



Институт за математику и информатику
Природно-математички факултет
Универзитет у Крагујевцу

Семинарски рад

STEM Learning Predictions

Ментор и професор

Драгутин Остојић

Вишња Симић

Студент

Марија Јоловић, 46/2021

Септембар, 2025.

Садржај

1. Увод.....	3
2. Опис података.....	4
2.1 Учитавање података	4
2.2 Приказ основних података о скупу.....	4
3. Опис обележја и анализа кроз графички приказ	8
3.1. Нумеричка обележја.....	8
Оцене	8
Изјаве.....	10
3.2. Категоријска обележја.....	10
Пол.....	10
Радионица.....	11
Искуство	12
3.3 Расподела циљне променљиве	13
3.4 Матрица корелације	13
5. Припрема података.....	15
5.1 Уклањање колоне <i>Id</i>	15
5.2 Енкодирање циљне променљиве.....	16
5.3 Детектовање аномалија коришћењем Isolation Forest алгорита	16
5.4 Детекција дупликата	18
5.5 Корекција колоне <i>experience</i>	20
6. Креирање модела машинског учења.....	20
6.1 Метрике за евалуацију модела	20
6.1.1 <i>Accuracy</i>	20
6.1.2 <i>F1 macro</i>	20
6.2 Крос-валидација	21
6.3 Модели.....	21
6.4 Резултати и анализа метрика добијених модела.....	22
Упоредна анализа	24
6.5 Фино подешавање	24
6.5.1 <i>GridSearchCV</i>	24
6.5.2 <i>RandomizedSearchCV</i>	25
7. Одабир најзначајнијих предиктора.....	27
8. Финални закључак	28
9. Литература	28

1. Увод

Савремено друштво се све више ослања на знања из области науке, технологије, инжењерства и математике (СТЕМ – *Science, Technology, Engineering and Mathematics*). Управо зато, развој интересовања ученика за ове области представља један од кључних изазова савременог образовања. Традиционални облици наставе често нису довољни да код ученика пробуде мотивацију и радозналост, па се у свету и код нас све више примењују интерактивне радионице и пројекти који повезују више области и омогућавају учење кроз игру, сарадњу и практичан рад.

У оквиру К1 пројеката Центра за промоцију науке Републике Србије током школске 2021/2022. године реализоване су радионице „Потрага за благом“ и „(Не)Могућа мисија“. Ови пројекти користили су СТЕМ приступ са нагласком на садржаје из математике, физике и информатике. Ученици су, кроз загонетке, практичне експерименте и примену савремених технологија, имали прилику да прошире своја знања и решавају неконвенционалне проблеме. Такав приступ показао се као ефикасан у подстицању креативности, сарадње и активног учења.

Осим практичног дела радионица, спроведено је и анкетање 208 ученика основних и средњих школа, са циљем да се процени њихово искуство и ставови о СТЕМ настави. Анкета је обухватила основне податке о ученицима (пол, узраст, оцене из предмета и претходно искуство), као и 14 изјава на Ликертовој скали које су се односиле на мотивацију, сарадњу, ангажованост и образовни значај радионица.

Циљ овог рада је предвиђање степена сагласности ученика са изјавом: „Верујем да на овај начин могу више да научим у односу на традиционалне методе наставе“ (означена као i8), на основу осталих доступних података. У раду ће бити примењене различите методе машинског учења, упоређени њихови резултати и изабран најбољи модел. Добијени резултати могу дати смернице за унапређење наставних метода и будуће примене СТЕМ приступа у школама.

Сав код који је коришћен за израду овог семинарског рада може се видети кликом на линк [ОВДЕ](#).

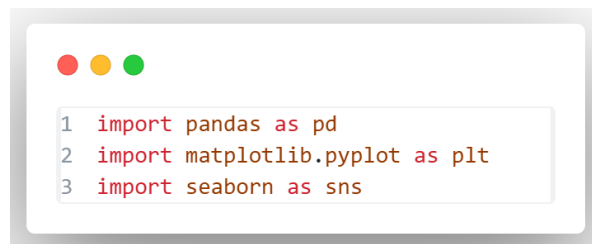
2. Опис података

Аналізу скупа података *STEM Anketa* започећемо описом целокупног скупа података, након чега ћемо пружити детаљнију анализу сваког од обележја. Скуп података садржи 208 редова и више различитих обележја који представљају кључне аспекте за анкету, као што су пол, на којој су радионици били, претходно искуство, оцене из предмета, и одговоре на изјаве.

За потребе анализе података користићемо **Python** програмски језик и низ специјализованих библиотека. Библиотека **Pandas** ће бити коришћена за учитавање скупа података у оквиру података (*data frames*) и за основну обраду. За графички приказ података ослонићемо се на **Matplotlib** и **Seaborn**, док ће за трансформацију и пре процесирање података, као и за креирање модела машинског учења, бити коришћен **Scikit-learn**. Поред ових, употребићемо и библиотеке попут **NumPy** (за рад са нумеричким подацима), и сл..

2.1 Учитавање података

Почињемо са анализом датог скупа података укључивањем потребних Python библиотека на следећи начин:



```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import seaborn as sns
```

Након што смо успешно укључили потребну библиотеку за рад са подацима, учитавамо скуп података о стем анкети:



```
1 df = pd.read_csv(
    "podaci/STEM_anketa.csv")
2 df.head()
```

Од параметара који се прослеђују функцији **read_csv** имамо само путању до локалног фајла са сировим (необрађеним) подацима.

2.2 Приказ основних података о скупу

Након учитавања скупа података, како би се уверили да је учитавање добро прошло и видели приказ података који се у њему налазе, можемо искористити функцију оквира података - **head**, чији је задатак да прикаже првих неколико редова нашег скупа података.

	# id	# workshop	# sex	# class	# math	# physics	# informatics	# experience
0	1	1	1	1	9	5	5	5
1	2	1	1	1	10	5	5	5
2	3	1	1	1	9	3	5	5
3	4	1	1	1	9	2	4	5
4	5	1	1	1	9	5	4	5

Резултат позива ове функције без задавања додатног параметра који означава колико редова желимо приказати резултује у приказу првих пет редова скупа података. Ову функцију је корисно извршити на почетку сваког рада се новим скупом података како би се стекао осећај о изгледу података. Алтернативни приступ био би преглед скупа података у неком од програма као што је Microsoft Excel, или другим сличним програмима који додатно нуде и могућност приказа одређених статистика о скупку података.

# 11	# 12	# 13	# 14	# 15	# 16	# 17	# 18
5	5	5	5	5	5	5	5
5	5	5	3	4	5	4	5
5	5	5	5	5	5	5	5
5	4	4	4	4	4	5	5
3	2	3	4	5	2	2	3

Следећа функција коју можемо искористити зарад детаљнијег увида у податке је **info**, која је задужена за испис сажетог резимеа података.

```

1 import sys
2
3 df.info(
4     verbose=True,
5     buf=sys.stdout,
6     memory_usage=True,
7     show_counts=True
8 )

```

Примећујемо да се и ова функција извршава над објектом оквира података са следећим излазом:

```


<class 'pandas.core.frame.DataFrame'>
RangeIndex: 208 entries, 0 to 207
Data columns (total 22 columns):
#   Column      Non-Null Count  Dtype
---  ---
0   Id           208 non-null   int64
1   workshop     208 non-null   int64
2   sex          208 non-null   int64
3   class        208 non-null   int64
4   math         208 non-null   int64
5   physics      208 non-null   int64
6   informatics  208 non-null   int64
7   experience    208 non-null   int64
8   i1           208 non-null   int64
9   i2           208 non-null   int64
10  i3           208 non-null   int64
11  i4           208 non-null   int64
12  i5           208 non-null   int64
13  i6           208 non-null   int64
14  i7           208 non-null   int64
15  i8           208 non-null   int64
16  i9           208 non-null   int64
17  i10          208 non-null   int64
18  i11          208 non-null   int64
19  i12          208 non-null   int64
20  i13          208 non-null   int64
21  i14          208 non-null   int64
dtypes: int64(22)
memory usage: 35.9 KB

```

Извршавањем функције **info** у стању смо да видимо проширене информације о скупку података. Наиме, можемо видети да се ради о скупку података који садржи **208** врста и **2** колоне (што се такође може видети и коришћењем својства оквира података – **shape**), као

и број вредности различитих од NA (*Not Available*) што нам већ у овом тренутку показује да овај скуп нема класичних недостајућих вредности. Додатно, постављањем параметра *memory_usage* на *True*, можемо видети да овај скуп података у меморији заузима скоро 36 килобајта. Такође, обзиром да тип података заузима веома важно место у анализи и раду са подацима, али и касније у овом раду, он је укључен у излаз ове функције. Можемо уочити да су елементи свих колона целобројног типа.

Оно што је за сваки скуп података веома важно је и број недостајућих вредности које се у њему појављују. Недостајуће вредности (*Missing values*) се могу проверити на више начина коришћењем уграђених функција оквира података, уз њихово увезивање ради прегледнијих резултата.




```
1 # Broj nedostajucih vrednosti za svaku kolonu
2 df.isnull().sum()
```

Id	0
workshop	0
sex	0
class	0
math	0
physics	0
informatics	0
experience	0
i1	0
i2	0

i3	0
i4	0
i5	0
i6	0
i7	0
i8	0
i9	0
i10	0
i11	0
i12	0

i13	0
i14	0

Резултат извршавања ове функције потврђује резултат функције **info**, да нема недостајућих вредности. Још неке од комбинација функција којима се може одговорити на питање да ли у одређеној колони постоје недостајуће вредности или то није случај су:



```
1 df.isna().any()
```

У нашем случају, очекиван резултат је за сваку колону вредност *False*, обзиром да смо претходно закључили да нема непостојећих вредности.

Id	False
workshop	False
sex	False
class	False
math	False
physics	False
informatics	False
experience	False
i1	False
i2	False

Опис података, заједно са дескриптивним статистикама попут броја вредности, средње вредности, стандардне девијације, минимума, максимума и карактеристичних перцентила (квартили) може се генерисати помоћу функције **describe**.

```

1 df.describe(
2     percentiles=[0.25, 0.5, 0.75],
3     exclude=[object]
4 )

```

Пример за првих пар колона, може се видети у табели испод:

	# Id	# workshop	# sex	# class
count	208.0	208.0	208.0	208.0
mean	104.5	1.4807692307692308	1.4951923076923077	8.346153846153847
std	60.18859249614221	0.5008354224706334	0.5011831041664266	1.4092121399718611
min	1.0	1.0	1.0	0.0
25%	52.75	1.0	1.0	7.0
50%	104.5	1.0	1.0	9.0
75%	156.25	2.0	2.0	9.0
max	208.0	2.0	2.0	10.0

Додатно, вредности за које ће се рачунати перцентици се могу задати посебно помоћу параметра *percentiles*, па је функцију *describe* могуће извршити на начин да је могуће израчунати и, рецимо, вредности 10% и 90% перцентици, поред стандардних 25%, 50% и 75%.

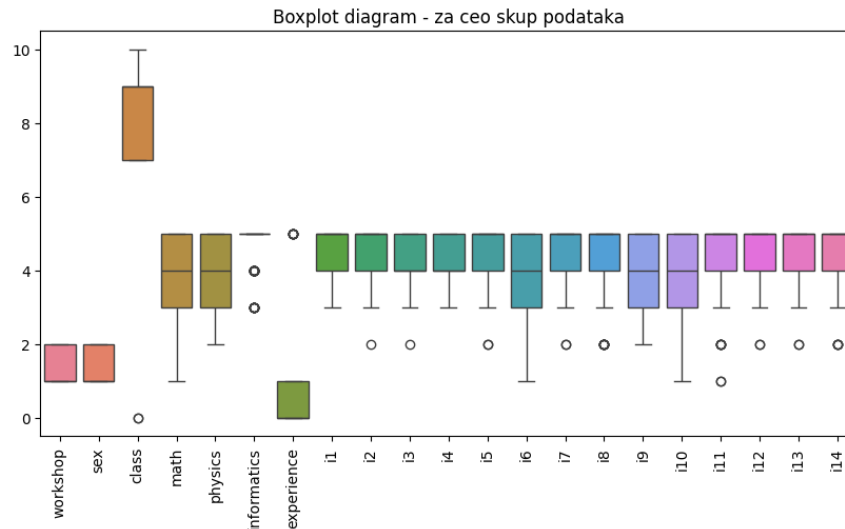
Познавајући резултат функције *describe*, као и појма шта представљају квартили и перцентици, можемо исцртати и боксплот дијаграме за цео скуп података.

```

1 # Vizuelizacija celog skupa podataka boxplot diagramom
2
3 plt.figure(figsize=(10,5))
4 sns.boxplot(data=df.drop(columns=['Id']))
5 plt.xticks(rotation=90)
6
7 plt.title("Boxplot diagram - za ceo skup podataka")
8 plt.show()

```

Са боксплот дијаграма, избацили смо колону која представља *Id* испитаника, обзиром да би њене вредности биле у опсегу [1, 208], што би довело до нескалираних података у односу на друге колоне.



Слика 1 Боксплот дијаграми свих обележја скупа података

3. Опис обележја и анализа кроз графички приказ

Графички приказ података представља један од кључних корака у анализи обележја унутар скупа података. Кроз визуелизацију лакше се могу идентификовати обрасци, односи и аномалије које би могле остати неприметне кроз једноставне нумеричке приказе.

Осим што олакшава разумевање понашања појединачних обележја, графичка анализа омогућава доношење одређених закључака који могу бити од кључног значаја за даљу обраду података и изградњу модела.

Закључци изведени из графичких приказа не само да олакшавају разумевање података, већ и усмеравају на кључне аспекте које би требало додатно истражити. У овом поглављу ће бити приказани различити графици за нека од обележја уз објашњење кључних запажања и закључака који се могу извући на основу приказа података.

3.1. Нумеричка обележја

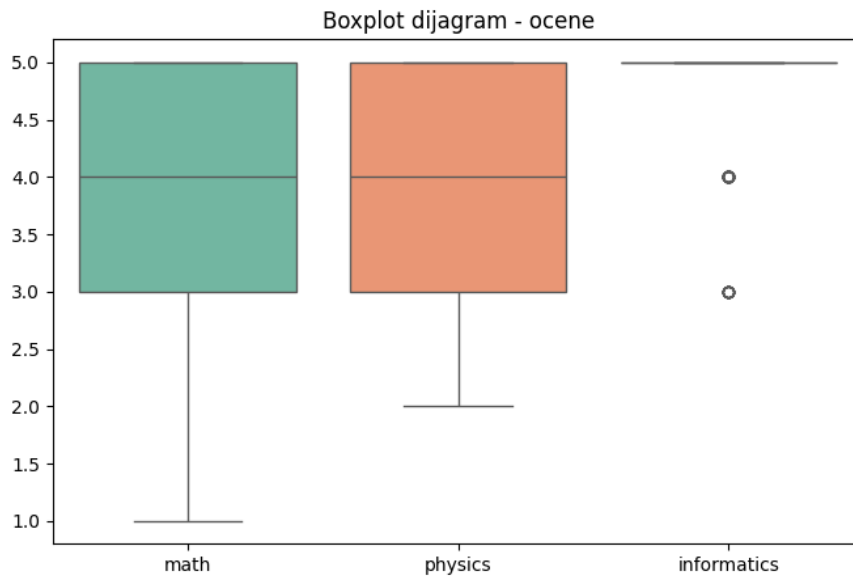
Оцене

Нумеричке колоне су оцене из математике, физике и информатике. Већина ученика има оцене 4 или 5, што потврђује и *boxplot* дијаграм.

```

1 plt.figure(figsize=(8,5))
2 sns.boxplot(data=df[["math","physics","informatics"]],
3             palette="Set2")
4 plt.title("Boxplot dijagram - ocene")
5 plt.show()

```

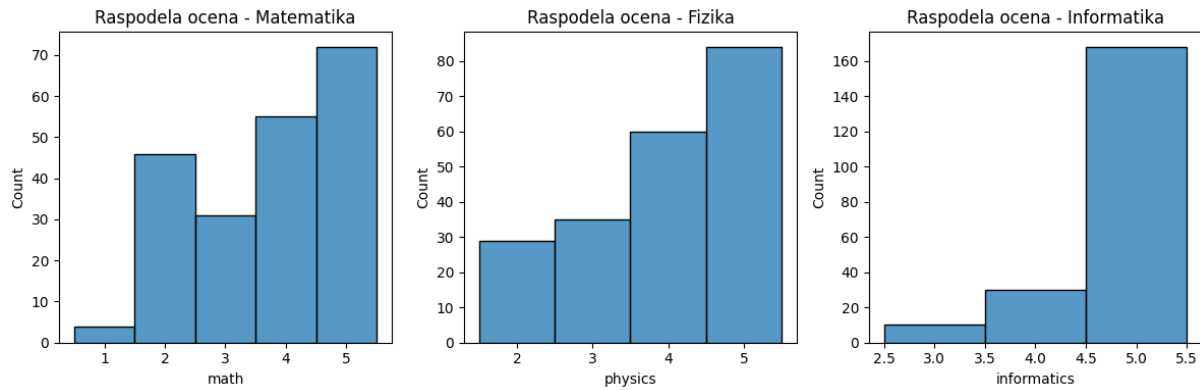
Оно што је додатно занимљиво је да већина ученика има оцену 5 из информатике, што показује да је то предмет у којем су најбољи. Највише се види разлика у знању из математике, где постоје и слаби и одлични ученици. Физика је стабилнија, али и даље са доста ученика у средњем рангу.

Додатно, можемо приказати и хистограме расподела оцена за сва 3 предмета.

```

1 # Histogram raspodele ocena
2
3 plt.figure(figsize=(12,4))
4
5 plt.subplot(1,3,1)
6 sns.histplot(df["math"], bins=5, discrete=True)
7 plt.title("Raspodela ocena - Matematika")
8
9 plt.subplot(1,3,2)
10 sns.histplot(df["physics"], bins=5, discrete=True)
11 plt.title("Raspodela ocena - Fizika")
12
13 plt.subplot(1,3,3)
14 sns.histplot(df["informatics"], bins=5, discrete=True)
15 plt.title("Raspodela ocena - Informatika")
16
17 plt.tight_layout()
18 plt.show()

```

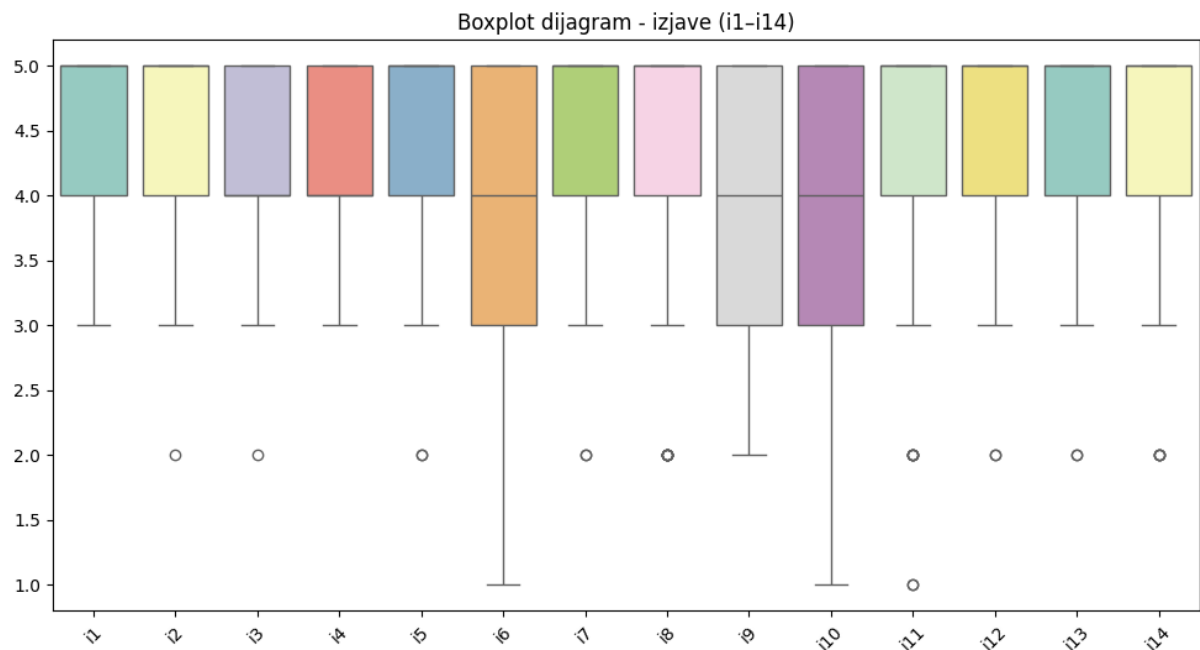


Изјаве

На основу графичког приказа за изјаве, можемо да закључимо да су ученици углавном позитивно оценили радионице.



```
1 plt.figure(figsize=(12,6))
2 sns.boxplot(data=df[[f"i{i}" for i in range(1,15)]], palette="Set3")
3 plt.title("Boxplot dijagram - izjave (i1-i14)")
4 plt.xticks(rotation=45)
5 plt.show()
```



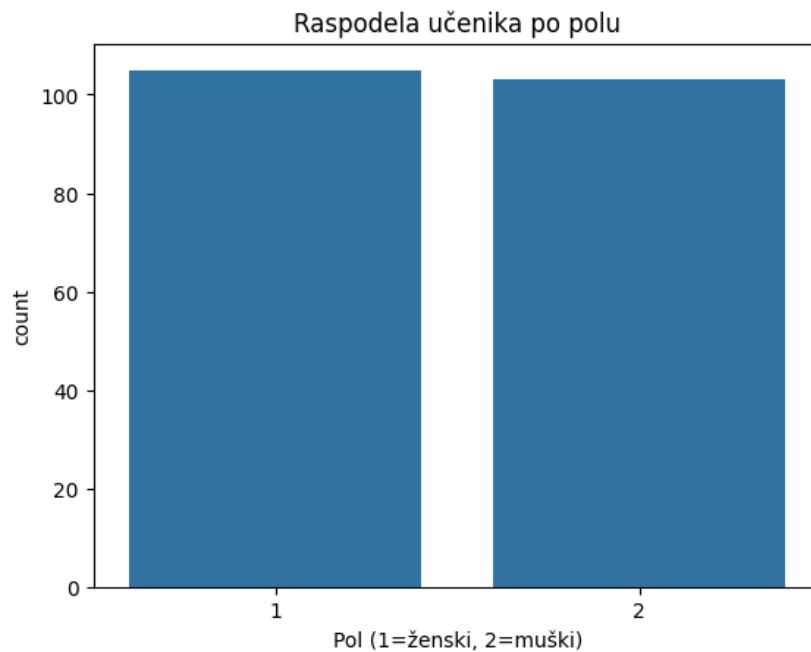
3.2. Категоријска обележја

Категоријска обележја код нас представљају пол – мушки и женски, радионица – (Не)могућа мисија и Потрага за благом, и искуство – има или нема искуство.

Пол

Што се тиче пола, имамо 105 ученица и 103 ученика.

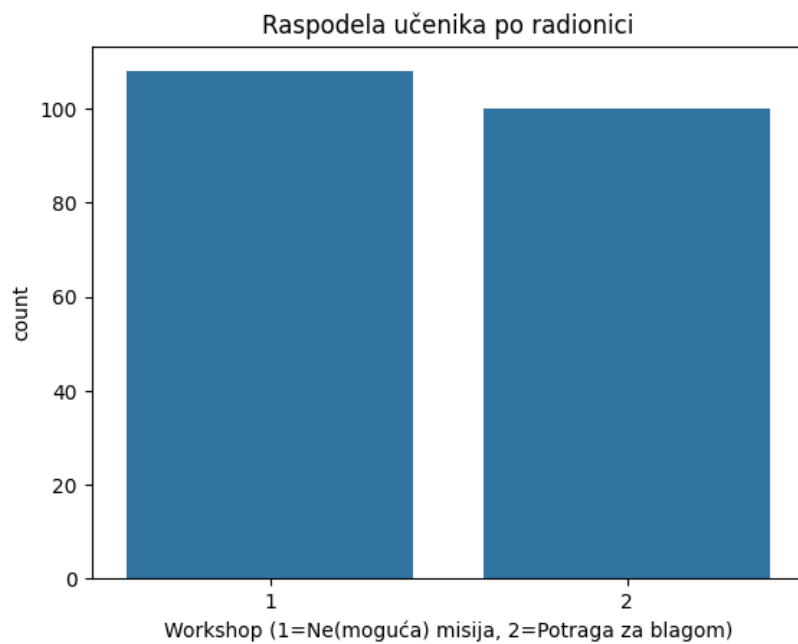
```
1 # Raspodela pola
2 sns.countplot(x="sex", data=df)
3 plt.title("Raspodela učenika po polu")
4 plt.xlabel("Pol (1=ženski, 2=muški)")
5 plt.show()
```



Радионица

Радионице имају 107 ученика на (Не)могућој мисији и 101 на Потрази за благом.

```
1 # Raspodela radionica
2 sns.countplot(x="workshop", data=df)
3 plt.title("Raspodela učenika po radionici")
4 plt.xlabel("Workshop (1=Ne(moguća) misija, 2=Potraga za blagom)")
5 plt.show()
```



Искуство

Овде можемо приметити да се јавља вредност које одступа од осталих, а то је вредност 5, па претпостављамо да је то ученик хтео да оцени као “има искуство”.



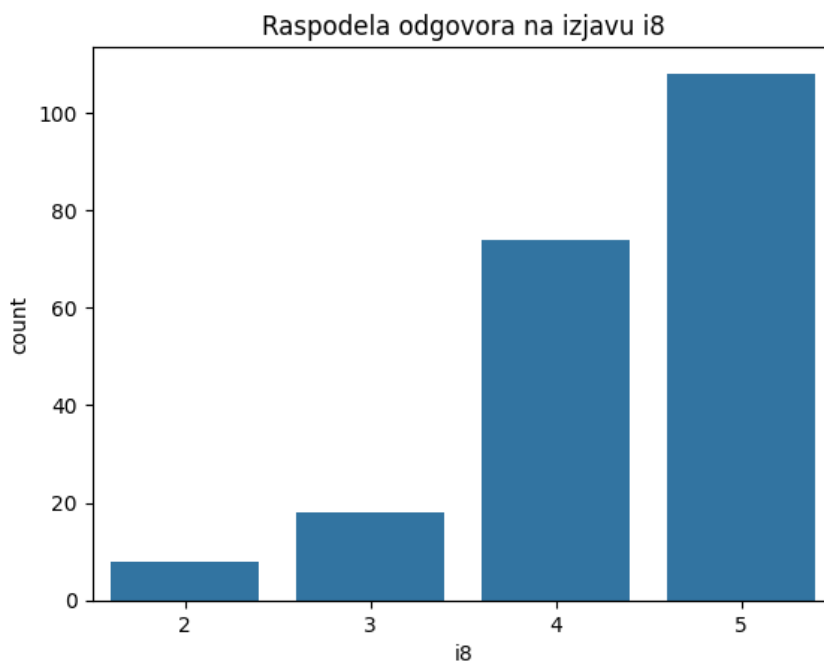
```
1 # Raspodela prethodnog iskustva
2 sns.countplot(x="experience", data=df)
3 plt.title("Prethodno iskustvo učenika sa sličnim radionicama")
4 plt.xlabel("Experience (0=nema, 1=ima)")
5 plt.show()
```



3.3 Расподела циљне променљиве

На основу расподеле циљне променљиве, можемо видети да она узима све вредности из опсега 1-5, осим јединице.

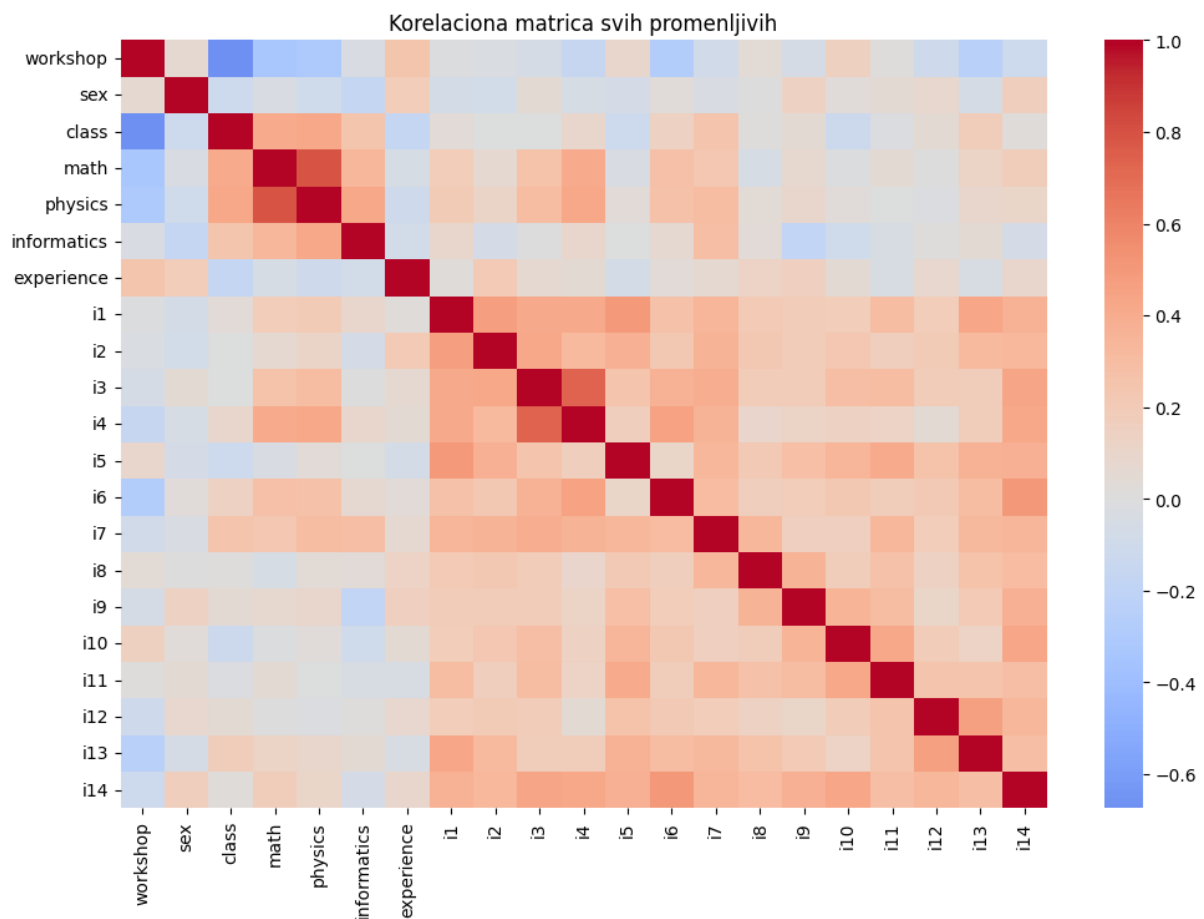
```
1 # Raspodela ciljne promenljive (i8)
2
3 sns.countplot(x='i8', data=df)
4 plt.title('Raspodela odgovora na izjavu i8')
5 plt.show()
```



3.4 Матрица корелације

Додатно, приказаћемо и иницијалну повезаност, тј. корелацију између свих обележја коришћењем корелационе матрице генерисане следећим кодом:

```
1 # Korelaciona matrica
2 plt.figure(figsize=(12,8))
3 sns.heatmap(df.corr(), annot=False, cmap="coolwarm", center=0)
4 plt.title("Korelaciona matrica svih promenljivih")
5 plt.show()
```



Слика 2 Матрица корелисаности обележја

Можемо видети и корелацију свих променљивих само са нашом циљном променљивом и8.



```
1 # Korelacija sa ciljnom promenljivom i8
2 corr_with_i8 = df.corr()["i8"].sort_values(ascending=False)
3 print("Korelacija svake promenljive sa i8:\n")
4 print(corr_with_i8)
```

Korelacija svake promenljive sa i8:

i8	1.000000
i9	0.364567
i7	0.339486
i14	0.306492
i11	0.270775
i13	0.262792
i2	0.223009
i5	0.220762
i1	0.203870
i3	0.190751
i10	0.175554
i6	0.167259
i12	0.133818
experience	0.127658
i4	0.097203
workshop	0.041399
physics	0.040976
informatics	0.031913
class	0.010250
sex	0.004300
math	-0.052921

Name: i8, dtype: float64

Највеће корелације са циљном променљивом, имају друге изјаве учесника: и9, и7, и14, и11. То значи да ученици који су осећали повезаност са групом и прилагођеност узрасту, као и важност сарадње, и активности за образовање, више верују да овакав начин учења даје боље резултате.

Такође, претходно искуство има малу али позитивну корелацију, што доводи да закључка да ученици који су раније имали сличне радионице мало чешће верују у изјаву и8.

5. Припрема података

Припрема података је кључни корак у процесу анализе, чији је циљ да осигура чистоћу података, релевантност и њихову спремност за даљу употребу. У овом делу, фокус ће бити на провери дупликата и елиминацију ирелевантних колона, што је од значаја за добијање поузданих резултата.

Током прегледања детаља о скупу података, извршена је провера дупликата и утврђено је да нема дупликата у овом скупу података. Ово представља важну информацију, јер дупликати могу да доведу до пристрасности и нетачних резултата у анализи. Пошто одабрани скуп података нема дупликате, можемо бити сигурни да сваки запис представља јединствен одговор ученика на анкету и да ће наша анализа бити тачна и поуздана.

5.1 Уклањање колоне *Id*

Колона *ID* садржи јединствене идентификаторе за сваког ученика, и иако је ова колона корисна за праћење и препознавање сваког појединачног записа (јединствени идентификатори се често користе у базама података како би јединствено одредили запис и зато су најчешће део сваког скупа података), она не пружа информације које су

релевантне за саму анализу одговора на анкету. Пошто нема аналитичку вредност и не доприноси моделирању или доношењу закључака, колона *Id* ће бити уклоњена из скупа података. На овај начин, поједностављујемо податке и задржавамо само она обележја која су од значаја за нашу анализу.

```
1 df.drop(columns='Id', inplace=True)
2 y = list(df['i8'])
3 features = list(df.columns)
4 features.remove('i8')
```

5.2 Енкодирање циљне променљиве

Обзиром да наша циљна променљива, као што смо малопре видели, садржи вредности 2,3,4 и 5, а не садржи оцену 1, ми ћемо извршити енкодирање на њу како би се те вредности енкодирале у вредности од 0 до 3, тј. 0, 1, 2 и 3.

```
1 from sklearn.preprocessing import LabelEncoder
2
3 label = LabelEncoder()
4 y_label = label.fit_transform(y)
5
6 # Fali nam 1 pa koristimo enkoder
7 print(y_label)
```

```
[3 3 3 3 1 2 3 1 2 2 3 3 3 2 2 1 2 2 1 2 3 2 0 3 2 0 2 3 2 3 3 3 3 1 2 2
 2 3 3 2 3 3 2 3 3 2 3 3 3 3 2 3 3 3 2 2 2 3 1 3 2 3 1 2 1 2 2 2 2 2 3 2
 3 3 3 3 3 3 3 1 3 3 2 3 3 2 3 3 1 2 3 3 3 3 2 3 2 2 2 2 3 0 3 0 3 3 2 3 0
 3 2 3 3 3 3 3 3 3 2 3 2 2 1 2 3 3 3 2 3 1 2 3 3 3 2 1 2 1 2 3 0 2 0 2 3 3
 3 3 1 2 2 2 3 3 2 3 2 3 3 3 2 2 2 3 1 3 2 3 2 2 2 2 3 3 3 3 1 3 2 3 3 1 2
 3 3 3 3 2 2 2 3 0 3 3 3 2 3 3 3 3 3 2 3 2 2 2]
```

5.3 Детектовање аномалија коришћењем Isolation Forest алгорита

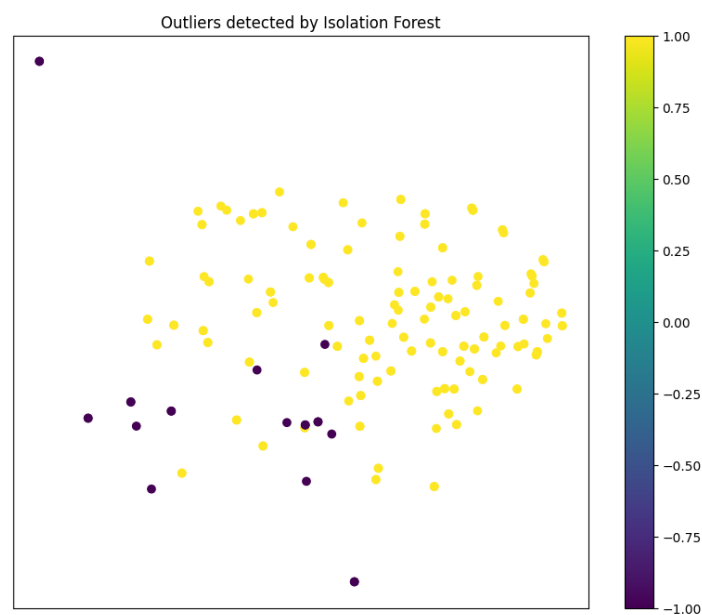
Нетипичне вредности (outliers) су подаци који значајно одступају од остатка података у скупу. Оне могу бити знатно веће или мање од типичних вредности, и често указују на специфичне или ретке догађаје и грешке приликом уноса података. Њихово препознавање и анализа су важни кораци у анализи података јер могу да утичу на статистичке резултате и моделе машинског учења. Због свог потенцијалног утицаја на тачност и стабилност модела, неопходно је посветити посебну пажњу идентификацији и третману аномалија.

Како бисмо додатно детектовали аномалије, одлучили смо се да пробамо и коришћење *Isolation Forest* алгорита, који изолира аномалије уместо да профилише нормалне податке и користи *Isolation Forest* стабла, која су слична стандардним стаблима одлучивања. У имплементацији *Isolation Forest* алгорита, користили смо

хиперпараметар контаминације са вредношћу 0.1, што значи да се очекује да око 10% података у нашем скупу буде означено као нетипично. Овај проценат је изабран на основу претпоставке удела аномалија у скупу података.

```
1 from sklearn.ensemble import IsolationForest
2 import numpy as np
3
4 out = IsolationForest(contamination = 0.1)
5 y_pred = out.fit_predict(X)
6 y_pred
```

На основу `y_pred`, правимо очишћене променљиве које садрже предикторе и циљну променљиву.



Слика 3 Графички приказ нетипичних вредности добијених помоћу Isolation Forest алгоритма

Љубичастом бојом на графику, приказане су тачке које су алгоритмом Isolation Forest означене као нетипичне у нашем скупу података.

```
1 X_clear = X[y_pred == 1]
2 y_clear = y_label[y_pred == 1]
3
4 print(X_clear)
5 print(y_clear)
```

[illegible]

```
1 print(X.shape)
2 print(X_clear.shape)
3 print("-----")
4 print(y_label.shape)
5 print(y_clear.shape)
```



```
1 duplikati = df.duplicated().sum()
2 print(f"Broj duplikata u tabeli: {duplikati}")
```

Broj duplikata u tabeli: 0

Оно што је резултат ове функције нам говори да скуп података нема дуплиране записе.

Наравно, поред броја дупликата врста, може нас занимати и колико јединствених вредности се налази у свакој колони. Ово је могуће одрадити користећи функцију **nunique**, која је у стању да изброји колико има јединствених вредности у зависности од осе која се посматра (у нашем случају ово значи да ли желимо да избројимо јединствене вредности по врстама или колонама).



```
1 print("\nBroj jedinstvenih vrednosti po kolonama:")
2 print(df.nunique())
```

```
Broj jedinstvenih vrednosti po kolonama:
Id                208
workshop           2
sex                2
class              5
math              5
physics           4
informatics        3
experience         2
i1                 3
i2                 4
i3                 4
i4                 3
i5                 4
i6                 5
i7                 4
i8                 4
i9                 4
i10                5
i11                5
i12                4
i13                4
i14                4
dtype: int64
```

5.5 Корекција колоне *experience*

```
1 df["experience"] = df["experience"].replace(5, 1)
2 print("Iskustvo nakon korekcije:", df["experience"].unique())
```

```
Iskustvo nakon korekcije: [0 1]
```

6. Креирање модела машинског учења

У овом поглављу биће речи о изабраним алгоритмима машинског учења који су коришћени за креирање модела који ће на најбољи начин радити закључивање.

6.1 Метрике за евалуацију модела

Обзиром да се ради о проблему класификације, метрике које можемо да користимо су следеће: *accuracy*, *precision*, *recall*, *f1*, *balanced_accuracy*, *roc_auc*. Одлучујем се за коришћење *accuracy* и *F1 macro* вредности.

6.1.1 *Accuracy*

Асигурање или тачност представља једну од основних метрика за оцену перформанси класификационих модела. Представља проценат исправно класификованих примера у односу на укупан број примера у скупу података. Формулу за рачунање тачности могуће је извести из матрице конфузије и она изгледа као:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

Тачност је метрика коју је могуће лако интерпретирати и могуће ју је брзо генерисати. Са друге стране, ова метрика има недостатак да у случају неуравнотеженог скупа података може дати лажни приказ перформанси модела. На пример, уколико је 95% примера из једне класе, модел који увек предвиђа ту класу ће имати тачност 95%, али неће бити ефикасан у препознавању друге класе. Такође, тачност не узима у обзир лажно позитивне и лажно негативне предикције које могу бити критичне у неким случајевима (нпр. При дијагностиковању у медицини). Због наведеног, тачност је најсигурније користити заједно са додатним метрикама.

6.1.2 *F1 macro*

F1 представља хармонијску средину између прецизности и одзива. Корисна је када постоји потреба за балансирањем између тачности позитивних предикција и способности модела да препозна све позитивне примере. Формула по којој се ова метрика рачуна је следећа:

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

F1 macro рачуна скор за сваку класу посебно и онда прави аритметичку средину. Све класе имају исти значај, без обзира на број примера.

```
1 scoring = {  
2     'accuracy': 'accuracy',  
3     'f1_macro': 'f1_macro'  
4 }
```

6.2 Крос-валидација

Пошто скуп података није велики (**187** примерака), модели су евалуирани применом **10-fold крос-валидације**. На овај начин добијена је поузданија процена перформанси, јер се сваки модел обучава и тестира више пута на различитим подскуповима података.

За крос валидацију, користимо функцију из библиотеке **scikit-learn**.

```
1 from sklearn.model_selection import cross_validate
```

```
1 # Funkcija za evaluaciju liste modela  
2 def evaluate_models(models, X, y, cv=10):  
3     results = {}  
4     for name, model in models:  
5         scores = cross_validate(model, X, y, cv=cv, scoring=scoring)  
6         results[name] = {  
7             "Accuracy": scores['test_accuracy'].mean(),  
8             "F1-macro": scores['test_f1_macro'].mean()  
9         }  
10    return pd.DataFrame(results).T.sort_values(by="Accuracy", ascending=False)
```

6.3 Модели

За израду модела коришћени су алгоритми: **Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, SVM, MLP** и **XGBoost**. Модели су одабрани тако да обухвате како класичне статистичке приступе (нпр. логистичка регресија), тако и савремене ансамбл методе које комбинују више стабала одлучивања (Random Forest, Gradient Boosting, XGBoost).

```
1 from sklearn.neural_network import MLPClassifier  
2 from sklearn.tree import DecisionTreeClassifier  
3 from sklearn.svm import SVC  
4 from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier  
5 from xgboost import XGBClassifier  
6 from sklearn.linear_model import LogisticRegression
```

```

1 classifiers = [
2     ("LogisticRegression", LogisticRegression(max_iter=1000)),
3     ("DecisionTree", DecisionTreeClassifier(random_state=42)),
4     ("RandomForest", RandomForestClassifier(random_state=42)),
5     ("GradientBoosting", GradientBoostingClassifier(random_state=42)),
6     ("SVM", SVC()),
7     ("MLP", MLPClassifier(max_iter=1000, random_state=42)),
8     ("XGBoost", XGBClassifier(use_label_encoder=False, eval_metric="mlogloss", random_state=42))
9 ]

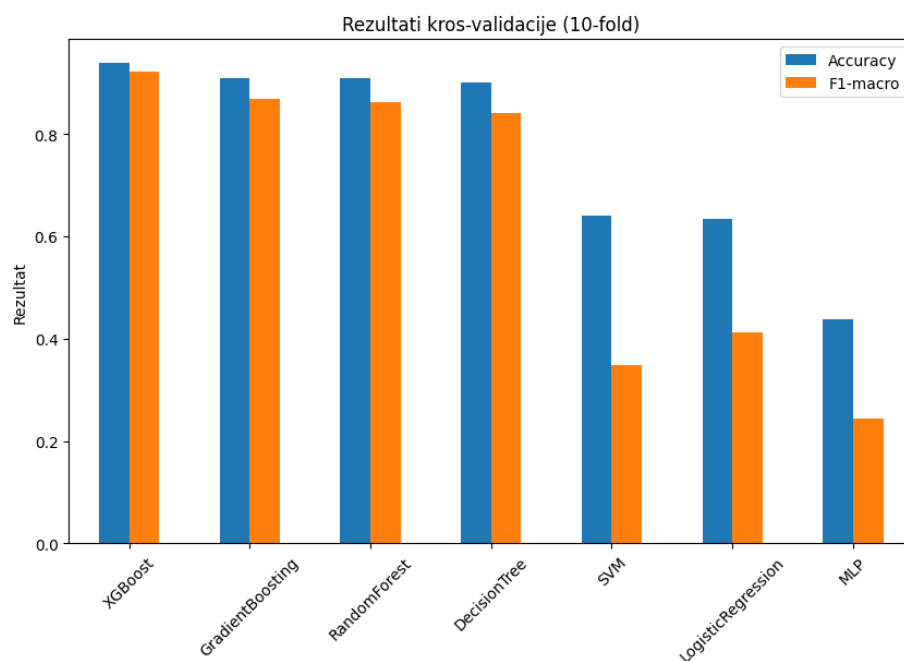
```

6.4 Резултати и анализа метрика добијених модела

Модели су најпре обучавани и тестирани на **непрочишћеном скупу података**. Најбоље резултате постигао је **XGBoost** са Accuracy ≈ 0.94 и F1-macro ≈ 0.92 . Одмах иза њега налазе се **Random Forest** и **Gradient Boosting**, са Accuracy око 0.91 и F1-macro између 0.86 и 0.87. **Decision Tree** је имао нешто слабије перформансе (Accuracy = 0.90, F1-macro ≈ 0.84), али и даље прихватљиве у односу на ансамбл методе.

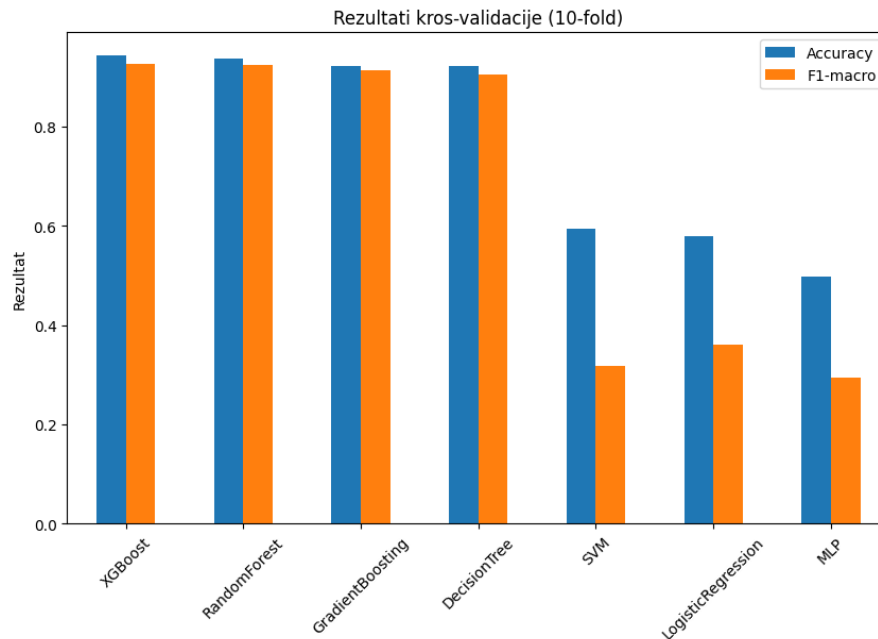
Са друге стране, једноставнији модели као што су **SVM** и **Logistic Regression** дали су значајно слабије резултате (F1-macro око 0.35–0.41), док је **MLP** постигао најслабије перформансе (F1-macro ≈ 0.24). Ово указује да за овај тип података једноставне линеарне и неуронске методе нису погодне, већ да ансамбл приступи боље моделирају зависности између обележја.

	Accuracy	F1-macro
XGBoost	0.938095	0.921952
GradientBoosting	0.909524	0.867867
RandomForest	0.909524	0.862718
DecisionTree	0.900000	0.841277
SVM	0.639524	0.349085
LogisticRegression	0.635000	0.411782
MLP	0.437381	0.243788



Затим је анализа поновљена на **прочишћеном скупу података**, добијеном применом Isolation Forest алгорита који је елиминисао око 10% аномалних примера. Ансамбл модели су задржали водећу позицију. **XGBoost** је поново био најуспешнији са Accuracy ≈ 0.94 и F1-macro ≈ 0.93 . Одмах иза њега налази се **Random Forest** (Accuracy ≈ 0.94 , F1-macro ≈ 0.92), што потврђује да оба модела доследно дају најбоље резултате на овом задатку.

	Accuracy	F1-macro
XGBoost	0.942105	0.925729
RandomForest	0.936842	0.923028
GradientBoosting	0.921053	0.912359
DecisionTree	0.921053	0.903897
SVM	0.594737	0.317187
LogisticRegression	0.578070	0.360510
MLP	0.496784	0.294536



Ово поређење показује да је **прочишћавање података позитивно утицало на тачност и стабилност ансамбл модела**, који су најбоље успели да искористе структуру података након уклањања нетипичних вредности.

Упоредна анализа

	Accuracy_original	F1-macro_original	Accuracy_cleaned	F1-macro_cleaned
XGBoost	0.938095	0.921952	0.942105	0.925729
GradientBoosting	0.909524	0.867867	0.921053	0.912359
RandomForest	0.909524	0.862718	0.936842	0.923028
DecisionTree	0.900000	0.841277	0.921053	0.903897
SVM	0.639524	0.349085	0.594737	0.317187
LogisticRegression	0.635000	0.411782	0.578070	0.360510
MLP	0.437381	0.243788	0.496784	0.294536

6.5 Фино подешавање

Након иницијалне евалуације модела, приступило се финој оптимизацији хиперпараметара најбољих модела – **Random Forest** и **XGBoost**. За ове експерименте примењене су методе **GridSearchCV** и **RandomizedSearchCV**.

6.5.1 GridSearchCV

GridSearchCV омогућава систематично испробавање свих комбинација задатих вредности параметара. Иако је временски захтевнија метода, на релативно малом скупу података показала се ефикасном.

GridSearchCV за RandomForest

```
1 # GridSearchCV за RandomForest
2
3 from sklearn.model_selection import GridSearchCV
4
5 rf = RandomForestClassifier(random_state=42)
6
7 param_grid_rf = {
8     "n_estimators": [50, 100, 200],
9     "max_depth": [None, 5, 10],
10    "min_samples_split": [2, 5, 10]
11 }
12
13 grid_rf = GridSearchCV(
14     estimator=rf,
15     param_grid=param_grid_rf,
16     cv=5,
17     scoring="f1_macro",
18     n_jobs=-1
19 )
20
21 grid_rf.fit(X_clear, y_clear)
22
23 print("RandomForest - Najbolji parametri (GridSearch):", grid_rf.best_params_)
24 print("RandomForest - Najbolji rezultat (GridSearch):", grid_rf.best_score_)
```

```
RandomForest - Najbolji parametri (GridSearch): {'max_depth': 10, 'min_samples_split': 2, 'n_estimators': 100}
RandomForest - Najbolji rezultat (GridSearch): 0.9538203969943101
```

GridSearchCV за XGBoost

```
1 # GridSearch za XGBoost
2
3 xgb = XGBClassifier(use_label_encoder=False, eval_metric="mlogloss",
4     random_state=42)
5
6 param_grid_xgb = {
7     "n_estimators": [50, 100, 200],
8     "max_depth": [3, 5, 7],
9     "learning_rate": [0.01, 0.1, 0.2]
10 }
11
12 grid_xgb = GridSearchCV(
13     estimator=xgb,
14     param_grid=param_grid_xgb,
15     cv=5,
16     scoring="f1_macro",
17     n_jobs=-1
18 )
19
20 grid_xgb.fit(X_clear, y_clear)
21
22 print("XGBoost - Najbolji parametri (GridSearch):", grid_xgb.best_params_)
23 print("XGBoost - Najbolji rezultat (GridSearch):", grid_xgb.best_score_)
```

```
XGBoost - Najbolji parametri (GridSearch): {'learning_rate': 0.2, 'max_depth': 5, 'n_estimators': 200}
XGBoost - Najbolji rezultat (GridSearch): 0.9431452203191334
```

6.5.2 RandomizedSearchCV

RandomizedSearchCV насумично испробава ограничен број комбинација из ширег простора параметара, међутим показао је нешто слабију поузданост.

RandomizedSearchCV za RandomForest

```

1 # RandomizedSearchCV za RandomForest
2
3 from sklearn.model_selection import RandomizedSearchCV
4
5 rf = RandomForestClassifier(random_state=42)
6
7 param_dist_rf = {
8     "n_estimators": [50, 100, 200, 300, 400],
9     "max_depth": [None, 5, 10, 15, 20],
10    "min_samples_split": [2, 5, 10, 20],
11    "min_samples_leaf": [1, 2, 4, 6]
12 }
13
14 random_rf = RandomizedSearchCV(
15     estimator=rf,
16     param_distributions=param_dist_rf,
17     n_iter=20,
18     cv=5,
19     scoring="f1_macro",
20     random_state=42,
21     n_jobs=-1
22 )
23
24 random_rf.fit(X_clear, y_clear)
25
26 print("RandomForest - Najbolji parametri (RandomizedSearch):", random_rf.best_params_)
27 print("RandomForest - Najbolji rezultat (RandomizedSearch):", random_rf.best_score_)

```

```

RandomForest - Najbolji parametri (RandomizedSearch): {'n_estimators': 400, 'min_samples_split': 5, 'min_samples_leaf': 1, 'max_depth': 20}
RandomForest - Najbolji rezultat (RandomizedSearch): 0.7300408843941426

```

RandomizedSearchCV za XGBoost

```

1 # RandomizedSearchCv za XGBoost
2
3 xgb = XGBClassifier(use_label_encoder=False, eval_metric="mlogloss", random_state=42)
4
5 param_dist_xgb = {
6     "n_estimators": [50, 100, 200, 300, 400],
7     "max_depth": [3, 5, 7, 10],
8     "learning_rate": [0.01, 0.05, 0.1, 0.2],
9     "subsample": [0.6, 0.8, 1.0],
10    "colsample_bytree": [0.6, 0.8, 1.0]
11 }
12
13 random_xgb = RandomizedSearchCV(
14     estimator=xgb,
15     param_distributions=param_dist_xgb,
16     n_iter=20,
17     cv=5,
18     scoring="f1_macro",
19     random_state=42,
20     n_jobs=-1
21 )
22
23 random_xgb.fit(X_clear, y_clear)
24
25 print("XGBoost - Najbolji parametri (RandomizedSearch):", random_xgb.best_params_)
26 print("XGBoost - Najbolji rezultat (RandomizedSearch):", random_xgb.best_score_)

```

```

XGBoost - Najbolji parametri (RandomizedSearch): {'subsample': 0.6, 'n_estimators': 100, 'max_depth': 5, 'learning_rate': 0.2, 'colsample_bytree': 0.6}
XGBoost - Najbolji rezultat (RandomizedSearch): 0.947114319736271

```

7. Одабир најзначајнијих предиктора

У циљу додатне анализе, испитано је да ли искључивање појединих обележја може довести до побољшања резултата. Уместо коришћења свих обележја (категоријских, оцена и изјава), задржане су само изјаве i1–i14, при чему је циљна променљива *i8* искључена из скупа предиктора. На овај начин тестирано је да ли се став ученика у вези са ефикасношћу радионица (*i8*) може поуздано предвидети само на основу других ставова.

```
1 # Features = samo izjave i1-i14 (bez i8)
2 features_i = [f"i{i}" for i in range(1,15) if i != 8]
3 X_i = df[features_i]
4 y_i = df["i8"]
5
6 label = LabelEncoder()
7 y_i_label = label.fit_transform(y_i)
8
9 print("Dimenzije:", X_i.shape, y_i_label.shape)
```

```
1 # Evaluacija samo sa i1-i14 features
2 results_i = evaluate_models(classifiers, X_i, y_i_label)
3
4 print("Rezultati modela (samo i1-i14):")
5 print(results_i.sort_values(by="Accuracy", ascending=False))
6
7 # Vizuelizacija
8 results_i.plot(kind="bar", figsize=(10,6))
9 plt.title("Rezultati (samo i1-i14 kao features)")
10 plt.ylabel("Rezultat")
11 plt.xticks(rotation=45)
12 plt.show()
```

Резултати су показали да су и у овом случају најбоље перформансе дали *Random Forest* и *XGBoost*. Ассурасу и *F1-macro* вредности остале су врло високе (око 0.93–0.95), док су једноставнији модели попут Logistic Regression, SVM и MLP поново имали знатно слабије резултате. Ово потврђује да категоријска обележја (пол, радионица, искуство) и школске оцене немају значајан утицај на перцепцију ученика.

Над овим редукованим скупом поновљен је поступак финог подешавања хиперпараметара за два најбоља модела – *Random Forest* и *XGBoost*, и то како на **непрочишћеним**, тако и на **прочишћеним подацима** (коришћењем *Isolation Forest* алгоритама). Резултати су показали да Random Forest и даље најбоље одговара *GridSearch* оптимизација, при чему је *F1-macro* остао око 0.95. *XGBoost* је и у овом случају постигао сличне резултате (*F1-macro* око 0.91), а најбољи параметри пронађени су такође *GridSearch* методом.

Овај експеримент потврдио је конзистентност добијених резултата: без обзира на то да ли се користе сви предиктори или само изјаве i1–i14, најбољи модели су Random Forest и XGBoost, при чему је Random Forest уз *GridSearchCV* подешавање хиперпараметара постигао највише перформансе.

8. Финални закључак

Резултати анализе показали су да је могуће са високом тачношћу предвидети став ученика у вези са изјавом „Верујем да на овај начин могу више да научим у односу на традиционалне методе наставе“ (i8) на основу података прикупљених током STEM радионица.

Ансамбл модели, нарочито *Random Forest* и *XGBoost*, постигли су најбоље резултате, са F1-масо метриком око 0.95. *Random Forest* се показао као најстабилнији модел, нарочито након *GridSearchCV* оптимизације хиперпараметара. *XGBoost* је такође дао врло добре резултате. Једноставнији модели као што су *Logistic Regression*, *SVM* и *MLP* показали су значајно слабије резултате.

Додатна анализа са редукцијом предиктора потврдила је да се наша циљна променљива може успешно предвидети искључиво на основу осталих изјава ученика. Ово јасно указује да је перцепција ефикасности радионица највише условљена личним искуством и ставовима учесника, а не њиховим полом, школским оценама или претходним искуством.

Закључује се да су STEM радионице имале позитиван ефекат на ученике, а примена ансамбл метода машинског учења омогућила је прецизну предикцију њихових ставова. Ови налази могу бити корисни у даљем планирању и унапређењу наставних метода које подстичу активно учење и интердисциплинарни приступ.

Визуелизација стабла најбољег модела *RandomForest*-а, фино подешеног коришћењем *GridSearchCV*-а.

