Математички факултет

Београд, Студентски трг 16

Семинарски рад из Истраживања података 1

ПРЕПОЗНАВАЊЕ СЛОВА

– класификација скупа података „Letter Recognition“ –

Професор:

проф. др Ненад Митић

Асистент:

Мирјана Маљковић

Студент:

Лазар Васовић, 99/2016

Београд, јун 2019

С А Д Р Ж А Ј

Страна

1. УВОДНА РЕЧ

1.1. Замисао рада

Истраживање података, као тема курса у оквиру којег је овај рад настао, јесте процес прикупљања података, њиховог чишћења и обраде, пропратне анализе и, најзад, добијања неких корисних сазнања о њима. На улазу у овај процес налазе се сирови подаци односно њихови извори, док излаз зависи од циља и потреба.

Битан аспект истраживања података који ће бити обрађен у овом раду јесте генерализација односно индуктивно закључивање. Циљ ће бити уочавање и прављење модела законитости у улазним подацима. Простим језиком реченом, на основу виђених података покушаће се формирање представе о узрочно-последичним односима у обрађеном скупу података, као и уопштавање закључака на невиђене податке.

Биће обрађен проблем класификације података о словима тако да излазни модел са великом прецизношћу разликује слова према њиховим особинама. Класификација је, дакле, проблем препознавања врсте објекта; у конкретном случају – препознавање које слово представљају неки подаци. Додатно, она је пример надгледаног машинског учења, што значи да ће заједно са скупом улазних података бити прослеђен и жељени излаз (класа) за сваку инстанцу. Примењени алгоритми ће, стога, приликом учења модела (укалупљивања података у модел) знати која инстанца представља које слово, а крајњи резултат ће бити тестиран на подацима који нису учествовали у тренирању.

1.2. Скуп података

Скуп података „Letter Recognition“ настао је 1991. године за потребе рада „Letter Recognition Using Holland-Style Adaptive Classifiers” америчких научника информатичара Дејвида Џ. Слејта (енгл. *David J. Slate*) и психолога Питера В. Фреја (енгл. *Peter W. Frey*). Обојица се баве разним проблемима вештачке интелигенције и машинског учења, а у поменутом раду су дискутовали прилагодљиве класификаторе.

Подаци над којима је класификација примењена јесу они о великим словима енглеске абецеде. Наиме, у питању су црно-беле правоугаоне растерске слике које је потребно препознати као једно од 26 могућих слова. Толики број класа чини овај задатак тешким, за разлику од уобичајеног случаја, који је између осталог обрађиван на настави из ИП1, када постоје само две односно тек неколико (мали број) класа.

При генерисању података, употребљено је двадесет различитих фонтова, намерно одабраних тако да обухвате што више различитих стилова и начина писања. Диверзитет и хетерогеност додатно су повећани насумичним изобличавањем слика.

Укупно је 20.000 инстанци. Свака од њих је добијена као резултат позива посебно написаног програма за генерисање слике слова. За одабир параметара попут врсте фонта, типа слова, величине слике и фактора изобличења (искривљења) коришћене су равномерно расподељене случајне променљиве/величине.

Додатни детаљи о коришћеним фонтовима и току генераторског програма могу се видети у поменутом раду. Корисно је засад још напоменути да су излаз програма биле слике просечних димензија 45x45 пиксела, који су искључиво имали вредности „укључено“ и „искључено“ („да“ и „не“, црно и бело, тачно и нетачно, суштина је да су у питању бинарни пискели, са само две вредности), те да су, упркос изобличењима, према процени ауторâ, сва слова са слика махом била препознатљива људима.

Пример генерисаних слова дат је на илустрацији која следи.

Слике, међутим, нису оно што чини овај скуп података, већ низ нумеричких вредности. Ти бројеви су у даљем процесу прављења података добијени систематским читањем слика пиксел по пиксел, те израчунавањем основних статистичких особина расподеле пискела, о чему ће детаљније бити говорено у поднаслову који следи.

1.3. Опис и визуелизација

Скуп података „Letter Recognition“, како је већ напоменуто, чини 20.000 слогова (инстанци, објеката, примера(ка), ентитета, стимулуса) распоређених у 26 категорија, које представљају одговарајуће велико слово енглеске абецеде које та инстанца описује. Подаци се складиште у фајлу „Letter Recognition.csv“, датотеци која чува запетом раздвојене вредности (CSV – comma-separated values), док се опис скупа налази у обичном текстуалном фајлу „Letter Recognition.txt“. Подаци су јавно и бесплатно доступни на интернет страници „http://archive.ics.uci.edu/ml/datasets/Letter+Recognition“, репозиторијуму за машинско учење Универзитета Калифорније у Ервајну.

Формат података је уобичајени вишедимензиони, у ком различита поља у подацима одговарају различитим мерљивим особинама које су тим пољима (атрибутима, димензијама) представљени. Атрибута има 17, од чега је 16 улазних атрибута квантитативног (нумеричког) типа. Посреди су цели бројеви, мада је дискутабилно да ли их је природније тако посматрати или као категорије, пошто узимају коначан број вредности. Преостаје још један излазни атрибут (класа, тип слова) квалитативног (категоричког), и то именског типа, пошто у овом контексту уређење слова у абецеди не игра никакву улогу. Јасно је, притом, да је и домен последњег атрибута коначан, те је и он дискретног типа, попут улазних атрибута.

Оно што овај скуп чини посебно атрактивним јесте чињеница да су сви нумерички атрибути стандардизовани (тј. нормализовани, употреба термина зависи од случаја). Наиме, сваки се налази у целобројном интервалу [0, 15]. Ово је постигнуто линеарним скалирањем, могуће по мин-макс формули типа Xi → (Xi – стари_минX) * (нови_максX – нови_минX) / (стари_максX – стари_минX) + нови_максX, а затим сажимањем у целе бројеве одсецањем или заокруживањем вредности. То доприноси компактности података и спречава одређене алгоритме да фаворизују неки атрибут само зато што он има већи распон. Осим тога, на тај начин се олакшава припрема података, односно избегава потреба за претпроцесирањем у контексту скалирања. С друге стране, проблем би био покушај опонашања програма за генерисање слова; иако би се добиле одговарајуће вредности атрибута, не би било јасно како их скалирати.

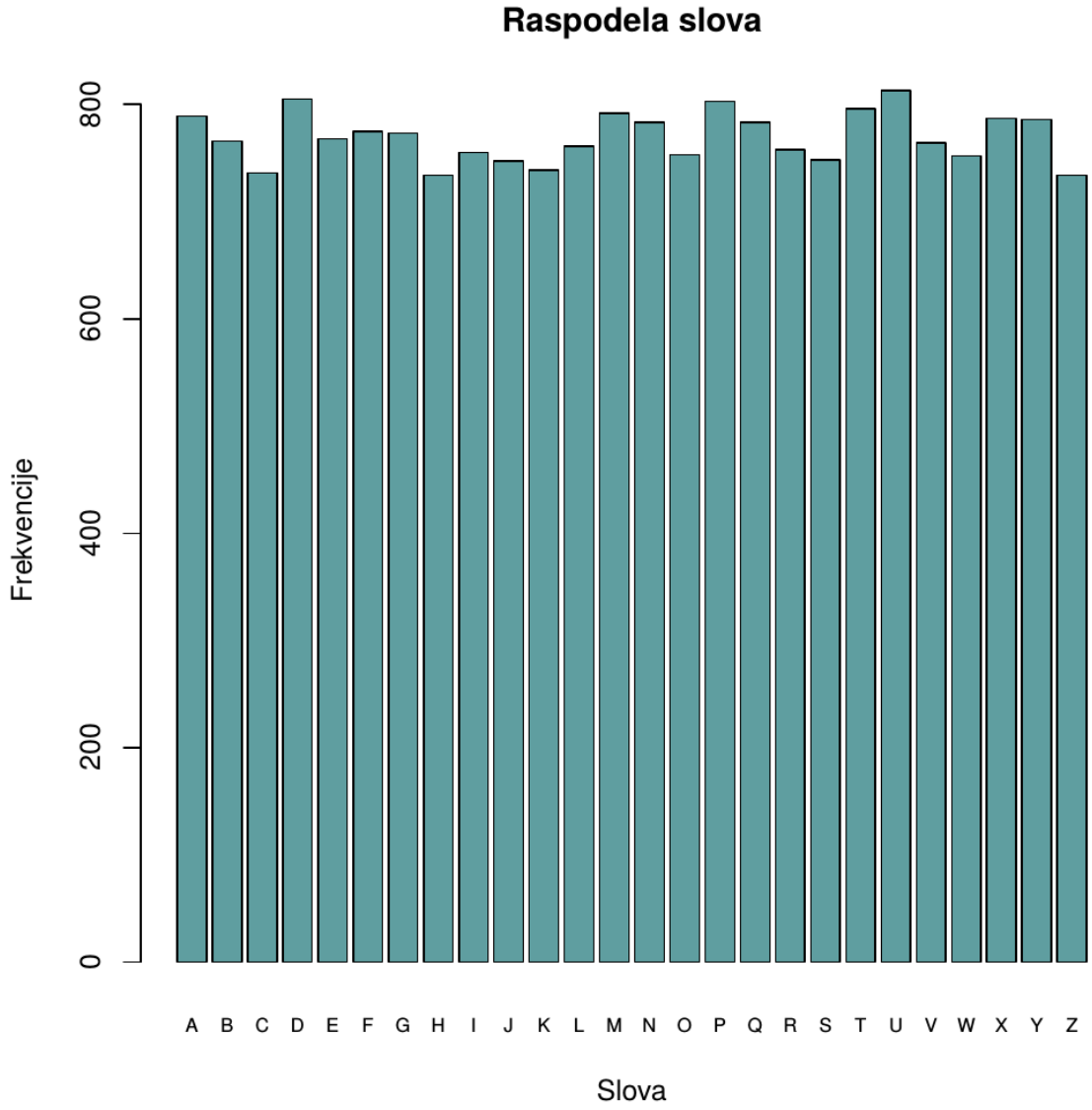
У скупу не постоје недостајуће нити бланко вредности, као ни некоректни нити дуплирани подаци, што додатно олакшава припрему података, као и сам рад са њима.

У наставку следи опис сваког атрибута односно, у случају улазних нумеричких вредности, његовог значења пре сабијања у нормализацијом ограничен интервал. Притом се при помињању координата мисли на уобичајен Декартов координатни систем са почетком у доњем левом углу; *x* оса расте надесно, док *y* оса расте нагоре:

1. тип слова, дискретна именска вредност из домена {A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z}; представља циљни атрибут, то јест, класу (категорију) инстанце којој је придружена,
2. водоравни положај (*x* координата) центра најмањег правоугаоника (кутије) који обухвата све „укључене“ пикселе [који се може нацртати тако да се сви „укључени“ пиксели налазе у њему],
3. усправни положај (*y* координата) центра те кутије,

4. ширина (број водоравних пиксела) те кутије,
5. висина (број усправних пиксела) те кутије,
6. број „укључених“ пиксела на слици,
7. средња вредност (математичко очекивање) водоравног положаја (*x* координате) „укључених“ пиксела у односу на центар кутије подељен ширином кутије (негативна вредност за нпр. налево померено „L“),
8. средња вредност (математичко очекивање) усправног положаја (*y* координате) „укључених“ пиксела у односу на центар кутије подељен висином кутије (негативна вредност за нпр. надоле померено „L“),
9. средња вредност квадратне водоравне удаљености (средњеквадратно одступање) „укључених“ пиксела од очекивања из седмог атрибута (дисперзија/варијанса, већа код нпр. хориз. раширених „W“ и „M“),
10. средња вредност квадратне усправне удаљености (средњеквадратно одступање) „укључених“ пиксела од очекивања из осмог атрибута (дисперзија/варијанса, већа код нпр. верт. раширених „E“ и „K“),
11. средња вредност производа водоравног и усправног одступања „укључених“ пиксела од очекивања из седмог и осмог атрибута (корелација, позитивна за линије облика y = x, негативна за y = -x),
12. средња вредност производа квадратног водоравног и усправног одступања „укључених“ пиксела од очекивања из седмог и осмог атрибута (корелација хоризонталне дисперзије и вертикалног положаја),
13. средња вредност производа квадратног усправног и водоравног одступања „укључених“ пиксела од очекивања из седмог и осмог атрибута (корелација вертикалне дисперзије и хоризонталног положаја),
14. средња вредност броја ивица („укључен“ пиксел одмах након [десно од] „искљученог“ или ивица/граница/крај слике) при читању (скенирању) слике слева надесно (разликовање нпр. „W“ или „M“ и „I“ или „L“),
15. збир усправних положаја ивица из претходног атрибута (корелација броја вертикалних ивица са хоризонталним положајем, веће за нпр. „Y“),
16. средња вредност броја ивица („укључен“ пиксел одмах након [изнад] „искљученог“ или ивица/граница/крај слике) при читању (скенирању) слике од доле нагоре (разликовање нпр. „E“ или „B“ и „I“ или „L“),
17. збир водоравних положаја ивица из претходног атрибута (корелација броја хоризонталних ивица са вертикалним положајем).

На наредној страни следи тракасти дијаграм апсолутних фреквенција циљног атрибута, као начин његове визуелизације и уверавања да су класе релативно равномерно распоређене. Слика је добијена помоћу приложеног R скрипта „Barplot.r“.



За остале атрибуте је опробана визуелизација у пару, као и поједине методе редукције. Резултати тога ће бити приказани касније, а засад је довољно напоменути да је скуп такав да је – без већег губитка прецизности – могуће смањити му димензионалност. Међутим, ситуација је таква да, и да то није било могуће, не би представљало проблем, пошто у сваком случају скуп не садржи велики број атрибута.

1.4. Припрема података

Као што је већ напоменуто, сви улазни атрибути су инхерентно стандардизовани (нормализовани) и скалирани у целобројне вредности из интервала [0, 15]. Са те стране, дакле, нема потребе ни за каквим акцијама. Евентуално би се ово могло још више смањити свођењем на нпр. реални интервал [0, 1], али за тим нема преке потребе, а потенцијално би изменило природу података и утицало на прецизност.

Могућности редукције биће детаљно дискутоване приликом описа тока класификације. Засад је довољно напоменути да ће најбитније бити нагласити алгоритмима улоге атрибута, као и њихове типове, мада је ово друго опционо.

Потребе за посебним претпроцесирањем података, дакле, нема.

2. КЛАСИФИКАЦИЈА У SPSS-У

SPSS Modeler је IBM-ом алат за моделовање, визуелизацију и уопштено истраживање података на веома комфоран и интуитиван начин – преко богатог графичког корисничког интерфејса и са једноставном „превуци и постави/пусти“ (drag and drop) парадигмом рада. Подаци, алгоритми, модели и пропратне радње представљени су чворовима, којима корисник манипулише на жељени начин.

Очекивано, овај алат подржава и велики број радњи везаних за класификацију. Редом ће бити описано и примењено њеколико њих, а биће наведени и успутни резултати и закључци о обрађиваним подацима о словима. Све написано доступно је читаоцу на проверу преко сачуваних токова. Први део је у датотеци „PCA, SVM.str“.

2.1. Уводна разматрања

За почетак, неопходно је стећи неки утисак о подацима, као и могућностима за њихову измену пре конкретног рада.

Првенствено је учитан скуп у радни простор помоћу чвора „Var. File“, након чега су, у оквиру њега, атрибути преименовани како би име адекватно осликавало значење. Затим су атрибутима учитане вредности и одређени типови – именски за класу, непрекидни за остале (остали су могли бити означени и нпр. као редни, али у том случају не би било могуће рачунати њихову корелацију и остале статистике) – као и улога у току (стриму) података – класа је циљни атрибут, док су остали улазни.

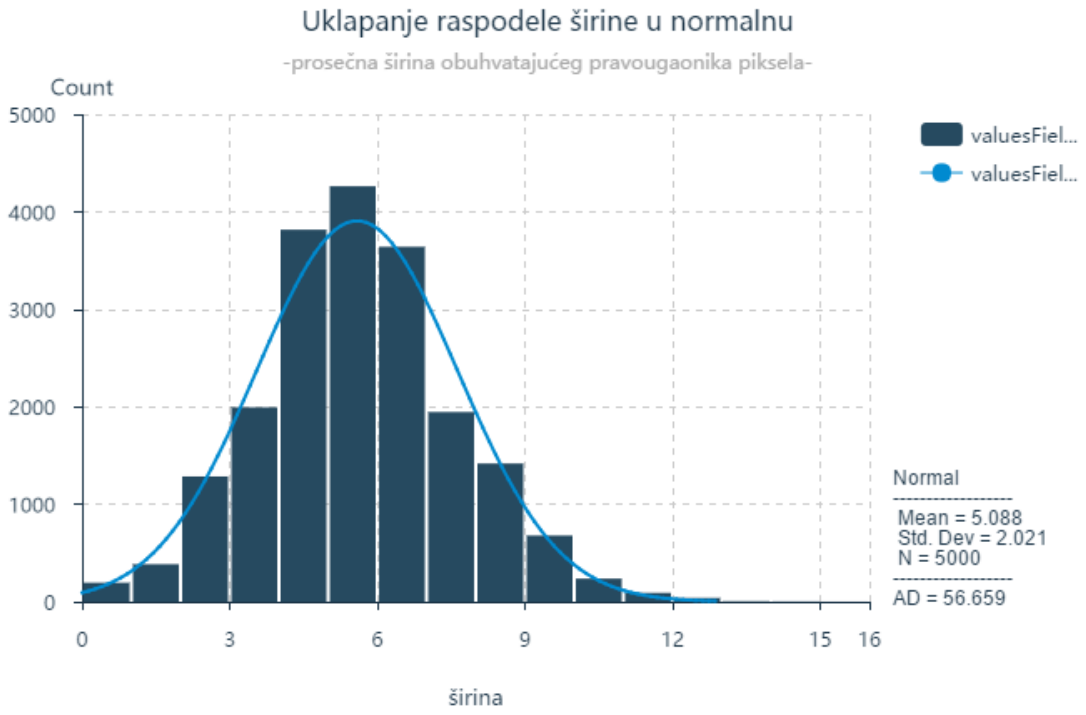
Напомена: ово је могло бити урађено и помоћу чворова „Type“ и „Filter“, али без њих је компактније и чини ток података краћим. С тим у вези, аутор ипак није ускраћен за вежбу са та два чвора, пошто су она коришћена у даљем раду.

Наредни корак био је анализа података, што је постигнуто помоћу чвора „Data Audit“, али и напредним техникама визуелизације и анализе доступним одабиром опције „View Data…“ након десног клика на чвор са подацима.

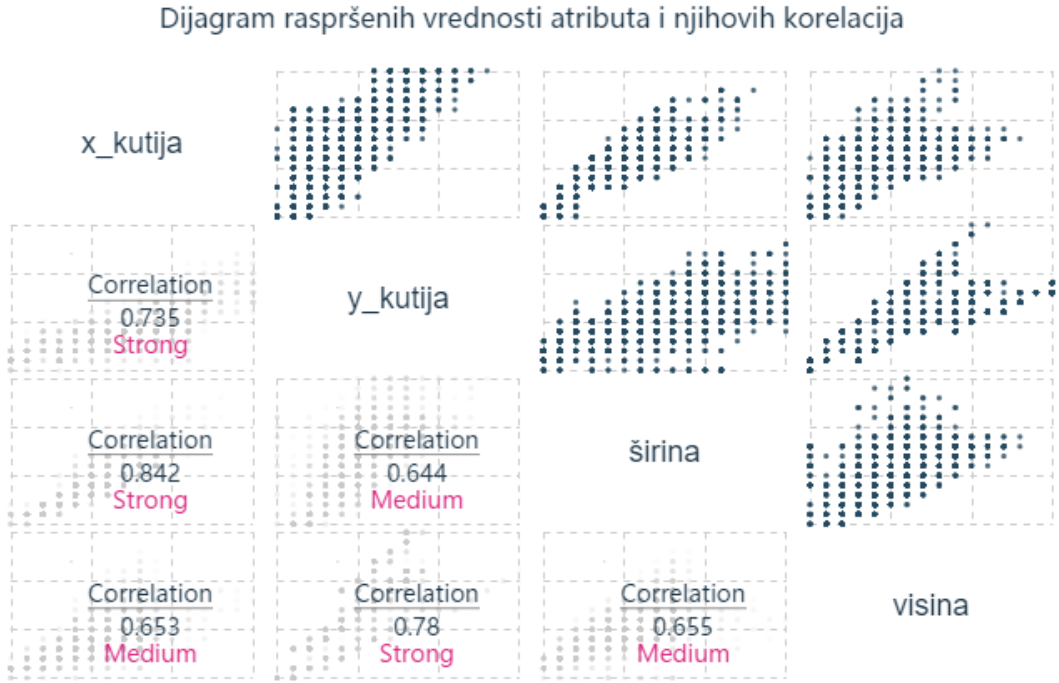
У оквиру резултата прве методе могла се видети расподела атрибута, број екстрема и аутлајера, као и потврдити већ позната чињеница да су сви подаци исправни и без недостајућих вредности.

Други приступ, иако ограничен на првих 5.000 инстанци, дао је још занимљивије резултате. Наиме, најзначајнија је била визуелизација атрибута, што независно (уз опцију аутоматског погађања расподеле), то у паровима (ради разматрања зависности), као и Пирсонова матрица корелација. Помоћу ње је утврђено да су атрибути махом независни у пару, те да највећа корелација (0,62–0,85) постоји између атрибута који представљају димензије, што је и очекивано; јасно је да ће нпр. ширина и висина кутије бити у тесној вези са *x* и *y* координатом њеног центра и слично. Ово је омогућило да се у даљој обради покуша са неком редукцијом атрибута, што ће и бити дискутовано.

На слици на следећој страни (није стала на тренутну) дат је пример уклапања ширине кутије у нормалну расподелу.



На наредним сликама дати су дијаграми корелације атрибута са највећом сличношћу, где се уочава њихова средња до јака (псеудолинеарна) зависност.



2.2. Подела и факторизација

Након анализе података, све је спремно за класификацију.

Прво је урађена факторизација атрибута анализом главних компоненти (PCA – principal component analysis). Овај алгоритам налази обрасце у подацима велике димензионалности. Основна замисао му је смањење димензионалности проблема ротацијом података у систем са осама таквим да је највећи број варијанси покривен најмањим бројем димензија. Ово се постиже анализом већ поменуте матрице корелација односно коваријанси, које представљају меру заједничке промене атрибута. За добијену трансформацију података важи да је сваки пар новодобијених атрибута некорелисан и ортогоналан, те да је систем уређен по величини одступања опадајуће.

Резултат овог алгоритма, примењеног помоћу чвора „PCA/Factor“, јесте пет фактора, сваки погодна линеарна комбинација улазних атрибута. Скуп података затим је прочишћен искључивањем првобитних улазних атрибута помоћу чвора „Filter“. У чвору „Type“ након тога су потврђени домен и улога преосталих шест атрибута.

Наредни корак био је подела (партиционисање) података на скуп за прављење (скуп за учење, тренинг скуп) и скуп за проверу (тестирање) модела помоћу чвора „Partition“. Могао се издвојити и посебан скуп за додатну проверу (валидацију), али то није учињено. Притом је узет уобичајени однос тренинг и тест скупа 70-30%, што је заправо декадно заокружен препоручен однос две трећине према једној.

На овако добијене податке примењен је метод потпорних вектора (SVM – support-vector machine, SVC – support-vector classifier) помоћу чвора „SVM“. Овај метод заснован је на статистичкој теорији учења и на идеји векторских простора. Модел је формула на основу које се израчунава класа. Алгоритам налази раздвајајућу хиперраван која раздваја категорије унутар векторског простора података, при чему максимизује размак између хиперравни и најближих јој инстанци које раздваја.

Резултати су, међутим, изостали, иако је алгоритам пуних десет минута покушавао да реши проблем. Наиме, перцентил погођених слова у оба скупа свега је 50%, што значи да је модел осетно потприлагођен. Осим тога, грешке су исувише распршене да би се уочило шта је тачан проблем, тако да се може претпоставити да је проблем у ниском степену информативности података које је проследила PCA. Могуће је да би били добијени бољи модели подешавањем „експертских“ параметара али то није испробано, пошто ће детаљније бити обрађено приликом рада у Python-у.

Results for output field slovo

Comparing $S-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	6,978	49.8%	2,956	49.36%
Wrong	7,033	50.2%	3,033	50.64%
Total	14,011		5,989	

Након овога, покушана је примена SVM на скуп пре трансформације. Резултати су били неупоредиво бољи, што казује да је проблем стварно у неадекватним параметрима факторизације. Касније је, ипак, покушана и примена других метода класификације на скуп из PCA, о чему ће бити речи у даљем току рада. Прецизност класификације на обе партиције података у другом покушају била је око 86%.

Results for output field slovo

Comparing $S-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	12,134	86.6%	5,139	85.81%
Wrong	1,877	13.4%	850	14.19%
Total	14,011		5,989	

Свеукупно гледано, добијени модел је веома добар до одличан, у зависности од критеријума. Ипак, приликом анализе података (чвор „Analysis“) консултована је и матрица коинциденције (конфузије), како би се уочило које су инстанце погрешно класификоване. Примећује се да је највише грешака прављено код слова која су стварно визуелно слична, па је део података о слову „P“ категорисан као „F“. Конфузија је настала и при разликовању „M“ и „W“, „H“ и „R“, „S“ и „Z“ али и, помало неочекивано, „O“ и „W“. Поправка оваквих појава биће покушана у другом делу рада.

Значајно је још напоменути и да је приликом покретања алгоритма назначено да се жели израчунавање важности предиктора, па је тако средњеквадратно одступање висине „укључених“ пиксела (десети атрибут) оцењено као убедљиво најбитније (коефицијент 0,45). Дупло мању важност (коеф. 0,23) показује број ивица при скенирању слева надесно (четрнаести атрибут), за чим следе средњеквадратно одступање ширине „укључених“ пиксела (девети атрибут, коеф. 0,10) и вертикални центар кутије (трећи атрибут, коеф. 0,08). Коефицијенти важности осталих предиктора износе 0,03 и мање, при чему број пиксела има најмању ненула битност (коеф. 0,01).

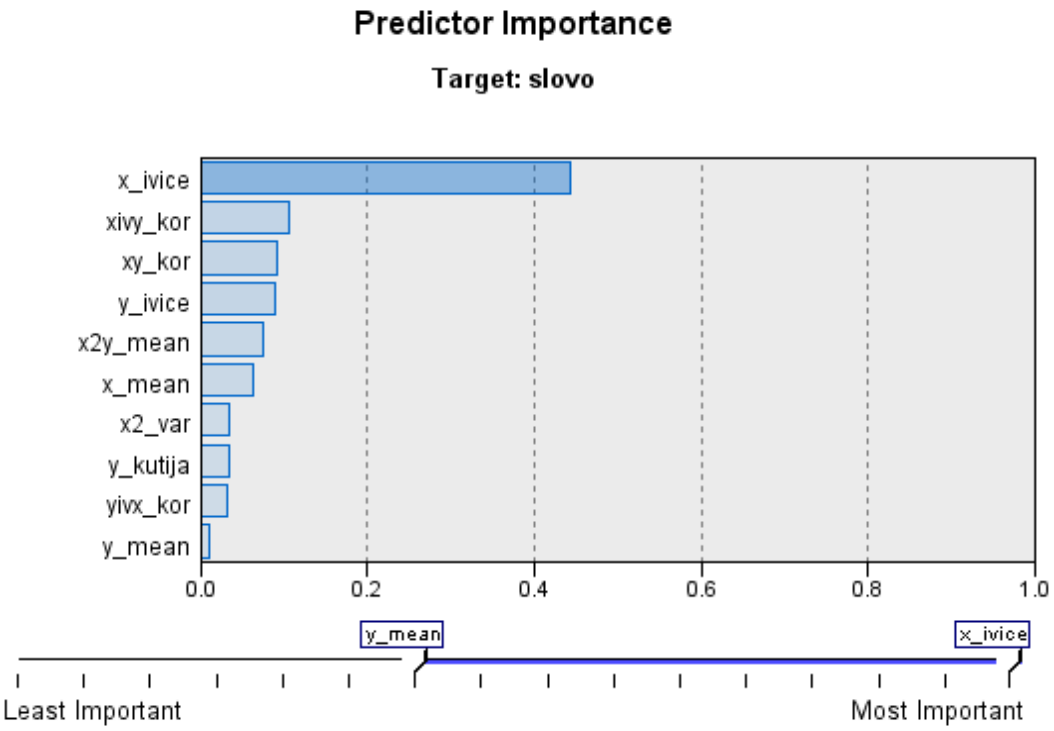
Разматрајући те коефицијенте, практично је шест атрибута проглашено за неважне у овом алгоритму. У питању су атрибути 4, 5, 8, 12, 13, 15 – димензије за које је већ откривено да су зависне од центра кутије, корелације одступања једне димензије од друге, као и још два додатна атрибута. У складу са тим, за крај је опробана примена SVM на скуп из ког су ова поља искључена, као и атрибути са најмањом ненула битношћу. На овај начин је преполовљена димензија проблема, са 16 улазних атрибута на 8, али је дошло до другачијег понашања модела, а на крају и губитка на прецизности, која је опала на око 64%, што и није толико лоше ако је најбитнија брзина рада. Уколико се искључе само небитни атрибути, прецизност се повећава на око 72%.

2.3. Дрвета одлучивања

Наредни примењени начин класификације јесу дрвета одлучивања. Алгоритми засновани на њима идејно прате основни Хантов алгоритам, који у сваком кораку дели слогове према тестном атрибуту који максимизује одређени критеријум. За разлику од претходно разматраног, овај метод није јако математички заснован, а донекле се понаша и као црна кутија (није најјасније зашто ради). Ипак, даје веома добре резултате у проналаску класификационе функције типа ако-онда (if-then), пошто резултујуће пресликавање није ништа горе него оно строго математичко, које се израчунава.

Први алгоритам овог типа који је размотрен јесте C5.0, помоћу чвора истог назива. Овај део одрађен је у току „C5.0.str“, у који су претходно учитани дорађени (са додатим именима атрибута) подаци извезени (експортовани) из претходног тока, а затим партиционисани на исти начин као и досад. Што се самог алгоритма тиче, у питању је проширење агоритма C4.5, који је даље проширење алгоритма ID3. Одликује га употреба ентропије као мере нечистоће података и информациона добит као критеријум поделе који се максимизује. У стању је да формира *n*-арно дрво над категоричким циљним атрибутима, користећи улазне атрибуте као тестне. Критеријуми заустављања – број инстанци у чвору испод доње границе, сви подаци у чвору припадају истој класи. У случају првог критеријума, класа се бира методом гласања. Могуће је накнадно поткресати резултујуће стабло. Могућа је и употреба појачавања (boosting) у циљу повећања прецизности, расејавања (winnowing) у циљу смањења димензије проблема, као и тежина атрибута (weighting) зарад бољег погађања.

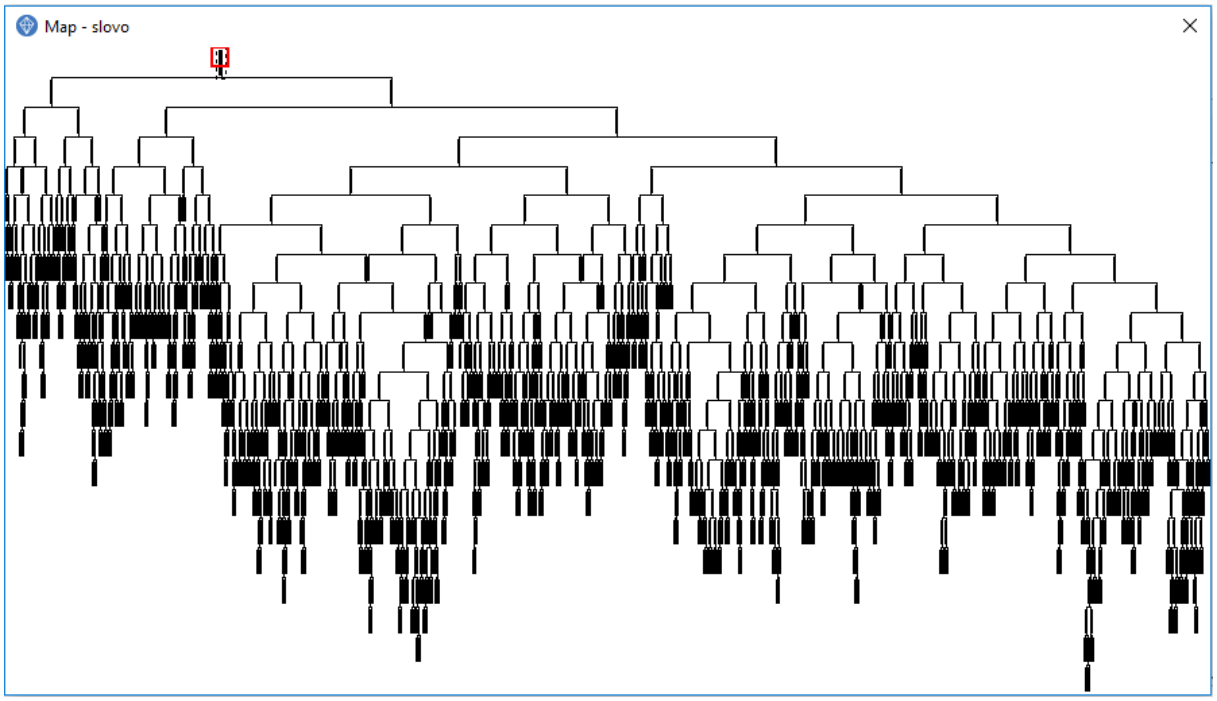
C5.0 примењен је над партиционисане што основне, то податке добијене анализом главних компоненти (PCA). Оба модела су се показала као прилично добра, с тим што је онај добијен над основним подацима супериорнији. Наиме, његова прецизност на тренинг скупу је око 96%, а на тест скупу 86%. Овај однос у другом случају је 90% према 66%, што не само што је мање, већ вуче на преприлагођавање, пошто је разлика између прецизности на ова два скупа осетна. У наставку следи визуелизација важности предиктора, прецизности и самог стабла првог модела.



Results for output field slovo

Comparing $C-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	13,392	95.58%	5,180	86.49%
Wrong	619	4.42%	809	13.51%
Total	14,011		5,989	



Како је могуће уочити, овај модел је знатно другачији од оног добијеног пре увођења стабала одлучивања. Наиме, најбитнији предиктор овде је број ивица при скенирању слева надесно (четрнаести атрибут), који је био други по важности у претходном моделу, док је средњеквадратно одступање висине „укључених“ пиксела (десети атрибут), оцењено током SVM као убедљиво најбитнији, овде у потпуности небитно. Приметно је и повећање прецизности на тренинг скупу (96% > 87%), док је понашање на невиђеним односно унапред непознатим подацима слично (~86%).

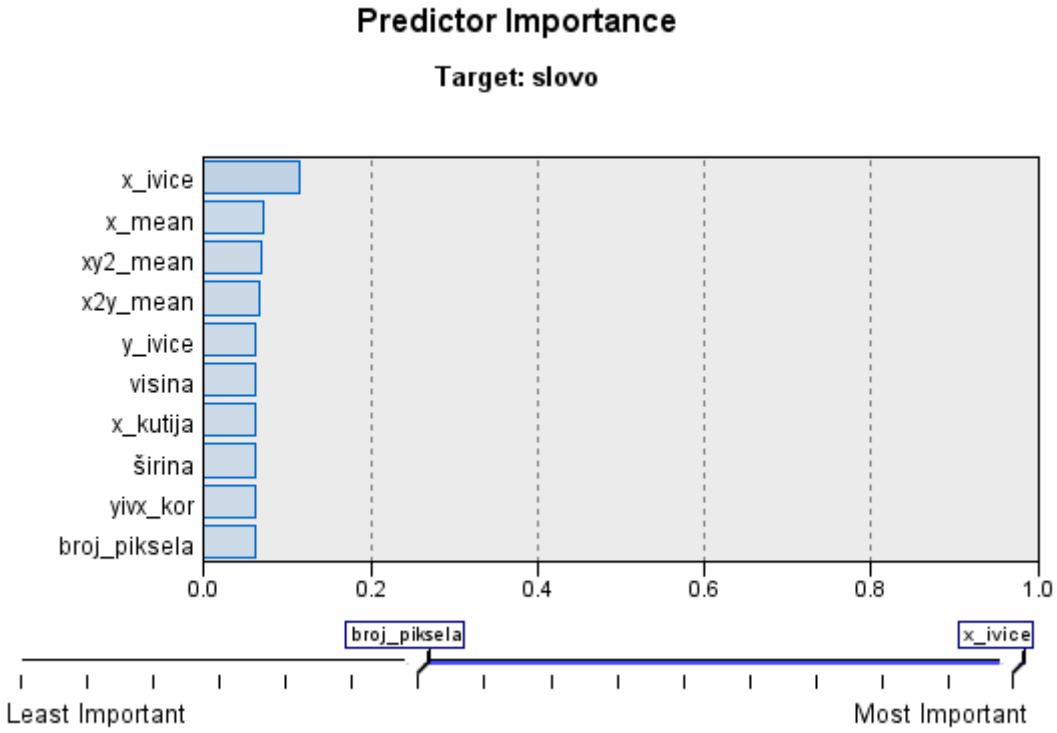
Иако је модел сасвим задовољавајући, не треба овде стати, па то и није учињено. Даље се покушало смањивање дубине стабла, у складу са принципом штедљивости („Окамова оштрица“), пошто је тренутно стабло прилично дубоко и исцрпно, али и детаљна анализа матрице конфузије. Од оба се ипак одустало, пошто је сваки покушај скраћивања водио лошијем резултату. Када је матрица цена у питању, грешка класификације је уместо уобичајених 1,0 постављена на 3,0 за парове слова код којих број погрешно класификованих није једноцифрен. Неки од њих су наведени већ приликом описа SVM, при чему је дискутована и њихова визуелна сличност. Комплетан списак следи: „F“ и „P“, „H“ и „P“, „H“ и „R“, „I“ и „J“, „K“ и „X“. Међутим, избегавањем тих грешака нагомилале су се нове, па је прецизност опала.

С друге стране, применом свих погодносни самог алгоритма (бустовање, расејавање, унакрсна валидација) добијен је скоро савршен модел, мада је свака погодност за себе успорила његову конструкцију. Прецизност побољшаног модела испала је фантастичних 99,79% на тренинг, а 93,94% на тест скупу. Погрешно су класификоване свега 392 инстанце, а највећи проблем и даље праве ситне грешке у разликовању „F“ и „P“ и других већ дискутованих сличних слова. Важности предктора донекле су сличне као код претходног модела, а оне и прецизност дати су у наставку.

Results for output field slovo

Comparing $C-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	13,982	99.79%	5,626	93.94%
Wrong	29	0.21%	363	6.06%
Total	14,011		5,989	



Поред овога, покушано је са елиминацијом небитних атрибута (полазног модела) – 2, 4, 5, 6, 10, 13 – према израчунатој важности предиктора. И то је довело до повећане прецизности, што је одлично када се узме у обзир да је у питању модел направљен након редукције скупа атрибута, заправо његовог половљења и тиме осетног смањења димензије проблема. Дакле, добит је двојака – и у презиности и у смислу редукције. Подсећања ради, није необично што су за небитне проглашени управо наведени атрибути (димензије и сличне зависне статистичке величине); још је приликом уводног разматрања примећено да су јако корелисани са другим атрибутима.

Results for output field slovo

Comparing $C-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	13,926	99.39%	5,453	91.05%
Wrong	85	0.61%	536	8.95%
Total	14,011		5,989	

Иако је већ конструисан суштински ненадмашив модел, примењени су и други алгоритми класификације засновани на дрветима одлучивања. Следећи на реду јесте CART односно C&RT, доступан преко чвора „C&R Tree“. Овај алгоритам је теоријски заснован и свеобухватан, а може се користити и за решавање проблема регресије, што је погађање нумеричких, а не само категоричких циљних атрибута. Гради бинарна стабла, при чему користи Гинијев индекс као меру нечистоће чвора, а информациону добит као оцену поделе. Механизмом претходника решава проблем дисбаланса класа, а

недостајуће вредности решава сурогатима. Поткресује стабло и ограничава његову дубину у току рада, а подржава и матрицу цене, као и тежине и важност атрибута.

Модел направљен овим алгоритмом веома је лош и потприлагођен. Наиме, иако је примењен бустинг, прецизност је свега око 53%. Једино занимљиво код њега јесте то што је прецизнији на тест подацима него на онима коришћеним при тренингу.

Results for output field slovo

Comparing $R-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	7,448	53.16%	3,208	53.56%
Wrong	6,563	46.84%	2,781	46.44%
Total	14,011		5,989	

Ово је донекле последица тога што је дубина дрвета ограничена на пет. Због тога је она повећавана, али најбољи резултат је и након тога био око 62% на оба скупа, што не ваља. Још лошији резултати добијени су на факторима. Све је у току „CART.str“.

Наредни примењени алгоритам дрвета одлучивања је CHAID (Chi-squared Automatic Interaction Detector), доступан преко истоименог чвора, а који се извршава узастопном применом три корака – упаривање, подела, заустављање – на сваки чвор, почевши од корена. Формирају се значајни парови атрибута и упарују у сложену категорију уколико је добијена p вредност статистичког теста (F мера или χ^2 тест) већа од прага. Током алгоритма формира се *n*-арно дрво. За прављење модела за класификацију слова употребљена је модификација овог алгоритма – тзв. исцрпни CHAID, који проверава све могуће поделе укрштањем атрибута и бира најбоље. Добијени модел сличне је (ниске) прецизности као и у случају C&RT.

Ништа се боље није показао ни QUEST (Quick, Unbiased, Efficient Statistical Trees), доступан преко истоименог чвора, а који је и иначе сличан CHAID-у, с тим што формира бинарно дрво, али зато подржава и линеарне комбинације подела у чвору. Оба ова алгоритма налазе се у радном току „CHAID, QUEST.str“.

2.4. Остале методе

Из куриозитета је примењено још неколико алгоритама који нису обрађивани на настави, а за које је аутор желео да испроба какве моделе дају, поготову пошто су неки од њих пример дистрибуираног учења у ансамблу (ensemble). Они се налазе у току „Ostalo.str“, а то су, по српским називима, чворовима и постигнутој прецизности:

- логистичка регресија (чвор „Logistic“) – прецизност ~78%,
- анализа дискриминанти (чвор „Discriminant“) – прецизност ~70%,
- вишеслојна неуронска мрежа (чвор „Neural Net“) – прец. ~86%,
- случајна дрвета (чвор „Random Trees“) – прецизност ~82%,
- случајна шума (чвор „Random Forest“, пример учења у ансамблу, прави се више модела на различите начине) – прецизност чак 99,89% на тренинг скупу (само 15 погрешних инстанци, које су вероватно аутлајери или некакав шум, па се нису уклопили), а 93,42% на скупу за тестирање,
- Бајесова мрежа (чвор „Bayes Net“) – прецизност ~71%.

Последњи примењени алгоритам, у истом току, био је *k* најближих суседа (k-nearest neighbors), доступан преко чвора „KNN“. У питању је лењи класификатор заснован на инстанцама. То значи да се приликом његове примене не гради

експлицитни модел, већ он постоји имплицитно. Како сам назив сугерише, објектима се додељује класа која је најраспрострањенија међу *k* најближих суседа, где се добар број *k* обично експериментално одређује, методом унакрсне валидације, док се близина процењује према погодно одабраној мери сличности односно различитости.

У конкретном случају класификације слова, као мера различитости узето је еуклидско растојање (L2 норма), док је k = {3, 4, 5}. При примени алгоритма није било никаквих проблема нити потребе за претпроцесирањем, пошто су, како је већ неколико пута напоменуто, сви улазни атрибути нумерички и сабијени у исти интервал.

Results for output field slovo

Comparing $KNN-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	13,731	98%	5,702	95.21%
Wrong	280	2%	287	4.79%
Total	14,011		5,989	

Како је свако слово у скупу у суштини добијено изобличавањем полазних слика из фонтова, није неочекивано што је овај приступ дао одличне резултате, са горе приказаним прецизностима. Посебно је занимљиво што је овај модел најпрецизнији на скупу за тестирање. Иако су C5.0 и случајна шума дали боље резултате на тренинг скупу, што их чини свеукупно прецизнијим, оволика прецизност на тест скупу је значајнија. Ово поготову важи у случају алгоритма *k* најближих суседа, чија је природа таква да тренинг скуп фактички „памти напамет“, тако да не треба превише разматрати прецизност на њему, већ углавном на непознатим подацима.

Нажалост, услед величине и димензионалности података, неуспешан је био покушај визуелизације и детаљнијег разматрања резултата овог алгоритма.

За сам крај рада са SPSS Modeler-ом примењен је алгоритам KNN за k = 1. Ово је урађено како би се проверило у којој мери долази до преприлагођавања када се врши класификација на основу само једног најближег суседа. Како је очекивано, прецизност на тренинг скупу је 100%, пошто је свака инстанца било ког скупа сама себи најближа, тако да је јасно да је класификација непогрешива. Оно што је, међутим, прилично занимљиво јесте чињеница да је овај приступ донео највећу прецизност и на тест скупу. Наиме, преко 95% тестних објеката је исправно класификовано. Ово, између осталог, значи да је класификација према првом најближем суседу веома ефикасна у овом случају (иначе важи за многе проблеме), те да ипак није изражено преприлагођавање.

Results for output field slovo

Comparing $KNN-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	14,011	100%	5,712	95.37%
Wrong	0	0%	277	4.63%
Total	14,011		5,989	

Ко и досад, највећи проблем прави разликовање „F“ и „P“, али и „F“ и „T“, „B“ и „R“, као и, за овај приступ специфично мешање, „E“ и „G“,„D“ и „H“, „B“ и „V“. Ови подаци могу се прочитати из одговарајуће матрице коинциденције. И даље важи претпоставка да до ове конфузије долази не због немогућности самих модела односно алгоритама, већ због шумова и аутлајера, што је и иначе главна слабост алгоритма 1NN.

Напослетку, проверене су перформансе и при другим односима партиционисања, како би се утврдило да, услед моћне природе алгоритма, нема преприлагођавања, упркос максималном укалупљавању у тренинг скуп. Следе добијене прецизности при издвајању 50%, 30%, 10% односно само 5% инстанци у скуп за учење.

Results for output field slovo
Comparing $KNN-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	9,976	100%	9,445	94.22%
Wrong	0	0%	579	5.78%
Total	9,976		10,024	

Results for output field slovo
Comparing $KNN-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	5,937	100%	12,979	92.29%
Wrong	0	0%	1,084	7.71%
Total	5,937		14,063	

Results for output field slovo
Comparing $KNN-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	1,932	100%	15,218	84.23%
Wrong	0	0%	2,850	15.77%
Total	1,932		18,068	

Results for output field slovo
Comparing $KNN-slovo with slovo

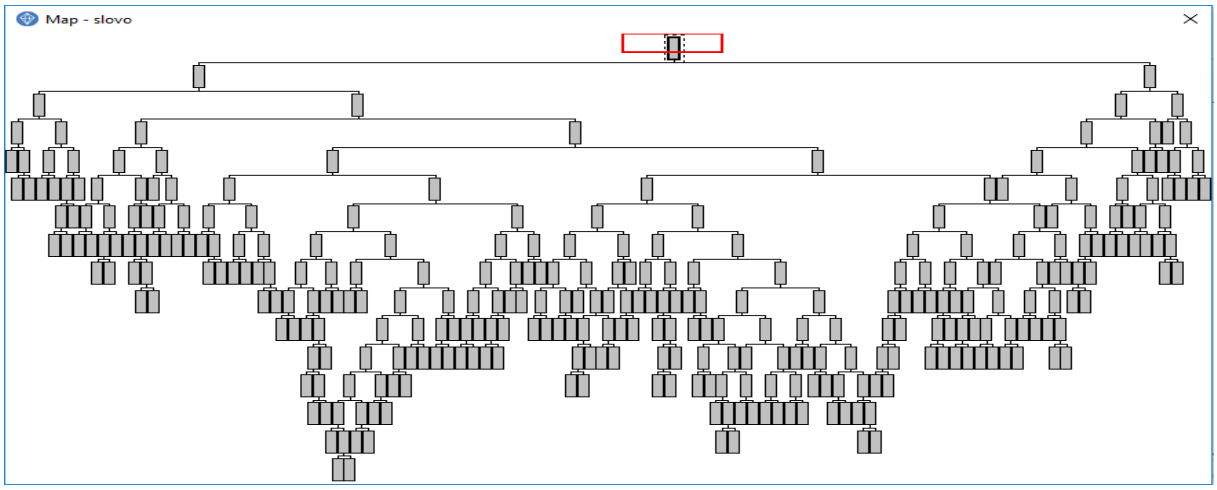
'Partition'	1_Training		2_Testing	
Correct	989	100%	14,639	77%
Wrong	0	0%	4,372	23%
Total	989		19,011	

Као што се може приметити, презизност постепено опада, али и даље је значајно висока када се узме у обзир колико је заправо тренинг скуп мали, а ипак довољно репрезентативан за добру, високопрецизну категоризацију.

Модели сличних прецизности добијају се применом алгоритма C5.0, који се у претходној анализи показао као најбољи са експлицитним моделом. Следе слике које илуструју прецизност класификације и изглед стабла за тренинг скуп величине 5%.

Results for output field slovo
Comparing $C-slovo with slovo

'Partition'	1_Training		2_Testing	
Correct	984	99.49%	14,154	74.45%
Wrong	5	0.51%	4,857	25.55%
Total	989		19,011	



3. КЛАСИФИКАЦИЈА У PYTHON-У

Python је програмски језик који омогућава брзо писање прегледних и захтевних програма, при чему подржава разне стилове и парадигме програмирања, као и велики број библиотека и структура односно типова података високог нивоа. Из тог разлога је један од најатрактивнијих на пољу машинског учења и истраживања података.

Кроз модуле попут numpy, pandas, sklearn (scikit-learn) и matplotlib подржава велики број радњи везаних за класификацију и уопштено рад са подацима. Редом ће бити описано и примењено њеколико њих, а биће наведени и успутни резултати и закључци о обрађиваним подацима о словима. Све написано доступно је читаоцу на проверу преко сачуваних скриптова, од којих је први описни „Analiza.py“.

3.1. Уводна разматрања

За почетак, као и у SPSS Modeler-у, неопходно је стећи неки утисак о подацима, као и могућностима за њихову измену пре конкретног рада.

Првенствено је учитан скуп података у програм и смештен у одговарајући оквир података (dataframe – дејтафрејм). Након тога је исписано првих пет инстанци скупа, опис скупа (статистике нумеричких атрибута), као и матрица корелација. Мада је испис мање илустративан него раније, закључак је исти, а то је да постоји неколико атрибута, углавном димензијских, са високим степеном корелације.

Напомена: услед проблема са енкодирањем у библиотеци pandas, атрибут „širina“ преименован је у „sirina“ за потребе рада у Python-у.

3.2. Дрвета одлучивања

Први примењени начин класификације овде су била дрвета одлучивања, већ описана у претходном делу рада. Све се налази у скрипту „Drvo.py“.

Након учитавања скупа података и издвајања улазних и циљног атрибута, подаци су подељени на скуп за учење и тестирање у уобичајеном односу 70-30%. Затим је инстанцирано дрво одлучивања и примењено на претходно подељене податке. Резултујуће стабло сачувано је у DOT формату и складиштено у датотеци „Drvo.dot“. Као и до сада, израчуната је и исписана важност предиктора, која је дата у наставку.

```
Vaznost prediktora:
x_kutija        0.010274
y_kutija        0.016941
sirina          0.010076
visina          0.014296
broj_piksela    0.017669
x_mean          0.048655
y_mean          0.055109
x2_var          0.068161
y2_var          0.119419
xy_kor          0.063463
x2y_mean        0.083342
xy2_mean        0.086849
x_ivice         0.137278
xivy_kor        0.099428
y_ivice         0.114702
yivx_kor        0.054338
dtype: float64
```

Примећује се да су важности у неку руку сличне онима из модела добијених преко SPSS Modeler-а, али да се у појединим аспектима разликују. И овде је број ивица при скенирању слева надесно (четрнаести атрибут) означен као најбитнији.

Што се тиче алгоритма који стоји иза поделе података на тренинг и тест скуп, једина подешена вредност била је величина скупова. Остале вредности су остављене подразумавене, као на пример семе за генерисање случајних бројева или чињеница да ће скуп бити промешан пре партиционисања. Имплицитно је одабрана нестратификована подела, пошто нема ретких класа – свако слово је отприлике подједнако заступљено, што је уочено приликом почетне анализе.

Када је у питању процес који стоји иза примењеног класификатора DecisionTreeClassifier, ниједан параметар није диран, што значи да је као мера нечистоће чвора коришћен Гинијев индекс. Осим тога, минималан број инстанци у чвору био је два, а дрво се ширило до чистих или минималних листова, без одсецања.

Након формирања, модел је анализиран на сличан начин као досад. Над тренинг и тест скупом рачунати су прецизност и матрица конфузије, а додатне статистике доступне су преко исписаног извештаја класификације, који следи у наставку.

Izvestaj klasifikacije:

	precision	recall	f1-score	support
A	0.88	0.96	0.92	232
B	0.82	0.80	0.81	235
C	0.93	0.88	0.90	183
D	0.82	0.82	0.82	219
E	0.87	0.87	0.87	247
F	0.81	0.82	0.82	239
G	0.81	0.85	0.83	205
H	0.77	0.83	0.80	224
I	0.90	0.93	0.92	240
J	0.91	0.88	0.89	209
K	0.88	0.81	0.84	226
L	0.95	0.86	0.90	223
M	0.94	0.92	0.93	249
N	0.91	0.91	0.91	231
O	0.79	0.83	0.81	245
P	0.86	0.84	0.85	239
Q	0.87	0.80	0.83	253
R	0.79	0.84	0.81	227
S	0.81	0.86	0.83	224
T	0.91	0.87	0.89	244
U	0.91	0.92	0.92	259
V	0.90	0.90	0.90	249
W	0.94	0.93	0.93	222
X	0.90	0.92	0.91	219
Y	0.89	0.90	0.89	231
Z	0.91	0.88	0.89	226
accuracy			0.87	6000
macro avg	0.87	0.87	0.87	6000
weighted avg	0.87	0.87	0.87	6000

Као што се може уочити, свеукупна прецизност на тест скупу је око 87%, што је у суштини врло добро, и веома личи на резултат добијен у SPSS Modeler-у помоћу алгоритма C5.0 без бустовања. На неким словима прецизност, одзив и F_1 скор/мера су већи, а на неким, попут већ дискутованих „F“ и „P“ или „H“ и „R“, мањи. Разлог мање

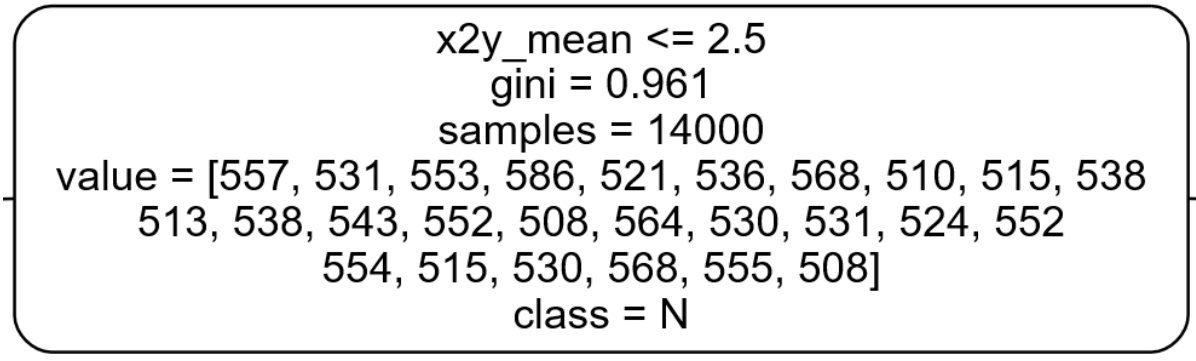
прецизности може се видети и у матрици коинциденције. С друге стране, тренинг скуп је максимално укалупљен, па је ту прецизност стопроцентна. У наставку следи исечак из матрице конфузије на тест скупу, која је сад лакше приказива него у SPSS Modeler-у.

```
Test skup:
Matrica konfuzije:
     A    B    C    D    E    F    G  ...    T    U    V    W    X    Y    Z
A  222    0    0    1    0    0    2  ...    0    0    0    0    0    0    0
B    1  188    0    3    1    0    2  ...    0    0    2    0    1    0    1
C    0    0  161    0    1    1    6  ...    0    3    0    0    1    0    0
D    1    1    0  180    0    1    1  ...    0    0    0    0    0    0    0
E    0    0    3    3  214    1    4  ...    3    1    0    0    3    0    4
F    1    2    2    0    0  197    1  ...    5    0    2    0    1    1    2
G    0    0    2    4    8    0  175  ...    2    2    0    0    0    0    0
H    0    2    1    9    2    2    2  ...    0    0    0    1    3    0    1
I    0    0    0    1    0    2    3  ...    0    0    0    0    0    0    2
J    1    0    0    0    0    3    0  ...    0    0    0    0    1    1    2
K    1    4    0    1    2    0    2  ...    0    0    1    0    6    0    0
L    2    3    0    0    2    0    5  ...    0    0    0    0    2    0    0
M    2    0    0    0    0    0    1  ...    0    6    1    3    0    0    0
N    3    2    0    2    0    0    0  ...    0    2    1    0    0    1    0
O    2    1    0    3    0    1    2  ...    0    6    1    2    0    2    0
P    0    4    0    3    1   19    0  ...    1    0    0    1    0    1    0
Q    1    2    0    5    4    1    2  ...    0    1    2    0    0    4    4
R    3    4    2    2    3    0    3  ...    0    0    2    0    1    0    0
S    6    6    3    0    1    1    0  ...    1    0    0    0    2    0    0
T    0    1    0    0    0    6    2  ...  212    0    2    0    1   12    3
U    0    1    0    1    0    1    2  ...    0  238    1    3    0    1    0
V    1    5    0    0    0    3    0  ...    2    0  223    4    0    3    0
W    1    0    0    0    0    2    0  ...    0    1    5  207    0    1    0
X    1    2    0    1    2    0    1  ...    1    0    0    0  202    0    1
Y    1    0    0    0    0    2    0  ...    5    1    6    0    0  208    0
Z    2    0    0    1    4    0    1  ...    2    0    0    0    1    0  198

[26 rows x 26 columns]
```

Након овога, покушана је промена неколико дифолтних параметара у циљу добијања бољег модела. Међутим, није се много тога променило. Задавањем ентропије као мере нечистоће, променио се редослед важних атрибута, али је прецизност само занемариљиво повећана – тек на око 88%. Ни повећање минималног броја чворова у листу нити остале измене нису утицале на повећање прецизности модела.

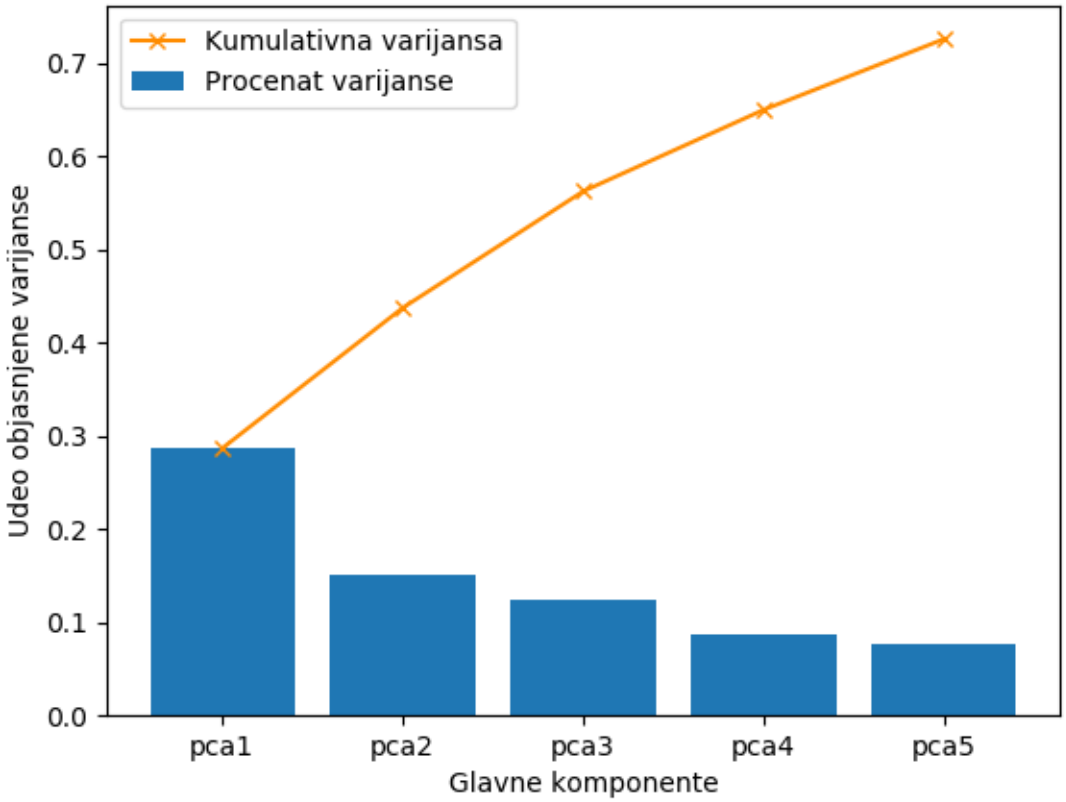
Само дрво одлучивања је помоћу програма Graphviz сачувано у формату скалабилне векторске графике (SVG – scalable vector graphics) у датотеку „Drvo.svg“. Стабло је веома исцрпно, тако да је у наставку приказан само корени чвор.



3.3. SVM и редукција

Наредни начин класификације био је помоћу редукционе методе PCA и помоћу SVM. Обе су дискутоване раније, а овде обрађене у скриптовима „PCA.py“ и „SVM.py“.

Што се првог дела тиче, након редукције, одабрано је првих пет фактора, који објашњавају око 70% варијансе. Подсећања ради, овај број је изабрао и SPSS Modeler.



Ови подаци затим су прослеђени методу подржавајућих вектора. Услед проблема са временом извршавања, није коришћена експлицитна унакрсна валидација, већ је она одрађена имплицитно, ручним мењањем кернела. Остали параметри нису мењани. Као најбољи показао се RBF, са 84% погодака на тренинг и 76% на тест скупу. Бољи резултати, добијени узимањем већег броја компонентни, налазе се на крају рада.

Након овога, SVM је примењен и на нередуковане податке, на којима се такође најбоље показао RBF, са прецизношћу око 99% на тренинг и 97% на тест скупу, што је уједно био и свеукупно најбољи модел. У њему су мешани нпр. „F“ и „P“, „D“ и „H“.

3.4. Неуронска мрежа

Следеће на списку метода биле су вишеслојне неуронске мреже, успут већ поменуте у оквиру додатно обрађених алгоритама. Скрипт је „Neuron.py“.

У конкретном случају класификације слова, како постоји чак 26 класа, биће неопходно конструисати већи број неурона, при чему ће се као коначан одговор бирати класа чији неурон да највећи излаз. Неурони ће се укрштати у скривеним слојевима. Функција активације ће бити одабрана унакрсном валидацијом, док ће решавач подразумевано примењивати методу стохастичког (случајног) градијентног спуста. Остали параметри биће занемарени односно користиће се подразумеване вредности.

Унакрсна валидација известила је да је најбољи модел трослојна мрежа са поправљеном линеарном идентичком функцијом активације (ReLU – rectified linear unit, f(x) = max{0, x}). Ово није неочекивано, пошто се ова функција показала као најбоља и најпопуларнија за савремене дубоке мреже неурона. Прецизност овако добијеног модела на тест скупу била је одличних 92%. Анализом матрице конфузије, закључује се да је модел донекле различит од досадашњих. Наиме, у њему не постоји проблем разликовања „F“ и „P“, али постоји мешање „R“ и „B“, „E“ и „G“, као и „K“ и „X“.

3.5. Лењи класификатори

Последњи примењени били су лењи класификатори, у које спадају већ обрађени метод *k* најближих суседа, као и вероватносни класификатори. Скрипт је „Lenji.py“.

У првом случају, унакрсна валидација је као најбољи модел оценила онај са четири суседа (могућ је био било који цео број између 1 и 10), при чему је одабрана метрика било растојање Минковског степена два (еуклидско растојање), а тежина суседа одабрана тако да буде обрнуто сразмерна растојању – што су суседи ближи, то више утичу на коначни одабир класе. За алгоритам је имплицитно одабрано *kd* дрво, пошто је број инстанци у тренинг скупу већи од 30, а број атрибута мањи од 20.

Постигнута је прецизност око 96%, што је веома блиско раније добијеном поготку од 95% код модела са једним најближим суседом, тако да су оба суштински подједнако добра. Примећено је још да одабир метрике за растојање Минковског степена већег од два не поправља прецизност модела, али чини грешке распршенијим – мање је иначе честих мешања „H“ са другим словима, али више неких других.

Када су у питању вероватносни класификатори, примењена су два типа наивних бајесовских класификатора, и то мултиномијални (MNB) и Гаусов (гаусовски) Бајес (GNB). Они користе знање из вероватноће, на основу којег у току учења модела рачунају априорне (претходне) вероватноће да се одређена вредност атрибута појави у некој класи. На основу тога, а преко Бајесове теореме, затим за сваку инстанцу одређују апостериорне (накнадне) вероватноће за сваку класу. Максимизирањем тих вредности (одабиром највеће), добија се коначна, додељена класа. Када су априори вероватноће у питању, у складу са својим називима, MNB претпоставља мултиномијалну, а GNB нормалну (Гаусову) расподелу на сваком нумеричком улазном атрибуту.

Ови класификатори махом добро раде на мањим скуповима, као и над текстуалним скуповима. Стога није неочекивано што су добијене прецизности биле тек 55% односно 65% на тест скупу. Чињеница да је други перцентил већи уклапа се у уводно разматрање према којем се многи атрибути уклапају у нормалну, гама или експоненцијалну расподелу. С друге стране, будући лењи и наивни, ови класификатори претпостављају висок степен независности расподела атрибута, што, већ је закључено, не важи на обрађиваном скупу података о словима. У складу са тим, боље су (96%, 58%, 67%) перформансе добијене уклањањем јако корелисаних атрибута – 2, 4, 5, 6. Добит је двојака, пошто се редукцијом улаза успут и смањује димензија проблема.

Напослетку, покушана је примена најбољих класификатора на факторе добијене анализом главних компоненти. Резултати су табеларно представљени вертикалним тројкама следећег облика: број компоненти, KNN прец., SVM прец. (скрипт „PCA.py“).

n	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
KNN	95	96	95	95	95	95	93	92	91	88	84	74	63	41	22	13
SVM	97	97	97	97	96	96	95	93	91	89	85	76	65	44	20	6

3.6. Додатак

До сада су коришћене само прераде скриптова доступних на асистенткињиној страници, али је из куриозитета написан и један додатни, који имплементира наивни бајесовски класификатор без икакве претпоставке о расподели.

То значи да су априорне вероватноће да атрибути узму неку од 16 вредности (у овом случају су, дакле, погодно посматрани као категорички, а не нумерички) при некој од 26 класа (што је њих 16 * 26 = 416) рачунате на основу тренинг скупа. Складиштене су у мапи мапа, при чему главна мапа као кључеве садржи категорије, док таблице у које се сликају класе мапирају атрибуте у нумеричку листу тј. низ. Низ је такав да се вероватноћа да атрибут узме неку вредност у тој класи налази на баш том месту у низу. Другим речима, у мапи се на месту probs[k][a][i] налази вероватноћа да у класи *k* атрибут *a* узме вредност *i*. Дозвољене су ситне вероватноће (0.00005) за вредности атрибута које се не налазе у тренинг скупу, мада то није утицало на прецизност.

Даље су за сваку инстанцу тест скупа формиране мапе вероватноћа, где се на месту odabir[k] за текући објекат налази апостериорна вероватноћа да он припада класи *k* – заправо вероватноћа класе *k* при том објекту. Она је израчуната на основу вредности атрибута убачених у Бајесову теорему, уз неизбежну наивну претпоставку о њиховој независности. Као додељена класа узимала се она која максимизира функцију веродостојности, односно она која се мапира у највећи број у таблици odabir. У случају једнаке вероватноће неких категорија, бира се прво слово у лексикографски уређеном низу. Примењене формуле доступне су на следећем исечку из презентације.

- Pretpostavka o nezavisnosti između atributa

$$P(C|A_1, A_2, ..., A_n) = \frac{\prod_{i=1}^{n} P(A_i|C) * P(C)}{P(A)}$$

- Određivanje klase

$$\widehat{C} = \arg\max_{C} \prod_{i=1}^{n} P(A_i|C) * P(C)$$

За потребе имплементације овог класификатора, формирана је и vrvk, мапа вероватноћа класа. Иако се могла претпоставити равномерна расподела, што би било у складу са уводним разматрањима, то је негативно утицало на прецизност, па је ипак избегнуто. Када је у питању претпоставка о међусобној независности атрибута, и овде је брзину и прецизност поправило искључивање јако корелисаних атрибута.

Све је доступно у скрипту „Bayes.py“, а следи извештај класификације, на коме се уочава 76-77% погодака на тест скупу, што је боље од уграђених MNB и GNB.

```
Izvestaj klasifikacije:
              precision    recall  f1-score   support

    accuracy                       0.76      6000
   macro avg       0.77      0.76      0.76      6000
weighted avg       0.77      0.76      0.76      6000
```

4. ЗАКЉУЧАК

Препознавање слова из скупа података „Letter Recognition“ показало се као лако савладив проблем, упркос постојању чак 26 класа (категорија). У раду са 20.000 инстанци и 16 улазних атрибута успешно је конструисано неколико модела који са великом прецизношћу погађају које је слово представљено којим објектом.

При раду у IBM-овом SPSS Modeler-у најбољи конструисани модел био је онај имплицитни (лењи), методом једног најближег суседа (1NN), са прецизношћу 95,37% на тест скупу од око 6.000 инстанци (30%). На истом тест скупу, веома слично се показао метод три до пет најближих суседа (KNN), са прецизношћу 95,21%. Тачност предикције је незнатно опала при замени перцентила слогова у тренинг и тест скупу.

Следећи најбољи модел био је експлицитни, и то дрво одлучивања формирано алгоритмом C5.0, при чему су искоришћене све његове погодности (бустовање, расејавање атрибута, унакрсна валидација). Постигнута прецизност на тест скупу била је ~94%. Добро се показала и случајна шума, са прецизношћу ~93% на истом скупу.

Математички заснованом методом потпорних вектора (SVM) постигнута је прецизност ~86% на тест скупу. Сличне перформансе имали су модели направљени помоћу вишеслојне неуронске мреже и логистичке регресије, док су алгоритми C&RT, CHAID, QUEST и други фомирали моделе са прецизношћу ~70% и мање.

Нешто нижа прецизност била је и код модела формираних помоћу пет фактора – атрибута добијених као излаз из анализе главних компоненти (PCA) – највећа ~70% на тест скупу. С друге стране, у случају сажетог (преполовљеног) скупа атрибута добијеног уклањањем оних који су се показали као јако корелисани и небитни предиктори, добијена је прецизност ~91% на скупу за проверу, што је веома високо.

При раду у Python-у, најбоље се показао SVM, који је уједно био и свеукупно најбољи модел, са прецизношћу ~99% на тренинг и ~97% на тест скупу. Неуронска мрежа дала је прецизност ~92%, док је ентропијско стабло погађало ~88% класа. И овде се KNN одлично показао, са прецизношћу ~96% при погодним параметрима, док су остали лењи класификатори били лошији, са ~76% код простог наивног бајесовског.

Што се тиче фактора добијених из PCA, успех је био >90% за узимање осам или више атрибута. Ови резултати, природно, подсећају на оне добијене искључивањем висококорелисаних атрибута. Највиша прецизност одржава се са најмање 13 фактора.

Свеукупан закључак је да није постојала препрека у формирању поузданих и прецизних модела класификације обрађиваног скупа. Наиме, у бољим моделима греши се тек свако 20-33. слово. Ако се говори о препознавању у контексту, проблем је још мањи. Примера ради, уколико би класификација слово по слово као резултат над подацима о некој речи вратила нпр. „CDMFUTEB“, већина програма за аутоматско исправљање грешака у куцању (autocorrect) и предлагање измена умела би да препозна ову реч као „COMPUTER“ или да је предложи кориснику. Уколико је још доступан и програм који препознаје контекст (нпр. Grammarly), спречен би био и добар део грешака где је замена слова стварно могућа, попут „POOL“ (базен) и „FOOL“ (будала).

ЛИТЕРАТУРА

1. Slate, David J.: „Letter Recognition Data Set” (доступно на интернет страници „http://archive.ics.uci.edu/ml/datasets/Letter+Recognition”). *UCI Machine Learning Repository*. Irvine, Center for Machine Learning and Intelligent Systems, Bren School of Information and Computer Science, University of California.
2. Frey, Peter W.; Slate, David J.: „Letter Recognition Using Holland-Style Adaptive Classifiers” (доступно на „http://www.cs.uu.nl/docs/vakken/mpr/Frey-Slate.pdf”). *Machine Learning, 6*. Boston, Kluwer Academic Publishers, 1991, стр. 161-182.
3. Документација за IBM SPSS Modeler и коришћене Python библиотеке.