Przewidywanie rozwodów

Celem projektu jest implementacja algorytmu lasu losowego na podstawie danych dotyczących rozwodów (http://archive.ics.uci.edu/ml/datasets/Divorce+Predictors+data+set). Dane te składają się z odpowiedzi na 54 pytania dotyczące wspólnych wartości, obowiązków w małżeństwie oraz życia codziennego. Klasyfikacja jest binarna (rozwiedziony lub nie). Algorytm został zaimplementowany w języku *Python* z wykorzystaniem bibliotek: *numpy, pandas, random.*

Funkcje (plik Functions.py)**:**

- train_test_split(df, test_size):
  funkcja dzieląca zbiór danych na dwa podzbiory – treningowy na którym algorytm będzie się uczył oraz testowy w celu ewaluacji.
  Parametry:
  df – zbiór danych,
  test_size = rozmiar zbioru testowego w skali (0,1).

  Zwraca: zbiór treningowy, zbiór testowy.

- calculate_accuracy(predictions, labels):
  sprawdza poprawność predykcji dokonanej przez las losowy.
  Parametry:
  predictions – klasy zwrócona przez algorytm,
  labels – rzeczywiste klasy.

  Zwraca: procentowy udział poprawnych klasyfikacji.

- sample_data(train_df, sample_size):
  wybiera losowe obserwacje z podanego zbioru. (Będą one wykorzystane do budowy jednego z drzew lasu losowego.)
  Parametry:
  train_df – zbior z którego losowane są obiekty,
  sample_size – ilość losowanych obiektów.

  Zwraca: zbiór wylosowanych obserwacji.

- unique_check(data):
  sprawdza czy wszystkie obiekty są tej samej klasy.
  Parametry:
  data - zbiór obiektów.

  Zwraca: odpowiednio *true* lub *false*.

- classify(data):
  klasyfikuje zbiór danych według najczęściej występującej klasy.
  Parametry:

data – zbiór obiektów.

Zwraca: wynik klasyfikacji.

- find_split(data, no_of_attributes):
  funkcja wybiera podaną ilość atrybutów, która będzie wykorzystana do stworzenia pojedynczego drzewa w lesie losowym. Znajduje również atrybut maksymalizujący zdobycz informacyjną oraz wartość danego atrybutu według której obserwacje zostaną podzielone.
  Parametry:
  data – zbiór obiektów,
  no_of_attributes – liczba atrybutów wykorzystana do stworzenia pojedynczego drzewa.

  Zwraca: atrybut, punkt podziału, podzbiór spełniający warunek, podzbiór niespełniający warunku.

- entropy_calc(data):
  funkcja licząca entropię podanego podzbioru.
  Parametry:
  data – zbiór obiektów.

- global_entropy_calc(data_true, data_false):
  funkcja licząca entropię po podziale zbioru według pewnego atrybutu.
  Parametry:
  data_true – część zbioru spełniająca warunek,
  data_false – część zbioru niespełniająca warunku.

- decision_tree(df, no_of_attributes, counter=0):
  funkcja budująca drzewo decyzyjne. W każdej swojej iteracji zwraca liść, jeśli obiekty danego podzbioru należą do tej samej klasy, w przeciwnym wypadku buduje poddrzewo.
  Parametry:
  df – zbiór danych;
  no_of_attributes - liczba atrybutów wykorzystana do stworzenia pojedynczego drzewa;
  counter – wbudowany licznik iteracji
  Zwraca: drzewo decyzyjne.

- predict(observation, tree):
  klasyfikuje dany obiekt wykorzystując podane drzewo decyzyjne.
  Parametry:
  observation – pojedyncza obserwacja,
  tree – drzewo decyzyjne

  Zwraca: klasę danego obiektu.

- predict_all(test_df, tree):

klasyfikuje obiekty danego zbioru, wykorzystując funkcję *predict*.
Parametry:
test_df – zbiór danych,
tree – drzewo decyzyjne.

Zwraca: wektor predykcji dla danego zbioru.

- random_forest(train_df, forest_size, sample_size, no_of_attributes):
  funkcja tworząca las losowy.
  Parametry:
  train_df – zbiór danych,
  forest_size – wielkość lasu,
  sample_size - ilość losowanych obiektów,
  no_of_attribues – liczba atrybutów wykorzystana do stworzenia pojedynczego drzewa

- predict_forest(test_df, forest):
  funkcja wykorzystująca las losowy do klasyfikacji obiektów.
  Parametry:
  test_df – zbiór danych,
  forest – las losowy.

  Zwraca: najczęściej występującą klasę

Plik Main.py zawiera przykładową instrukcję wywołania algorytmu w pętli. Dla stu iteracji uzyskaliśmy średnią poprawność predykcji na poziomie 96%.