Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie

Dokumentacja do projektu

„Drzewko decyzyjne”

z przedmiotu
Programowanie w Języku Python

Katarzyna Pióro
Jakub Szymański
Wojciech Wołosz

Elektronika PL, III rok

prowadzący: dr inż. Maciej Wielgosz

23.01.2022 r.

1. Opis projektu

Celem projektu było zrobienie drzewka decyzyjnego. Polega ono na estymowaniu przez program odpowiedzi ‘tak’ lub ‘nie’ na podstawie dostarczonych danych.

2. Project description

The goal of the project was to make decision tree. It is based on the software estimating the ‘yes’ or ‘no’ answer on the basis of provided data.

3. Opis algorytmu „Drzewa decyzyjnego”

Algorytm został opisany na podstawie kodu

1) `dataset = pandas.read_csv("titanic.csv")`
wczytywanie danych do programu z pliku titanic.csv

2)
```
dataset.drop(['PassengerId', 'Name', 'SibSp', 'Parch',
'Ticket', 'Cabin', 'Embarked'], axis='columns',
inplace=True)
```
usuwanie niepotrzebnych informacji z pliku z danymi

3)
```
d = {'male': 0, 'female': 1}
dataset['Sex'] = dataset['Sex'].map(d)
```
zmiana niektórych typów danych na takie, które przyjmują funkcje tworzenia drzewa

4) `dataset.Age = dataset.Age.fillna(dataset.Age.mean())`
nie znamy wszystkich danych więc wypełniamy puste pola średnią złożoną z reszty danych tego samego typu

5)
```
inputs = dataset.drop('Survived',axis='columns')
target = dataset.Survived
```
podział danych na parametry oraz wynik

6) `dtree = DecisionTreeClassifier(max_depth=4)`
tworzenie drzewka decyzyjnego:

- **używany jest tu algorytm CART**
- **dobieranie argumentu, który najlepiej rozdziela dane**
- **rozdzielenie danych na 2 części (2 gałęzie)**
- **dla każdej gałęzi jest powtarzany ten sam proces rozdzielania danych aż algorytm uzna, że przez dalsze rozdzielanie danych nic nie zyska**

7)
```
data = tree.export_graphviz(dtree, out_file=None,
feature_names=dataset.columns[1:5],
class_names=['Died', 'Survive'], rounded=True,
filled=True)
graph = pydotplus.graph_from_dot_data(data)
graph.write_png('titanic.png')
```

```
img = pltimg.imread('titanic.png')
```
zapisywanie gotowego grafu do pliku titanic.png

4. Zastosowane biblioteki

Do prawidłowego działania programu potrzebne były następujące biblioteki:

- `pandas`
- `sklearn.tree`
- `sklearn.tree.DecisionTreeClassifier`
- `pydotplus`
- `matplotlib.pyplot`
- `matplotlib.image`

5. Działanie programu

Do działania jest potrzebna pewna ilość danych początkowych, na których program może się nauczyć przewidywać odpowiedzi. Dane te są pobierane z pliku titanic.csv. Program robi potrzebne obliczenia na podstawie dostarczonych danych po czym wyświetla obraz titanic.png, na którym widać korzeń, gałęzie i liście naszego drzewa.

6. Kompilacja

Żeby uruchomić program wystarczy standardowa kompilacja.
Projekt był testowany w systemie Windows 10.

7. Pliki źródłowe

Projekt składa się z następujących plików źródłowych

- `decisive-tree.py` - skrypt realizujący algorytm
- `titanic.csv` - baza danych