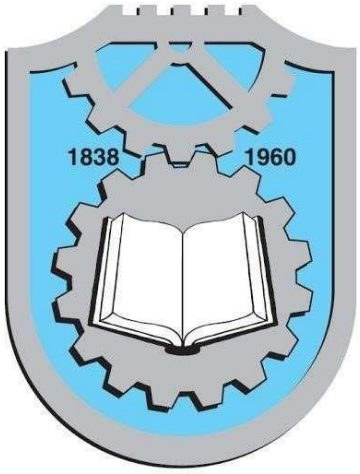
Универзитет у Крагујевцу
Факултет инжењерских наука



Машинско учење

Пројектни задатак:

Предикције фудбалских утакмица

Студенти:
Кристина Спасојевић 639/2019
Павле Зорић 641/2019

Предметни професор:
Владимир Миловановић

1. Опис пројектног задатка

Енглеска Премијер лига (EPL) сматра се за једно од најпопуларнијих фудбалских такмичења на свету.

У ЕПЛ-у за трофеј се такмичи 20 тимова, од којих три најгоре пласирана сваке сезоне испадају из такмичења и бивају замењена најбољим тимовима из лиге нижег ранга. Сваки тим игра против свих осталих два пута, једном на домаћем терену и једном као гост.

Предмет овог рада је примена метода машинског учења (енг. Machine learning) у предвиђању крајњег исхода утакмица ЕПЛ-а. Могући исходи подразумевају победу домаћег тима, победу гостујућег тима, као и нерешен резултат. Како је наведени скуп дискретан, у питању је проблем који подразумева додељивање класе (категорије) одређеном скупу података. Овакви проблеми се у теорији машинског учења називају проблеми класификације. Велики број сусрета завршених нерешеним исходом представља један од кључних изазова у овом раду, будући да они драстично повећавају неодређеност модела.

У првом делу презентовања говорићемо о популарним методама класификације који се користе у науци и индустрији и који ћемо ми користити у нашем пројекту.

У другом делу представљени су делови кода као што су уводни параметри као и решења задатих синтакси који нам омогућавају да лакше одредимо исход мечева ЕПЛ-а.

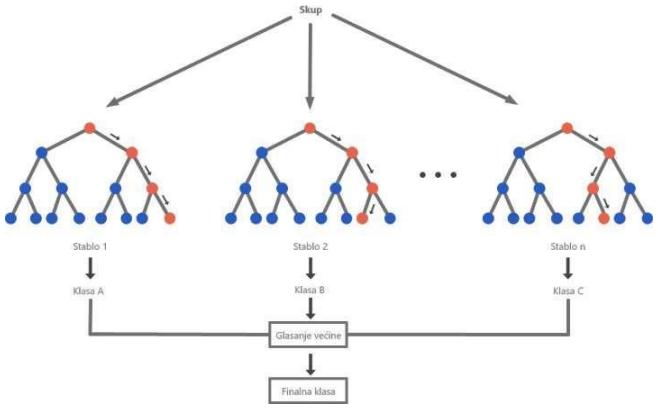
2. Модел стабла одлуке и случајне шуме

Техника насумичне шуме појавила се у последњих неколико година као моћан начин за анализу великих скупова података уз избегавање неких замки других метода руковања подацима. Заснива се на идеји да се неки будући догађај може одредити у стаблу одлучивања у којем се исход израчунава у свакој грани, позивајући се на скуп података.

Стабло одлуке се састоји од чворова, грана и листова. Сваки чвор представља једну особину решеног проблема. Гране које иду од чворова представљају сваку могућу вредност коју чвор може да има у зависности од вредности тог атрибута за случај који се испитује. На крају су ту и листови који представљају класе у које можемо сврстати одређене улазне податке.

Међутим, стабло одлуке пати од познатог проблема. Наиме у другој фази процеса разграњавања, одлуке могу бити озбиљно искривљене помоћу података о обуци који су ретки и склони великим варијацијама. Проблем се назива префитовање (overfitting).

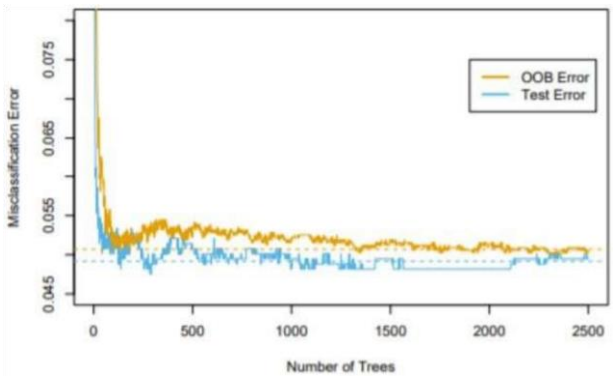
Приступ насумичних шума је другачији. Уместо израчунавања исхода у свакој грани, процес израчунава исход случајних грана. Ово ради много пута, сваки пут са различитим скупом случајно одабраних грана. Коначни резултат представља просек свих ових насумично конструисаних стабала одлучивања. Алгоритам случајних шума је модификација bagging методе који гради велики број неколинеарних стабала. Циљ алгоритма случајних шума је редуковати варијансу тако што ће смањити корелацију између стабала. Тај циљ се постиже кроз изградњу стабала помоћу насумично одабраних узорака.



Случајна шума - класификација

Out Of Bag

Важна ставка у алгоритму случајних шума је "Out Of Bag" скуп података који представља отприлике 33% података који се занемарују при узорковању. Процена грешке је готово једнака као унакрсна валидација стабла, па је задовољавајућа и за процену грешке случајне шуме.



Приказ грешке OOB скупа и грешке скупа за тестирање

3. Увод

Основни циљ ове теме је да направимо систем за предвиђање фудбалских утакмица.

За ову тему нам је неопходан скуп података који ћемо да истренирамо коришћењем RandomForest(коришћење случајних шума). Учитаћемо податке из фајла E0.csv и исте те податке ћемо да истренирамо. Подаци су преузети са следећег сајта https://www.football-data.co.uk/englandm.php.

4. Дефинисање параметара

Почетни параметри:

1. утакмице
2. тимови1, тимови2

Додати параметри:

1. број одиграних утакмица
2. број победа домаћег тима
3. број победа гостујућег тима
4. број нерешених исхода

Додато је и још пар колона (параметара), како бисмо могли што боље и ефикасније да истренирамо модел.

```
import pandas as pd

utakmice = pd.read_csv("E0.csv", index_col=0)
utakmice.head()

timovi1 = utakmice["HomeTeam"].value_counts()
timovi2 = utakmice["AwayTeam"].value_counts()
print(timovi1)
print(timovi2)

utakmice.dtypes

utakmice["target"] = (utakmice["FTR"] == "D").astype(int)
utakmice["GD"] = abs((utakmice["FTHG"] - utakmice["FTAG"]).astype(int))
utakmice["HBP"] = 10 * utakmice["HY"] + 25 * utakmice["HR"]
utakmice["ABP"] = 10 * utakmice["AY"] + 25 * utakmice["AR"]
utakmice["FTG"] = (utakmice["FTHG"] + utakmice["FTAG"]).astype(int)
utakmice["HTG"] = (utakmice["HTHG"] + utakmice["HTAG"]).astype(int)

#neresene ishode - 1, pobede domacina ili gosta - 0
br_odigranih_utakmica = utakmice.shape[0]
br_pobeda_domacin = len(utakmice[utakmice.FTR == 'H'])
br_pobeda_gost = len(utakmice[utakmice.FTR == 'A'])
br_neresenih = len(utakmice[utakmice.FTR == 'D'])

print("Ukupno odigranih utakmica: ", br_odigranih_utakmica)
print("Broj pobeda domacih timova: ", br_pobeda_domacin)
print("Broj pobeda gostujucih timova: ", br_pobeda_gost)
print("Broj neresenih ishoda: ", br_neresenih)

utakmice["Hour"] = utakmice["Time"].str.replace(":.+", "", regex=True).astype("int")
utakmice
```

5. Тренирање података

```
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=50, min_samples_split=2, random_state=1)
train = utakmice[utakmice["Time"] < '7']
test = utakmice[utakmice["Time"] > '7']
parametri_za_predvidjanje = ["FTHG", "FTAG", "HTHG", "HTAG", "HS", "AS", "AC", "HY", "AY", "HR", "AR", "HBP", "ABP"]
rf.fit(train[parametri_za_predvidjanje], train["target"])
predvidjanje = rf.predict(test[parametri_za_predvidjanje])
```

Пре него што почнемо да тренирамо податке, морамо да дефинишемо параметре класификатора (број гранања стабала, минималан број узорака потребних за гранање, параметар за контролу случајних шума). Истренираћемо и тестирање податке у зависности од параметра Time. Узели смо одређен број параметара за предвиђање (пожељно је имати што више параметара, како би побољшали прецизност тренирања модела) и скалираћемо коначни исход утакмице (target колона) у зависности од параметара за предвиђање.

6. Резултат тренирања

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(test["target"], predvidjanje)
print("Random Forest Classifier: ", accuracy)
```

✓ 0.4s

```
Random Forest Classifier:  0.6153846153846154
```