**Podstawy Sztucznej Inteligencji**

Projekt 2: Uczenie maszynowe

Emilia Gosk, nr albumu 266557

1. **Zadanie projektowe**

Zadanie polega na implementacji algorytmu ID3 i wykorzystaniu go do przewidywania rozwodów na podstawie zbioru danych dot. małżeństw ze strony [Divorce+Predictors+data+set](http://archive.ics.uci.edu/ml/datasets/Divorce+Predictors+data+set).

1. **Opis zbioru danych**

* 170 próbek - zbiór danych stanowią odpowiedzi 170 osób w wieku 20–63 na ankietę (DPS – Divorce Prediction Scale) dotyczącą ich relacji z obecnym lub byłym małżonkiem
* 54 atrybuty - ankieta zawierała 54 stwierdzenia
* Dozwolone wartości atrybutów: 0, 1, 2, 3, 4 - uczestnicy badania oceniali jak bardzo każde stwierdzenie jest prawdziwe dla ich relacji w skali 0-4 (0 – nieprawda, 4 – prawda)
* Klasy: 0, 1 - każda próbka zawierała informację o klasie do której należy (1, gdy uczestnik jest rozwiedziony, 0 – małżeństwo trwa).
* Rozkład klas:
  + 1 – rozwiedziony – 49% badanych, czyli 84 osoby
  + 0 – w małżeństwie – 51% badanych, czyli 86 osób
* W zbiorze danych nie ma brakujących wartości

Informacje dodatkowe:

* Dane zostały zebrane w Turcji, ponad połowa z badanych małżeństw była aranżowana (56,5%).
* W grupie badawczej było 86 kobiet (51%), 84 mężczyzn (49%).
* 74,7% uczestników ma dzieci.
* 60,56% uczestników ma wykształcenie wyższe

1. **Podział prac**

Wykonanie implementacji oraz jej przebadanie – Emilia Gosk

1. **Decyzje projektowe**

* Ze względu na to, że zbiór danych testowych nie był wyróżniony, zastosowałam walidację krzyżową, aby oszacować skuteczność modeli i zapobiec problemom przetrenowania i niedopasowania. Wybrałam k-krotną walidację krzyżową, gdyż ten rodzaj został najdokładniej opisany podczas wykładów.
* Wczytane do programu dane przechowywałam w strukturze DataFrame (tzw. ramka danych) dostarczanej przez bibliotekę pandas. Obiekty tej klasy są 2 wymiarową tabelą danych o uporządkowanym zbiorze kolumn, podobną do tabeli w bazie danych, przez co praca z DataFrame jest dość intuicyjna i dobrze sprawdza się w tym projekcie.

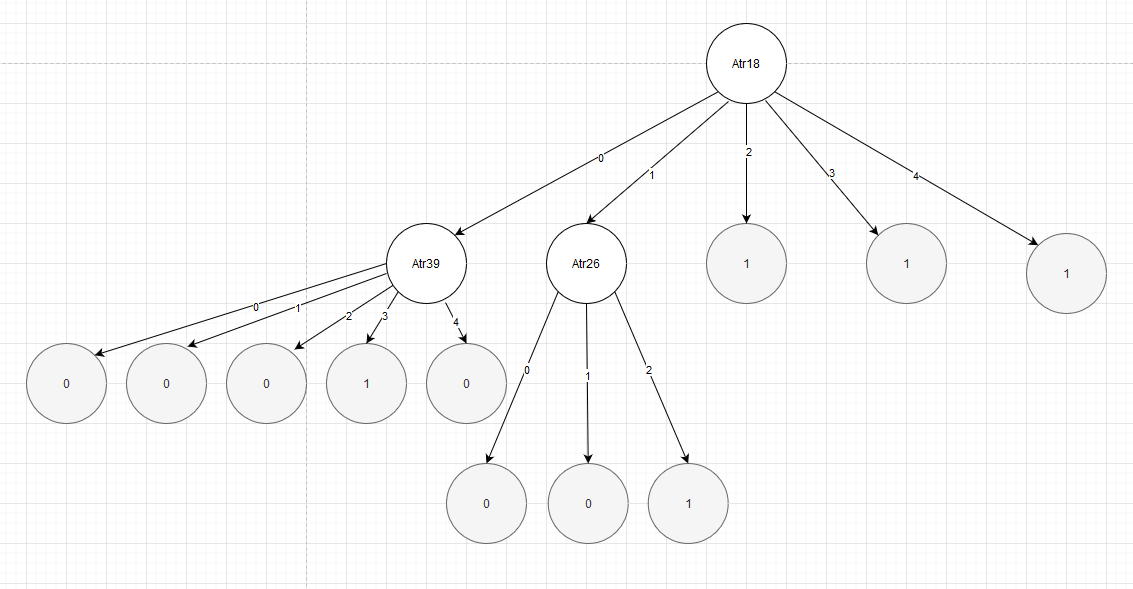
1. **Wykorzystane narzędzia i biblioteki**

* IDE: PyCharm
* Biblioteki: Pandas, Numpy, Sklearn

1. **Cele i tezy przeprowadzonych badań**

* Ze względu na stosunek ilości klas do ilości atrybutów, nie wszystkie atrybuty zostaną uwzględnione w zbudowanym modelu (w węzłach drzewa znajdzie się tylko kilka atrybutów).
* Z przeprowadzonych wcześniej badań na tych samych danych ([artykuł](http://static.dergipark.org.tr/article-download/64bb/dc31/aaee/5d17d27a14b62.pdf?)) wynika, że atrybuty Atr2, Atr6, Atr11, Atr18, Atr26, Atr40 mają największy poziom istotności (value of significance), zatem powinny znaleźć się w węzłach zbudowanego modelu.
* Spodziewana precyzja modeli (success rate) z przedziału 95-100% (podobna do otrzymanej podczas wcześniejszych badań).

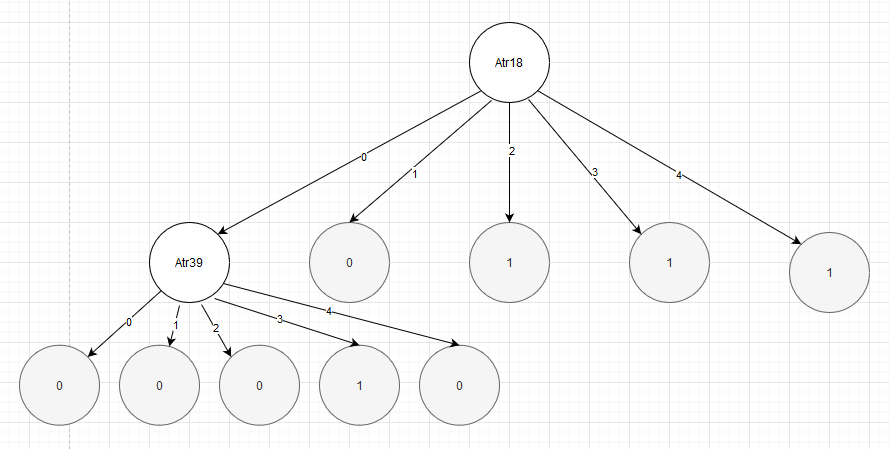
1. **Wyniki eksperymentów + omówienie**
2. Pierwszym z przeprowadzonych badań było zbudowanie modelu przy wykorzystaniu pełnego zestawu danych (bez podziału na zbiór treningowy i testowy). Poniżej wizualizacja takiego modelu.

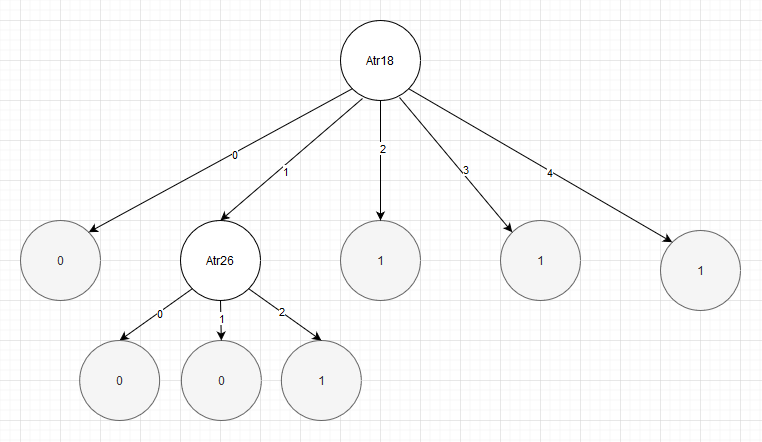


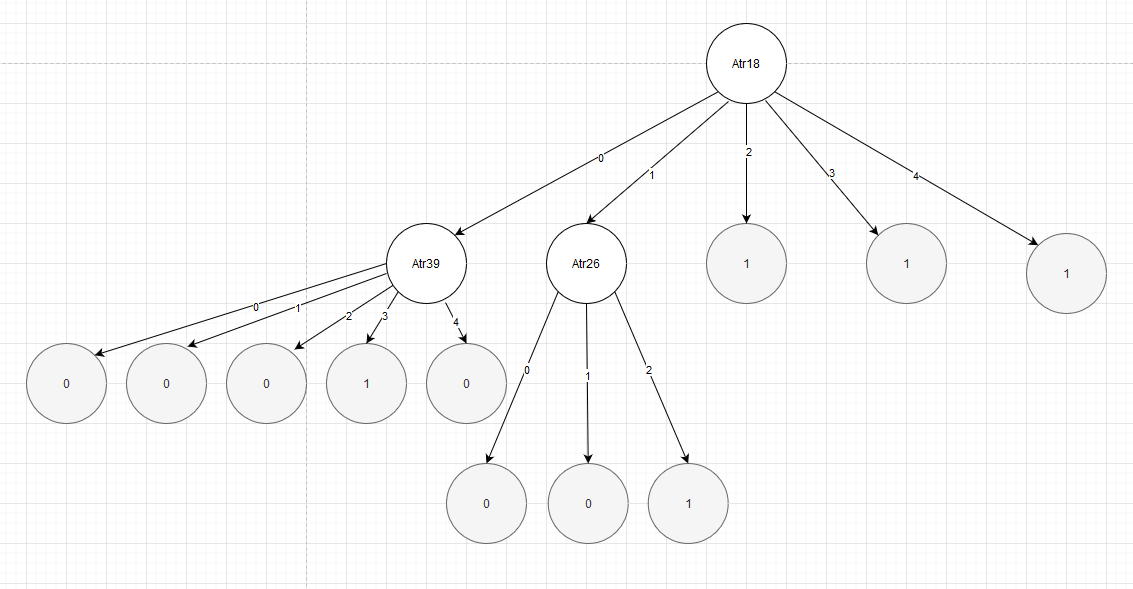
1. Żeby zapobiec problemowi przetrenowania dzieliłam dane na zbiór treningowy i testowy. By wybrać sposób podziału (stosunek ilości próbek testowych do treningowych, czyli wartość k w walidacji krzyżowej k-krotnej) badałam wartość średniej straty modelu dla k∈<2, 19>. W tabeli i na wykresie poniżej przedstawiam wartości średniej straty modelu dla różnych wartości k.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| k | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| q\_mod | 0.259 | 0.041 | 0.029 | 0.047 | 0.136 | 0.142 | 0.124 | 0.114 | 0.106 |
| k | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 |
| q\_mod | 0.077 | 0.094 | 0.088 | 0.089 | 0.101 | 0.094 | 0.1 | 0.088 | 0.076 |

1. Zdecydowałam się na k = 4, tak więc dzieliłam zbiór danych w stosunku 3:1 na dane treningowe i testowe. Poniżej 3 przykładowe modele zrealizowane w ten sposób.



Wartość straty modelu: 4,65%. Precyzja modelu: 95,35%.  Wartość straty modelu: 2,3%. Precyzja modelu: 97,7%.



Wartość straty modelu: 0% . Precyzja modelu: 100%.

Wśród modeli zbudowanych na części danych, znalazł się taki sam jak model zbudowany na całym zestawie danych. Dla takiego modelu wartość straty – jest równa 0.

1. Wyniki badania entropii atrybutów przedstawiam w poniższej tabeli – tabela zawiera wyniki 6 atrybutów o najmniejszej entropii. 50% z nich pokrywa się z atrybutami wymienionymi w artykule jako te o najwyższym stopniu istotności.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Atrybut | Atr18 | Atr20 | Atr40 | Atr17 | Atr19 | Atr11 |
| Entropia  atrybutu | 0.061 | 0.071 | 0.076 | 0.083 | 0.089 | 0.091 |

1. **Wnioski**

* W węzłach zbudowanych modeli najczęściej znajdowały się 2-3 atrybuty. Ze względu na to, że każdy z tych atrybutów mógł przyjąć aż 5 różnych wartości drzewa nie musiały byś rozbudowywane w wiele poziomów węzłów.
* Wśród najczęściej polecanych w literaturze wartości k dla k-krotnej walidacji krzyżowej znalazłam k = 5 , k = 10. Z moich badań wynika (wykres q\_mod(k)), że najskuteczniejsze były podziały dla k = 4 i k =11. Są to wartości zbliżone do ogólnie polecanych
* W modelach zbudowanych przy k = 4, najniższa precyzja modelu wynosiła 95,35%, a najwyższa 100%. Są to wartości zbliżone do otrzymanych przy wykorzystaniu sztucznych sieci neuronowych, RBF i lasu losowego, przedstawionych we wspomnianym wcześniej artykule (wartości z zakres 87,64% - 98,82%). Z tego powodu oceniam jakość zaproponowanych przeze mnie modeli jako bardzo dobrą, a samą metodę ID3 za odpowiednią do tego typu zadań.

**Instrukcja pozwalająca odtworzyć uzyskane wyniki:**

Uruchom program main.py za pomocą komendy:

python main.py

Po uruchomieniu program wypisuje uzyskane wyniki do konsoli.

Konieczne do działania programu są zaimportowane biblioteki:

pip install pandas

pip install sklearn

pip install numpy

Dodatkowo plik z danymi (w tym przypadku divorce.csv) musi znajdować się w tym samym folderze co main.py.