Универзитет у Крагујевцу

Системи за подршку одлучивању

Домаћи задатак 1

Класификациони алгоритми са надгледаним учењем

Студент: Кристина Старић

20/2019

Професор: ред. проф. др Ненад Филиповић

Асистент: Тијана Шуштершич

Крагујевац, 2020.

Садржај:

1. Логистичка регресија (Logistic Regression)

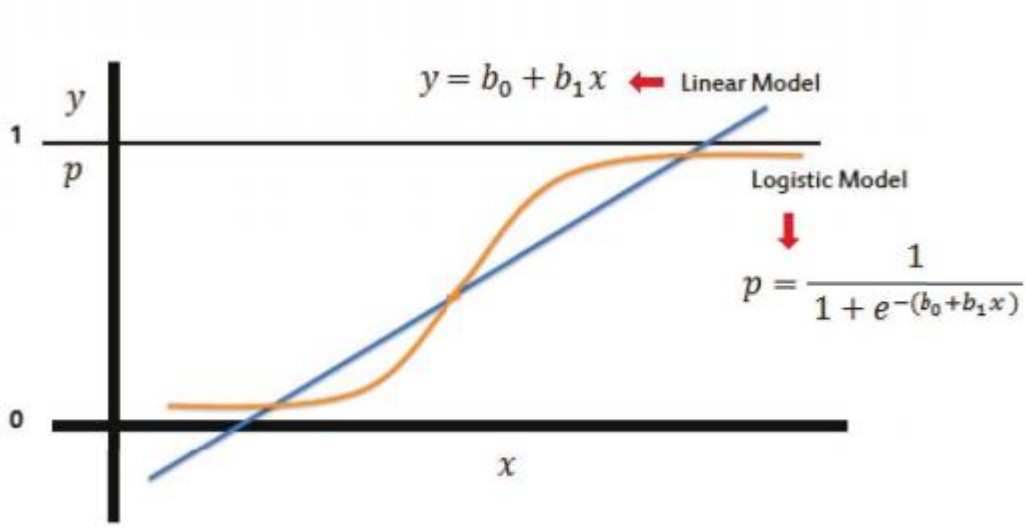
Логистичка регресија је један од најпопуларнијих алгоритама класификације. Основна верзија алгоритма служи за бинарну класификацију. Најчешће се једна класа означава са y = 1, а друга са y = 0. Представио га статистичар Дејвид Кокс (David Cox) 1958. године. Широко је распостарањен у економским, социолошким и медицинским анализама. Саставни елемент многих архитектура неуралних мрежа. Име је добио по логистичкој/сугмоидалној функцији која се користи у хипотези и има облик:

$$g\,(z) = \frac{1}{1 + e^{-z}}$$

Логистичка функција сужава интервал $(\infty, -\infty)$ на опсег [0,1]. То омогућује да се на излазу добије вредност која представља вероватноћу да је y = 1.
Хипотеза логистичке регресије је:

$$h\,(x) = \frac{1}{1+e^{-(Wo+W1X1+W2X2+\cdots WnXn)}};$$

где је n број одлика које се користе у моделу [1].



СЛИКА 1 – РАЗЛИКА ИЗМЕЂУ ЛИНЕАРНЕ И ЛОГИСТИЧКЕ ФУНКЦИЈЕ [1]

Због краће нотације обично се уводи фиктивна одлика $x_0 = 1$, тако да хипотеза постаје:

$$h(x) = \frac{1}{1+e^{-(WoXo+W1X1+\cdots WnXn)}} = \frac{1}{1+e^{-\sum_{i=0}^{n} WiXi}} = \frac{1}{1+e^{-W\cdot X}} = \frac{e^{W\cdot X}}{e^{W\cdot X}+1} = \frac{e^{W^T X}}{e^{W^T X}+1};$$

Вредност h (x) представља вероватноћу да је y = 1:

$$P(y = 1|x) = h(x) = \frac{e^{W^T X}}{e^{W^T X} + 1}$$

Вероватноћа за класу y = 0 се добија као:

$$P(y = 0|x) = 1 - P(y = 1|x) = 1 - h(x) = 1 - \frac{e^{W^T X}}{e^{W^T X} + 1} = \frac{1}{e^{W^T X} + 1}$$

Нов податак се класификује у класу која је за њега вероватнија, то значи да се податак класификује у класу $y = 1$ за $h(x) > 0.5$, а у класу $y = 0$ за $h(x) < 0.5$

Логистичка регресија у Пајтону (Python-у)

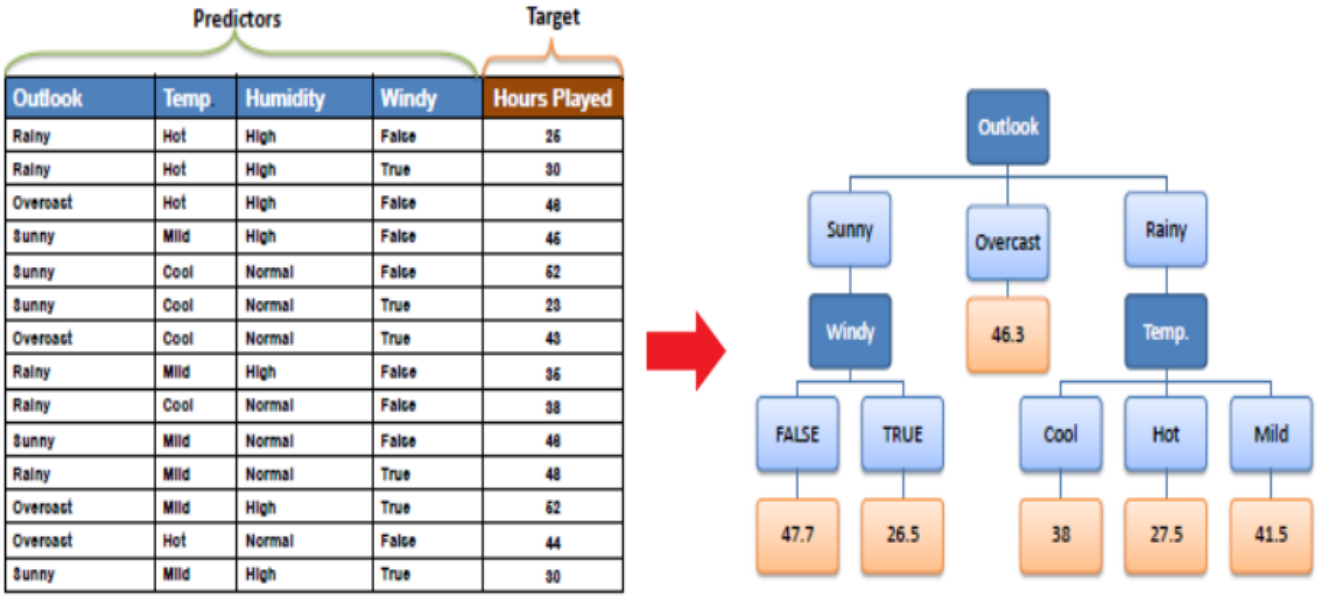
За логистичку регресију користи се библиотека Sklearn. Помоћу библиотеке Sklearn могу се раздвојити подаци на трениг и тест, тј. у две класе и израчунати перформансе модела

2. Стабло одлучивања (Decision Tree Learning)

Стабло одлучивања је тип надгледаних алгоритама (енг. supervised learnig). Најчешће се користи код проблема класификације. Циљ овог алгоритма је модел који предвиђа излазну вредност на основу улазних вредности неколико параметара. Сваки чвор одговара улазном параметру, свака грана (која потиче из посматраног чвора) одговара некој вредности тог параметра. Листови представљају излазне параметре у зависности од вредности улазних параметара на датом путу кроз стабло, од корена до посматраног листа (то су обележја класе, тзв. class label). Обележја класе: да или не, + или -, итд. Према типу излазне вредности:

- класификациона стабла
- регресивна стабла

Код класификационих стабла излазна променљива је категоричког типа, док код регресивних стабала је континуалног типа.Оба стабала деле простор за предвиђање у непреклапајуће регије [2].



СЛИКА 2 – ПРИМЕР СА СТАБЛОМ ОДЛУЧИВАЊА [2]

Што се тиче саме примене класификације, користи се у медицинској дијагностици, анализи употребе кредитних картица, детекцији превара у електронском пословању, откривање вируса и нежељеног садржаја, препознавање говора, препознавање слика, препознавање рукописа

Изградња стабла

Стабла се граде од корена, ка листовима, тзв. „похлепним “ приступом, тј. рекурзивним бинарним дељењем. На почетку, све инстанце припадају једној регији, а простор се сукцесивно (једнан за другим) дели на регионе. За дељење кажемо да је похлепно зато што се при сваком кораку најбоља подела одређује на основу стања у посматраном кораку, односно не узима се у обзир како ће се подела извршити у наредним корацима и која би подела могла довести до бољих резултата у наредним корацима. Поступак поделе се наставља док се не досегне до дефинисаног критеријума за заустављање дељења. Код оба типа стабала се на крају овог поступка добија комплетно разгранато стабло, где су листови достигли дефинисани критеријум заустављања. На тачност предвиђања стабла највише утиче метод којим се врши рачвање у чворовима. Критеријум поделе је различит у зависности од тога да ли се гради регресивно или класификационо стабло. Избор алгоритма се врши на основу стабла које се гради.

Како изабрати „најбољи атрибут“?

- Random: случајним избором изабрати било који атрибут
- Least-values: изабрати атрибут који има најмањи број могућих вредности
- Most-values: изабрати атрибут са највећм бројем могућих вредности
- Max-Gain: изабрати атрибут који има највећу очекивану добит информација (information gain)

Важност стабала одлучивања

Стабла одлучивања су добра за графичко представљање и једноставно интерпретирање посматраног модела.

Предност: можемо користити и уколико поједини атрибути имају недостајуће вредности.

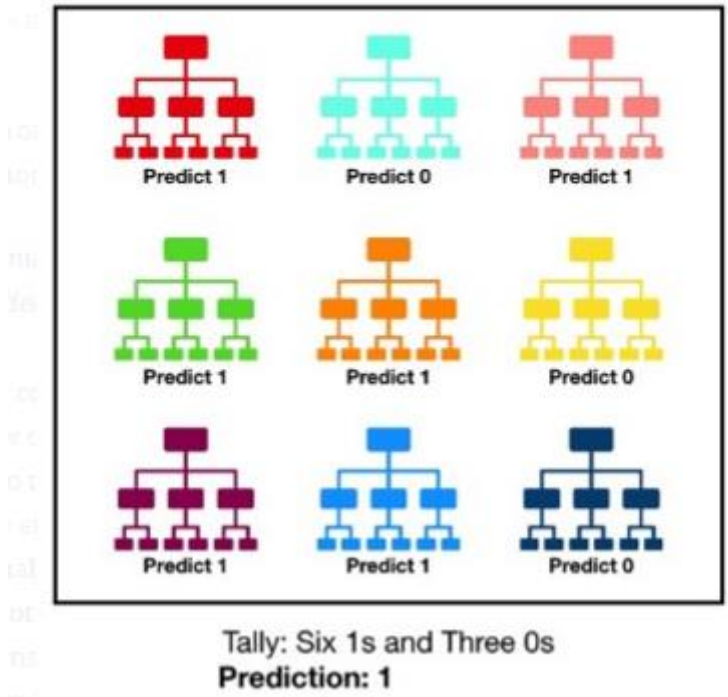
Недостатак: слабија тачност предикције у поређењу са другим приступима машинског учења [2].

Стабло одлучивања у Пајтону (Python-у)

Стабло се учи методом CART из библиотеке csikit-learn. За приказ стабла се користе методи из библиотеке graphviz.

3. Алгоритам случајних шума (Random Forest)

Random Forest састоји се од великог броја индивидуалних стабала. Свако појединачно стабло у случајној шуми даје предвиђање класе. Класа са највише гласова постаје предвиђање нашег модела (види слику доле) [3].



СЛИКА 3 – RANDOM FOREST [3]

Основни концепт је једноставан, али моћан – мудрост гужви. У науци о подацима, разлог зашто случајни модел шуме тако добро функционише је велики број релативно неусклађених модела (стабала), који делују као одбор надмашиће било који од појединачних конститутивних модела. Кључна је ниска повезаност између модела. Иако нека стабла могу бити у криву, многа ће друга стабла бити у праву, тако да се као група стабла могу кретати у правом смеру. Дакле, предуслови да би случајна шума добро функционисала су:

- У нашим карактеристикама мора постојати неки стварни сигнал, тако да модели направљени користећи те функције раде боље него случајно нагађање.
- Предвиђања (и грешке) која су направила појединачна стабла морају имати малу међусобну ниску повезаност [3].

Општи алгоритам алгоритма случајних шума:

- Одабрати број стабала n, број атрибута из којих ће се генерисати стабла m и број узорака за тренирање N.
- Одабрати N узорака из тренинг скупа користећи случајни одабир с понављањем.
- Изградити стабло користећи претходно одабране податке и величину m за тренутни чвор. Од m атрибута узети онај који даје највећу информацијску добит.
- Понављати корак 2 и 3 док се не изгради n стабала.

- Наћи категорију коју је за резултат дала највећи број стабала и ту категорију дати као резултат алгоритма.

Random Forest у Пајтону (Python-у)

За Random Forest у оквиру језика Python користи се RandomForestClassifier класа из библиотеке sklearn.ensemble.

4. Класификатор Наивни Бајес (Naive Bayes Classifier)

Класификатор Наивни Бајес користи се за класификационе проблеме и заснива се на Бајесовој теореми:

$$P(H \mid E) = \frac{P(E \mid H) \cdot P(H)}{P(E)}$$

где је : H – хипотеза;

E – опажај везан за хипотеза H, тј. подаци на основу којих би требало да потврдимо или одбацимо хипотезу H;

P(H) – веровања хипотезе H (prior probability);

P(E) – вероватноћа опажаја тј. стања на које указују прикупљени подаци;

P(E | H) – (условна) вероватноћа опажаја E уколико важи хипотеза H;

P(H | E) – (условна) вероватноћа хипотезе H уколико имамо опажај E.

Наивни Бајес уводи две „наивне“ претпоставке над атрибутима:

1. сви атрибути су а приопри подједнако важни
2. сви атрибути су статистички независни (вредност једног атрибута нам не говори ништа о вредности другог атрибута) [4].

Ове претпоставке најчешће никада нису тачне, али се у пракси ипак добијају добри резултати [4].

Предности Наивног Бајесовог класификатора

Главне предности су једноставност, брзина учења и класификације – довољан један пролаз кроз податке. Такође се добро се понаша када постоји већи број подједнако важних одлика. Већа тежина придружена одређеној одлици значи да је она важнија при

одлучивању, интерпретабилност. Директно је примењив на вишекласну класификацију. Модел се лако може ажурирати новопристиглим подацима [5].

Мане Наивног Бајесовог класификатора

Мане су (нетачна) претпоставка о статистичкој независности одлика – када су две одлике корелисане, долази до дуплог бројања што даје искривљене процене о важности одлика, такође и вредности апостериорних вероватноћа које модел даје су често искривљене у корист највероватније класе.

Употреба Наивног Бајесовог класификатора

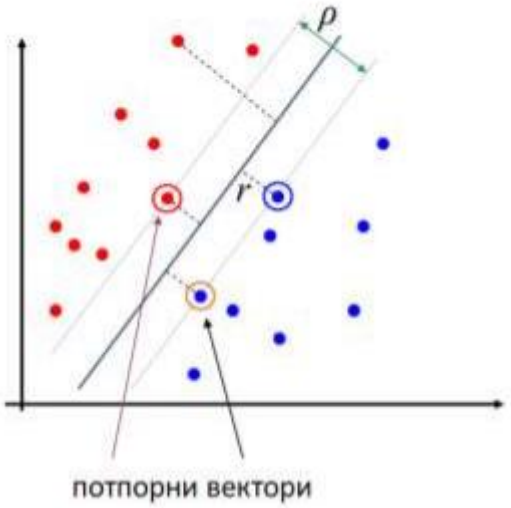
Често се користи као baseline због своје једноставности и брзине. Постиже боље перформансе од сложенијих модела када је доступно мало података за обучавање. Због брзине је компетитиван и када је доступно пуно података. Такође је распрострањен алгоритам у класификацији тексуалних података (детекција спама,..). Користи се у медицинској дијагностици, анализи геномских података... [5].

Наивни Бајес у Пајтону (Python-у)

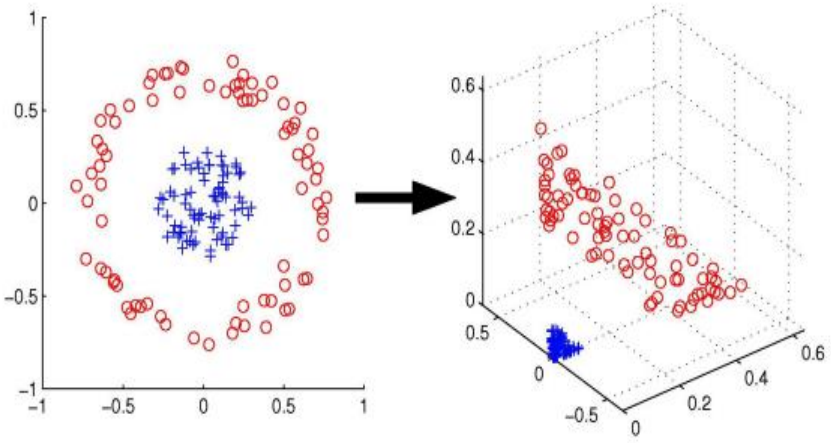
За имплементацију Наивног Бајеса користи се библиотека sklearn.naive_bayes.

5. Алгоритам потпорних вектора (Support Vector Machine)

Основну варијанту методе потпорних вектора су представили Владимир Вапник и Алексеј Червоненкис 1963. године. 1992. године је група аутора предложила проширење на нелинеарне случајеве коришћењем кернелског трика (енг. kernel trick). Кортез и Вапник су 1993. године предложили, 1995. године објавили варијанту алгоритма са меким маргинама (енг. soft margin). Циљ је пронаћи хипер-раван у N-димензионалном простору (N – број карактеристика) који јасно класификује тачке података. При класификацији података потребно је пронаћи хиперраван која раздваја податке који припадају различитим класама. Ако су подаци линеарно сепарабилни могуће је пронаћи бесконачно много хиперравни које су у стању да изврше раздвајање података различитих класа и да се између њих не налази ни један податак. Област простора између ових замишљених хиперравни се назива маргином. Циљ обучавања модела у методи потпорних вектора је да се пронађе максимална маргина, тј. максималну удаљеност између тачака података обе класе. Максимално повећавање маргине омогућава неко појачање тако да се будуће тачке података могу класификовати са већим поуздањем. Маргина ρ је ширина раздвајања између класа коју треба максимизовати и рачуна се као $\rho = 2r$.



СЛИКА 4 – МАРГИНЕ



СЛИКА 5 – ХИПЕР РАВНИ У ДВОДИМЕНЗИОНАЛНОЈ И ТРОДИМЕНЗИОНАЛНОЈ РАВНИ

Обе замишљене хиперравни ефективно належу на један или више података своје класе – ти подаци су потпорни вектори (енг. support vectors). Потпорни вектори „подупиру“ замишљене хиперравни. Назив вектор потичу од посматрања сваког податка као вектора/ тачке у n-димензионалном простору.

Интуитивно гледано, хипераван раздвајања која пролази кроз средину празнине између података двеју класа има више смисла од оне која се налази близу података било једне било друге класе.

Хипотеза методе потпорних вектора

У методи потпорних вектора модел представља једначину хиперравни раздајања. Ако је n број одлика које се користе у моделу тј. број димензија простора, онда је хипотеза облика:

$$h(x) = w_0 + w_1x_1 + w_2x_2 + \cdots + w_nx_n = W^TX + w_0$$

Често се уместо ознаке Wo користи ознака b (eng. bias term).

Једначина хиперравни раздвајања је:

$$h(x) = W^TX + w_0 = 0$$

Вектор W је вектор нормале на хипераван раздвајања.

Support Vector Machine у Пајтону (Python-у)

У језику Пајтон за имплементацију алгоритма Support Vector Machine користи се фукција SVC библиотеке Scikitlearn.

6. Алгоритам К најближих суседа (K Nearest Neighbour (KNN))

Алгоритам машинског учења, који спада у технике надгледаног учења. Може се користити у класификацији и регресији, али су у индустрији чешћи класификациони проблеми предикције.

Сваки алгоритам машинског учења треба да води рачуна о 3 аспекта:

- једноставно тумачење излазних података
- време извршавања алгоритма
- колико је добра моћ предвиђања

KNN у класификацијји: класификује тачку посматрања у односу на то како су суседи класификаковани. Алгоритам је заснован на сличности карактеристика (особина). Избор праве вредности за фактор К је процес који се назива „подешавање параметара“. к у kNN алгоритму представља параметар који означава број накближих суседа који су укључени (у процесу гласања). Прецизност нашег модела ће бити боља што бољи фактор к одредимо.

Избор фактора К:

- $\sqrt{n}$, где је n укупан број свих инстанци (примерака) у скупу података
- бира се непарна вредност к, да би се избегла конфузија којој од две класе припада, [2].

Већа вредност к има мању шансу за грешку. За проналажења најближих суседа најчешће се користи Еуклидско растојање.

Еуклидско растојање у дводимензионалном простору рачуна се као:

$$dist = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

Псеудо код алгоритма kNN

1. учитај податке
2. израчунај вредност фактора К
3. Да добијете предикитну класу, итерирајте од првог до последњег примерка у оквиру трениг скупа података
   3.1. Израчунати раздаљину између тражене инстанце (тачке) и свих инстанци из трениг скупа

3.2. Сортирати израчунате раздаљине у растућем поретку.
3.3. Узети највиших К редова из сортираног низа
3.4. У К редова анализирати колико је број појављивању класа.
3.5. Добијени резултат је предиктивна класа за тражену инстанцу [2]

KNN у Пајтону (Python-у)

За имплементацију KNN алгоритма користи се класа KNeighborsClassifier библиотеке sklearn.neighbors.

7. Литература

[1] Вук Батановић, Обрада природних језика, Београд, 2018/19, Приступљено: 25.11.2019. године, линк: https://rti.etf.bg.ac.rs/rti/ms1opj/

[2] Вук Батановић, Дражен Драшковић, Проналажење скривеног знања, Београд, 2018/19, Приступљено: 25.11.2019. године, линк: https://rti.etf.bg.ac.rs/rti/ms1psz/

[3] Tony Yiu, Understending Random FOrest, приступљено: 4.4.2020. године, линк: https://towardsdatascience.com/understanding-random-forest-58381e0602d2

[4] Никола Миликић, Класификација- Наивни Бајес, приступљено: 4.4.2020. године, линк:http://ai.fon.bg.ac.rs/wp-content/uploads/2015/04/Klasifikacija-Naivni-Bajes-2015.pdf

[5] Вук Батановић, Проналажење скривеног знања, приступљено: 4.4.2020. године, линк: https://rti.etf.bg.ac.rs/rti/ms1psz/pdf/Naivni%20bajesovski%20klasifikator.pdf

8. Списак слика