Domača naloga - 2.del

Meta učenje

Vito Rozman

2. april 2023

1 Izbira podatkovij

Pri iskanju najboljšega modela sem najprej pridobil vsa podatkovja iz OpenML, za katera obstajajo naloge (ang. tasks) za nadzarovano kasifikacijo. Izbral sem podatkovja s število primerov med $500 - 3000$ in s $20 - 150$ značilk, ker drugače obdržim premalo podatkovij. Potem sem odstarnil tista, ki niso primerna za iskanje podobnosti. Torej da gre za kasifikacijo, vsebujejo samo eno napovedno spremenljivko, značilke so numeričnega tipa in podatkovja se ne ponavljajo. Od vseh podatkovij, ki sem jih pridobil sem obdržal 50 podatkovij.

2 Meta prostor

Izbrana podatkovja sem opisal z meta značilkami štirih tipov. To so *splošnega*, *stetističnega*, *info-teoreticnega* in *modelskega* (ang. general, statistical, info-theory and model-based). Pri tem sem naletel na težavo z nedefiniranimi vrednostmi *nan*, kar sem rešil z metodo imputacij. Nedefinirane vrednosti sem napolnil s povprečno vrednostjo.

2.1 Izbira podobnih podatkovij in njihovih modelov

Za izbiro podobnih podatkovij sem izbral pristop z iskanje k najbližjih sosedov za $k = 3$. Tako sem dobil tri podatkovja PieChart2, cardiotocography in wdbc. Za vsakega od njih sem pridobil njegove ocene (ang. evaluations) in posledično model z najboljšim izidom metrike AUC. Ker je za vsako podatkovje več taskov sem tako dobil več kot tri modele. Ker so nekatere modeli implementacije iz drugih programskih jezikov, sem za nekatere uporablil *sklearn* verzijo ali pa pa jih izločil. Njaboljši modeli so **Gaussian Naive Bayes**, **Decision Tree**, **Adaptive Boosting**(*base-estimator=DecisionTreeClassifier*) in **XGBoost**.

3 Zmogljivost modelov

Najprej sem razdelil podatke na učne in testne, na učnih sem model učil na testnih pa preveril njegovo zmogljivost z metriko AUC (ploščina pod ROC krivuljo). Opisan potopek sem najprej izvedel na neskaliranih podatkih, potem pa še na skaliranih ter primerjal rezultate. Iskazalo se je, da so skalirani podatki bolje obnesli, vendar pri izbiri najboljšega modela niso privedli do večjih razlik.

Modeli	AUC - cross validation	AUC test set
Gaussian Naive Bayes	0.848005	0.740600
Decision Tree	0.753321	0.739688
Adaptive Boosting	0.798328	0.715982
XGBoost	0.926251	0.813512

Hiperparametre modelov ki sem jih dobil nisem optimiziral, morda bi bilo smiselno izvesti še ta kotak za odločitev najboljšega modela iz meta učenja. Kot končni model sem izbral model z najboljšim izidom AUC na testni podatakih, to je bil **XGBoost** z AUC na testni množici 0.813512.

4 Zaključek

	AUC - cross validation	AUC test set
Random Forest - ročno	0.928039	0.887989
Random Forest - avtomatizirano	0.928101	0.847335
XGBoost - meta učenje	0.926251	0.813512

Pristop z meta učenjem je bil dokaj učinkovit, saj je zmogljivost izbranega modela primerljiva ročnemu iskanju in avtomatiziranem iskanju. Ob ponovnem iskanju najboljšega modela, bi najprej uporabil meta učenje tako bi dobil okvirno modele, ki so primerni za moje podatke, nato bi za nekaj izbranih kandidatov izvedel avtomatizirano izbiro hiperparametrov. Po dani konfiguraciji bi še ročno preveril okolico nastavitve parametrov.