Code review grupy KTR

Tomasz Siudalski

May 2022

1 Wstęp

Moim zadaniem było ocenienie kodu funkcji napisanej przez grupę 5 ('KTR'), której celem było przygotowanie danych do użycia przez pakiet FLAML.

2 Recenzja kodu

2.1 Czy kod osiąga cel, który postawiono?

Stworzona przez grupę 5 funkcja wykonuje następujący preprocessing:

1. usunięcie zmiennych które prawdopodobnie są id
2. usunięcie zmiennych o zerowej wariancji
3. uzupełnienie braków danych
4. zastąpienie outlierów wartościami skrajnymi
5. przeskalowanie do rozkładu normalnego
6. zamienienie kolumn kategorycznych ich WOE

Jest on moim zdaniem wystarczający, zatem funkcja osiąga swój cel, czyli przygotowanie danych pod pakiet FLAML.

2.2 Czy w kodzie są jakieś oczywiste błędy logiczne?

Klasa idRemover usuwa kolumny które mają same unikalne wartości w celu pozbycia się kolumn które potencjalnie są id. Jednak w przypadku zmiennych numerycznych a szczególnie zmiennoprzecinkowych istnieje duża szansa, że wszystkie wartości będą unikalne i poprawne kolumny zostaną usunięte. Moim zdaniem nie warto w ten sposób tracić cennych informacji.

2.3 Czy patrząc na wymagania zawarte podczas prezentacji są one w pełni zaimplementowane?

Na prezentacji nie znalazłem specyficznych wymagań poza koniecznością wykonania jakiegoś preprocessingu gdyż pakiet FLAML tego nie oferuje i wymaganie to jak pokazałem wcześniej zostało spełnione.

2.4 Czy kod jest zgodny z istniejącymi wytycznymi stylistycznymi (PEP 8)?

W większości kod jest zgodny ze standardem PEP 8, różne obiekty są oznaczone poprawnie, komentarze są dobrze napisane, jedynie w niektórych miejscach istnieją drobne niezgodności z tym standardem takie jak m.in: zbyt długie linijki (ponad 79 znaków), za mały odstęp między linijkami kodu, brak spacji w odpowiednich miejscach. Ponadto według standardu PEP 8 w linijce nr 33 i 47 pliku *ignorer.py*:

$$assert(self.\_broken == False)$$

przyrównanie do False powinno się wykonać przy użyciu is a nie ==. Wymienione rzeczy nie wpływają jednak moim zdaniem zbyt negatywnie na czytelność kodu.

2.5 Czy są jakieś obszary, w których kod mógłby zostać poprawiony (skrócić, przyspieszyć, itp.)?

Nie udało mi się znaleźć takich miejsc, uważam że kod jest napisany poprawnie.

2.6 Czy dokumentacja i komentarze są wystarczające?

Kod jest dobrze opisany. Bez większych problemów udało mi się zrozumieć czym się zajmują poszczególne funkcje.

2.7 Czy udało się odtworzyć zamieszone przykłady w kodzie?

Przetestowałem funkcję na 5 wybranych przeze mnie zbiorach wykorzystanych przez grupę 5. W przypadku AUC uzyskałem bardzo podobne wyniki przy 4 zbiorach, jedynie na zbiorze vehicle otrzymałem wzrost o prawie 0.4. Jeśli chodzi o czas trenowania najlepszej konfiguracji to wyniki znacznie się różnią, na zbiorze vehicle zmalał on ponad trzykrotnie.

Dataset	AUC (oryginalne)	Czas (oryginalny)	AUC (testowe)	Czas (testowy)
credit-g	0.799907	0.180996	0.798956	0.026212
kr-vs-kp	0.999419	0.132000	0.999815	0.051860
blood-transfusion-service-center	0.750273	0.096999	0.760811	0.173639
kc1	0.826784	0.184034	0.830505	0.024177
vehicle	0.554425	0.427996	0.590712	0.132688

2.8 Czy udało się użyć przygotowanych kodów na nowych danych?

Na końcu przetestowałem funkcję na nowych danych `https://data.world/exercises/logistic-regression-exercise-1` których celem było przewidzenie czy dany zawodnik utrzyma się ponad 5 lat w NBA. Uzyskałem $AUC = 0.741$ co wydaje się być przyzwoitym wynikiem.

3 Podsumowanie

1. Kod spełnia swój cel
2. Jest dobrze opisany
3. Miejscami nie spełnia standardu PEP 8 ale i tak jest czytelny
4. Osiągane wyniki są nie do końca powtarzalne, czsami istnieją rozbieżności
5. Można go wykorzystać na nowych danych