**Primjena Random Forest algoritma mašinskog učenja za detekciju znakovnog jezika**

Podgorica 2023.

Nikola Radenović Mentor:

Milivoj Radulović Zoja Šćekić

Sadržaj

**Uvod2**

Funkcionalni zahtjevi3

Nefunkcionalni zahtjevi3

Etička pitanja3

Biznis aspekt3

**1. Alati i metode4**

1.1 Paketi4

1.2 Dataset5

1.3 Fajlovi projekta6

**2. Detekcija simbola znakovnog jezika 7**

2.1 Mašinsko učenje7

2.2 Stablo odlučivanja (Decision tree)7

2.2.1 Kriterijumi razdvajanja u stablima odlučivanja i Ginijeva nečistoća 8

2.3 Konstrukcija stabla odlučivanja 9

2.4 Random Forest Classifier10

2.4.1 Konstrukcija RFC11

**3. Primjena scikit-learn RandomForestClassifier modela na naš primjer 11**

3.1 Prag detekcije12

**Zaključak13**

Problem sa M i N13

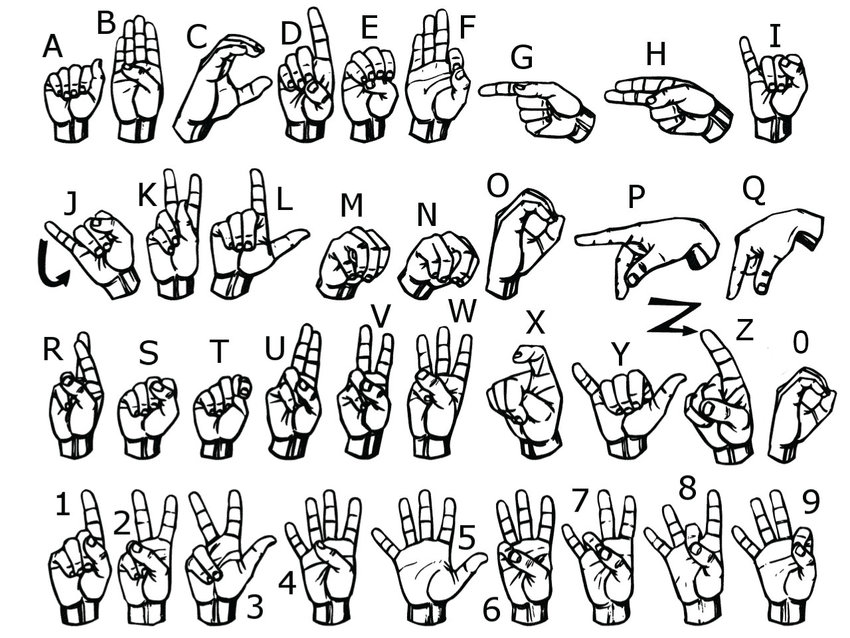
Literatura14

**Uvod**

Glavni cilj ovog projekta je naučiti korisnika simbolima za brojeve i slova na znakovnom jeziku u cilju njegovog osposobljavanja za jednostavnu komunikaciju sa gluvonijemim osobama uz primjenu algoritama za kompjutersku viziju.

Znakovni jezik se sastoji od mnogo simbola koji uključuju pokrete, izraze lica i druge elemente, ali ovaj projekat će se baviti samo simbolima za slova i brojeve zbog nedostatka dataset-ova koji uključuju riječi na znakovnom jeziku i zbog vremenskog ograničenja. On je, primarno, dokaz koncepta.

Takođe, zbog jednostavnosti i dostupnosti resursa, projekat će se baviti Američkim znakovnim jezikom.



U cilju realizacije projekta, korišćen je programski jezik Python.

Python je programski jezik visokog nivoa poznat po svojoj jednostavnosti, čitkosti i širokoj primjeni. Postao je izuzetno zastupljen i koristi se u raznim oblastima razvoja software-a, kao što su: razvoj web aplikacija, data science, vještačka inteligencija i mnoge druge.

U ovom radu će se koristiti Scikit-learn i OpenCV paketi.

**Funkcionalni zahtjevi**

**1. Detekcija landmark-ova:** Sistem koristi kompjutersku viziju da prepozna i locira ključne landmark-ove. Tako se položaj ruke predstavlja digitalno.

**2. Prepoznavanje simbola:** Sistem mora imati mogućnost da prepozna i klasifikuje već definisani skup simbola koji odgovaraju slovima i ciframa.

**3. Obrađivanje inputa u realnom vremenu:** Sistem mora da obradi video ulaz u realnom vremenu kako bi se rezultat klasifikacije prikazanog simbola odmah prikazao korisniku.

**4. Interfejs:** Korisnički interfejs programa mora biti jasan i jednostavan i mora prikazati detektovane landmark-ove sa ruke i informaciju o tome koji je simbol prikazan.

**5. Prilagodljivost:** Sistem treba da se prilagodi različitim uslovima kao različiti nivo osvjetljenja, različite pozadine, male razlike u položaju ili nagibu šake, pritom održavajući visoku preciznost detekcije.

**Nefunkcionalni zahtjevi**

**1. Preciznost:** Sistem mora imati visok stepen preciznosti detekcija landmark-ova, kao i visok stepen preciznosti klasifikacije da bi se osigurala tačnost krajnjih predikcija modela.

**2. Nisko kašnjenje:** Sistem treba da smanji vrijeme između prikazivanja simbola od strane korisnika i njegove detekcije i klasifikacije.

**3. Otpornost:** Sistem mora da bude stabilan i ne smije da prestane sa radom pri različitim uslovima kao osvjetljenje, veličina šake korisnika, različite pozadine itd.

**Etička pitanja**

Program bi pomogao u obrazovanju lica i njihovom osposobljavanju za jednostavnu komunikaciju sa licima sa invaliditetom, što bi dovelo do njihove bolje integracije u društvo. Takođe, može uvećati pristupačnost tehnologije licima sa invaliditetom tako što uređaji prepoznaju simbole koje oni prikazuju.

Pri kreiranju dataset-a za trening ovog modela korišćene su samo slike sa web-kamere njegovih autora, što znači da nije došlo do narušavanja autorskih prava korišćenjem slika preuzetih sa interneta.

Program ne čuva slike prikupljene sa web-kamere korisnika, niti bilo kakve informacije o njemu, tako da nema problema sa privatnosti.

**Biznis aspekt**

Ovaj program je primarno namijenjen za obrazovne ustanove. U njegovom daljem razvoju bi se napravila web aplikacija koja bi olakšala pristup modelu i radila na širem spektru uređaja.

Takođe, postoji prilika za njegovu prodaju kao komponentu drugih programa koji žele da poboljšaju pristup osobama sa invaliditetom.

**1. Alati i metode**

**1.1 Paketi**

* **Scikit-learn** je biblioteka za mašinsko učenje u Python-u i postala je jedan od ključnih alatki za mašinsko učenje. Razvijena je na osnovu drugih Python biblioteka kao NumPy i SciPy i omogućava pristup interfejsu za široki skup algoritama mašinskog učenja, od klasičnih tehnika kao što su linearna regresija i Support Vector Machine, do novijih pristupa mašinskom učenju kao što je duboko učenje. Njegov interfejs je pristupačan i dobro dokumentovan. Scikit-learn je pogodan za klasifikaciju, regresiju, clustering i druge primjene.
* **OpenCV** (Open-source Computer Vision) je moćna biblioteka za kompjutersku viziju i obradu slike koja nam daje alate za primjenu kompjuterske vizije u realnom vremenu. Široko se koristi u robotici, medicini i augmentovanoj realnosti. OpenCV omogućava obradu slika i videa, detekciju objekta, izvlačenje odlika objekta i slično. Njegov veliki broj optimizovanih algoritama omogućava programerima da efikasno rešavaju probleme u vezi sa kompjuterskom vizijom.
* **Mediapipe** je framework koji je napravljen da pojednostavi razvoj aplikacija koje koriste kompjutersku viziju i mašinsko učenje. Nudi veliki broj gotovih komponenti za praćenje šake, prepoznavanje lica, procjenu poze, augmentovanu realnost itd. Njegova modularna arhitektura omogućava laku integraciju ovih komponenti i njihovu adaptaciju za potrebe programera.
* **NumPy** je osnovna biblioteka za obradu numeričkih podataka u Python-u. Nudi podršku za velike, multidimenzionalne nizove i matrice i skup matematičkih funkcija koje efikasno rade sa ovim nizovima. Njegova široka primjena ga čini idealnim za obradu naučnih podataka, mašinsko učenje i analizu podataka.

**1.2 Dataset**

Dataset je skup podataka koji omogućava treniranje modela. Cilj našeg projekta je prepoznavanje 26 slova abecede i 10 cifara, što znači da ima 36 klasa za tih 36 mogućih simbola. Naš model treba da odredi kojoj od ovih klasa je najsličniji simbol koji korisnik prikaže. Svaka od 36 klasa ima svoj folder (pod imenom 0-35, indeksiranje kreće od nule), a svaki od tih foldera sadrži po 250 slika određenog simbola.

Slike su preuzete sa web kamere pokretanjem skripte collect\_imgs.py. Pri prikupljanju slika, vođeno je računa o prikupljanju raznovrsnih podataka: simboli su prikazivani na različitim odstojanjima od kamere, sa različitim stepenom rotacije i nagiba šake, tako da augmentacija u klasičnom smislu nije rađena.

Skripta create\_dataset.py iz ovih slika izvlači ključne tačke, tj. landmark-ove. U ovom slučaju, landmark-ovi su vrhovi prstiju, zglobovi na šaci, tj. sve tačke gdje se šaka može saviti. Praćenjem ovih tačaka se detektuje šaka. Landmarkova ima 21 po šaci.

Nakon detektovanja landmark-ova, njihove x i y koordinate se uvršćuju u Python dictionary kao vrijednosti. Tim vrijednostima se pridružuje ključ koji se sastoji od imena foldera u koji je skripta collect\_image.py upisala slike (0-35). Ovaj rječnik se upisuje u data.pickle fajl koji služi za treniranje modela.

Ovako izgleda jedna od slika iz dataset-a (slovo Q):



Ovako izgleda niz landmarkova izvučenih iz slike (42 atributa):

[0.027667999267578125, 0.3312065005302429, 0.12462975084781647, 0.3438516855239868, 0.22152568399906158, 0.23423141241073608, 0.25709863007068634, 0.10223713517189026, 0.2772873789072037, 0.013156592845916748, 0.18951289355754852, 0.07663619518280029, 0.21433322131633759, 0.02361232042312622, 0.18838848173618317, 0.13124924898147583, 0.1720416396856308, 0.15213939547538757, 0.12435998022556305, 0.05461639165878296, 0.1481214314699173,

0.0, 0.1345830112695694, 0.1288493275642395, 0.12485949695110321, 0.13851338624954224, 0.06179390847682953, 0.06001311540603638, 0.08557407557964325, 0.01041308045387268, 0.08241923153400421, 0.13597038388252258, 0.07587696611881256, 0.15100976824760437, 0.0, 0.07258683443069458,

0.022750124335289, 0.04665932059288025, 0.030477166175842285, 0.13131484389305115, 0.027804046869277954, 0.14994841814041138]

Koordinate zauzimaju vrijednosti između 0 i 1. Mediapipe hands klasa vraća koordinate normalizovane tako da zauzimaju vrijednosti između 0 i 1.

U dictionary-u se ovom nizu pridružuje i naziv klase.

80% dataseta su podaci za treniranje, 20% podaci za testiranje.

**1.3 Fajlovi projekta:**

Projekat se sastoji iz sledećih fajlova:

1. **collect\_imgs.py** je skripta koja služi da prikupi slike za dataset sa web-kamere i upiše ih u foldere sa nazivom 0,1,2,3…35 u folderu data koji je ona takođe kreirala.

2. Folder **data** koji sadrži slike dataset-a razvrstane po klasama 0-35, što odgovara ukupnom broju simbola koje model prepoznaje: 0-25: slova A-Z i 26-35: cifre 0-9.

3. **create\_dataset.py** je skripta koja služi da iz prikupljenih slika izvuče landmark-ove i njima pridruži label 0-35, tako kreirajući dictonary. Ovo radi tako što petljom prolazi kroz sve slike u svakom folderu 0-35.

4. **data.pickle** fajl je rezultat create\_dataset.py skripte i u njemu je smješten dictonary sa landmarkovima i pridruženim label-ovima.

5. **train\_classifier.py** je fajl u kojem se inicijalizuje klasa RandomForestClassifier paketa scikit-learn kojoj se prosleđuju podaci iz data.pickle. Model se trenira i čuva u lokalnom folderu.

6. **model.p** je rezultat skripte train\_classifier.py i to je fajl u kojem je smješten istrenirani Random Forest model.

7. **inference\_classifier.py** je skripta koja izvlači koordinate landmark-ova simbola koji je korisnik prikazao sa web kamere i prosleđuje ih modelu koji pravi predikciju. Na feed-u sa web kamere se uokviruje šaka korisnika, prikazuje se predikcija modela i sigurnost predikcije.

**2. Detekcija simbola znakovnog jezika**

**2.1 Mašinsko učenje**

Mašinsko učenje je polje vještačke inteligencije koje omogućava računarima da uče pravila, predviđaju, odlučuju ili prepoznaju objekte bez eksplicitnog programiranja. U suštini, mašinsko učenje se sastoji od „treniranja“ modela na velikim skupovima podataka - dataset-ovima, nakon čega će on moći da pravi generalizacije i pravi predviđanja za nove podatke sa kojima do sada nije radio.

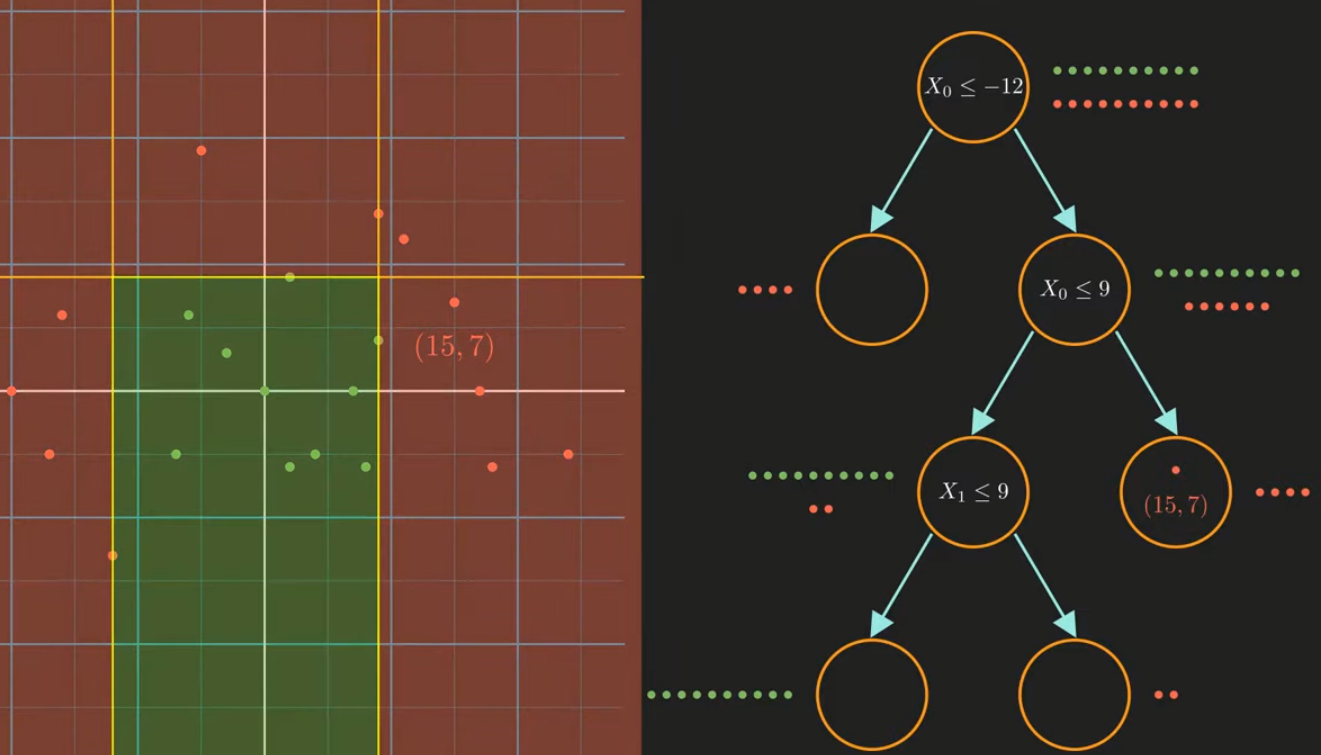
Algoritam mašinskog učenja koji ćemo koristiti u ovom projektu je Random Forest Classifier (RFC). On se sastoji od velikog broja stabala odlučivanja. Da bismo razumjeli RFC, prvo moramo razumjeti stabla odlučivanja.

**2.2 Stablo odlučivanja (Decision tree)**

Stablo odlučivanja je jedan od najpopularnijih algoritama nadgledanog učenja. Koristi se za klasifikaciju i regresiju. Sastoji se od čvorišta za odluke koja nam služe za podjelu podataka i listova koji služe za određivanje klase podatka.

Stablo odlučivanja radi tako što rekurzivno razdvaja ulazne podatke na osnovu nekog kriterijuma u čvorištu. Svaki čvor u stablu je testni slučaj za neko svojstvo, a svaka ivica koja se spušta sa čvora je jedno od njegovih rešenja.

U ovom primjeru imamo istrenirano stablo odlučivanja koje treba da odredi da li je neka tačka (u ovo primjeru 15,7) crvena ili zelena.



U korijenom čvorištu se učitava cijeli dataset ili njegov dio koji se trenutno razmatra. Na osnovu kriterijuma u čvorištima, podaci u njima se razdvajaju sve dok se u listovima ne nađu podaci željene „čistoće“. Nakon ovog procesa, stablo odlučivanja je trenirano i spremno za upotrebu.

Uvrštavamo tačku (15,7): ne zadovoljava kriterijum u korijenom čvoru i šalje se u sledeći čvor, čiji uslov podjele takođe ne zadovoljava i svrstava se u list koji je „čisto“ crven, i zaključujemo da je i naša tačka crvena.

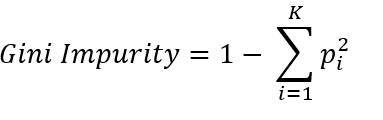
**2.2.1 Kriterijumi razdvajanja u stablima odlučivanja i Ginijeva nečistoća**

Ovdje se nameće pitanje: kako biramo kriterijume razdvajanja u čvorištima? Upravo je ovo cilj treniranja stabala odlučivanja - pronaći najbolje podjele u čvorovima. U stablima odlučivanja, „kvalitet“ kriterijuma se određuje Ginijevom nečistoćom (Gini impurity) ili dobitkom informacija (information gain). Random Forest Classifier koji primjenjujemo koristi Ginijevu nečistoću, stoga ćemo nju u nastavku razmatrati (iako model nudi mogućnost korišćenja dobitka informacija).

Nečistoća je, u ovom kontekstu, vjerovatnoća da će nasumično odabran član u skupu biti netačno klasifikovan u određenom čvoru.

Podjela je čista ako su svi elementi tačno razdvojeni u klase. Vrijednost Ginijeve nečistoće se kreće od 0, nema nečistoće - svi elementi pripadaju istoj klasi i 1, nečistoća je maksimalna - elementi su jednako raspoređeni među klasama. Vrijednost nečistoće za ostale vrijednosti se kreće između 0 i 1, što označava različite stepene izmiješanosti klasa.

Ginijeva nečistoća se računa formulom:



Gdje je:

* K: broj klasa
* pi: vjerovatnoća da član pripada klasi.

**2.3 Konstrukcija stabla odlučivanja**

1. **Inicijalizacija:** U korijenom čvorištu se nalazi cijeli dataset. Algoritam procjenjuje sve moguće podjele za sve atribute podatka i bira onu koja najviše umanjuje Ginijevu nečistoću.
2. **Čvorovi-djeca:** Odabrana podjela kreira dva čvora-djecu (stabla odlučivanja u scikit-learn Random Forest Classifier-u su binarna drva, što znači da svako čvorište ima dva čvora kao djecu, drugačiji algoritmi mogu imati više). Svaki čvor-dijete ima u sebi podskup dataset-a. Novonastali čvor čija je Ginijeva nečistoća jednaka nuli je „čist“ i ne dijeli se dalje, što taj čvor čini listom. U praksi nije uvijek moguće dostići nultu nečistoću, tako da su listovi često izmiješani, ali obično u takvim listovima dominira jedna klasa u odnosu na druge. Čvor koji nije „čist“ postaje novi čvor sa podjelom.
3. **Dalja podjela:** proces se ponavlja rekurzivno za svaki čvor-dijete. Za svaki novi čvor, algoritam procjenjuje sve moguće podjele za sve atribute podataka i bira, ponovo, onu koja najviše umanjuje Ginijevu nečistoću.

Kandidati za podjele su različiti za različite vrste podataka: za Boolean tip podatka, postoje samo dva kandidata za podjelu True i False (podjela bi izgledala if atribut == True). Za numeričke podatke, kandidati za podjelu se određuju inkrementima (atribut je npr. godine, i imamo inkrement od 5, tj. kandidate 20, 25, 30, 35 itd., podjela bi izgledala if godine >=30) ili računanjem prosjeka vrijednosti numeričkog atributa susjednih redova u datasetu (nakon sortiranja po toj numeričkoj vrijednosti). Za ovako pronađene kandidate, bira se onaj koji naviše umanjuje Ginijevu nečistoću.

1. **Završetak procesa** : proces se završava kada se dostigne određeni kriterijum za prestanak konstrukcije drveta. Odabir ovog kriterijuma je ključan zbog sprečavanja overfitting-a, tj. pada tačnosti predikcije modela zato što se previše adaptirao podacima za trening.

Neki od čestih kriterijuma za prestanak konstrukcije drveta:

* Maksimalna dubina: ograničavanje nivoa drveta (u gornjem primjeru, dubina drveta je 4).
* Minimalni broj ulaznih podataka: nakon velikog broja podjela, broj ulaznih podataka u novonastalom čvoru bio bi previše nizak da bi doveo do značajnog uvećanja u tačnosti predikcija.
* Minimalni broj članova u listu: ako se nakon velikog broja podjela u listu nađe previše mali broj članova, taj list neće moći dobro da generalizuje.
* Prag nečistoće: ako dalje podjele ne umanjuju značajno Ginijevu nečistoću, nije ih smisleno dalje dijeliti. Zato se uvodi prag nečistoće.
* Maksimalni broj listova: ograničava veličinu i kompleksnost drveta.

**Najzastupljenija klasa nekog lista je klasna predikcija tog lista.**

**2.4. Random Forest Classifier**

Random Forest Classifier (RFC) je vrsta ansambl (ensemble) učenja. To je tehnika mašinskog učenja u kojoj se kombinuju rezultati više stabala odlučivanja kako bi se postigla veća tačnost od pojedinačnih modela. Ansambl učenje uključuje korišćenje više modela za rešavanje istog problema i zatim kombinovanje njihovih predikcija kako bi se dobio krajnji rezultat. Koristi se i za regresiju i za klasifikaciju. Ovo je projekat klasifikacije, tako da će biti korišten u tom svojstvu.

**Prednosti i mane:**

+ nije osjetljiv na outliere

+ može da radi sa nedostajućim podacima

+ koristi se za inferencije raznih vrsta npr. određivanje važnosti nekog atributa dataset-a

- sporiji i koristi dosta memorije

**2.4.1 Konstrukcija RFC:**

1. Bootstrapping: članovi se nasumično biraju iz dataset-a i od njih se pravi tzv bootstrapped dataset, čija veličina ne mora nužno biti ista kao veličina originalnog dataset-a, ali je u večini slučajeva slične veličine. Isti članovi se mogu ponavljati u bootstraped dataset-u. Zbog toga, pri kreiranju bootstrapped dataset-a, jedan dio originalnog dataset-a biva neupotrijebljen. Taj dio se zove Out Of Bag dataset i obično čini jednu trećinu originalnog dataseta.
2. Konstrukcija stabala odlučivanja: pri određivanju kandidata za podjelu u čvorištima, posmatra se samo nasumični podskup atributa. Veličina ovog podskupa je najčešće kvadratni korijen ukupnog broja atributa (ovako je u scikit-learn RFC koji se koristi u ovom projektu).

Nakon ovoga, stabla se generišu na gore definisani način i dobija se skup stabala koja su veoma različita (bootstrapping i izdvajanje atributa su rađeni da bi se postigao baš ovaj efekat).

1. Agregacija: kada se ovako generisani RFC primijeni za klasifikaciju novog podatka, svako od stabala odlučivanja će napraviti predikciju klase. **Klasa koja ima najviše „glasova“ stabala odlučivanja je predikcija RFC modela.**

**3. Primjena scikit-learn RandomForestClassifier modela na naš primjer**

1. Naš dataset od 42 atributa (x i y koordinate 21-og landmark-a šake) se učitava u scikit-learn RandomForestClassifier model. Pravi se 100 bootstrap-ovanih dataset-ova za 100 stabala odlučivanja.
2. Pri svakoj podjeli u stablu, za kandidate podjele se uzimaju nasumično odabrani atributi. Njihov broj je jednak kvadratnom korijenu ukupnog broja atributa. To je za ovaj slučaj: . Ova vrijednost se zaokružuje na manju vrijednost, tako da se za kandidate podjele nasumično bira 6 atributa.
3. Drva se konstruišu tako što se podaci u čvorovima rekurzivno dijele na osnovu odabranih atributa. Za kriterijum podjele bira se ona podjela koja najviše umanjuje Ginijevu nečistoću.
4. Drvo prestaje sa konstrukcijom kada se dosegnu ovi kriterijumi:

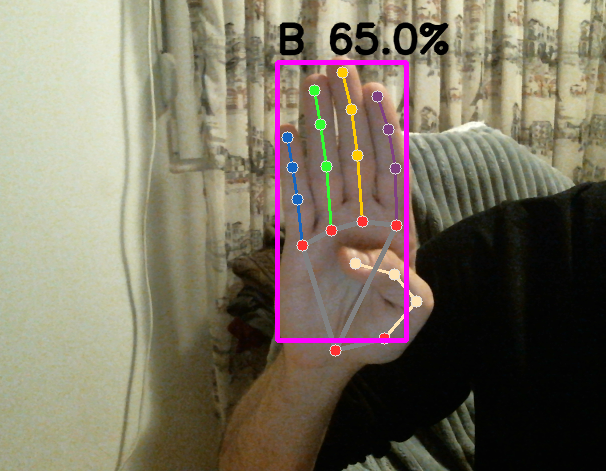
- Maksimalna dubina drveta je 10 (max\_depth).

- Najmanji broj članova u čvoru da bi došlo do podjele mora biti 2 (min\_samples\_split).

- Najmanji broj članova u listu da bi se on tretirao kao list mora biti 1 (min\_samples\_leaf).

- U čvoru će doći do podjele ako podjela umanjuje Ginijevu nečistoću za vrijednost veću od 0.0 (min\_impurity\_decrease).

1. Nakon završetka konstrukcije stabala, u listovima se određuje najzastupljenija klasa. Ta klasa je predikcija tog lista.
2. Nakon što su sva stabla konstruisana, pri primjeni modela na neki novi, do sada nepoznati podatak, svako drvo će napraviti određenu predikciju (0-35) za skup landmark-ova koji im stiže sa slike gdje je korisnik pokazao neki simbol. **Klasa koja osvoji najviše „glasova“ je predikcija cijelog modela, tj. to je simbol koji je korisnik prikazao.**



Nakon treniranja modela, tačnost predikcija na testnom dijelu dataseta je bila 99.8%

**3.1 Prag detekcije**

Da nije postavljen prag detekcije, zbog karakteristika sistema, model bi detektovao bilo koju poziciju šake kao neki simbol jer su stabla odlučivanja „glasanjem“ odlučila da je ta pozicija šake najbliža tom simbolu. Zbog toga je uveden prag detekcije od 30%, što znači da model mora biti barem 30% siguran u svoju predikciju da bi se ona prikazala korisniku.

**Zaključak**

Rezultat ovog projekta je model koji uspješno raspoznaje simbole znakovnog jezika koje korisnik pokaže primjenom algoritma mašinskog učenja Random Forest i korisniku prikazuje koji je simbol prikazan, zajedno sa informacijom o tome koliko je model siguran u svoju predikciju. Iako je model uglavnom uspješan, postoji nedostatak:

**Problem sa M i N**

Znakovni simboli za slova M i N su veoma slični. Isprva, ovo je bio problem sam po sebi jer model nije dobro raspoznavao M, N i druge simbole sa zatvorenom šakom kao A, E, S... Ovaj problem je uglavnom prevaziđen pravljenjem većeg i raznovrsnijeg dataset-a.



Ali problem raspoznavanja M i N i dalje nije prevaziđen zbog karakterističnog položaja palca u ovim simbolima. Naime, pri prikazivanju simbola za M i N, palac većinom prekrivaju ostali prsti šake, što onemogućava objektu klase mediapipe hands da uspješno prati položaj palca. Zbog ovog problema, model ima poteškoća u raspoznavanju simbola za M i N.

Literatura:

Computer vision engineer

26.01.2023.

*Sign language detection with Python and Scikit Learn | Landmark detection | Computer vision tutorial* [Video]

Youtube

<https://www.youtube.com/watch?v=MJCSjXepaAM&t=721s&ab_channel=Computervisionengineer>

StatQuest with Josh Starmer

26.04.2021.

*Decision and Classification Trees, Clearly Explained!!!* [Video]

Youtube

<https://www.youtube.com/watch?v=_L39rN6gz7Y&t=270s&ab_channel=StatQuestwithJoshStarmer>

StatQuest with Josh Starmer

05.02.2018.

*StatQuest: Random Forests Part 1 - Building, Using and Evaluating* [Video]

Youtube

<https://www.youtube.com/watch?v=J4Wdy0Wc_xQ&ab_channel=StatQuestwithJoshStarmer>