SPRAWOZDANIE

Zajęcia: Nauka o danych I
Prowadzący: prof. dr hab. Vasyl Martsenyuk

Laboratorium Nr 7 Data 21.12.2024 Temat: Klasyfikacja danych przy użyciu algorytmów uczenia maszynowego Wariant drugi (2)	Bartosz Bieniek Informatyka II stopień, stacjonarne, 1 semestr, gr.A

1. Polecenie: wariant drugi zadania

Wariant 2: Zbiór danych Iris

Zbiór Iris zawiera 150 próbek różnych gatunków irysów (Setosa, Versicolor, Virginica). Celem zadania jest klasyfikacja gatunku irysa na podstawie 4 cech: długość i szerokość kielicha oraz płatka. Link do danych: https://archive.ics.uci.edu/ml/datasets/iris.

2. Opis programu opracowanego [Kod źródłowy github.com/mindgoner]

Wyżej wymieniony link zawiera dane dane dotyczące Irysów oraz ich klasyfikacji według gatunków. Zmieniono nazwę plików *iris.data* na *dane-treningowe.csv* oraz *bezdekIris.data* na *dane-nieoznaczone.csv*.

Dane wymagały przygotowania do ćwiczenia. Plik *dane-nieoznaczone.csv* otworzono za pomocą edytora arkuszy kalkulacyjnych oraz dodano nową kolumnę z losowymi liczbami. Następnie posortowano dane według kolumny z losowymi liczbami:

	A	B	C	D	E	F	G
1	5.5	2.5	4.0	1.3	Iris-versicolor	0	
2	5.4	3.9	1.7	0.4	Iris-setosa	1	
3	5.8	2.7	5.1	1.9	Iris-virginica	1	
4	4.6	3.6	1.0	0.2	Iris-setosa	3	
5	5.5	2.6	4.4	1.2	Iris-versicolor	5	
6	5.5	2.3	4.0	1.3	Iris-versicolor	7	
7	6.3	2.5	4.9	1.5	Iris-versicolor	9	
8	6.7	3.3	5.7	2.5	Iris-virginica	9	
9	5.1	3.8	1.9	0.4	Iris-setosa	10	
10	5.8	2.6	4.0	1.2	Iris-versicolor	10	
11	4.8	3.1	1.6	0.2	Iris-setosa	11	
12	6.3	3.3	6.0	2.5	Iris-virginica	11	
13	6.4	2.8	5.6	2.1	Iris-virginica	11	

Rys. 1. Pomieszane dane.

Plik *dane-nieoznaczone.csv* zapisano i skopiowano do *dane-oryginalne.csv*. Plik *dane-oryginalne.csv* pozwoli na koniec zestawić dane i zweryfikować, czy poprawnie dokonano klasyfikacji na końcu ćwiczenia. Z pliku dane-nieoznaczone.csv usunięto kolumnę klasyfikacji i liczb losowych.

	A	B	C	D	E	F	G
1	5.5	2.5	4.0	1.3			
2	5.4	3.9	1.7	0.4			
3	5.8	2.7	5.1	1.9			
4	4.6	3.6	1.0	0.2			
5	5.5	2.6	4.4	1.2			
6	5.5	2.3	4.0	1.3			
7	6.3	2.5	4.9	1.5			
8	6.7	3.3	5.7	2.5			
9	5.1	3.8	1.9	0.4			
10	5.8	2.6	4.0	1.2			
11	4.8	3.1	1.6	0.2			
12	6.3	3.3	6.0	2.5			
13	6.4	2.8	5.6	2.1			

Rys. 2. Usunięto etykietę w celu rzetelnego przebiegu ćwiczenia.

Plik *dane-treningowe.csv* pozostawiono bez zmian. Plik ma identyczną budowę, jak *dane-nieoznaczone.csv*, jednak każdy rekord jest sklasyfikowany gatunkiem:

A	B	C	D	E	F	G	H
5.1	3.5	1.4	0.2	Iris-setosa			
4.9	3.0	1.4	0.2	Iris-setosa			
4.7	3.2	1.3	0.2	Iris-setosa			
7.0	3.2	4.7	1.4	Iris-versicolor			
6.4	3.2	4.5	1.5	Iris-versicolor			
6.9	3.1	4.9	1.5	Iris-versicolor			
5.5	2.3	4.0	1.3	Iris-versicolor			
5.8	2.7	5.1	1.9	Iris-virginica			
7.1	3.0	5.9	2.1	Iris-virginica			
6.3	2.9	5.6	1.8	Iris-virginica			
6.5	3.0	5.8	2.2	Iris-virginica			

Rys. 3. Dane treningowe.

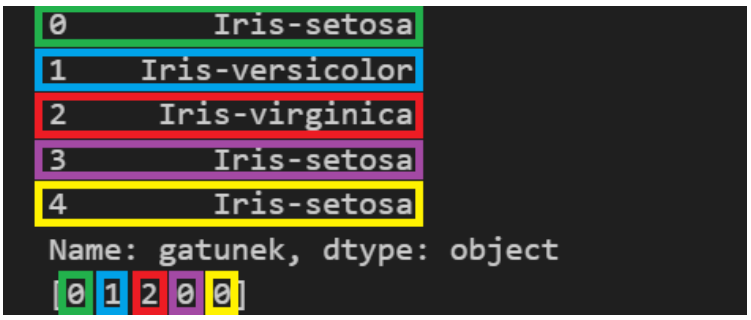
Dane treningowe posłużyły do stworzenia modelu klasyfikującego dane niesklasyfikowane (nieoznaczone).

Przystąpiono do wykonywania celu ćwiczenia. W pierwszym kroku zaimportowano odpowiednie biblioteki: sklearn oraz pandas.

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestClassifier
```

Rys. 4. Dane treningowe.

Moduł *LabelEncoder* posłużył do zamienienia gatunków (etykiet) na gatunki w postaci liczby (etykiety liczbowe). Jak widać na poniższym rysunku, każdy gatunek ma numeryczny odpowiednik. *Iris-setosa* ma identyfikator 0, *Iris-versicolor* posiada identyfikator 1 a *Iris-virginica* posiada przypisany numer 2.



Rys. 5. Zasada działania LabelEncoder - konwersja mnemoniczna na liczbową.

Drugą zaimportowaną biblioteką jest RandomForrestClassifier. Jest to podstawowy moduł pozwalający w łatwy sposób dokonać klasyfikacji w uczeniu maszynowym.

Podstawową operacją w klasyfikowaniu danych jest utworzenie dataframeów z danymi treningowymi (z etykietami gatunków) i nieoznaczonymi (bez etykiet).

```
dane_treningowe = pd.read_csv("dane-treningowe.csv", header=None)
dane_treningowe.columns = ["dlugosc_kielicha", "szerokosc_kielicha", "dlugosc_platka", "szerokosc_platka", "gatunek"]

dane_nieoznaczone = pd.read_csv("dane-nieoznaczone.csv", header=None)
dane_nieoznaczone.columns = ["dlugosc_kielicha", "szerokosc_kielicha", "dlugosc_platka", "szerokosc_platka"]
```

Rys. 6. Tworzenie dataframeów Pandas i oznaczanie nagłówkami kolumn.

W kolejnym kroku przystąpiono do przygotowania danych:

```
dane_nieoznaczone_z_nazwami_kolumn = dane_treningowe.iloc[:, :-1]
etykiety = dane_treningowe.iloc[:, -1]
label_encoder = LabelEncoder()
etykiety_numeryczne = label_encoder.fit_transform(etykiety)
```

Rys. 7. Przygotowywanie danych, oznaczanie.

Dane zostały podzielone na cechy (wszystkie kolumny poza ostatnią) i etykiety (ostatnia kolumna) przy użyciu funkcji iloc. Następnie inicjalizowano LabelEncoder, który zakodował tekstowe etykiety na wartości numeryczne, co umożliwiło ich użycie w algorytmach uczenia maszynowego.

```
model = RandomForestClassifier(random_state=42, n_estimators=100)
model.fit(dane_nieoznaczone_z_nazwami_kolumn, etykiety_numeryczne)
```

Rys. 8. Przygotowywanie danych, oznaczanie.

Na rysunku ósmym znajduje się fragment kodu, w którym zainicjalizowano model klasyfikatora Random Forest z ustaloną liczbą drzew (n_estimators=100) oraz losowym stanem (random_state=42) dla powtarzalności wyników. Następnie model został wytrenowany na danych wejściowych (cechy) i odpowiadających im numerycznych etykietach.

Wartość random_state została wybrana, aby wyniki były powtarzalne przy kolejnych uruchomieniach kodu. Ustalenie random_state gwarantuje, że losowe operacje wykonywane przez model (np. wybór próbek do tworzenia drzew) będą zawsze prowadziły do tych samych wyników. Liczba 42 jest często używana w przykładach, ale w praktyce dowolna liczba spełni ten cel.

Liczba drzew równa 100 jest standardową wartością początkową, często wybieraną jako kompromis między dokładnością modelu a czasem jego trenowania. Większa liczba drzew zwiększa dokładność modelu, ale kosztem większych zasobów obliczeniowych. W przypadku problemów o umiarkowanej złożoności, takich jak klasyfikacja irysów, 100 drzew zazwyczaj wystarcza, aby uzyskać dobre wyniki.

```
prawdopodobienstwa = model.predict_proba(dane_nieoznaczone)
prawdopodobienstwa_df = pd.DataFrame(
    prawdopodobienstwa,
    columns=[f"procent_{label}" for label in label_encoder.classes_]
)
```

Rys. 9. Klasyfikacja danych nieoznaczonych i przekształcenie wyników do ramki.

Model obliczył prawdopodobieństwa przynależności każdego rekordu w danych nieoznaczonych do poszczególnych klas za pomocą metody predict_proba. Wyniki zostały zapisane w DataFrame, gdzie każda kolumna reprezentuje procentowe prawdopodobieństwo przynależności do danej klasy, a ich nazwy zostały dynamicznie wygenerowane na podstawie zakodowanych etykiet klas. Na poniższym rysunku znajduje się kod przypisujący wyżej pozyskane dane do ramki danych wyników oraz dodanie kolumny “gatunek” w której zapisano najbardziej prawdopodobny (z największą wartością procentową prawdopodobieństwa) gatunek.

```
wyniki = pd.concat([dane_nieoznaczone, prawdopodobienstwa_df], axis=1)
wyniki["gatunek"] = label_encoder.inverse_transform(prawdopodobienstwa.argmax(axis=1))
```

Rys. 10. Konkatenacja danych w ramce danych.

W ostatnim kroku zapisano dane z pliku *dane-nieoznaczone.csv* do pliku *dane-oznaczone.csv* wraz z odpowiednim oznaczeniem.

Na poniższym rysunku zaprezentowano zestawienie danych z pliku *dane-oryginalne.csv* i wyjściowego pliku *dane-oznaczone.csv*. Jak można zauważyć dane oryginalne [kolumny A-D] pokrywają się z danymi wyjściowymi [kolumny G-J]. “Rzeczywisty gatunek” pokrywa się z kolumną wygenerowaną przez klasyfikator “gatunek”. W kolumnach rozpoczynających się od “PP_” znajduje się prawdopodobieństwo procenowe (z zakresu 0-1 gdzie 0 to 0% a 1 to 100%), które zostało wyliczone dla tych wymiarów kielicha i wymiarów płatków.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1					Rzeczywisty gatunek						PP_Iris-setosa	PP_Iris-versicolor	PP_Iris-virginica	gatunek	**POPRAWNIE?**	
2	5.5	2.5	4.0	1.3	Iris-versicolor		5.5	2.5	4.0	1.3	0.0	1.0	0.0	Iris-versicolor	TAK!	
3	5.4	3.9	1.7	0.4	Iris-setosa		5.4	3.9	1.7	0.4	1.0	0.0	0.0	Iris-setosa	TAK!	
4	5.8	2.7	5.1	1.9	Iris-virginica		5.8	2.7	5.1	1.9	0.0	0.0	1.0	Iris-virginica	TAK!	
5	4.6	3.6	1.0	0.2	Iris-setosa		4.6	3.6	1.0	0.2	1.0	0.0	0.0	Iris-setosa	TAK!	
6	5.5	2.6	4.4	1.2	Iris-versicolor		5.5	2.6	4.4	1.2	0.0	1.0	0.0	Iris-versicolor	TAK!	
7	5.5	2.3	4.0	1.3	Iris-versicolor		5.5	2.3	4.0	1.3	0.0	0.98	0.02	Iris-versicolor	TAK!	
8	6.3	2.5	4.9	1.5	Iris-versicolor		6.3	2.5	4.9	1.5	0.0	0.83	0.17	Iris-versicolor	TAK!	
9	6.7	3.3	5.7	2.5	Iris-virginica		6.7	3.3	5.7	2.5	0.0	0.0	1.0	Iris-virginica	TAK!	
10	5.1	3.8	1.9	0.4	Iris-setosa		5.1	3.8	1.9	0.4	1.0	0.0	0.0	Iris-setosa	TAK!	
11	5.8	2.6	4.0	1.2	Iris-versicolor		5.8	2.6	4.0	1.2	0.0	1.0	0.0	Iris-versicolor	TAK!	
12	4.8	3.1	1.6	0.2	Iris-setosa		4.8	3.1	1.6	0.2	1.0	0.0	0.0	Iris-setosa	TAK!	
13	6.3	3.3	6.0	2.5	Iris-virginica		6.3	3.3	6.0	2.5	0.0	0.0	1.0	Iris-virginica	TAK!	
14	6.4	2.8	5.6	2.1	Iris-virginica		6.4	2.8	5.6	2.1	0.0	0.0	1.0	Iris-virginica	TAK!	
15	6.9	3.1	5.1	2.3	Iris-virginica		6.9	3.1	5.1	2.3	0.0	0.01	0.99	Iris-virginica	TAK!	
16	5.4	3.0	4.5	1.5	Iris-versicolor		5.4	3.0	4.5	1.5	0.02	0.96	0.02	Iris-versicolor	TAK!	
17	5.0	2.3	3.3	1.0	Iris-versicolor		5.0	2.3	3.3	1.0	0.0	1.0	0.0	Iris-versicolor	TAK!	
18	7.1	3.0	5.9	2.1	Iris-virginica		7.1	3.0	5.9	2.1	0.0	0.0	1.0	Iris-virginica	TAK!	

Rys. 10. Zestawienie danych.

3. Wnioski

Jak można zauważyć z zestawienia wyników, klasyfikator nie zawsze był w 100% pewny, że dane wymiary sugerują przynależność do konkretnego gatunku. Najniższą wartością prawdopodobieństwa otrzymał kwiat o wymiarach 4.9, 2.5, 4.5 oraz 1.7. Został on przez klasyfikator oznaczony jako Iris-virginica (w 63%) oraz Iris-versicolor (w 37%). Większościowo program jednak przypisał go do grupy Iris-virginica, co było poprawną odpowiedzią.

Spośród 150 rekordów otrzymano tylko dwa rekordy, które we wszystkich kolumnach PP miały wartość >0. Obydwa rekordy otrzymały odpowiednio 1% i 2% oraz 2% i 1% w kolumnach Iris-virginica i Iris-setosa oraz 98% w kolumnie Iris-versicolor.

Proces kodowania etykiet przy użyciu LabelEncoder umożliwił efektywne przetwarzanie danych kategorycznych przez model. Generowanie prawdopodobieństw dla każdej klasy pozwoliło nie tylko na klasyfikację, ale również na ocenę pewności modelu względem każdej klasy, co jest szczególnie przydatne w analizach wymagających interpretowalności wyników.