Katowice 10.07.2020r.

Projekt

Drzewo decyzyjne autentyczności banknotu

Systemy uczące

Marcin Satława

Uniwersytet Ekonomiczny w Katowicach

Wydział Informatyki i Komunikacji

Kierunek: Informatyka

Stopień II

Rok 1

Semestr 2

Prowadzący: dr Przemysław Juszczuk

Opracowanie teoretyczne

Wstęp

Program napisano w języku Python z wykorzystaniem narzędzia Jupyter Notebook. Kod został opisany z metodami, klasami, zmiennymi w języku angielskim. Do kodu dołączone komentarze w języku polskim. Program krok po kroku pokazuje wykonanie Drzewa decyzyjnego autentyczności banknotu

Drzewo decyzyjne – na czym polega ?

Drzewo decyzyjne jest graficzną metodą, która pomaga w procesach decyzyjnych, będąca jedną z najczęstszych technik używanych analizy danych. Drzewa decyzyjne składają się z korzenia oraz gałęzi prowadzące z korzenia do następnych wierzchołków. Wierzchołki, z których wychodzi co najmniej jedna krawędź, są nazywane węzłami, a pozostałe wierzchołki – liśćmi.

Drzewo decyzyjne – zastosowanie

Drzewa decyzyjne znajdują praktyczne zastosowanie w różnego rodzaju problemach decyzyjnych, szczególnie takich gdzie występuje dużo rozgałęziających się wariantów a także w warunkach ryzyka. Drzewa decyzyjne znalazły zastosowanie m.in. w takich dziedzinach jak medycyna czy botanika, ekonomii.

Drzewo decyzyjne – zalety

Istnieje kilka zalet korzystania z drzew decyzyjnych do analizy predykcyjnej:

- Drzewa decyzyjne mogą być używane do przewidywania zarówno wartości ciągłych, jak i dyskretnych, czyli działają dobrze zarówno dla zadań regresji, jak i klasyfikacji.
- Wymagają one stosunkowo mniej wysiłku do szkolenia algorytmu.
- Mogą one służyć do klasyfikowania nieliniowo rozdzielnych danych.
- Są bardzo szybkie i wydajne w porównaniu do KNN i innych algorytmów klasyfikacji.

Ważnymi pojęciami w projekcie są: wariancja, skośność, entropia, kurtoza, macierz błędów i wynik F1.

Wariancja informuje o tym, w jak dużym stopniu jest zróżnicowanie wyników w danym zbiorze wyników (zmiennej), czyli czy wyniki są bardziej skoncentrowane wokół średniej, czy

są małe różnice pomiędzy średnią a poszczególnymi wynikami czy może rozproszenie wyników jest duże, duża jest różnica poszczególnych wyników od średniej.

Skośność informuje o tym jak wyniki dla danej zmiennej kształtują się wokół średniej. Czy większość zaobserwowanych wyników jest z lewej strony średniej, blisko wartości średniej czy z prawej strony średniej? Innymi słowy, czy w naszym zbiorze obserwacji więcej jest wyników, które są niższe niż średnia dla całej grupy, wyższe czy równe średniej.

Entropia – termodynamiczna funkcja stanu, określająca kierunek przebiegu procesów spontanicznych (samorzutnych) w odosobnionym układzie termodynamicznym. Entropia jest miarą stopnia nieuporządkowania układu i rozproszenia energii oraz jest wielkością ekstensywną. Zgodnie z drugą zasadą termodynamiki, jeżeli układ termodynamiczny przechodzi od jednego stanu równowagi do drugiego, bez udziału czynników zewnętrznych, to jego entropia zawsze rośnie.

Kurtoza jest to względna miara koncentracji i spłaszczenia rozkładu. Określa rozmieszczenie i koncentrację wartości (zbiorowości) w pobliżu średniej. Występuje on w postaci stosującej moment centralny czwartego rzędu.

$$K = \frac{m_4}{s^4}\Big|$$

Gdzie:

m4 oznacza moment centralny rzędu czwartego

s2 jest odchyleniem standardowym podniesionym do czwartej potęgi.

Tablica pomyłek (macierz błędów swoje zastosowanie ma przy ocenie jakości klasyfikacji binarnej (na dwie klasy). Dane oznaczone etykietami: pozytywną i negatywną poddawane są klasyfikacji, która przypisuje im predykowaną klasę pozytywną albo predykowaną klasę negatywną. Możliwa jest sytuacja, że dana oryginalnie oznaczona jako pozytywna zostanie omyłkowo zaklasyfikowana jako negatywną. Wszystkie takie sytuacje przedstawia tablica pomyłek.

		Klasa rzeczywista	
		pozytywna	**negatywna**
Klasa predykowana	**pozytywna**	prawdziwie pozytywna (TP)	fałszywie pozytywna (FP)
	negatywna	fałszywie negatywna (FN)	prawdziwie negatywna (TN)

Wynik F1 jest harmoniczną średnią precyzji i przypomnień, gdzie wynik F1 jest najlepszy, gdy wynosi 1 (doskonała precyzja i przypomnieć). Bierze pod uwagę zarówno dokładność p, jak i wycofanie r testu w celu obliczenia wyniku: p jest liczbą prawidłowych pozytywnych wyników podzieloną przez liczbę wszystkich pozytywnych wyników zwróconych przez klasyfikator, natomiast r jest liczbą prawidłowych pozytywnych wyników podzieloną przez liczbę wszystkich odpowiednich próbek (wszystkie próbki, które powinny zostać zidentyfikowane jako pozytywne).

Spis bibliotek używanych w projekcie:

Pandas - biblioteką oprogramowania napisaną dla języka programowania Python do manipulowania i analizy danych. W szczególności oferuje struktury danych i operacje do manipulowania tabelami numerycznymi i szeregami czasowymi. Jest to wolne oprogramowanie wydane na podstawie trzypunktowej licencji BSD.

Numpy - jest biblioteką dla języka programowania Pythona, dodając obsługę dużych, wielowymiarowych tablic i macierzy, wraz z dużą kolekcją funkcji matematycznych wysokiego poziomu do pracy na tych tablicach.

Matplotlib – biblioteka do tworzenia wykresów dla języka programowania Python i jego rozszerzenia numerycznego NumPy. Zawiera ona API "pylab" zaprojektowane tak aby było jak najbardziej podobne do MATLAB’a, przez co jest łatwy do nauczenia przez jego użytkowników.

Opracowanie praktyczne

Projekt ma za zadanie sprawdzić, czy banknot jest autentyczny czy może fałszywy w zależności od następujących atrybutów: wariancja, skośność, kurtoza, entropia oraz klasa. Dane, które są wykorzystywane w projekcie zostaną dodane do załącznika projektu.

Kroki jakie zostały wykonane w projekcie to import bibliotek i zestawu danych, analiza danych, podział danych na zestawy szkoleniowe i testowe, wyszkolenie algorytmu oraz ocena wydajności algorytmu w zestawie danych

Importowanie bibliotek

Poniżej zrzut ekranu zaimportowanych bibliotek

```
: import pandas as pd
  import numpy as np
  import matplotlib.pyplot as plt

  %matplotlib inline
  # import bibliotek
```

Importowanie zestawu danych

Z racji, że plik jest w formacie CSV, musimy użyć metody pandas do odczytu pliku z zestawem danych. Skrypt wygląda następująco:

```
dataset = pd.read_csv("D:\Pulpit\dataset.csv") #odczyt danych z pliku
```

Analiza danych

```
dataset.shape #dlugosc zestawu danych
```

```
(1372, 5)
```

Wykonanie powyższego polecenia wyświetla liczbę wierszy oraz kolumn z naszego zestawu danych. Otrzymane dane pokazują 1372 rekordy oraz 5 atrybutów.

```
dataset.head(10) # wypisanie pierwszych dziesieciu kolumn wraz z wartosciami
```

	Variance	Skewness	Curtosis	Entropy	Class
0	3.62160	8.6661	-2.80730	-0.44699	0
1	4.54590	8.1674	-2.45860	-1.46210	0
2	3.86600	-2.6383	1.92420	0.10645	0
3	3.45660	9.5228	-4.01120	-3.59440	0
4	0.32924	-4.4552	4.57180	-0.98880	0
5	4.36840	9.6718	-3.96060	-3.16250	0
6	3.59120	3.0129	0.72888	0.56421	0
7	2.09220	-6.8100	8.46360	-0.60216	0
8	3.20320	5.7588	-0.75345	-0.61251	0
9	1.53560	9.1772	-2.27180	-0.73535	0

Powyższe polecenia pokazuje dane dla pierwszych dziesięciu rekordów zestawu danych.

Przygotowanie danych

```
# podzielenie danych na atrybuty i etykiety
X = dataset.drop('Class', axis=1) # axis = 1 zastosowanie metody w każdym wierszu
y = dataset['Class']

# X zawiera wszystkie kolumny, natomiast  Y zawiera wszystkie odpowiednie etykiety
```

Następnie otrzymane dane trzeba podzielić na atrybuty i etykiety. Zmienna X zawiera wszystkie kolumny, natomiast zmienna Y zawiera etykiety, w tym przypadku tylko Class. Metodę Axis= 1 stosuje się w każdym wierszu lub do etykiet kolumn.

Podział danych na zestawy szkoleniowe i testowe

Z biblioteki Scikit-Learn importujemy metodę train_test_split.

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
#okreslenie zestawu testowego: 20% dane zestawu testowego, 80% dane zestawu szkoleniowego
```

Następnie jak w kodzie powyżej parameter test_size określa stosunek zestawu testów, który jest używany do podzielenia 20% do testu i 80% do szkolenia.

Szkolenia i przewidywanie

Po podzieleniu zestawu danych na szkoleniowe i testowe można dokonać szkolenia algorytmu drzewa decyzyjnego przy użyciu posiadanych danych oraz można dokonać prognozy. Biblioteka Scikit-Learn zawiera bibliotekę, która posiada wbudowane metody oraz klasy dla rożnych algorytmów drzewa decyzyjnego.

```
from sklearn.tree import DecisionTreeClassifier


classifier = DecisionTreeClassifier()
classifier.fit(X_train, y_train)
#import klasy DecisionTreeClassifier i wyszkolenie algorytmu
```

```
DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                       max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort=False,
                       random_state=None, splitter='best')
```

Powyższa metoda klasy jest wywoływana do szkolenia algorytmu na dane szkoleniowe, który przekazywany jest metodzie jako parametr.

Następnie, gdy klasyfikator został przeszkolony następnym krokiem jest zrobienie prognozy na danych testowych. Poniższy kod pokazuje jej składnie.

```
y_pred = classifier.predict(X_test)
#prognoza danych testowych
```

Ocena algorytmu

Dotychczas w naszym projekcie został przeszkolony algorytm oraz zostały dokonana prognoza. Następnie zostanie sprawdzony jak dokładny jest algorytm. W przypadku wykonania zadania klasyfikacyjnego często używanymi metrykami są m.in. macierz błędów, dokładność, odwołanie oraz wynik F1. W tym przypadku wraz z biblioteką Scikit-Learn użyjemy metod. classification_report oraz confusion_matrix

```
#import klas classification_report, confusion_matrix wraz z wynikami macierzy błedów,
#dokładności, odwołania i wyniku F1

from sklearn.metrics import classification_report, confusion_matrix

print(confusion_matrix(y_test, y_pred)) #wynik macierzy błędów
print(classification_report(y_test, y_pred)) #wynik  dokładności, odwołania oraz wyniku F1
```

Wywołanie powyższego kodu daje następujący wynik:

```
[[165   2]
 [  2 106]]
              precision    recall  f1-score   support

           0       0.99      0.99      0.99       167
           1       0.98      0.98      0.98       108

    accuracy                           0.99       275
   macro avg       0.98      0.98      0.98       275
weighted avg       0.99      0.99      0.99       275
```

Z macierzy można zauważyć, że z 275 wystąpień testowych stworzony algorytm błędnie przyjął tylko 4 wartości. Dokładność wynosi 98,5%, co daje bardzo dobry wynik.

Załączniki:

1. Anaconda- Jupyter

https://www.anaconda.com/products/individual#windows

2. Zestaw danych wykorzystanych w projekcie – autentycznośc bankota

https://archive.ics.uci.edu/ml/datasets/banknote+authentication

Bibliografia:

1. http://home.agh.edu.pl/~pmarynow/pliki/iwmet/drzewa.pdf

2. https://mfiles.pl/pl/index.php/Drzewo_decyzyjne

3. https://www.naukowiec.org/wiedza/statystyka/wariancja_719.html

4. https://pl.wikipedia.org/wiki/Entropia

5. https://www.naukowiec.org/wiedza/statystyka/skosnosc_714.html

6. https://mfiles.pl/pl/index.php/Kurtoza

7. https://pl.wikipedia.org/wiki/Tablica_pomy%C5%82ek

8. https://en.wikipedia.org/wiki/F1_score