Предмет: Структуре података и алгоритми

Професор: Доц. др Димитрије Д. Чвокић

**ПРОЈЕКАТ**

Тема: Изградити модел који може да класификује вијести у различите категорије као што су политика, спорт и забава.

**Идеја**:

Моја идеја је била да направим један фајл(*skrejpovanjePodataka.py*) који показује како се скрејпују информације са веб страница и вади кључне ријечи из њих, свакој категорији вијести сам дао по 4 кључне ријечи које се најчешће спомињу у вијестима тих категорија. Када скрејпује и преброји колико пута се свака од тих кључних ријечи помиње у тексту, бројаче свих тих кључних ријечи смјешта у један ред *csv* фајла. То урадим за још 9 вијести и све смјестим у један *csv* фајл(*brojacRijeciPoUrlovima.csv*).

Даље, правим нови фајл(*generisanjePodataka.py*) који узима тих 10 постојећих редова и генеришем још 90 редова гдје се насумично додјељује сваком бројачу неки број. Затим, додајем још једну колону – класа, која одређује којој класи припада одговарајућа вијест тј. ред. Ако су бројачи у некој од прве четири ријечи, додјељујем класу 1 тј. спорт. Ако су бројачи у некој од друге четири ријечи додјељујем класу 2 – политика и ако су бројачи у посљедње четири ријечи – класа 3 тј. забава. Затим свих тих 100 редова смјештам у нови *csv* фајл(*generisaniPodaci.csv*) који ћу користити у следећем фајлу.

Коначно, креирам трећи фајл(*klasifikacijaVijesti.py*) који узима све податке из *generisaniPodaci.csv* фајла, мијеша их и дијели их на податке преко којих ће програм тренирати и на податке који ће служити као тест програму, да видимо може ли програм класификовати правилно вијести(80% тренинг скуп и 20% тест скуп). Затим преко функције еуклидског растојања и алгоритма k најближих сусједа пројекат класификује вијести у одговарајуће категорије. Такође сам уврстио и већ уграђену функцију *classification\_report* да нађе додатне параметре који показују тачност класификације.

**Напомена:** *generisanjePodataka.py* креира 100% тачне резултате и убацује их у *generisaniPodaci.csv*, па сам ја након креирања тог *csv* фајла ручно промијенио пар класа да класификација која се извршава у *klasifkacijaVijesti.py* фајлу не би сваки пут била 100% тачна.

**Израда:**

У фајлу *skrejpovanjePodataka.py* сам импортовао библиотеке: *requests* за слање *HTTP* захтјева, *BeautifulSoup* за парсирање *HTML*-а, *csv* ѕа рад са *CSV* фајловима, *Counter* за бројање ријечи и *stanza* за лематизацију текста(претварање ријечи у њихов основни облик). Затим сам креирао листу кључних ријечи и листу линкова које ћу скрејповати, као и празан рјечник гдје ћу чувати бројања кључних ријечи за сваку вијест. Пролазим кроз све линкове, извлачим текстове чланака, лематизујем ријечи и конвертујем све у мала слова. Након тога пребројавам кључне ријечи у лематизованом тексту и ажурирам бројач за сваки *URL*. Коначно, уписујем све у *brojacRijeciPoUrlovima.csv* фајл и исписујем поруку о успјешном чувању у фајл.

У фајлу *generisanjePodataka.py* сам импортовао још библиотеку *random* коју сам користио за генерисање насумичних бројева и за насумичан избор елемената из листе. Прво дефинишем кључне ријечи и број њихових понављања у оних 10 сајтова које сам претходно скрејповао. Правим функцију *generisi\_dodatne\_redove* која насумично одабере кључну ријеч и додјели јој насумичан број понављања. Са њом генеришем још 90 редова који представљају још 90 фајлова које би могли скрејповати помоћу прошлог фајла да би укупно имао 100 *csv* редова. Такође, додајем још једну колону назива класа поред колона са називима кључних ријечи. У колони класа се исписују цифре 1, 2 или 3 у зависности од од тога које кључне ријечи садржи текст. 1 – спорт, 2 – политика и 3 – забава. Све то додајемо у нови креирани *generisaniPodaci.csv* фајл и исписујемо поруку о успјешном чувању података.

У фајлу *klasifikacijaVijesti.py* сам још имплементовао библиотеке *numpy* која пружа подршку за рад са нумеричким подацима, *pandas* коју користим за учитавање података, манипулацију и анализу података као и припрему података за тренирање и тестирање модела. Модул *sklearn.metrics* из библиотеке *scikit-learn* пружа функције за евалуацију перформанси модела. *Accuracy\_score* користим за израчунавање тачности модела, а *classification\_report* пружа детаљан извјештај о прецизности, одзиву и другим метрикама класификације. Када смо импортовали све ове библиотеке, почињемо са учитавањем података и дјелимо податке на све колоне сем колоне „klasa“ и на податке из колоне „klasa“. Овако сам припремио податке за даљу анализу и тренирање модела машинског учења. Податке затим мијешам како бих осигурао разноликост у тренинг и тест скуповима. Податке дијелим 80% за тренирање модела и 20% за тестирање. Такође одвајам атрибуте од циљне класе како бих могао да их користим у тренирању и тестирању. Правим функцију за рачунање еуклидског растојања. **Еуклидско** **растојање** између двије тачке је дужина најкраћег пута који их повезује. Затим правим функцију за креирање алгоритма к најближих сусједа. {**Алгоритам к најближих сусједа** је алгоритам који се може користити за класификацију или регресију. Изаберем број к(у мом случају 5) и користим еуклидско растојање како бих израчунао удаљеност између података. Тражим к инстанци са најмањим растојањем до дате тачке. У мом случају, класификацији, претпостављам да је класа нове тачке иста као и класа већине њених к најближих сусједа. Овај алгоритам такође тражи да се чува цијели тренинг скуп, што може представљати проблем ако је скуп података велик.} Након креирања *kNN* функције, креирам функцију *predict* која на основу листе најближих сусједа одређује којој класи припада тест инстанца. На крају позивам функцију за предвиђање класе(*kNN*), користим *accuracy\_score* за израчунавање тачности класификације и на крају исписујем детаљан извјештај о прецизности, одзиву, F1-score-у, подјели између 20 тест инстанци, као и макро и тежинском просјеку помоћу *classification\_report* модула.

**Закључак:**

Овај пројекат има потенцијал за унапређење процеса аутоматске класификације вијести. Међутим, овакав модел захтијева редовно ажурирање како би се одржала тачност у окружењу вијести са интернета.

Такође, овај пројекат може послужити као основа за развој система који прате и анализирају новости из различитих извора, што даље доприноси ефикаснијем праћењу догађаја и садржаја на интернету.

Студент: Никша Кашћелан, 79/21, Природно-математички факултет, Бања Лука