TAiO - projekt

Jakub Gocławski, Janusz Lewandowski, Michał Wójcik

# Moduł 1

**Generowanie zbiorów danych: zbioru uczącego i zbioru testowego.**

Generowane dane o rozkładzie normalnym z ustalonym odchyleniem standardowym. Zbiory powinny być zapisywane do formatu CSV. Dane należy wczytać i znormalizować. Następnie przekształcić dane numeryczne do postaci symbolicznej (liczby -> litery).

### Potrzebne struktury danych

**Zbiór danych**: macierz o *N* wierszach i *(C + 1)* kolumnach, gdzie: *N* - liczba elementów w zbiorze, *C* - liczba cech.

|  |  |  |  |
| --- | --- | --- | --- |
| **Klasa** | **Cecha 1** | **Cecha ...** | **Cecha *C*** |
| „7” | a | ... | d |
| „1” | b | ... | u |
| ... | ... | ... | ... |

Potrzebne dwa zbiory: zbiór uczący i zbiór testowy.

# Moduł 2

**Generowanie automatu deterministycznego**

Zakładając, że automat ma rozpoznawać *C* cech (każda o *P* wartościach symbolicznych) oraz *K* klas, należy wygenerować automat deterministyczny mający *K* stanów akceptujących oraz *P* przejść. Stanom odpowiadają klasy, do których ma zostać zaklasyfikowany element. Przejściom natomiast odpowiadają litery opisujące cechy elementów.

### Potrzebne struktury danych

**Automat:** *P* macierzy *K x K* opisujących przejścia automatu - macierze te służą do ustalania przejścia pomiędzy dwoma stanami. Stany reprezentowane jako wektor długości *K*.

Początkowo automat należy zainicjalizować losowo tak, aby w każdej kolumnie każdej macierzy była tylko jedna wartość 1. Dzięki temu automat będzie deterministyczny.

# Moduł 3

**Poprawianie automatu za pomocą PSO**

Iteracyjnie należy uruchamiać automat (klasyfikator) na zbiorze testowym i sprawdzać jego „błąd” - czyli stosunek poprawnie rozpoznanych elementów do liczby wszystkich elementów. W celu „poprawiania” automatu, należy wykorzystać funkcję PSO, która powinna zwrócić inne macierze przejść (inny automat), optymalizując funkcję błędu.