Laboratorium maszynowej analizy danych

Laboratorium 3

Wprowadzenie do uczenia nadzorowanego. Problem przewidywania klas. Klasyfikacja binarna i wieloklasowa.

Wprowadzenie

Ogólny schemat postępowania przy projekcie uczenia maszynowego:

- zdefiniowanie celu/problemu do rozwiązania
- pozyskanie i przygotowanie danych
- wybór modelu
- wytrenowanie modelu na danych uczących
- dostrojenie modelu
- wykorzystanie wytrenowanego modelu do prognozowania wyników dla nowych przypadków
- wdrażanie, monitorowanie, utrzymanie i konserwacja systemu

Wybór/selekcja modelu

W zależności od problemu, dysponując odpowiednio przygotowanymi danymi, można przystąpić do wyboru modelu uczenia maszynowego. Problem przewidywania klas jest to zdanie klasyfikacyjne, które może być realizowane za pomocą dowolnego modelu klasyfikacyjnego:

- Modele nieparametryczne:
  - algorytm k-najbliższych sąsiadów (ang. *k-Nearest Neighbors, kNN*),
  - drzewo decyzyjne (ang. *Decision Tree*),
  - metody zespołowe (ang. *Ensemble Methods*):
    - klasyfikatory głosujące (ang. *Voting Classifier*),
    - zespoły agregujące (ang. *Bagging*),
    - lasy losowe (ang. *Random Forest*),
    - wzmacnianie (ang. *Boosting*):
      - adaptacyjne (ang. *Adaptive Boosting*),
      - gradientowe (ang. *Gradient Boosting*).
    - kontaminacja (ang. *Stacking*).
- Modele parametryczne i liniowe:
  - **regresja logistyczna**
- Modele nieliniowe:
  - sieci neuronowe
- Metody bayesowskie:

    - naiwny klasyfikator bayesowski (parametryczny)
- Maszyny wektorów nośnych (ang. *Support Vector Machines, SVM*)

Rodzaje problemów klasyfikacyjnych:

- klasyfikacja binarna,
- klasyfikacja wieloklasowa,
- klasyfikacja wieloetykietowa,
- klasyfikacja wielowyjściowa (każda etykieta może być wieloklasowa).

Regresja logistyczna

$$\sigma(t) = \frac{1}{1 + e^{-t}}$$

$$t = w_0 + w_1 x_1$$

W regresji logistycznej model liniowy został zawarty w funkcji logistycznej:

$$p(y_i = 1|X) = \frac{1}{1 + e^{-(w_0 + w_1 x_1)}}$$

gdzie $p(y_i = 1|X)$ określa prawdopodobieństwo, że wartość docelowa i-tej obserwacji y_i należy do klasy 1. X to dane uczące. W efekcie działania funkcji logistycznej wynik zawiera się w przedziale 0 i 1 i oznacza prawdopodobieństwo. Jeżeli $p(y_i = 1|X)$ ma wartość większą niż 0.5, prognozowana jest klasa 1. W przeciwnym razie jest prognozowana klasa 0.

Funkcja kosztu regresji logistycznej to logarytmiczna funkcja straty.

W modelu *LogisticRegression* za siłę regularyzacji odpowiada nie hiperparametr *alpha*, tak jak w innych modelach liniowych, lecz jego odwrotność: *C*. Im większa jego wartość, tym mniejszy stopień regularyzacji modelu.

Dostrojenie modelu/optymalizacja modelu

Większość algorytmów uczenia maszynowego wymaga określenia wartości tzw. hiperparametrów. W zależności od algorytmu może to być np.:

- liczba *k* w algorytmie k-najbliższych sąsiadów,
- maksymalna głębokość drzewa decyzyjnego,
- liczba drzew w lesie losowym,
- liczba warstw w sieci neuronowej.

Wartości hiperparametrów należy ustalić przed dopasowaniem modelu do danych. Przeszukiwanie zakresu hiperparametrów w celu wyboru najlepszego modelu można realizować za pomocą:

- własnoręcznego doboru wartości hiperparametrów,
- przeszukiwania gridowego (ang. *grid search*),

- przeszukiwania losowego (ang. *random search*),
- optymalizacji bayesowskiej (ang. *Bayesian optimization*),
- optymalizacji opartej na gradientach (ang. *Gradient-based Optimization*).

Wykorzystanie wytrenowanego modelu do prognozowania wyników dla nowych przypadków/ewaluacja modelu

Mając wytrenowany model klasyfikacji jego jakość można ocenić za pomocą:

Macierz błędów (ang. *confusion matrix*) inaczej tablica pomyłek zawiera wszystkie możliwe informacje o dokonanych klasyfikacjach i ich poprawności. Dla przypadku binarnego:

Wartość rzeczywista	Wartość przewidywana		
	P (1)	N (0)	
P (1)	PRAWIDZIWIE POZYTYWNE (TP)	FAŁSZYWIE NEGATYWNE (FN)	$TPR = \frac{TP}{TP + FN}$
N (0)	FAŁSZYWIE POZYTYWNE (FP)	PRAWDZIWIE NEGATYWNE (TN)	$TNR = \frac{TN}{TN + FP}$
	$PPV = \frac{TP}{TP + FP}$	$NPV = \frac{TN}{TN + FN}$	

Przypadki rozpatrywane w macierzy błędów odnoszą się do klasyfikacji:

Prawdziwie pozytywnych (*TP*) – przypadki należące do klasy pozytywnej, które zostały poprawnie sklasyfikowane.
Prawdziwie negatywnych (*TN*) – przypadki należące do klasy negatywnej, które zostały poprawnie sklasyfikowane.
Fałszywie pozytywnych (*FP*) – przypadki należące do klasy negatywnej, które zostały błędnie sklasyfikowane jako należące do klasy pozytywnej.
Fałszywie negatywnych (*FN*) – przypadki należące do klasy pozytywnej, które zostały błędnie sklasyfikowane jako należące do klasy negatywnej.

Dokładność (ang. *accuracy*) określa stosunek poprawnie sklasyfikowanych obiektów do wszystkich sklasyfikowanych obiektów:

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

Czułość (ang. *recall, sensitivity*, *True Positive Rate, TPR*) wskazuje, jaki jest udział prawidłowo zaprognozowanych przypadków pozytywnych wśród wszystkich przypadków pozytywnych:

$$TPR = \frac{TP}{TP + FN}$$

Swoistość (ang. *specificity, true negative rate, TNR*) wskazuje, jak dużo ze wszystkich negatywnych przypadków zostało rzeczywiście zaklasyfikowanych do tej kategorii:

$$TNR = \frac{TN}{TN + FP}$$

Precyzja (ang. *precision, positive predicted value, PPV*) ile wśród przykładów zaprognozowanych pozytywnie jest rzeczywiście pozytywnych:

$$PPV = \frac{TP}{TP + FP}$$

F-miara (ang. *F-measure, F1 score*) jest próba zrównoważenia precyzji i czułości. W najprostszej postaci wyznacza się ją na podstawie stosunku podwojonego iloczynu precyzji i czułości do sumy tych dwóch miar:

$$F1\ score = \frac{2TP}{2TP + FP + FN}$$

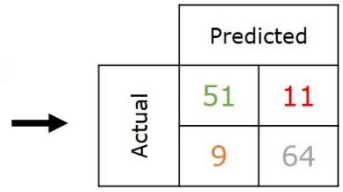
W przypadku klasyfikacji wieloklasowej macierz błędów można przekształcić np. do macierzy typu jeden kontra wszyscy. Konwersja macierzy na macierz typu „jeden do wszystkich" dla klasy 1 danych wygląda następująco:

Wartość rzeczywista	Wartość przewidywana		
	1	0	-1
1	51	4	7
0	2	35	3
-1	7	1	25

FN

TN

FP



Czułość i precyzję dla poszczególnych klas można wyliczyć z poniższych wzorów:

$$recall = \frac{1}{n}\sum_{i=1}^{n}\frac{TP_i}{TP_i + FN_i}$$

$$precision = \frac{1}{n}\sum_{i=1}^{n}\frac{TP_i}{TP_i + FP_i}$$

Krzywa charakterystyki roboczej odbiornika ROC (ang. *receiver operating characteristic*) – krzywa ROC jest graficzną reprezentacją wydajności klasyfikatora binarnego, wskazującą kompromis między współczynnikiem prawdziwie pozytywnych obserwacji (TPR) a współczynnikiem fałszywie pozytywnych obserwacji (FPR, FPR = 1- TNR) przy różnych progach, przy czym:

$$FPR = \frac{FP}{FP + TN}$$

ROC AUC (ang. *area under the ROC curve*) - obszar pod krzywą ROC. Im bliższa 1 wartość ROC AUC, tym lepszy model.

Logarytmiczna funkcja straty (ang. *log loss*) – wskaźnik stosowany w problemach klasyfikacji binarnej i wieloklasowej, zwłaszcza w przypadku modeli, które generują wyniki w postaci prawdopodobieństw dla każdej klasy (np. regresja logistyczna, sieci neuronowe):

$$LogLoss = -\frac{1}{N}\sum_{i=1}^{N}(y_i \cdot \log(p_i) + (1 - y_i) \cdot \log(1 - p_i))$$

gdzie: N – to liczba obserwacji. y_i – to rzeczywisty wynik binarny (0 lub 1) dla i-tej obserwacji, p_i –to przewidywane prawdopodobieństwo, że i-ta obserwacja należy do klasy 1.

Materiały dodatkowe

Aby użyć Scikit-Learn do klasyfikacji, należy postępować według następujących kroków:

1. Wybrać klasę modelu poprzez import odpowiedniej klasy z ScikitLearn.
2. Wybrać hiperparametry.
3. Zapisać dane w macierzy cech (*X*) i wektorze wartości docelowych (*y*).
4. Podzielić dane na zbiór treningowy i testowy.
5. Utworzyć model i dopasować model do danych. Należy wywołać metodę *fit()*.
6. Zastosowanie modelu na nowych danych. Do znalezienia etykiet dla nieznanych danych można wykorzystać metodę *predict()*.
7. Ewaluacja modelu m. in. z wykorzystaniem raportu z klasyfikacji oraz macierzy błędów.

Zadania

Proszę o pobranie danych ze źródła:

https://www.kaggle.com/c/titanic

Zbiór danych dotyczący pasażerów statku rejsowego Titanic, będzie wykorzystany w celu predykcji czy dany pasażer przeżył katastrofę:

PassengerId - numer identyfikacyjny pasażera,
Survived - zmienna określająca, czy dany pasażer przeżył katastrofę (1) czy nie (0),
Pclass - zmienna określająca, czy pasażer podróżował w klasie 1, 2 czy 3,
Name - imię i nazwisko pasażera,
Sex - płeć pasażera,
Age - wiek pasażera,
SibSp - liczba członków rodziny znajdująca się na pokładzie statku (rodzeństwo/małżonkowie),
Parch - liczba członków rodziny znajdująca się na pokładzie statku (rodzice/dzieci),
Ticket - sygnatura biletu,
Fare - cena biletu,
Cabin - numer zajmowanej kajuty,
Embarked - zmienna określająca czy pasażer wsiadł na pokład w porcie Cherbourg, Queenstown lub Southampton.

Zadanie 1.

1.1.Import modułów (należy zaimportować klasę *LogisticRegression* z biblioteki *sklearn.linear_model*).

1.2. Otwarcie pliku z danymi (*titanic_train.csv*). Należy zaimportować nazwy kolumn i stworzyć obiekt *df*.

1.3. Sprawdzenie podstawowych statystyk.

1.4. Sprawdzenie kompletności danych.

1.5. Wizualizacja:

1.5.1. Podział liczby osób ze względu na przetrwanie katastrofy,

1.5.2. Podział przeżycia osób względem płci,

1.5.3. Podział przeżycia osób, względem klasy biletów,

1.5.4. Podział pasażerów ze względu na wiek,

1.5.5. Podział pasażerów ze względu liczbę współpasażerów z rodziny,

1.5.6. Podział biletów ze względu na cenę.

2. Przygotowanie danych:

2.1. Sprawdzenie brakujących danych.

2.2. Wizualizacja:

2.2.1. Heatmapa ze względu na liczbę brakujących danych.

2.3. Data Cleaning:

2.3.1. Age - trzeba uzupełnić brakujące dane;

2.3.2. Cabin - usunąć kolumnę (usunięcie całego atrybutu);

2.3.3. Embarked - usunąć wiersz NaN.

2.4. Przekształcenie danych kategorialnych:

2.4.1. Dummying: płeć i miejsce wejścia na pokład.

2.4.2. Usunięcie kolumn 'Sex', 'Embarked', 'Name', 'Ticket' z danych.

2.4.3. Stworzenie obiektu *df_titanic* z przetworzonych danych treningowych + zmienna dotycząca płci + zmienna dotycząca embark.

2.4.4. Usunięcie kolumny 'PassengerId'.

3. Podział zbioru danych na treningowy i testowy:

3.1. Do zmiennej *X* należy zapisać wszystkie kolumny z zestawu danych *df_titanic* (macierz cech) oprócz kolumny *Survived* (można skorzystać z metody *drop* z argumentem *axis=1*).

3.2. Do zmiennej *y* należy zapisać dane z kolumny *Survived*.

3.3. Do zmiennych *X_train, X_test, y_train, y_test* należy zapisać dane powstałe z podziału *X* i *y* na dane uczące i testowe (z biblioteki *sklearn.model_selection* należy zaimportować *train_test_split*). Zbiór testowy ma stanowić 20% zbioru danych, a *random_state* = 101).

4. Wybór modelu:

Należy stworzyć obiekt regresji logistycznej *lr_model.* Dla obiektu *lr_model* należy wywołać metodę *fit()*, która ma nauczyć model w jaki sposób odgadywać, czy dany pasażer statku Titanic przeżył katastrofę.

5. Wykorzystanie wytrenowanego modelu do prognozowania wyników dla nowych przypadków. W zmiennej *y_pred* należy zapisać wynik predykcji dla *X_test*.

6. Ewaluacja modelu:

6.1. Ewaluacja modelu za pomocą raportu z klasyfikacji oraz macierzy błędów (z biblioteki *sklearn.metrics* należy zaimportować odpowiednie wskaźniki wydajności).
6.2. Krzywa ROC.
6.3. Krzywa czułość/precyzja.

Zadanie 2.

Proszę o pobranie danych ze źródła:

https://archive.ics.uci.edu/dataset/53/iris

Zbiór danych dotyczy kilku rodzajów irysów. Każdy irys jest opisany za pomocą 4 cech (długość i szerokość kielicha, długość i szerokość płatka). Wykorzystując regresję logistyczną należy przeprowadzić klasyfikację wieloklasową.

Zadanie 2a.

Należy stworzyć model regresji logistycznej rozpoznający gatunek *Iris virginica* na podstawie szerokości płatka.

2.1. Wczytanie danych.
2.2. Sprawdzenie kompletności danych.
2.3. Sprawdzenie statystyk podstawowych, informacji nt. danych.
2.4. Wykres punktowy zależności długości płatka w funkcji szerokości płatka, dane różnicowane za pomocą gatunku/rodzaju.
2.5. Wykres punktowy zależności długości kielicha w funkcji szerokości kielicha, dane różnicowane za pomocą gatunku/rodzaju.
2.6. Wykres porównawczy składający się z macierzy par wszystkich zmiennych, różnicowanych na podstawie kolumny 'species'.
2.7. Mapa (heatmapa) korelacji.
2.8. Należy stworzyć model regresji logistycznej rozpoznający gatunek *Iris virginica* na podstawie szerokości płatka.
2.9. Do zmiennej X1 należy zapisać szerokość płatka z zestawu danych *Iris*, a do y1 gatunek *'Iris virginica'*.
2.10. Do zmiennych *X_train, X_test, y_train, y_test* należy zapisać dane powstałe z podziału *X* i *y* na dane uczące i testowe, a *random_state* = 101).
2.11. Trening modelu regresji logistycznej (*random_state* = 101).
2.12. Nowe dane do predykcji *X_new* (1000 równomiernie rozłożonych punktów pomiędzy wartościami 0 a 3). Każdy punkt reprezentuje hipotetyczną szerokość płatka.
2.13. Przewidywanie prawdopodobieństw. Wywołanie metody *predict_proba()* zwraca prawdopodobieństwa dla każdej próbki w *X_new*. Model przewiduje prawdopodobieństwo dla obu klas (nie *Iris virginica* oraz *Iris virginica*).
2.14. Należy znaleźć granicę decyzyjną, czyli punkt, w którym model przechodzi od przewidywania klasy *"inne"* do przewidywania klasy *"Iris virginica"*.

Zadanie 2b.
Wykorzystując regresję logistyczną należy przeprowadzić klasyfikację wieloklasową.

2.1. Do zmiennej *X* należy zapisać macierz cech, a do *y* kolumnę 'species'.
2.2. Podział zbioru *Iris* na podzbiór treningowy i testowy: zbiór testowy stanowi 25% całego zbioru danych, *random_state*=101.
2.3. Standaryzacja danych.
2.4. Budowa modelu *logr* wieloklasowej regresji logistycznej:
    2.4.1. dostrojenie modelu za pomocą metody przeszukiwania siatki lub przeszukiwania losowego: C = np.logspace(0, 4, 10), penalty = ['l1', 'l2']. Dla modelu *logr*, solver potrzebny do obsługi kary L1 i L2 to ‘liblinear’.
    2.4.2. Wytrenowanie modelu.
    2.4.3. Wypróbowanie modelu na zbiorze testowym.
    2.4.4. Ewaluacja modelu na podstawie raportu z klasyfikacji i macierzy błędów.
    2.4.5. Krzywa ROC.
    2.4.6. Krzywa czułość/precyzja.

Zadanie dodatkowe

Dla zestawu danych Iris można przeprowadzić walidację krzyżową (np. cv=5) i porównać wyniki z zdaniem 2b.

Literatura

1. A. Géron, *Uczenie maszynowe z użyciem Scikit-Learn, Keras, i TensorFlow,* Wydanie III, Helion, 2023.

2. S.J. Russell, P. Norvig, *Sztuczna inteligencja. Nowe spojrzenie,* Wydanie IV, Helion, 2023.

3. A. Król-Nowak, K. Kotarba, *Podstawy uczenia maszynowego,* Wydawnictwa AGH, Kraków, 2022.

4. K. Gallatin, K. Albon, *Uczenie maszynowe w Pythonie. Receptury. Od przygotowania danych do deep learningu*, Wydanie II, Helion, 2023.

5. J. VanderPlas, *Python Data Science. Niezbędne narzędzia do pracy z danymi,* Wydanie II, Helion, 2023.