Izbirno tekmovanje iz umetne inteligence
Praktični del

24. januar 2026

Ali so gobe užitne?

Opis. Dana je podatkovna množica *Mushroom*[1], v kateri je vsak primer opisan z 22 kategoričnimi značilkami, ciljna spremenljivka pa pove, ali je goba *užitna* (`e`) ali *strupena* (`p`). Naloga je zgraditi klasifikator, ki dobro napoveduje razred, hkrati pa uporablja čim manj izvornih značilk.

Podatki

- Datoteka: `dataset_24_mushroom.arff`.
- Število primerov: $n = 8124$.
- Število izvornih značilk: $d = 22$ (vse kategorične).
- Ciljna spremenljivka: stolpec `class` z vrednostma `e` in `p`.
- V stolpcu `stalk-root` se pojavlja vrednost `?`; obravnavajte jo kot posebno kategorijo.
- V izvorni ARFF datoteki je atribut `bruises?` lahko zapisan kot `bruises%3F`; ocenjevalnik to ime normalizira v `bruises?`.

Namig. Datoteko s podatki (tipa `arff`) odpreš takole:

```
import pandas as pd
from scipy.io import arff

data, meta = arff.loadarff("dataset_24_mushroom.arff")
df = pd.DataFrame(data)
```

Naloga

Zgradite postopek učenja, ki za podane značilke napove $\hat{y} \in \{\texttt{e}, \texttt{p}\}$.

Dovoljene metode. Uporabite lahko poljuben model in poljubno predobdelavo (npr. kodiranje kategorij, regularizacija, izbira modela), vendar mora biti rešitev deterministična (nastavite seme, če uporabljate naključnost) in mora uporabljati izključno podatke, ki jih prejme prek vmesnika `fit/predict`.

[1]The Audubon Society Field Guide to North American Mushrooms (1981). G. H. Lincoff (Pres.), New York: Alfred A. Knopf. OpenML id = 24 (`https://www.openml.org/search?type=data&status=active&id=24&sort=runs`)

Evalvacija: 5-kratno stratificirano prečno preverjanje

Ocenjevanje poteka s stratificiranim 5-kratnim prečnim preverjanjem (5-fold stratified CV). Primarni protokol je enakovreden klicu:

`StratifiedKFold(n_splits=5, shuffle=True, random_state=42)`.

Naj bo za $k = 1, \ldots, 5$ učna množica $(X_{\text{train}}^{(k)}, y_{\text{train}}^{(k)})$ in validacijska množica $(X_{\text{val}}^{(k)}, y_{\text{val}}^{(k)})$. Na vsakem foldu naučite model na učnih podatkih in napovejte oznake na validacijskih podatkih.

Macro-F_1 mera. Za fold k izračunamo F_1 posebej za oba razreda in ju povprečimo:

$$F_{1,\text{macro}}^{(k)} = \frac{F_1^{(k)}(\texttt{e}) + F_1^{(k)}(\texttt{p})}{2}.$$

Za posamezen razred $c \in \{\texttt{e}, \texttt{p}\}$ obravnavamo razred c kot "pozitiven", drugega pa kot "negativen". Tedaj velja

$$\text{Precision}_c^{(k)} = \frac{TP_c^{(k)}}{TP_c^{(k)} + FP_c^{(k)}}, \qquad \text{Recall}_c^{(k)} = \frac{TP_c^{(k)}}{TP_c^{(k)} + FN_c^{(k)}},$$

$$F_1^{(k)}(c) = \frac{2\,\text{Precision}_c^{(k)}\text{Recall}_c^{(k)}}{\text{Precision}_c^{(k)} + \text{Recall}_c^{(k)}}.$$

Če je imenovalec v zgornjih ulomkih 0, definiramo ustrezno vrednost kot 0 (enako kot standardne implementacije). Končni rezultat kakovosti je povprečje po foldih:

$$\overline{F_{1,\text{macro}}} = \frac{1}{5}\sum_{k=1}^{5} F_{1,\text{macro}}^{(k)}.$$

Kaznovanje kompleksnosti (število značilk)

Naj bo $\mathcal{F}$ množica *uporabljenih izvornih značilk* in $m = |\mathcal{F}|$. Končna ocena je

$$S = \overline{F_{1,\text{macro}}} - \lambda m, \qquad \text{kjer je} \qquad \lambda = 0.001.$$

Rezultat S se ne obrezuje (lahko je tudi negativen).

Definicija uporabljenih značilk.

- Uporabljene štejemo izvorne značilke iz podatkov (pred one-hot kodiranjem ipd.).
- Če iz ene izvorne značilke ustvarite več stolpcev (npr. one-hot), k m prispeva natanko 1.
- m se računa na dedupliciranem seznamu: $\mathcal{F} = \text{set}(\texttt{USED_FEATURES})$.
- Veljati mora $1 \le m \le 22$. Neveljavne oddaje (npr. neznane značilke) se ocenijo z rezultatom $S = 0$.

Oddaja (za avtomatsko ocenjevanje)

Oddate program (npr. `submission.py`), ki implementira naslednji minimalni vmesnik (glej Dodatek B):

- `USED_FEATURES`: seznam imen izvornih značilk (glej Dodatek C),
- razred `Model` z metodama `fit(X, y)` in `predict(X)`.

Zahteve za vmesnik.

- Ocenjevalnik bo v `X` obdržal samo stolpce `USED_FEATURES` in jih podal v `fit/predict`. Rešitev naj ne predpostavlja, da so prisotni drugi stolpci.
- `predict` mora vrniti natanko $|X|$ napovedi; vsaka napoved mora biti niz `e` ali `p`.
- Ocenjevanje lahko izvaja posamezne folde v izolaciji (ponovni uvoz modula ali ločen proces); rešitev se ne sme zanašati na globalno stanje med foldi.

Cilj. Maksimizirajte S: visoka napovedna kakovost in čim manj uporabljenih značilk.

A Protokol ocenjevanja

A.1 Protokol in obravnava robnih primerov

Ocenjevanje izvedemo deterministično in tako, da preprečimo prenos informacij med foldi:

1. Naložimo `dataset_24_mushroom.arff` in normaliziramo imena stolpcev (npr. `bruises%3F` $\rightarrow$ `bruises?`). Vrednost `?` v `stalk-root` obravnavamo kot kategorijo.
2. Preverimo strukturo oddaje: obstoj `USED_FEATURES` in razreda `Model`, veljavnost imen značilk ter da `USED_FEATURES` vsebuje 1–22 različnih izvornih značilk. V tem koraku določimo $m = |\text{set}(\texttt{USED_FEATURES})|$. Strukturno neveljavna oddaja dobi končni rezultat $S = 0$.
3. Pripravimo stratificiran 5-fold razrez z `shuffle=True` in `random_state=42`.
4. Za vsak fold k izvedemo ocenjevanje v svežem okolju (ponovni uvoz modula ali ločen proces):
   (a) ponovno uvozimo oddajo in preverimo, da je množica `USED_FEATURES` enaka kot v koraku 2; sicer je oddaja neveljavna in dobi $S = 0$;
   (b) ustvarimo novo instanco `Model`, pokličemo `fit` na učnem delu in `predict` na validacijskem delu;
   (c) če pride do napake, prekoračitve časa ali neveljavnih izhodov (npr. napačno število napovedi ali oznake izven {`e`,`p`}), fold dobi $F^{(k)}_{1,\text{macro}} = 0$;
   (d) sicer izračunamo $F^{(k)}_{1,\text{macro}}$.
5. Izračunamo $\overline{F_{1,\text{macro}}}$ in nato $S = \overline{F_{1,\text{macro}}} - 0.001m$.

B Minimalni primer oddaje (`submission.py`)

Listing 1: Minimalni primer oddaje

```
USED_FEATURES = []

class Model:
    def __init__(self):
        pass

    def fit(self, X, y):
        return self

    def predict(self, X):
        return ['e'] * len(X)
```

C Seznam izvornih značilk

Kanonična imena stolpcev za uporabo v `USED_FEATURES`:

Cilj	`class`
Značilke	`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`, `population`, `habitat`.