Гузенко А.М. Группа 7.2.

Лабораторная работа №1

Энтропия и взаимная информация

Цель

Практическая реализация методов нахождения энтропии, взаимной и собственной информации для дискретных источников и каналов передачи информации.

Задание

Написать программу, реализующую нахождение энтропии, взаимной и собственной информации, условной энтропии и других информационных характеристик для произвольного XY- ансамбля. В текстовом файле input.txt содержится матрица вероятностей $a_{ij} = p(x_i, y_j)$. Программа должна выводить в файл output.txt величины $H(X), H(Y), H(X|Y), H(Y|X), H(X,Y), I(X;Y)$. Проверить работу программы на контрольном примере.

Результат выполнения работы

1. Импортируем нужные пакеты.

```
import numpy as np
import os
import pandas
```

2. Объявим константы.

```
""" CONST """
PATH = os.path.dirname(os.path.abspath(__file__)) + '\\'
```

3. Напишем функцию для чтения файла.

```
def read_file(name):
    """ Function for reading file """
    return np.loadtxt(PATH + name)
```

4. Напишем функцию для получения значений ансамбля.

```
def get_values(matrix):
    """ Return values of our ensemble """
    def create_list(m):
        for i in range(m.shape[0]):
            yield np.sum(m[[i]])

    return [elem for elem in create_list(matrix)]
```

5. Напишем функцию для вычисления энтропии.

```
def independent_entropy(matrix):
    """ Return independent entropy of ensemble H"""
    return -1 * sum(map(lambda x: x * np.log2(x), matrix))
```

6. Напишем функцию для создания матрицы условных p.

```
def create_dependent_matrix(matrix):
    """ Create matrix with dependent values of ensemble """
    res, values = [], get_values(matrix)
    for i in range(matrix.shape[0]):
        res.append(list(map(lambda matrix_row_elem, value: matrix_row_elem / value,
matrix[i], values)))
    return np.asarray(res)
```

7. Напишем функцию для вычисления условной энтропии.

```
def dependent_entropy(matrix, matrix_dependent):
    """ Return dependent entropy like (H(X|Y), H(Y|X) """
    def calc_elements(m, m_d):
        """ Summary of y | x """
        return m * np.log2(m_d) if m_d != 0 else 0

    sum_x, sum_y = 0, 0
    for i in range(matrix.shape[0]):
        sum_x += sum(map(calc_elements, matrix[i], matrix_dependent[i]))
        sum_y += sum(map(calc_elements, matrix.transpose()[i],
matrix_dependent.transpose()[i]))
    return abs(sum_x), abs(sum_y)
```

8. Напишем функцию для вычисления общей энтропии.

```
def total_entropy(matrix):
    """ Return total entropy H(X,Y) """
    sum_x = 0
    for i in range(matrix.shape[0]):
        sum_x += sum(map(lambda matrix_elem: matrix_elem * np.log2(matrix_elem) if
matrix_elem != 0 else 0, matrix[i]))
    return -1 * sum_x
```

9. Напишем функцию для вычисления взаимной информации.

```
def mutual_information(matrix, matrix_dependent, values):
    """ Return mutal_information I(X,Y) """
    def calc_elements(m, m_d, val):
        """ Summary of y """
        if m_d != 0 and values[i] != 0:
            return m * np.log2(m_d / val)
        else:
            return 0

    sum_x = 0
    for i in range(matrix.shape[0]):
        sum_x += sum(map(calc_elements, matrix[i], matrix_dependent[i], [values[i]
for _ in range(matrix.shape[1])]))
    return sum_x
```

10. Напишем функцию для вывода в файл полученных значений.

```
def output(name, matrix, x, y, matrix_dependent):
    """ Function for output data in file """
    with open(PATH + name, 'w') as f:
        dependent_entropy_res = dependent_entropy(matrix, matrix_dependent)
        f.write(pandas.DataFrame(data={'H(X)': [independent_entropy(x)], 'H(Y)':
[independent_entropy(y)],
                                       'H(X|Y)': [dependent_entropy_res[0]], 'H(Y|X)':
[dependent_entropy_res[1]],
                                       'H(X,Y)': [total_entropy(matrix)],
                                       'I(X,Y)': [mutual_information(matrix,
matrix_dependent, x)]}).to_string())
```

11. Используем написанные функции для получения нужных значений.

```
if __name__ == '__main__':
    ensemble_matrix = read_file('input.txt')
    x_values, y_values = get_values(ensemble_matrix), get_values(ensemble_matrix)
    dependent_matrix = create_dependent_matrix(ensemble_matrix)
    output('output.txt', ensemble_matrix, x_values, y_values, dependent_matrix)
```

Результат работы

input.txt

```
0.1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.1 0.0 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.1 0.0 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.1 0.0 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.1 0.0 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.1 0.0 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.1 0.0 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.1 0.0
0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.1
```

output.txt

	H(X)	H(Y)	H(X|Y)	H(Y|X)	H(X,Y)	I(X,Y)
0	3.321928	3.321928	0.0	0.0	3.321928	3.321928

Код и файлы ввода/вывода находятся вместе с отчетом в архиве.