bertatik tweet-ak modu automatiko batean lortzeko, *snscrape* erabili da.

Snscrape *scraping*[10] egiteko komando-lerro tresna bat da, eta sare sozial ezberdinetatik informazioa lortzeko erabiltzen da. Tresna honi esker, bilaketa aurreratuak egin daitezke, erabiltzaile-profilei buruzko informazio zehatza lortu, argitalpenak eta iruzkinak bildu, eta argitalpenen metadatuak atera (hala nola, *retweet*-ak, *like*-ak, eta abar).

Bilaketak modu automatizatu batean egiteko, pelikularen izenburua eta estreinaldi-data erabiliz, programa bat egin da. Programa honek datu-basetik pelikulen izenburuak iteratiboki hartzen ditu, eta estreinaldi-data eta 3 hilabete lehenagoko denbora tartean, Twitter-en izenburua duten tweet-ak bilatzen ditu.

Aurkitutako tweet-a benetan pelikula bati buruz hitz egiten ari dela ziurtatzeko, izenburuaz gain, 'peli' edo 'trailer' terminoak dituzten tweet-ak soilik erabili dira. 'peli' terminoa gehitzea erabaki da Twitter-en erabiltzen den idazkeran, hitzak gehienetan laburpenak direlako, eta errazagoa da 'peli' duten tweet-ak aurkitzea 'película' dutenak baino. 'trailer' terminoa erabiltzea ere erabaki da orokorrean estreinaldia baino hilabete batzuk lehenago filmaren trailerra ateratzen delako, eta ondorioz, 'trailer' hitza duten tweet dexente egongo direla pentsatu delako. Modu honetan, adibidez, 'AKELARRE' pelikulari buruzko tweet-ak bilatzean, 'he ido a ver el akelarre de Zugarramurdi' bezalako tweet-ak baztertzen dira, ez baitaude filmari buruz hitz egiten.

Hala ere, 10 tweet baino gutxiago aurkitzen badira, tweet-ak 'peli' eta 'trailer' terminoak gabe bilatzea erabaki da, bestela oso tweet gutxi lortuko liratekeelako eta izan daiteke tweet batzuk termino horiek ez eduki arren, pelikulari buruz hitz egiten egotea. Hortaz, 10 tweet baino gutxiago lortzekotan, pelikula horren bilaketa berri bat egingo da, baina 'peli' eta 'trailer' terminoak baztertuz.

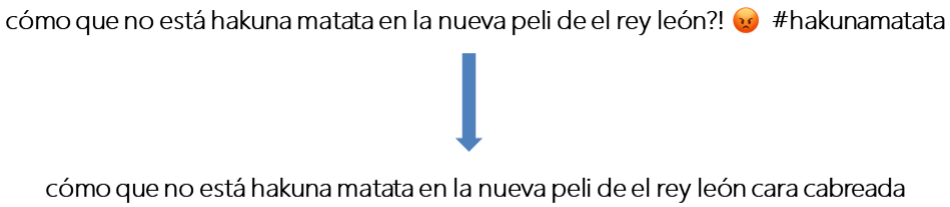
Aipatzekoa da bilaketan ipinitako hitzak tweet-ean agertuko direla, baina ez dutela zertan ordenean agertu behar. Adibidez, 'EL GRAN BAÑO' pelikulan 'el susto de ayer en

[10] *Scraping* ingelesezko termino bat da, web orri bateko datuak automatikoki erauzteko teknikari erreferentzia egiten diona.

el baño hubiera sido un gran vlog' bezalako tweetak ager daitezke, non 'baño' eta 'gran' jarraian ez dauden. Kasu honetan tweet-ak ez du pelikularen inguruan hitz egiten.

Pelikula bakoitzaren tweet-en informazio guztia *csv* fitxategi batean gorde da; fitxategi bat pelikula bakoitzeko. Fitxategi bakoitzak pelikularen izenburua izango du izen bezala, eta terminorik gabeko bilaketak egin badira, hau da, 'peli' eta 'trailer' terminoekin 10 tweet baino gutxiago lortu badira eta, ondorioz, bilaketa berri bat terminorik gabe egin bada, izenaren bukaeran 'noterm' ipiniko zaio, terminorik gabeko tweetak dituzten filmak identifikatu ahal izateko. Lortutako datu-base horretan, tweet horren esteka, tweet-a idatzi duen erabiltzailearen informazioa, tweet-aren edukia edo izandako *retweet*-ak bezalako informazioa agertzen da. Lan honetarako soilik tweet bakoitzaren edukia behar denez, modu automatizatu batean *csv* fitxategi bakoitzetik tweet-en edukia hartu da eta, berriz ere, pelikula bakoitzeko testu-fitxategi bat sortuz, tweet-ak testu-fitxategietan idatzi dira; tweet bat lerro bakoitzeko.

Behin testu-fitxategiak izanik, tweet bakoitzaren garbiketari ekin zaio. Tweet askok hashtagak, estekak, arrobak (erabiltzaile-aipamenak), lerro-jauziak eta testuaren analisirako garrantzitsuak ez diren karaktere kateak dauzkate. Horregatik, pythoneko *re* liburutegia[11] erabiliz ezabatu egin dira, bestela, tweet-ak betore numerikoetan transformatzean zarata izango lukete. Halaber, zenbait tweet emotikonoak izan ditzakete, eta hauek, informazio esanguratsua eman dezakete erabiltzailearen sentimendua edo espresioaren inguruan. Horregatik, hauek ezabatu beharrean, hitzetan transformatzea erabaki da. Horretarako, *emoji* liburutegia erabili da. 3.2 irudian tweet batek jasandako aldaketen adibidea ikus daiteke.



3.2 Irudia: Tweet baten garbiketaren adibidea.

Modu honetan, pelikula bakoitzak tweet garbiak ditu, baina ikusi da zenbait fitxategi hutsik daudela. Hau tweet-ik aurkitu ez deneko seinalea da, eta ondorioz, informaziorik ematen ez duten pelikulak dira. Horrenbestez, lagin hauek ezabatu daitezke, baina hau egin aurretik, corpusaren analisi bat egingo da, 3.3 atalean deskribatzen dena.

Behin tweet gabeko pelikulak ezabatu eta tweet-ak garbitu direla, prest egongo dira RoBERTa lengoaia-ereduan erabiltzeko. Horretarako *hugginface* web-orritik eredua inportatu da. Modu automatizatu batean, pelikula bakoitzetik tweet guztiak 768 tamainako embedding batean transformatu dira eta hauen batezbestekoa egin da, pelikula bakoitzak 768 tamainako bektore numeriko bat izan dezan. Bektore hauekin, *csv* fitxategi bakar bat sortu da, hurrengo zutabeak dituena: filmaren embedding-a eta filmaren klasea (1, 2, 3

[11]Pythoneko *re* liburutegia karaktere-kateetan testu-patroiak bilatzeko, erauzteko eta manipulatzeko erabiltzen da.

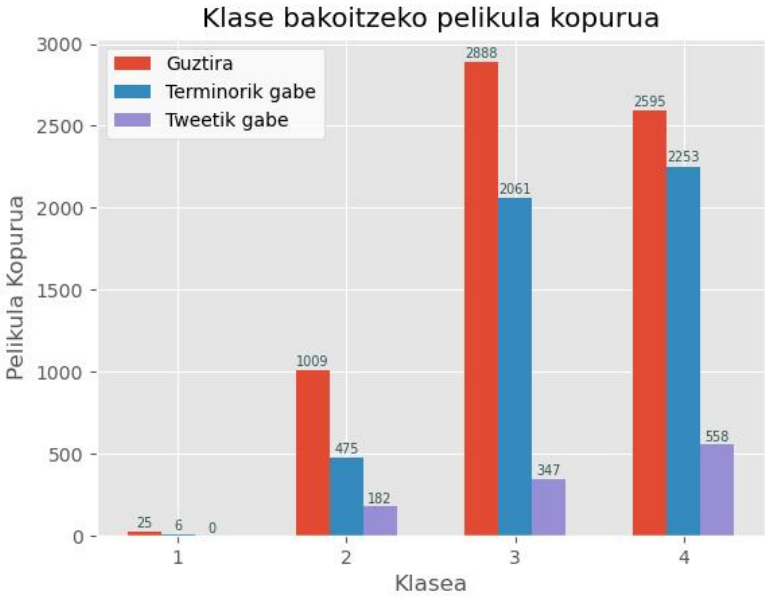
edo 4).

Hortaz, neurona-sarea egiteko erabiliko den datu-basearen ezaugarria embedding-a izango da eta etiketa, berriz, klasea. Dena prest dago neurona-sarea eraikitzeko.

3.3. Corpusaren analisia

Lortutako datuak nolakoak diren aztertzeko, pythoneko *matplotlib* liburutegia erabiliz, hainbat grafiko egin dira.

Hasteko, barra-grafiko bat egin da klase bakoitzean dauden pelikuletatik tweet kopuru egoki bat zenbatek lortu duten aztertzeko (ikusi 3.3 irudia eta 3.2 taula).



3.3 Irudia: Klase bakoitzeko pelikula-kopuruen barra-grafikoa.

Grafikoan, gorriz, klase bakoitzak dituen testu-fitxategien kopurua adierazten da. Pelikula bakoitzeko testu-fitxategi bat dagoenez, gorriz klase bakoitzak duen pelikula-kopurua azaltzen da. Lehen aipatu den bezala, zenbait fitxategien izenetan, pelikularen izenburuaz gain, *noterm* identifikatzailea gehitu zaie. Urdinez, identifikatzaile hori duten fitxategi-kopurua adierazten da, hau da, tweet-etan 'peli' eta 'trailer' terminoak ez dituzten pelikula-kopurua adierazten da. Azkenik, morez, hutsik dauden fitxategien kopurua adierazten da, hots, tweet-ik ez dituzten pelikulak. Hutsik daudenak, noski, izenean *noterm* identifikatzailea izango dute ere, terminorik gabeko bilaketak egin baitira, nahiz eta azkenean pelikula horri buruzko tweet-ek ez aurkitu.

Laburbilduz, gorriz, klase bakoitzeko pelikula-kopurua ageri da, urdinez, pelikula-kopuru horren barruan terminorik gabeko bilaketak izan dituzten pelikula-kopurua adierazten da, eta morez, terminorik gabeko bilaketa horien artean, tweet-ik lortu ez dituzten pelikula-kopurua azaltzen da.

Beraz, zutabe gorria eta urdinaren arteko diferentzia, 3.2 taulako lehenengo zutabea izango litzateke, hots, 'peli' eta 'trailer terminoak dituzten tweet-en pelikula-kopurua. Era berean, zutabe urdina eta zutabe morearen arteko diferentzia, 3.2 taularen bigarren zutabea izango litzateke, hau da, terminorik gabeko tweet-ak dituzten pelikula-kopurua.

	1. klasea	2. klasea	3. klasea	4. klasea
Tweet-etan 'peli' eta 'trailer' terminoak dituzten pelikulak	19 (%76.0)	534 (%52.9)	827 (%28.7)	342 (%13.2)
Tweet-etan 'peli' eta 'trailer' terminoak ez dituzten pelikulak	6 (%24.0)	293 (%29.1)	1714 (%59.3)	1695 (%65.3)
Tweetik gabeko pelikulak	0 (%0)	182 (%18.0)	347 (%12.0)	558 (%21.5)
Guztira	25	1009	2888	2595
	6517			

3.2 Taula: Klase bakoitzeko pelikula-kopuruen eta ehunekoen taula.

Lehenengo zutabean geroz eta pelikula-kopuru handiagoa izan, orduan eta hobe, 'peli' eta 'trailer' terminoak dituzten tweet-ek pelikula bati erreferentzia egiten ari diotela ziurtatzen baitute.

Terminorik gabeko tweet-ak dituzten pelikula kopuruari dagokionez, taulan ikus daiteke lehenengo klasea, proportzioan, kopuru txikiena duen klasea dela. Klasean gora egin ahala, terminorik gabeko tweetak dituzten pelikulen kopurua handitzen doa. Horrek esan nahi du pelikula batek geroz eta diru-sarrera handiagoak lortu, orduan eta gehiago hitz egin dela pelikulari buruz bere estreinaldi-data baino lehen. Azkeneko bi klaseen pelikulen erdiak baino gehiagok ez ditu guztiz egokiak diren tweet-ik, eta horrek eragina izango du neurona-sarea eraikitzerakoan.

Tweet-ik gabeko pelikula kopuruari erreparatuz, 3.2 taularen ehunekoen arbera, lehenengo eta laugarren klasearen artean tweet hutsen proportzioaren aldea oso nabarmena da. Lehenengo klaseak tweet-ik gabeko pelikulak ez dituen bitartean, laugarren klaseak 558 pelikula dauzka tweet-ik gabe; baina, zorionez, pelikula gehien dituen klasea laugarrena da, eta ondorioz, nahiz eta tweet-ik gabeko 558 pelikula izan, tweet-ak dituzten pelikula asko izaten jarraitzen du, eta ondoriz, datu-basea ez da oraindik gehiago desorekatuko. Hirugarren klaseari buruz berdina esan liteke, baina bigarren klasean, aldiz, tweet-ik gabeko 182 pelikula egoteak desoreka handitu dezake, klase horretan ez baitaude hirugarren eta laugarren klasean bezain beste pelikula.

Hortaz, orain, lehenengo eta bigarren klaseak pelikula gutxi izaten jarraitu arren, kalitate handiko tweetak dituzte, eta hirugarren eta laugarren klaseak, aldiz, pelikula asko dituzten arren, tweeten kalitatea ez da hain ona izango.

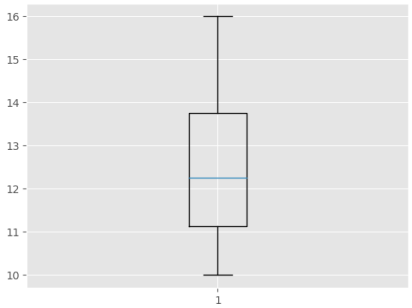
Tweet-ik gabeko pelikulak datu-basetik kendu dira, tweet-ik gabe ez baitago informaziorik. Ondorioz, horrela geratu da datu-basea:

1. klasea	2. klasea	3. klasea	4. klasea	Guztira
25	827	2541	2037	5430

3.3 Taula: Analisia egin ondoren aukeratu den pelikula kopurua.

Argi eta garbi, klaseen arteko pelikula-kopuruaren desoreka bere horretan jarraitzen du.

Datu-basearen izaera hobeto ezagutzeko nahian, klase bakoitzeko *box plot* bat egin da. Grafiko mota hau kaxa baten itxura dauka (ikusi 3.4 irudia), eta datuen banaketaren inguruan informazioa ematen du.

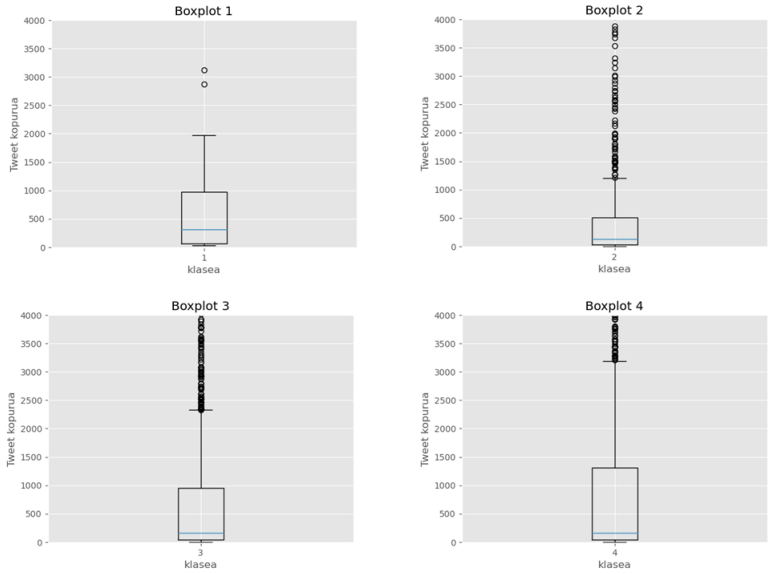


3.4 Irudia: *Box plot* baten adibidea.

Box plot batek bost parametro erakusten ditu. Lan honen kasuan, erabili diren datuak pelikulen tweet-ak direla kontuan hartuz, parametro horietako bakoitzak ondokoa adierazten du:

- Media: Klase bateko pelikula guztien tweet-kopuruaren batezbestekoa da. Parametro hau kaxaren marra urdinak adierazten du.
- Lehenengo kuartila: Klase bateko pelikulek lortutako tweet-kopuruak balio txikienetik handienera antolatzen badira, txikienetik erdiko balioraino dauden kopuruen batezbestekoa da. Parametro hau kaxaren beheko ertzak adierazten du.
- Hirugarren kuartila: Klase bateko pelikulek lortutako tweet-kopuruak balio handienetik txikienera antolatzen badira, handienetik erdiko balioraino dauden kopuruen batezbestekoa da. Parametro hau kaxaren goiko ertzak adierazten du.
- Behe-muturra: $1,5 \times IQR$-ko luzera duen muturra da, non IQR lehenengo eta hirugarren kuartilen arteko diferentzia den. Parametro hau bertikalki kaxaren beheko ertzetik ateratzen den marraren muturrak adierazten du. Mutur horretatik kanpo geratzen diren balioak, balio atipikoak izango dira.
- Goi-muturra: $1,5 \times IQR$-ko luzera duen muturra da, non IQR lehenengo eta hirugarren kuartilen arteko diferentzia den. Parametro hau bertikalki kaxaren goiko ertzetik ateratzen den marraren muturrak adierazten du. Mutur horretatik kanpo geratzen diren balioak, balio atipikoak izango dira [17].

Klase bakoitzeko *box plot* bat egin da, 3.5 irudian ikus daitekeen bezala.



3.5 Irudia: Klase bakoitzaren *box plot*-a.

Beha daiteke tweet-en banaketa pelikulen artean oso desorekatuta dagoela. Izan ere, balio atipiko handi asko daudela ikus daiteke (goi-muturraren gainean dauden puntuak), eta horrek esan nahi du datu-basea balio atipiko anitzez osatua dagoela. Ohartu ere, kaxa (eta batez ere media) zero baliotik oso hurbil dagoela, eta ondorioz, balio tipikoenak oso baxuak dira, hots, ohikoena 200 tweet artean dituzten pelikulak dira. Aipatu beharrekoa da ere, balio atipikoen ondorioz goi- eta behe-muturraren luzerak distortsionatu direla. Hemendik ondorioztatu daiteke, pelikulen gehiengoak tweet-kopuru oso txikia duela eta klase bakoitzak dituen pelikulen artean, desoreka handia dagoela tweet-kopuruari dagokionez.

Analisi hauekin guztiekin, argi geratzen da datu-basea ez dela oso ona, datuak bai klaseen artean (klase bakoitzak duen pelikula kopurua) bai klaseen barnean (pelikula bakoitzak duen tweet kopurua) desorekatuak baitaude. Hau normalean ikasketa automatikoan datu-baseekin gertatzen da, eta lortutako emaitzak datu-base artifizialekin erreplikatzea ez da zeregin erraza izaten. Beraz, honek eragin handia izango du ereduak eraikitzerakoan.

4. Esperimentuak

Atal honetan, lehendabizi, pelikula baten arrakasta iragartzeko gai den neurona-sarea eraiki nahi izan da (A neurona-sarea). Eredua sortzeko hartutako erabakiak azaldu eta honen ebaluaketa egingo da, eta emaitzak ikusita, zenbait hobekuntza proposatuko dira. Hala ere, problemari beste ikuspegi bat ematea erabaki da: etiketatutako lagin bakoitza pelikula bat izan beharrean, tweet-ak izango dira, hots, tweet bat zein klaseri dagokion iragartzea izango da helburu berria. Horretarako, beste neurona-sare artifizial bat erabiltzea erabaki da (B neurona-sarea), eta ondoren, *mayority voting* bat aplikatuko zaio, hau da, pelikula baten tweet bakoitzaren klasea B neurona-sarearekin iragarriko da, eta ondoren, tweet multzo horretan gehien errepikatzen den klasea, pelikula horren klasea dela aukeratuko da.

4.1. A Neurona-sarea

Esan bezala, neurona-sare honek pelikula baten embedding bat sarrera bezala hartuko du eta irteeran zein klaseri dagokion iragartzen ikasi beharko du.

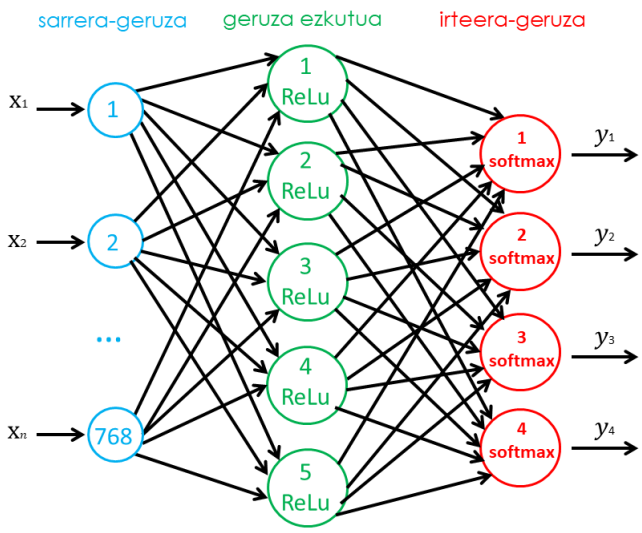
4.1.1. Entrenamendu eta testatze datuak

Neurona-sare artifiziala entrenatzen hasi aurretik, garrantzitsua da datuen zati bat neurona-sarea ebaluatzeko gordetzea. Modu honetan, datuen zati bat nerona-sarea entrenatzeko erabil daiteke eta beste zati bat eraikitako ereduaren eraginkortasuna aztertzeko.

Ahalik eta datu gehien erabili dira entrenamendurako, izan ere, entrenamendu-datu kopurua handitzeak neurona-sarearen parametroak egokiagoak izatea eragingo du. Hala ere, kontuan hartu behar da izan daitekela testatzeko datu gutxirekin geratzea. Kasu honetan, entrenamendurako, datuen %90 erabili da (4887 lagin), eta, ondorioz, eredua testatzeko, datuen %10 (543 lagin). Entrenamendurako erabiliko diren datuak *csv* fitxategi batean gorde dira, eta testatzeko datuak beste *csv* batean. Noski, banaketa hau klaseka egin da, hots, klase bakoitzaren pelikulen %10 testatzeko erabili da, hain zuzen ere: lehenengo klaserako 3 lagin, bigarren klaserako 92, hirugarrenerako 241 eta laugarrenerako 207. Banaketa honen kodea *KODEA.zip* artxibo konprimatuan ikus daiteke, *3.Klaseen Banaketa* karpetan.

4.1.2. Arikitektura

Sortutako neurona-sarearen eskema 4.4 irudian ikus daiteke.



4.1 Irudia: A neurona-sarearen eskema.

Sarearen sarrera 768 tamainako bektore numeriko bat denez, sarrera-geruzak 768 neurona izan behar ditu, eta hortaz, irudiko x_i parametroak bektorea osatzen duten zenbakiak dira. Geruza ezkutu bakarra erabili da, bost neuronaz osatutakoa. Hasiera batean bi geruza-ezkutu erabili ziren, neurona gehiagorekin, baina beste konbinazio batzuk frogatuz, bost neuronaz osaturiko geruza ezkutu bakarra ipintzea egokia dela ikusi da. Geruza- eta neurona-kopurua aukeratzean kontuan hartu behar da oreka bat egon behar dela datuetatik ezaugarri esanguratsuak ikasteko ahalmenaren eta datu berriak orokortzearen artean. Geruza eta neurona gehiegi aukeratuz gero, *overfitting*-a gertatzeko arriskua egongo da. Geruza ezkutuaren neuronek ReLu aktibazio funtzioa erabiltzen dute, aktibazio honekin emaitza hoberenak lortu baitira. Izan ere, aktibazio funtzio honek ereduak patroi konplexuagoak antzematea ahalbidetzen du.

Azkenik, lau klase daudenez, irteera-geruza 4 neurona osatzen dute; baina, oraingoan, softmax aktibazio-funtzioarekin, probabilitate banaketa bat lortzeko. Ereduaren arabera, irudiko y_i parametroetatik handiena dena zehaztuko du pelikula zein klasetakoa den. Adibidez, 'AGUR ETXEBESTE!' filmari dagokion embedding-a neurona-sarean sartzen bada eta irteera-geruzatik [0.08,0.21,**0.41**,0.30] bektorea ateratzen bada, ereduaren arabera, film hori hirugarren klasekoa izango da, hots, 10.000€ eta 1.000.000€ arteko diru-sarrerak izango ditu.

Optimizazioaren aldetik, erabilitako algoritmoa Adam izan da. Bere hiperparametroak ukituz, 4.1 taulako balioak hartzea erabaki da, balio horiekin emaitza hoberenak lortu baitira.

η	β_1	β_2	ϵ
0,00095	0,99	0,999	10^{-7}

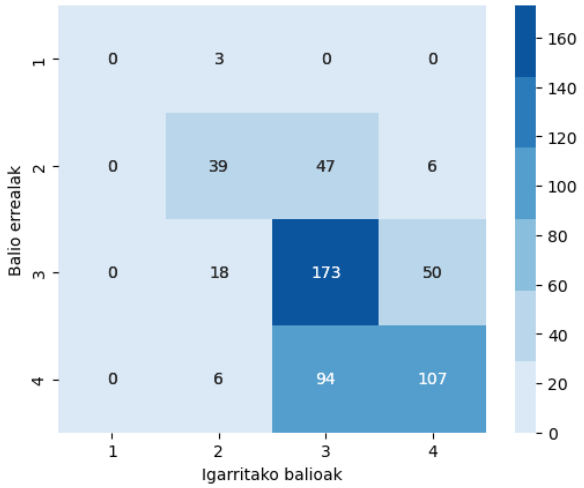
4.1 Taula: A sarearen Adam optimizaziorako hartutako hiperparametroen balioak.

4.1.3. Emaitzak

A neurona-sarearen emaitzak aztertzeko, aurretik azaldu diren ebaluazio metrikak erabiliko dira, hots, konfusio matrizea eta F1 *score*.

- **Konfusio matrizea**

  Konfusio matrizea aztertuz ondorio asko atera daitezke ereduaren inguruan.



4.2 Irudia: A neurona-sarearen konfusio matrizea.

4.2 irudian eraikitako ereduaren matrizea beha daiteke, *seaborn* eta *matplotlib* liburutegiak erabiliz egindakoa. Hasteko, ikus daiteke ez dituela lehenengo klaseko pelikularik hauteman. Eredua lehenengo klaseko 3 pelikulekin testatu da, eta bigarren klasekoa bezala klasifikatu ditu. Horrek esan nahi du eredua lehenengo klaseko 22 pelikularekin entrenatu dela, eta ez direla lagin nahikoak izan ereduak lehenengo klaseko pelikulak antzematen ikas dezan. Hortaz, klase horretako hain lagin gutxi izanda, esan daiteke ereduan zarata eragiten dutela.

Horregatik, beste eredu bat sortu da eta datuak lau klasetan banatu beharrean, hirutan banatu egin dira, lehenengo eta bigarren klaseak konbinatuz. Modu honetan, lehenengo klaseak sor dezakeen zarata leundu daiteke, klaseak konbinatzean ez baita soilik 25 pelikula dituen klaserik egongo. Hau egitean, ez dira emaitza hoberik lortu, eta horregatik lan honetan ez da honi buruz gehiago sakonduko.

A neurona-sarearekin jarraituz, hirugarren klasekoak izan dira gehien iragarri diren pelikulak, baina egia da, orokorrean, testatze-datu dexente hirugarren klasekoak bezala hauteman dituela. Honek esan nahi du ereduak espezifikotasun txikia duela, hots, hirugarren klaseari dagokionez, egiazko negatibo asko gaizki klasifikatzen dituela.

Laugarren klaseari dagokionez, klase horretako datuen erdia ondo klasifikatu dituela ikus daiteke, eta gaizki klasifikatuak, hirugarren klasera esleitu ditu.

Hortaz, argi ikusten da eredua hirugarren klasea nahikotxo orokortzen ari dela, klase horri ez dagozkien datu asko bertan esleitu baititu. Honen kausa, aurretik ikusi den bezala, datuen-desoreka bat edo eredua sinpleegia izatea izan daiteke. Eredua hasiera batean konplexuagoa egin da, neurona gehiagotako bi geruza ezkutu ipiniz, eta antzeko emaitzak ematen zituen arren, hau hobea dela ikusi da. Gainera, hirugarren klaseak besteak baino lagin gehiago dauzka, eta ondorioz, hirugarren klasea orokortzeko arrisku handia dago. Hortaz, esan daiteke datu-desoreka dela orokortze horren jatorria.

Zehaztasuna kalkulatzen bada,

$$Accuracy = \frac{0 + 39 + 173 + 107}{543} = 0{,}587 \Rightarrow \boxed{\text{Accuracy} = \%58.7}$$

- **F1 *score***

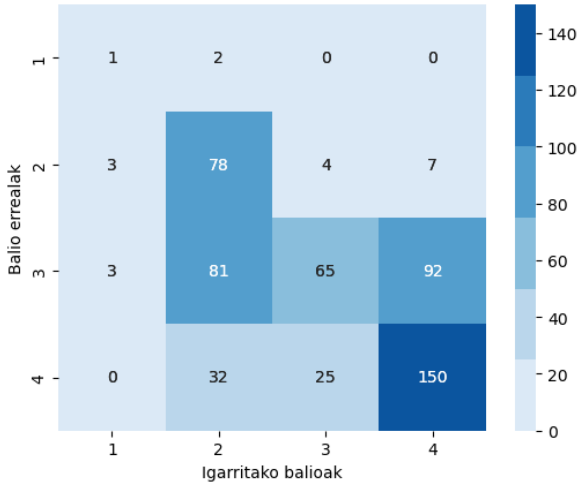
Metrika hau *tensorflow* liburutegiarekin erraz kalkula daiteke *KODEA.zip* artxiboko *9.ANeuronaSareak* $\boxed{F1 = 0.581}$ Balio oso handia ez denez, eraikitako eredua ez da oso egokia izango. Aurretik aipatu den bezala, datu-desorekak asko eragin ahal izan du emaitza hauetan.

4.1.4. Oversampling

Eredua hobetzeko nahian, datu-desoreka leuntzen saiatu nahi izan da. Horretarako, hainbat hautabide daude, baina, kasu honetan, *oversampling* bat egitea erabaki da, hots, pelikula gutxi dauzkaten klaseen embedding-ak erreplikatzea, klase guztiek lagin kopuru bera izan dezaten. *Oversampling*-a soilik entrenamendu datuetan egiten da, testatzeko gorde diren datuak berdin mantenduz. Hortaz, klase guztiek pelikula gehien duen klasearen pelikula-kopurua izango dute; kasu honetan, hirugarren klasea entrenamendu-lagin gehien dituena da (2300 pelikula) eta, ondorioz, klase guztiek 2300 entrenamendu-lagin izango dituzte.

Behin hau eginda, entrenamendurako datu-baseak 9200 lagin izango ditu. Neuronasarea arkitektura eta hiperparametro berdinak mantenduz entrenatzean eta testatzean, 4.3 irudiko konfusio matrizea lortu da, eta zehaztasuna kalkulatzen bada,

$$Accuracy = \frac{1 + 78 + 65 + 150}{543} = 0{,}541 \Rightarrow \boxed{\text{Accuracy} = \%54.1}$$



4.3 Irudia: A neurona-sarean *oversampling*-a aplikatzean lortutako konfusio matrizea.

F1 *score*-aren balioa, aldiz, **0.517** izan da.

Argi ikus daiteke eredua txartu egin dela, bai *accuracy* bai F1 metrikak behera egin baitute. Hala ere, aipatzekoa da hirugarren klasearekin izan ezik, gainontzeko klaseen detekzioan hobekuntzak egon direla. Konfusio matrizearen diagonal nagusian ikus daiteke lehenengo klaseko 3 laginetatik bat ondo antzeman dela, eta bigarren klaseko lagin gehienak ondo klasifikatu dira. 4.2 taulan klase bakoitzaren *precission*-a ikus daiteke.

	1. klasea	2. klasea	3. klasea	4. klasea
Oversampling gabe	0.0	0.59	0.72	0.52
Oversampling-rekin	0.33	0.85	0.27	0.72

4.2 Taula: Klase bakoitzeko *precission*-a *oversampling*-a egin baino lehen eta ondoren.

Nahiz eta *precission*-a onak izan, esan beharra dago *recall* metrika nahiko txarra izango dela. Izan ere, hirugarren klaseari dagozkion pelikula asko bigarren eta laugarren klasekoak bezala hauteman ditu eta, horregatik, *precission* onak dauzkate. Izan ere, orokorrean, *precission*-a handitzen denean, *recall*-a txikitu ohi da (eta alderantziz).

Orokorrean, datu-base batean *oversampling* egitean, emaitzak hobetu egiten dira, baina kasu honetan ez da horrela izan. Horren arrazoia sareak neurona-kopuru eskasa izatea izan daiteke, datu-basea handitzean, ereduaren konplexutasuna handitzea komenigarria izan baitaiteke. Sarearen geruza ezkutua 26 neuronarekin osatu da eta *oversampling* gabeko ereduaren antzeko metrikak lortu dira; %59,5-eko *accuracy*-a lortu da eta F1 *score*-a 0.589 da.

4.2. B Neurona-sarea

Aurreko ereduarekin lortutak emaitzak guztiz egokiak izan ez direnez, sarrera bezala pelikula baten embedding-a erabili beharrean, tweet independenteen embedding-ak erabiltzea erabaki da. Modu honetan, pelikula baten tweet-en embedding-en batezbestekoak

egin beharrean, embedding horiei klasearen etiketa ipini zaie.

Ideia da B neurona-sarea klase jakin bati dagokion pelikula baten tweet-ak, klase horri dagozkiela aurresateko gai izatea. Ondoren, neurona-sare hori mayority voting-a egiteko erabiliko da: pelikula baten tweet bakoitzaren klasea iragarriko da, eta ondoren, gehien errepikatzen den klasea pelikularen klasea bezala aukeratuko da. Adibidez, pelikula batek 15 tweet baditu eta neurona-sarearen arabera horietatik 6 bigarren klaseari dagozkion tweetak eta beste 9-ak hirugarren klaseari dagozkionak badira, pelikula hirugarren klasekoa dela esango da.

Oraingoan lagin askoz gehiago egongo dira, pelikulekin lan egin beharrean, tweet-ekin lan egingo delako. Lagin nahiko egongo direnez, soilik 'peli' eta 'trailer' terminoak dituzten tweet-ak erabiliko dira; modu honetan zarata murriztuko da. Noski, hau egitean, pelikula kopurua ere txikituko da. Hortaz, datu-base berri honetan klase bakoitzak izango duen lagin kopurua hurrengoa da:

	1. klasea	2. klasea	3. klasea	4. klasea	Guztira
Pelikula kopurua	25 (%0.7)	666 (%19.2)	2047 (%59.0)	730 (%21.1)	3468
Tweet kopurua	16817 (%4.6)	127175 (%35.0)	175191 (%48.2)	44352 (%12.2)	363535

4.3 Taula: B eredurako lagin kopurua.

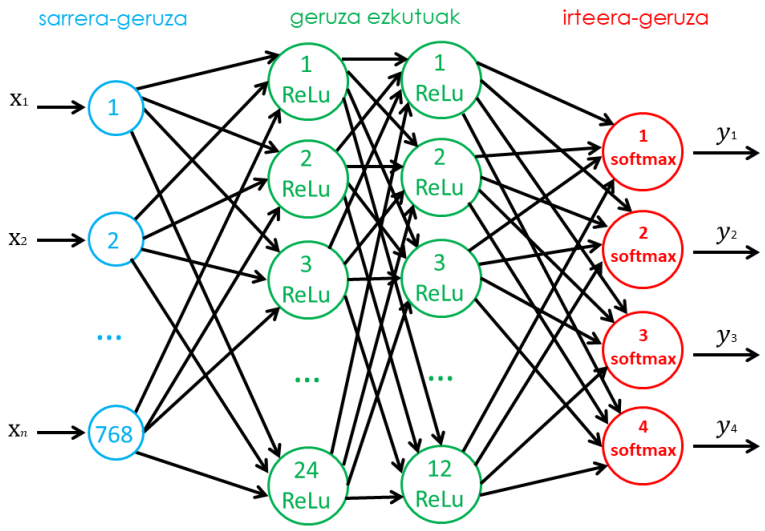
Argi ikus daiteke, berriz ere, datuen desoreka bat dagoela.

4.2.1. Entrenamendu eta testatze datuak

Nahiz eta B neurona-sarea tweet independenteekin entrenatu eta testatu, ondoren mayority voting-a inplementatuko denez, entrenamendu- eta testatze-datuak pelikula-kopuruaren arabera banatuko dira. Pelikulen %90 entrenamendurako gordeko dira eta, beraz, B neurona-sarea pelikula multzo horren tweet-ekin entrenatuko da. Ondoren, geratzen den pelikulen %10 horren tweet-ak, neurona-sarea testatzeko erabiliko dira; baina, *mayority voting*-a inplementatzean lortzen dena pelikula bati (eta ez tweet bati) dagokion klasea denez, beste konfusio matrize bat egingo da egindako iragarpenen eta pelikulen %10 horren etiketekin. Laburbilduz, neurona-sarea tweet-en iragarpenaren arabera ebaluatuko da, baina *mayority vouting*-a egitean, pelikulen arabera ebaluatuko da. Eredu honetarako, guztira 3468 pelikula daudenez eta testatzeko klase bakoitzetik pelikulen %10 hartuko denez, ereduaren ebaluaketarako 349 pelikula erabiliko dira. 349 pelikula horien tweet-kopurua 30101 da, eta horiek neurona-sarea testatzeko erabiliko dira.

4.2.2. Arkitektura

Sortutako neurona-sarearen eskema 4.4 irudian ikus daiteke. Neurona-sare honek soilik tweet-en klasea iragartzen ditu; aurrerago *mayority voting*-a egiteko kodea gehituko zaio.



4.4 Irudia: B neurona-sarearen eskema.

Sarearen sarrera 768 tamainako bektore numeriko bat izaten jarraitzen duenez, sarrera-geruzak 768 neurona izan behar ditu, eta hortaz, irudiko x_i terminoak bektorea osatzen duten zenbakiak dira. Bi geruza ezkutu erabili dira, lehenengoa 24 neuronakoa eta bigarrena 12 neuronakoa. A neurona-sarea baino geruza eta neurona gehiago erabili behar izan dira, oraingoan datu-kopurua handiagoa baita, eta ondorioz eredua konplexuagoa izan baitaiteke. Berriz ere, geruza ezkutuen neuronek ReLu aktibazio funtzioa erabiltzen dute, patroi konplexuagoak antzeman ahal izateko. Irteerako geruzak, noski 4 neurona ditu, softmax aktibazio-funtzioarekin.

Optimizazioaren aldetik, berriz ere Adam aukeratu da. Bere hiperparametroak ukituz, 4.4 taulako balioak hartzea erabaki da, balio horiekin emaitza hoberenak lortu baitira.

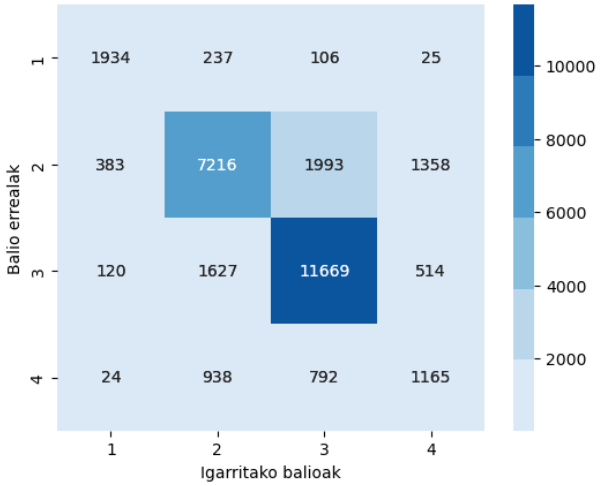
η	β_1	β_2	ϵ
0,00096	0,899	0,999	10^{-7}

4.4 Taula: B sarearen Adam optimizaziorako hartutako hiperparametroen balioak.

4.2.3. Emaitzak

B neurona-sarearen emaitzak aztertzeko, aurrekoarekin egindako ebaluazioa errepikatuko da.

- **Konfusio matrizea**



4.5 Irudia: B neurona-sarearen konfusio matrizea.

Sare honen kasuan, hirugarren klasea da gehien hautematen dena; baina, zentzuzkoa da, hirugarren klasea datu-basearen erdia baita, eta, ondorioz, klase horretako lagin asko iragarriko dira. Aurretik ikusi den bezala, hau A sarearekin ere gertatu da. Hala ere, oraingoan, lehenengo klaseari dagozkion zenbait tweet ondo antzeman ditu. Izan ere, klase guztiak detektatzen oso ondo jokatzen du, laugarren klasea izan ezik.

Zehaztasuna kalkulatzen bada,

$$Accuracy = \frac{1934 + 7216 + 11669 + 1165}{30101} = 0{,}730 \Rightarrow \boxed{\text{Accuracy} = \%73.0}$$

- **F1 *score***

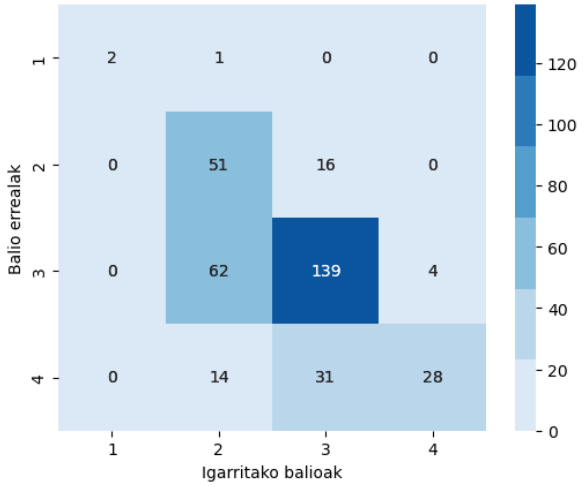
  Esan bezala, metrika hau *tensorflow* liburutegiarekin kalkula daiteke eta hau da lortutako balioa:

  $$\boxed{\text{F1} = 0.729}$$

Lortutako emaitzak nahiko onak dira. Hala ere, ezin daiteke esan eredu on bat denik, lagin asko ez baititu iragarri. Esan bezala, erabilitako datu-basea ez dago orekatuta. Hala ere, emaitza onak direnez, *mayority voting*-a inplementatzean pelikulen iragarapena A neurona-sarekoa baino hobea izateko aukerak daude.

4.2.4. Mayority Voting

Testatzeko gorde den datu-basearen %10-rekin, *mayority voting*-a inplementatu da. 4.6 irudian lortutako konfusio matrizea beha daiteke, hau da, eredu osoaren konfusio matrizea.



4.6 Irudia: Mayority voting-aren konfusio matrizea.

Konfusio matrizea behatuz ikus daiteke gutxi gora behera klaseak nahiko ondo iragartzen dituela, izan ere, hau da lortzen den *accuracy*-a:

$$Accuracy = \frac{2 + 51 + 139 + 28}{348} = 0{,}632 \Rightarrow \boxed{\text{Accuracy} = \%63.2}$$

Laugarren klasea detektatzen nahiko txarra da, baina gainerakoak oso ondo hautematen ditu.

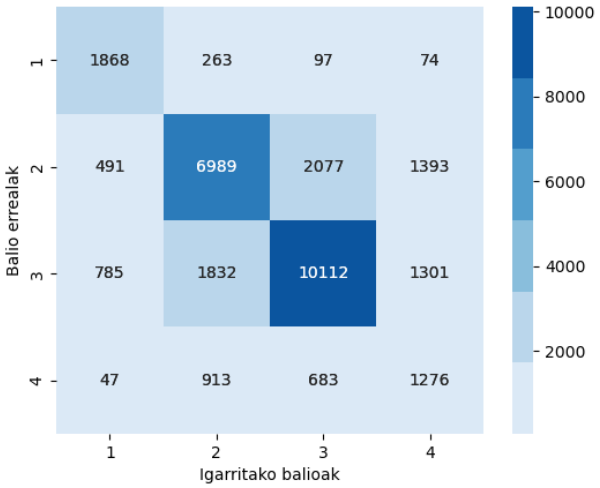
F1 *score*-ari dagokionez, **0.638** balioa lortzen da.

Hortaz, metrika hauek ikusita, esan daiteke eredu egoki bat dela. Egia da eredu hau sortzeko erabili den neurona-sarea nahiko ona dela, eta bukaerako inplementazio honekin metrikek ez dutela balio hoberenak erakutsi, baina %63-ko zehaztasuna ez dago gaizki.

4.2.5. Oversampling

Neurona-sarea hobetzeko nahian, berriz ere *oversampling* teknika egin da. Klase guztiek entrenamenduan lagin kopuru bera izan dezaten, lagin gutxi dauzkaten klaseen embeddingak erreplikatu dira. Hortaz, klase guztiek tweet gehien duen klasearen tweet-kopurua izango dute; kasu honetan hirugarren klasea entrenamendu-lagin gehien dituena da.

Berriz ere B neurona-sarearen arkitektura eta hiperparametro berdinak erabili dira, eta lortu den konfusio matrizea 4.7 irudian beha daiteke.



4.7 Irudia: B neurona-sarea hobetzean lortutako konfusio matrizea.

Zehaztasuna kalkulatzen bada,

$$Accuracy = \frac{1868 + 6989 + 10112 + 1276}{30101} = 0{,}672 \Rightarrow \boxed{\text{Accuracy} = \%67.2}$$

F1 *score*-aren balioa, aldiz, **0.677** da.

Oversampling-ak eredua okertu du, eta *mayority voting*-a inplementatzean aurreko ereduaren emaitza bera lortu da, hots, konfusio matrizea, berriz ere, 4.6 irudikoa da. Neurona-sarean aldaketa gutxi egon denez, normala da oraingo konfusio matrizea aurrekoaren berdina izatea.

Hala ere, aurreko ereduarekin gertatu den moduan, seguraski sarearen neurona-kopurua handituko balitz, metrikak emaitza hobeak erakutsiko lukete.

4.3. Ereduen konparaketa

Ikusi den bezala, A eredurako pelikula bakoitzetik bektore bat erabili da, eta B eredurako, berriz, tweet bakoitzetik bektore bat erabili da, ondoren *mayority voting*-a eginez pelikularen klasea aurresateko.
Soilik neurona-sareak konparatzen badira, ikusi da B neurona-sareak A neurona-sarea baino emaitza askoz hobeak lortu dituela, eta zentzuzkoa da, B neurona-sarearen laginak tweet-en embedding-ak baitziren eta A neurona-sarearenak pelikulen embedding-ak. Tweet-kopurua pelikula-kopurua baino handiagoa izatean, B neurona-sarearen datu-basea askoz handiagoa da, eta datu-base handi bat erabiltzean, orokorrean, emaitza hobeak lortzen dira. Gainera, B neurona-sarea soilik pelikuletaz hitz egiten duten tweet-ekin ('peli' eta 'trailer' terminoak dituztenak) entrenatu da, eta A neurona-sarearen datu-basea, berriz, pelikuletaz hitz egiten ez duten zenbait tweet dauzka, zarata sortzen dituztenak. Halaber, A neurona-sarerako erabilitako datu-basea askoz desorekatuagoa dago, batetik, klaseen artean pelikula-kopurua ezberdina delako, eta bestetik, pelikulen artean tweet-kopurua ezberdina delako. B neurona-sarearen datu-basearen kasuan, soilik desoreka bakarra dago, hain zuzen ere, klaseen arteko tweet-kopuruan.

Hala ere, ezin dira soilik neurona-sareak konparatu. Ereduak bere osotasunean alderatuz gero, emaitzak desberdinak dira. Izan ere, B ereduan *mayority voting*-a erabiltzean A ereduaren emaitzetatik ez da asko aldentzen, baina, hala ere, zehaztasun eta F1 *score* metrikak hobeak dira. Gainera, esan daiteke B ereduak klase guztiak neurri batean ondo detektatzen dituela, baina A eredua ez da lehenego klaseko pelikulak detektatzeko gai.

A Neurona-sarea		B Neurona-sarea + MV	
Konfusio Matrizea	F1 *score*	Konfusio Matrizea	F1 *score*
$\begin{pmatrix} 0 & 3 & 0 & 0 \\ 0 & 39 & 47 & 6 \\ 0 & 18 & 173 & 50 \\ 0 & 6 & 94 & 107 \end{pmatrix}$ **%58.7**	0.581	$\begin{pmatrix} 2 & 1 & 0 & 0 \\ 0 & 51 & 16 & 0 \\ 0 & 62 & 139 & 4 \\ 0 & 14 & 31 & 28 \end{pmatrix}$ **%63.2**	0.638

4.5 Taula: A eta B ereduen konparaketa.

Halaber, aipatu beharrekoa da bi neurona-sareak *oversampling* bat eginez hobetu nahi izan direnean, lortu diren emaitzak okerrera egin dutela. Horregatik, 4.5 taulan *oversampling*-ik gabeko ereduak konparatu dira. Hala ere, sareen neurona kopurua handitu egin da eta *oversampling*-a erabiltzean emaitzak hasierako neurona-sareak baino pixka bat hobeak direla ikusi da. Hala ere, ez dira askoz gehiago hobetu.

Beraz, bi ereduetatik B eredua hobeto jokatzen du, eta horrenbestez, bat aukeratzekotan, B eredua izango litzateke. Hala ere, kontuan hartzekoa da A ereduak entrenamendu-eta iragarpen-denbora gutxiago behar duela.

5. Ondorioak

Lan hau, pelikula batek, bere estreinaldia baino lehen, Espainian izango duen arrakasta aurresatea izan du helburu bezala. Hau lortzeko, Espainian estreinatutako pelikula ezberdinen informazioa duen datu-base bat erabiliz, pelikula bakoitzari buruz jendeak Twitter-en izandako iritziak bildu dira, eta datu horiek guztiak prozesatuz, bi eredu eraiki dira: A eredua eta B eredua.

A eredua eraikitzeko, neurona-sare bat egin da, baina, ikusi den bezala, ez da problema honetarako egokia den datu-base bat lortu. Batetik, pelikulak diru-sarreraren arabera banatzean, klase ezberdinen pelikula-kopurua oso ezberdina dela ikusi da. Desoreka hau konpontzeko, lehenengo eta bigarren klaseak batu dira, baina emaitzak ez dira hobetu, eta hortaz, esan daiteke klaseak mugatzen dituzten diru-sarreren balioak aldatzeak ez duela eragin handirik.

Ondorioz, datu-basea hobetzeko lagin gehiago lortu beharko lirateke, baina, ingurune soziokultural batean zentratzean, datuen lorpena ingurune horretan estreinatzen diren pelikula-kopuruak mugatzen du. Ez badira arrakasta handiko pelikula asko estreinatzen, ezingo dira lehenengo klaserako pelikula asko lortu. Horregatik, lan honetan, datu-basean oversampling egitea erabaki da. Teknika hau erabiltzean orokorrean emaitzak hobetzen diren arren, lan honetan metrikak okertu egin dira. Izan ere, klaseen artean dagoen lagin-kopuruaren aldea oso handia denean, oversampling-ak ez du nahitaez informazio berria edo garrantzitsuagoa sortuko ereduaren iragarpen-gaitasuna hobetzeko. Gainera, klase minoritarioentzako lagin sintetikoak sortzean, datuetan zarata edo patroi artifizialak sor daitezke, eta horrek ereduaren errendimendua oker dezake. Hala ere, erreplikatutako datuekin entrenatutako ereduari neuroa-kopurua handitzean, hobeto jokatu du, eta ondorioz, izan daiteke emaitza okerragoak lortzearen arrazoia neurona-sarearen sinpletasuna izatea.

Hala ere, datu-baseak errealitatea den bezala adierazten du: klase arrakastatsuenetan pelikula kopurua txikia da, eta eguneroko bizitzan arrakasta handia duten pelikulak gutxi dira. Gainera, arrakasta gutxi izan duten pelikulen tweet gutxi daude, eta errealitatean, jendeak ez du pelikula horietaz hitz egiten. Beraz, nahiz eta planteatutako problemari soluzio on bat emateko desegokia den datu-basea izan, errealitatea oso ondo adierazten du.

Bestetik, tweet-ak bilatzean, zenbait pelikulentzako tweet asko lortu dira, baina beste askorentzako ez dira hainbeste lortu. Desoreka hau konpontzeko, beste sare sozialetako iritziak lor litezke. Adibidez, YouTuben, pelikula bakoitzeko trailer-ean dauden iruzkinak har litezke, baina hau modu automatiko batean egitea zaila izango litzateke trailer bakoitzaren esteka jakin gabe.

B neurona-sarearen datu-baseari dagokionez, A neurona-sarean erabilitakoa baino askoz handiagoa da. Izan ere, laginak pelikula bakoitzeko embedding bat izan beharrean, tweet-en embeddingak izan dira. Klaseen arteko tweet-kopurua oso desberdinak direnez, datu-basea desorekatuta egon da. Hala ere, emaitza nahiko onak lortu dira, baina, B eredua eraikitzeko *mayority voting*-a inplementatu da eta, nahiz eta B neurona-sareari pelikula kopuruaren desoreka eragin ez dion, *mayority voting*-ean eragina izan du. Izan ere, pelikula baten 3 tweet-en artean klase bat erabaki behar izan duen bitartean, beste pelikula batekin 3000 tweet-en artean erabaki behar izan du. Hortaz, pelikularen arabera,

errore-ratioa handiagoa edo txikiagoa da.
B neurona-sarearekin lortutako emaitzei dagokienez, oso onak izan dira. *Mayority voting*-a inplementatzean ez dira hain onak izan, baina onargarriak direla esan daiteke. Hala ere, B neurona-sarearen emaitzak ikusita, etorkizuneko lanetarako kontuan hartzekoa da tweet-ak klaseetan klasifikatzeak pelikulak klaseetan klasifikatzea baino emaitza hobeak ematen dituela.

Hortaz, esan daiteke, Espainiako estreinaldi zinematografikoen arrakasta aurresatea neurri batean lortu dela, B ereduarekin 100 pelikuletatik 63 ondo klasifikatuko baitira. Kontuan hartuz 4 klase daudela, eta beraz, ausaz 100 pelikuletik 25 ondo klasifikatuko liratekeela, eredu onargarri bat dela esan daiteke. Hala ere, egindako analisi eta lortutako ondorioei esker beste lanetarako hobekuntzak egin litezke.

Ereduen errendimendua hobetzeko eta etorkizuneko lanetara begira, ondo legoke balaztatuta dagoen datu-base bat lortzea. Halaber, interesgarria izan daiteke erabilitako iritziak beste informazio mota batekin konbinatzea. Izan ere, lan honen sarreran aipatu den bezala, gaur egungo eredu zehatzenak informazio ugari erabiltzen dute iragarpen onak lortzeko. Informazio osagarri hori korrelazio handia izan beharko luke irabazitako dirusarrerarekin. Adibidez, pelikularen aktoreen zerrenda erabil liteke, non aktore bakoitza zenbaki batekin errepresenta liteke. Noski, horretarako, informazio hori duen datu-base bat beharko litzateke.

Bukatzeko, aipatu beharrekoa da lanean zehar hainbat arazo eta zailtasun egon direla. Nabarmenena prozesatzeko baliabide mugatua izan da. Erabilitako ordenagailuak ez du prozesatzeko ahalmen handirik eta GB askotako datu-baseak lortzeko edota tweet-ak embedding-etan bilakatzeko ordenagailua hainbat egunez prozesatzen utzi behar izan da. Lanean zehar suertatu diren beste errore batzuen ondorioz datu-baseak berriz eraikitzeak eta datuak berriz exekutatzeak lan honen bukatze-data atzeratu du.
Beste arazo nabarmen bat egon da. Lehenengo sarea eraikitzean eta emaitzak oso onak ez zirela ikusita, erabaki da berriz tweet-ak bilatzea, baina soilik pelikulekin erlazionaturik dauden termino gehiagorekin: 'peli' eta 'trailer' terminoez gain, 'reparto', 'cine', 'estreno', 'escena', 'taquilla', 'director' eta abar. Tweet horiek bilatzeko kodea exekutatzean, ez zen tweet-en informaziorik gordetzen, eta horren arrazoia Twitter-en aktualizazio berria izan da. Sare sozialean aldaketak egin dira eta orain ezin da *scraping*-ik egin ordaintzen ez bada. Hortaz, ezin izan zen tweet gehiago lortu, eta momentura arte lortutakoekin aurrera egin zen.

Nolanahi ere, lan hau mundu zinematografikorako interes handikoa dela pentsatzen dut, eta nire aburuz, Espaina mailan, etorkizuneko lanetarako oinarri gisa erabil daitekeen lana da.

Ingeniaritza Elektronikoko Graduko ikasle bezala, lan honi esker graduan zehar eskuratutako trebetasunak praktikara eraman ahal izan ditut; batez ere, programazioarekin zerikusia dutenak. Prozesuan zehar ikasitakoaren artean, nabarmenki datu-baseak lantzen, datuak prozesatzen eta aztertzen, ikasketa automatikoan erabiltzen diren tresnak erabiltzen eta ereduak eraikitzen ikasi dut. Gainera, programazio-trebetasunak hobetu ditut, eta akademikoki nahiz pertsonalki bete nauen lan bat izan da.

Bibliografia

[1] MathWorks, "Prevención del sobreajuste de modelos de Machine Learning." [Online]. Available: https://es.mathworks.com/discovery/overfitting.html

[2] N. Darapaneni, C. Bellarmine, A. R. Paduri, S. Entoori, A. Kumar, S. V. Vybhav, and K. Mondal, "Movie Success Prediction Using ML," *IEEE*, 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9298145

[3] J. D. Kelleher, B. M. Namee, and A. D'Arcy, *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies.* Cambridge, Massachusetts: The MIT Press, 2015.

[4] A. Géron, *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow.* O'Reilly Media, 2019.

[5] A. C. Mueller and S. Guido, *Introduction to Machine Learning with Python.* O'Reilly Media, 2016.

[6] W. Badr, "Auto-Encoder: What Is It? And What Is It Used For? (part 1)," *Towards Data Science*, Apr 2019. [Online]. Available: https://towardsdatascience.com/auto-encoder-what-is-it-and-what-is-it-used-for-part-1-3e5c6f017726

[7] P. Bhandari, "Correlation coefficient: Types, Formulas & Examples," *Scribbr*, Aug 2021. [Online]. Available: https://www.scribbr.com/statistics/correlation-coefficient/

[8] A. Pathak, "Introduction to Neural Networks [+ 7 Learning Resources]," *Geekflare*, Febrary 2023. [Online]. Available: https://geekflare.com/neural-networks

[9] A. Oppermann, "Optimization in Deep Learning: AdaGrad, RMSProp, ADAM," *Monstroid2*, October 2021. [Online]. Available: https://artemoppermann.com/optimization-in-deep-learning-adagrad-rmsprop-adam/

[10] J. Brownlee, "How to choose an activation function for deep learning," *Machine Learning Mastery*, Jan 2021. [Online]. Available: https://machinelearningmastery.com/choose-an-activation-function-for-deep-learning/

[11] D. Radečić, "Softmax activation function explained," *Towards Data Science*, Jun 2020. [Online]. Available: https://towardsdatascience.com/softmax-activation-function-explained-a7e1bc3ad60

[12] IBM, "¿Qué es el procesamiento del lenguaje natural (NLP)?" [Online]. Available: https://www.ibm.com/es-es/topics/natural-language-processing

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. [Online]. Available: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *Proceedings of the 2019 Conference of the North American Chapter of the Association for*

Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019. [Online]. Available: https://www.semanticscholar.org/reader/df2b0e26d0599ce3e70df8a9da02e51594e0e992

[15] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019. [Online]. Available: http://arxiv.org/abs/1907.11692

[16] A. Mishra, "Metrics to Evaluate your Machine Learning Algorithm," *Towards Data Science*, Feb 2018. [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

[17] PANDAS, "pandas.dataframe.corr." [Online]. Available: https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html