Примена на машинско учење за класификација на отровни и неотровни печурки

Илија Мижимакоски, Ведрана Петреска, Ангела Секуловска
Факултет за електротехника и информациски технологии,
Универзитет „Св. Кирил и Методиј“- Скопје, РС Македонија
ksiar892021@feit.ukim.edu.mk, kti1572021@feit.ukim.edu.mk, ksiar892021@feit.ukim.edu.mk

Со текот на времето печурките како растенија се многу популарна тема кај луѓето. Поради тоа решивме да направиме анализа на собрани податоци за повеќе видови на печурки за да добиеме информација дали се отровни или не.

Нашиот проект се заснова на база на податоци која е претставена пред околу 30 години на UCI репозиториумот за машинско учење. Со помош на собраните податоци може да се предвиде дали печурките се смртоносни или не и кои карактеристики на печурките се најважни за да се донесе таа одлука.

За таа цел искористивме повеќе алгоритми на машинско учење, така што преку споредба на резултатите од сите алгоритми, селектирање на карактеристики, стандардизација, и справување со null вредностите, добивме модел кој успешно ги класифицира печурките.

Клучни зборови: машинско учење, алгоритми, класификација, печурки

I. Вовед

Нашиот живот во ова модерно време во голема мера зависи од компјутерите. Речиси е невозможно да се размислува за живот без компјутери. Вештачката интелигенција е широко користена во најразлични сфери, почнувајќи од индустријата, науката, економијата, па дури и во здравството. Иако човекот има сомнежи за точноста што може да се постигне со вештачката интелигенција, со развојот на технологијата таа станува дел од секојденвниот живот на сите.

Отровите од печурките се секундарни метаболити произведени од габата. Труењето со печурки обично е резултат на ингестија на диви печурки по погрешна идентификација на токсичната печурка како јадлив вид. Најчеста причина за оваа погрешна идентификација е блиската сличност во однос на бојата и општата морфологија на видовите токсични печурки со видовите за јадење. За да се спречи труење со печурки, собирачите на печурки се запознаваат со печурките што имаат намера да ги соберат, како и со сите токсични видови со сличен изглед. Но тоа може да е процес што трае многу долго и не може човек што не ги познава печурките добро да познае дали печурката е безбедна. За таа цел машинското учење е добар начин за да се предвиде дали некоја печурка е отровна или не.

Започнавме со 21 карактеристика(вклучувајќи ја и класата) на печурките и 61 069 инстанци, кои ги анализиравме, па со нивно редуцирање и додавање на нови карактеристики упеавме да добиеме модел кој што е ефикасно ќе ги класифицира печурките. Cap-shape, cap-surface, cap-color, bruises, odor, gill-attachment, gill-spacing, gill-size, gill-color, stalk-shape, stalk-root, stalk surface above ring, stalk surface below ring, stalk color above ring, stalk color below ring, veil type, veil color, ring-number, ring type, spore print color, habitat, season се карактеритиките со помош на кои ги класифицираме печурките. Почнавме со справување со нул вредностите на два начини, после кое што ни се намалија инстанците и карактеристиките. Продолживме со енкодирање на податоците, feature engineering, стандардизација, тренирање на алгоритмите: Dummy, Random Forest, Support Vector Machine Classifier, K-Nearest Neighbors Classifier, AdaBoost Classifier.

	class	cap-diameter	cap-shape	cap-surface	cap-color	does-bruise-or-bleed	gill-attachment	gill-spacing	gill-color	stem-height	...	stem-root	stem-surface	stem-color	veil-type	veil-color	has-ring	ring-type	spore-print-color	habitat	season
0	p	15.26	x	g	o	f	e	NaN	w	16.95	...	s	y	w	u	w	t	g	NaN	d	w
1	p	16.60	x	g	o	f	e	NaN	w	17.99	...	s	y	w	u	w	t	g	NaN	d	u
2	p	14.07	x	g	o	f	e	NaN	w	17.80	...	s	y	w	u	w	t	g	NaN	d	w
3	p	14.17	f	h	e	f	e	NaN	w	15.77	...	s	y	w	u	w	t	p	NaN	d	w
4	p	14.64	x	h	o	f	e	NaN	w	16.53	...	s	y	w	u	w	t	p	NaN	d	w
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
61064	p	1.18	s	s	y	f	f	f	f	3.93	...	NaN	NaN	y	NaN	NaN	f	f	NaN	d	a
61065	p	1.27	f	s	y	f	f	f	f	3.18	...	NaN	NaN	y	NaN	NaN	f	f	NaN	d	a
61066	p	1.27	s	s	y	f	f	f	f	3.86	...	NaN	NaN	y	NaN	NaN	f	f	NaN	d	u
61067	p	1.24	f	s	y	f	f	f	f	3.56	...	NaN	NaN	y	NaN	NaN	f	f	NaN	d	u
61068	p	1.17	s	s	y	f	f	f	f	3.25	...	NaN	NaN	y	NaN	NaN	f	f	NaN	d	u

61069 rows × 21 columns

Слика 1. База на податоци

На слика 1. е прикажана целата база на податоци, со сите карактеристики и инстанци. Може да забележиме дека првичниот дата сет е со 21 колона (карактеристики) и 61069 редови (инстанци).

II. Претпроцесирање

Претпроцесирање на податоците е процес на генерирање необработени податоци за моделите за машинско учење. Ова е првиот чекор во креирањето на модел за машинско учење и е најкомплексен. Потребно е претпроцесирање на податоците во алгоритмите за машинско учење за да се намали нивната сложеност.

```
class                    2
cap-diameter          2571
cap-shape                7
cap-surface             11
cap-color               12
does-bruise-or-bleed     2
gill-attachment          7
gill-spacing             3
gill-color              12
stem-height           2226
stem-width            4630
stem-root                5
stem-surface             8
stem-color              13
veil-type                1
veil-color               6
has-ring                 2
ring-type                8
spore-print-color        7
habitat                  8
season                   4
dtype: int64
```

Слика 2. Уникатни вредности по колона

Со слика 2 е прикажан бројот на уникатни вредности по колона. Можеме да приметиме дека vail - type има само една вредност што значи дека не ни дава доволно информации, па одлучивме да ја отстраниме таа карактеристика.

II.I) Справување со null – вредности

```
class                        0
cap-diameter                 0
cap-shape                    0
cap-surface              14120
cap-color                    0
does-bruise-or-bleed         0
gill-attachment           9884
gill-spacing             25063
gill-color                   0
stem-height                  0
stem-width                   0
stem-root                51538
stem-surface             38124
stem-color                   0
veil-color               53656
has-ring                     0
ring-type                 2471
spore-print-color        54715
habitat                      0
season                       0
dtype: int64
```

Слика 3. Null-вредности по колона

Од оваа слика може да забележиме дека во некои колони има голем број null вредности, па би било корисно тие карактеристики кои имаат повеќе од 25 000 null-вредности да се отстранат. Како што може да видиме cap-surface исто така има доста вредности што недостасуваат, така што ќе го отфрлиме. Gill-attachment има вредности што недостасуваат (16,2%), што не е многу, затоа одлучивме да ги отфрлиме инстанците чии што вредности недостасуваат во оваа карактеристика. Додека поради малиот број на null-вредности кај ring-type, истите ги пополнивме.

После овие промени дата сетот се состои од 51185 инстанци и 14 карактеристики, во кои што веќе нема null-вредности.
Резултат на ова е тоа што имаме подобрена точност и тоа кај KNN - 99%, Decision Tree- 68%, SVM -81%.

```
class                    0
cap-diameter             0
cap-shape                0
cap-color                0
does-bruise-or-bleed     0
gill-attachment          0
gill-color               0
stem-height              0
stem-width               0
stem-color               0
has-ring                 0
ring-type                0
habitat                  0
season                   0
dtype: int64
```

Слика 4

II.II) Енкодирање на податоците

Колоните со категорични карактеристики често имаат повеќе категории. На пример, обликот на капа може да има категории како ['звонче', 'конусно', 'конвексно', 'рамно', 'потонато',].

One-hot encoding ја трансформира секоја категорија во бинарна колона. На пример, ако формата на капа има 6 уникатни категории, таа ќе се трансформира во 6 бинарни колони каде што само една колона ќе има 1 за секое набљудување, а другите ќе бидат 0. Ова кодирање го спречува моделот погрешно да ги толкува кодираните вредности. На пример, ако сте користеле LabelEncoder на cap-shape, може да завршите со цели броеви [0, 1, 2, 3, 4, 5], а некои модели може погрешно да заклучат дека повисоките вредности имаат посилен или различен ефект, што е не е точно за категорични податоци. One-hot кодирањето е од суштинско значење за да се осигураме дека моделот за машинско учење ја третира секоја категорија како независна карактеристика без да имплицира каков било редослед или хиерархија.

Label encoding е погодно за целната променлива бидејќи ги трансформира класите во формат со кој можат директно да работат многу алгоритми за машинско учење (т.е. нумерички формат). Поважно, кодирањето на етикетите се користи овде бидејќи моделот нема да ги толкува нумеричките ознаки како да имаат некаква врска (т.е. нема да претпостави дека една класа е поголема или помала од друга).
Имплентација на One-hot encoder ја подобрува точноста кај KNN - 99%, Decision Tree- 71%, SVM -88%.

	class	cap-diameter	stem-height	stem-width	cap-shape_b	cap-shape_c	cap-shape_f	cap-shape_o	cap-shape_p	cap-shape_s	...	habitat_h	habitat_l	habitat_m	habitat_p	habitat_u	habitat_w	sea
0	1	15.26	16.95	17.09	False	False	False	False	False	False	...	False	False	False	False	False	False	
1	1	16.60	17.99	18.19	False	False	False	False	False	False	...	False	False	False	False	False	False	
2	1	14.07	17.80	17.74	False	False	False	False	False	False	...	False	False	False	False	False	False	
3	1	14.17	15.77	15.98	False	False	True	False	False	False	...	False	False	False	False	False	False	
4	1	14.64	16.53	17.20	False	False	False	False	False	False	...	False	False	False	False	False	False	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
61064	1	1.18	3.93	6.22	False	False	False	False	False	True	...	False	False	False	False	False	False	
61065	1	1.27	3.18	5.43	False	False	True	False	False	False	...	False	False	False	False	False	False	
61066	1	1.27	3.86	6.37	False	False	False	False	False	True	...	False	False	False	False	False	False	
61067	1	1.24	3.56	5.44	False	False	True	False	False	False	...	False	False	False	False	False	False	
61068	1	1.17	3.25	5.45	False	False	False	False	False	True	...	False	False	False	False	False	False	

51185 rows × 78 columns

Слика 5. Кодирани категоричните карактеристики со One-hot encoding и Label encoding

II.III) Генерирање нови карактеристики

Генерирање нови карактеристики (Feature Engineering) е процес на селектирање, манипулација и трансформација на податоци во карактеристики кои што можат да се употребат во Supervised learning со цел подобрување на перформансите на моделот.

Додадовме 3 карактеристики и тоа:

Stem ratio = Сооднос на ширината на стеблото (Stem Width) и висината на стеблото (Stem Height): Овој сооднос потенцијално може да укаже на различни видови печурки или модели на раст поврзани со токсичност. Искривен или невообичаен сооднос може да сугерира токсичност, но повторно, тоа може да варира меѓу видовите.

Stem diameter (Дијаметар на стеблото): Отровните печурки може да имаат подебели или потенки стебла во споредба со неотровните поради различните структурни барања или еволутивните адаптации. Сепак, оваа корелација можеби не е силна или конзистентна кај сите видови.

Stable ring types (Стабилни типови на прстени) обично се посуштински и поиздржливи, честопати опстојуваат додека печурката созрева, а понекогаш дури и останува прицврстена за стеблото или остава остаток или лузна. Нестабилните типови прстени, од друга страна, обично се деликатни и ефемерни, често исчезнуваат или се распаѓаат релативно брзо по созревањето на печурката.

II.IV) Стандардизација

Стандардизацијата е статистичка техника која се користи при претходна обработка на податоците за да се направат различните променливи поспоредливи. Тоа е како сите овие различни „јазици“ на податоци да се преведат на еден универзален дијалект. нестандардизираните податоци може да доведат до неточни анализи.

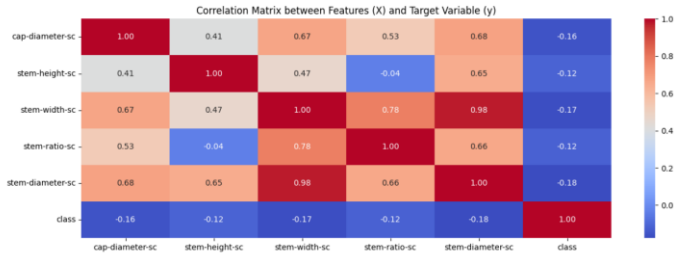
	cap-diameter	stem-height	stem-width	stem-ratio	stem-diameter
0	15.26	16.95	17.09	1.008260	17.020
1	16.60	17.99	18.19	1.011117	18.090
2	14.07	17.80	17.74	0.996629	17.770
3	14.17	15.77	15.98	1.013316	15.875
4	14.64	16.53	17.20	1.040532	16.865
...	...	...	...	...	...
61064	1.18	3.93	6.22	1.582697	5.075
61065	1.27	3.18	5.43	1.707547	4.305
61066	1.27	3.86	6.37	1.650259	5.115
61067	1.24	3.56	5.44	1.528090	4.500
61068	1.17	3.25	5.45	1.676923	4.350

51185 rows × 5 columns

Слика 6. Стандардизирани карактеристики

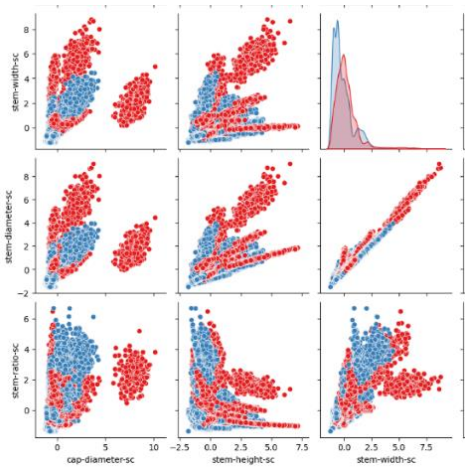
III. Визуелизација

Визуелизацијата на податоците има голема примена во машинското учење за подобро да се разбере и анализира сложена база на податоци преку нивно презентирање во лесно разбирлив формат. Визуелизацијата на податоците е суштински чекор во подготовката и анализата на податоците бидејќи помага да се откријат трендовите и моделите во податоците што може да се пропуштат при другите форми на анализа. Алгоритмите за машинско учење најдобро функционираат кога имаат висококвалитетни податоци, а визуелизацијата на податоците може да помогне да се идентификуваат и отстранат сите недоследности или аномалии во податоците. За таа цел, најпрво, со помош на корелациона матрица, ќе ја прикажеме корелацијата помеѓу нумеричките карактеристики и класата. Притоа, може да се воочи дека класата има мала корелација со секоја од нив.



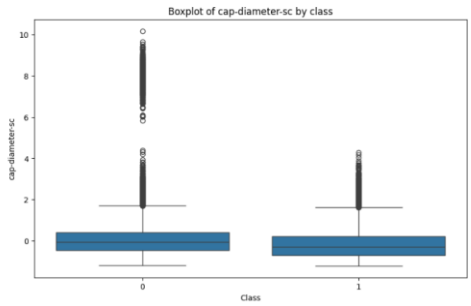
Слика 7. Корелациона матрица на на карактеритика (x) и класата(y)

Pair plot, исто така познат како матрица на расфрлање, е корисна алатка за визуелизација во истражувачката анализа на податоци, особено во контекст на машинското учење. Тоа овозможува да се визуелизираат односите помеѓу повеќе променливи во базата на податоци со создавање на scatter plots за секој пар на променливи. Ова може да помогне во разбирањето на дистрибуцијата на податоци, откривање на корелации и идентификување на потенцијалните outliers (вредности кои значително отстапуваат од другите во дата сетот).

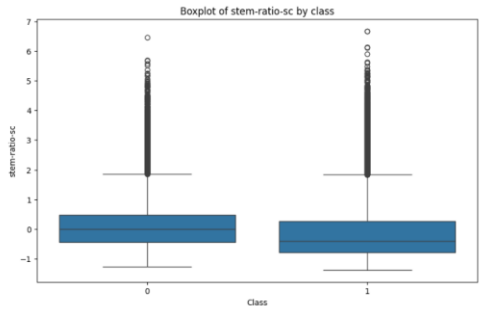


Слика 8. Pair plot график

Преку овој дијаграм забележавме дека има outliers на карактеристиките, па наредно тоа го проверивме преку boxplot за секоја нумеричка карактеристика.

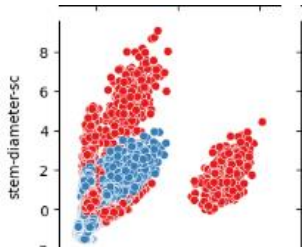


Слика 9. Boxplot на cap-diameter

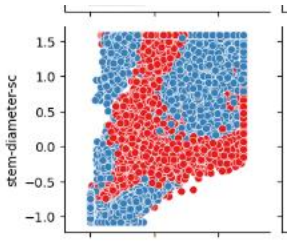


Слика 10. Boxplot на stem-ratio

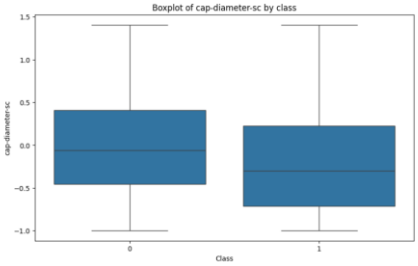
Преку овие boxplots ни се потврди дека има многу outliers. Со овој проблем се справивме со помош на винсоризација. Тоа е техника која се имплементира со функцијата winsorize која се користи за ограничување на екстремните вредности во базата на податоци за да се намали влијанието на потенцијалните outliers која со прилагодување на екстремните вредности на одреден перцентил помага да се направат поробусни наспроти влијанието на outliers.



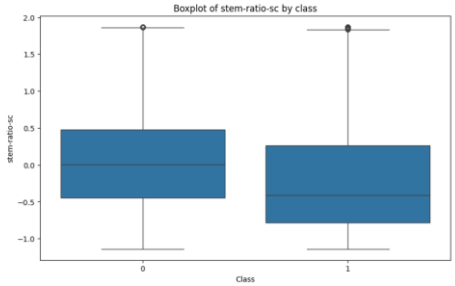
Слика 11. Пред справување со outliers



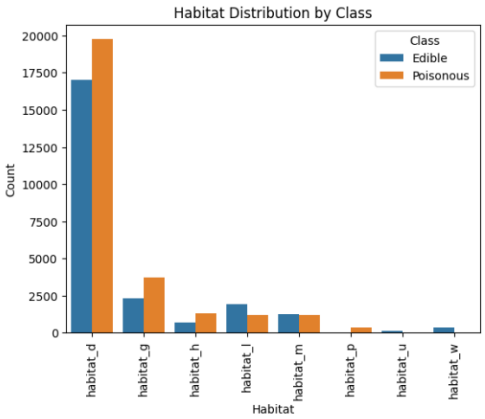
Слика 12. После справување со outliers



Слика 13. Boxplot на cap diameter после винсоризацијата



Слика 14. Boxplot на stem ratio после винсоризацијата.



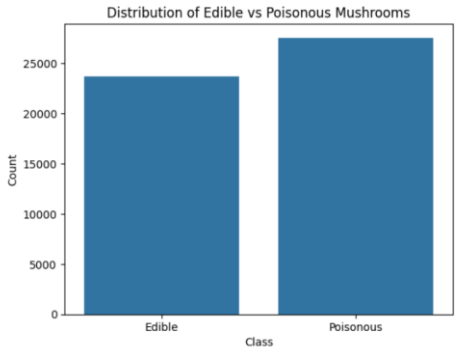
Слика 15. Од овој столбест дијаграм може да забележиме дека печурките од овој дата сет најмногу се застапени во habitat_d (шуми)

IV. АЛГОРИТМИ НА МАШИНСКО УЧЕЊЕ И ЕВАЛУАЦИЈА

Алгоитми кои ги користевме за тренирање на дата сетот се следниве:

- Dummy
- Random Forest
- Support Vector Machine Classifier
- K-Nearest Neighbors Classifier
- AdaBoost Classifier
- Decision Tree Classifier

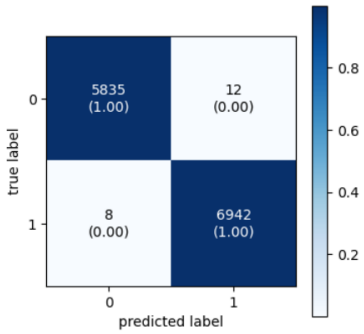
Dummy алгоритмите нудат основни перформанси што треба да ги надминат пософистицираните модели. Со разбирање и користење на dummy алгоритмите, може подобро да се оцени ефективноста на моделите за машинско учење и да се осигураме дека тие навистина учат од податоците наместо да се потпираат на поедноставени правила или случајност. Овој класификатор прави предвидувања користејќи основни стратегии без да се земат предвид влезните карактеристики



Слика 16. Распределба на отровни и неотровни печурки

Од резултатот на dummy класификаторот може да забележиме дека дата сетот е балансиран со 53% отровни печурки и 47% неотровни печурки.

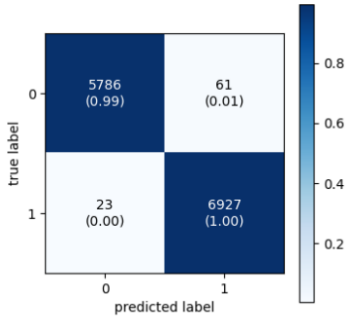
Random Forest е популарен и моќен алгоритам за машинско учење. Тоа е ансамбл метод, што значи дека гради повеќе дрва на одлучување и ги спојува заедно за да се добие попрецизно предвидување.



Слика 17. Матрица на конфузност на Random Forest

Од оваа матрица на конфузност можеме да видиме дека овој класификатор предвидел точно 5835 неотровни печурки, додека12 од нив ги предвидел како отровни. Исто така, предвидел 6942 отровни како такви, а само 8 од нив ги погрешил. Со овој алгоритам постигнавме точност од 99.84371337032117%.

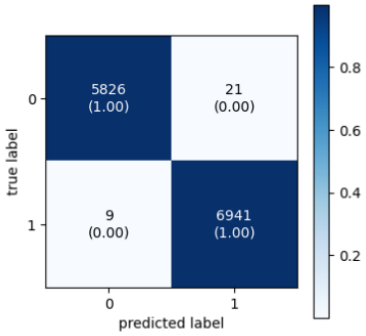
Support Vector Machine Classifier (SVM) е разноврсна и моќна алатка за класификација. Тој е погоден за линеарни проблеми каде што со помош на хиперрамнина ги разделува позитивната и негативната класа. Со искористување на „kernel trick“, SVM може да се справат со нелинеарни врски, што го прави вредна алатка за машинско учење.



Слика 18. Матрица на конфузност на SVM

Од оваа матрица на конфузност можеме да видиме дека овој класификатор предвидел 5786 неотровни печурки како неотровни и 61 од нив како отровни. Исто така, предвидел 6927 отровни како такви, а 23 од нив како за јадење. Точноста со овој алгоритам е 99.34359615534891%.

K-Nearest Neighbors Classifier - KNN е јасен и ефективен алгоритам за машинско учење, особено корисен за мали до средни групи на податоци. Иако може да биде интензивен и чувствителен на ирелевантни карактеристики, правилното подесување и преобработка (како што се скалирање на карактеристики и избор на карактеристики) може да помогне да се ублажат овие проблеми.

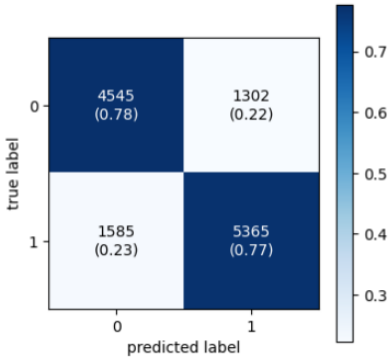


Слика 19. Матрица на конфузност на KNN

Од оваа матрица на конфузност можеме да видиме дека овој класификатор предвидел 5826 неотровни печурки како неотровни и 21 од нив како отровни. Исто така, предвидел 6941 отровни како такви, а 9 од нив како за јадење. Точноста со овој алгоритам е 99.76557005548176%.

AdaBoost е техника во машинското учење што се користи како метод на ансамбл. Најчестиот проценувач што се користи со AdaBoost се Decision Tree со едно ниво, што значи Decision Tree со само 1 поделба. Овие дрвја се нарекуваат и Decision Stumps.

Она што го прави овој алгоритам е тоа што гради модел и им дава еднакви тежини на сите точки на податоци. Потоа им доделува поголеми тежини на точките кои се погрешно класифицирани. Сега на сите точки со поголема тежина им се придава поголемо значење во следниот модел. Ќе ги задржи моделите за обука додека не се добие помала грешка.

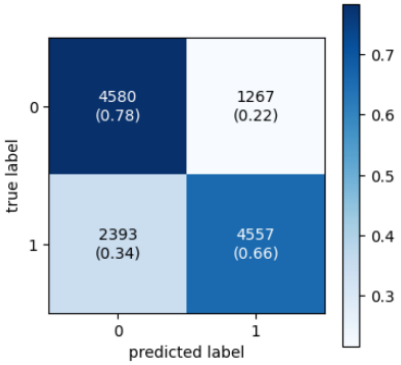


Слика 20. Матрица на конфузност на Decision Tree

Можеме да видиме дека овој класификатор предвидел 4545 неотровни печурки како неотровни и 1302 од нив како отровни. Исто така, предвидел 5365 отровни како такви, а 1585 од нив како за јадење. Од конфузиската матрица може да се воочи дека овој класификатор не е доволно прецизен. Поради тоа со помош на оптимизација на параметри целиме кон подобрување на резултатите од овој алгоритам.

Decision Tree Classifier е алгоритам на машинско учење кој се состои од јазли кои се репрезент за одлуките, гранки кои го претставуваат резултатот од овие одллуки и лисја кои ги претставуваат конечните исходи, т.е предвидувањата. Притоа со

помош на некои метрики како ентропија или gini се одредува кој е најдобар атрибут за да се постави како корен на дрвото и од него да се почне со разгранување.



Слика 21. Матрица на конфузност на Decision Tree

Можеме да видиме дека овој класификатор предвидел 4580 неотровни печурки како неотровни и 1267 од нив како отровни. Исто така, предвидел 4557 отровни како такви, а 2393 од нив како за јадење. Од конфузиската матрица може да се воочи дека овој класификатор не е доволно прецизен. Поради тоа со помош на оптимизација на параметри целиме кон подобрување на резултатите од овој алгоритам.

Работевме две евалуциони техники, со првата го поделивме целиот дата сет на 75% за тренирање и 25% за тестирање, а со втората користевме Cross Validation каде што дата сетот го поделивме на 10 ‘folds’. Двете евалуации дадоја прилично слични резултати.

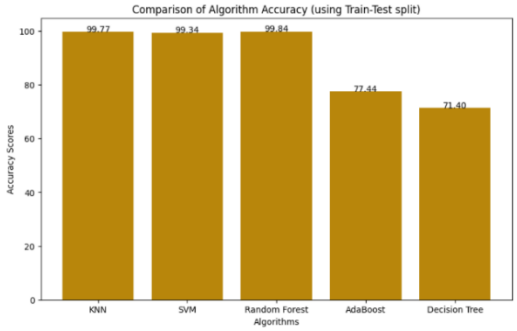
За да се подобрат резултатите од AdaBoost и Decision Tree искористивме оптимизација на хиперпараметри. Оптимизацијата на хиперпараметрите, позната и како подесување на хиперпараметри, е процес на пронаоѓање на најдобриот сет на хиперпараметри за модел на машинско учење. Хиперпараметрите се параметри чии вредности се поставени пред да започне процесот на учење и го контролираат однесувањето на алгоритмот за обука и структурата на моделот.

За AdaBoost го искоритисвме Grid Search алгоритмот за пронаоѓање на најдобрите естиматори и брзина на учење (learning rate) по што добивме бројот на естиматори да е 180, а брзината на учење 0.001. Со оваа оптимизација на параметри точноста се подобри на 99.3045244979292%.

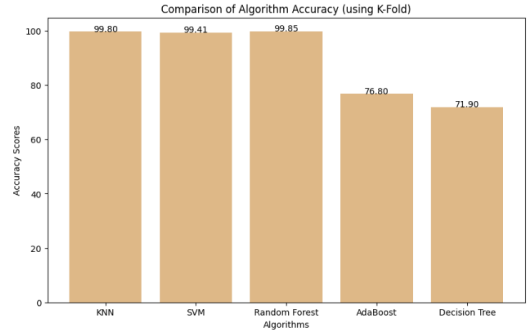
За Decision Tree ќе ги подесиме параметрите на критериумот, max_depth, min_samples_split и min_samples_leaf. Исто така искористивме Grid Search за одредување на овие параметри по што добивме Ентропија како критериум на одлучување, max_depth none (дрвото да се разгранува се додека дозволува дата сетот), min samples leaf: 1, min samples split: 5. Со оваа оптимизација на парамтерите добивме точност 99.35922481831679%.

V. Споредба на добиените резултати

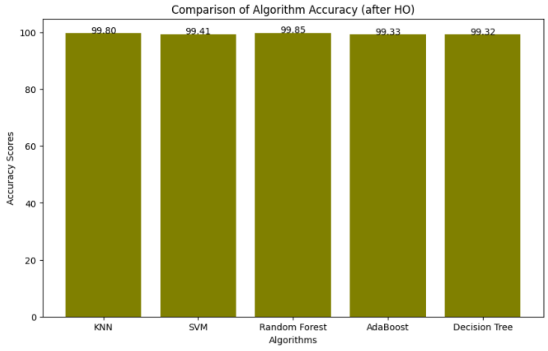
Од добиените резултати можеме да заклучиме дека двете евалуциони метрики дадоа многу слични резултати.
Додека оптимизацијата на хиперпараметрите на Decision Tree и AdaBoost значително ги подобрија истите со скок од дури 20%.
Според овие резултати може да заклучиме дека постигнавме оптималност за сите алгоритми и имаме конзистентност на нивните резултати.



Слика 22. Споредба на алгоритмите со користење на евалуциската техника ‘Train-Test split’



Слика 23. Споредба на алгоритмите со користење на евалуциската техника ‘K-fold’



Слика 24. Споредба на алгоритмите после оптимизација со хиперпараметри

VI. Заклучок

Преку овој проект можеме да согледаме дека машинското учење и вештачката интелегенција наоѓаат примена како во науката така и во секојдневниот живот.

Сепак, иако наизглед, алгоритмите даваат одлични резултати, кон ова треба да пристапиме внимателно. Постои потенцијална опасност од преголема доверба во резултатите од алгоритмите без соодветна верификација. Во конкретен пример, класификацијата на печурките, алгоритмите можат да направат грешки кои можат да имаат сериозни последици. На пример, алгоритамот може да класифицира отровни печурки како добри за конзумација што во реалноста не би смеело да се случи. Ваквите грешки може да доведат до сериозни здравствени ризици, па дури и смрт.

Овој пример ја нагласува важноста на човечкиот надзор и валидацијата на резултатите од машинското учење. Иако алгоритмите може да бидат моќни алатки, тие не се непогрешливи. Затоа, резултатите добиени од алгоритмите за машинско учење секогаш треба да се проверат и потврдат од експерти во соодветната област.