Programming in Python:

Final Project

Converting high-dimensional tensor and returns its projection onto a low-dimensional space

Marta Preis, nr 285129

1.The aim of the project

The goal of the project was to write a Python program for reducing data dimensionality and check the similarity of the given projection of selected data with the effect received by machine learning. There was given the high-dimensional representation of data X. Next, there was implemented a Python function which preprocesses and projects the data onto two-dimensional space, and plots the data in the low-dimensional embedding Y.

2.Realisation

Code starts with import of libraries:
*matplotlib.pyplot – collection of command style functions that make matplotlib work like MATLAB, allows to create a figure and make some changes on the plot,
*numpy – for large, multi-dimensional arrays and matrices; allows for a large collection of high-level mathematical functions and to operate on these arrays,
*mdshare – allows to download the high-dimensional data from a public FTP server at Freie Universität Berlin,
*math – access to the mathematical functions defined by the C standard,
*argparse – makes it easy to write user-friendly command-line interfaces,
*textwrap – text wrapping and filling,
*os.path – implements useful functions on pathnames,
*yaml – data serialization format designed for human readability and interaction with scripting languages,
*warnings – management of warning messages,
*scipy.stats – statistical functions of kernel-density estimation using Gaussian kernels,
and function from other library:
*sklearn.manifold.TSNE – machine learning (Manifold learning) – t-distributed Stochastic Neighbor Embedding (t-SNE).
Further code is divided for other parts where definitions of functions may be found:

- Density estimation:

Function called “density_estimation(m1, m2)” defined here allows to estimate the probability density function of a random variable in a non-parametric way. Given parameters m1, m2 by written commands allow to get X,Y,Z values.

- Converting high-dimensional tensor and returns its projection onto a low-dimensional space:

Function called “fit(Y)” defined here allows to reduce data from high dimension to low dimension. It uses global parameters and here is implemented machine learning method TSNE. Firstly, there is a linear interpolation – Y_norm. Next, the density is estimated. Then, there is created a figure, which contains density, contour lines and points.

- Main:

Function called “Main()” is a main function, which uses above functions. In the beginning there are implemented global parameters and some of them may be changed in the console. Next part sets variables from console using package “argparse”. Last part involves a check if file is available and finally fitting function.

- Executing program:

Last three lines contain functions “if”, “warnings.filterwarnings()“ and “Main()”.

To open program Anaconda console or Windows console can be used. Using Anaconda console, first step is to change directory for that one where the program is located – it can be done by using “cd” command. Second step is to write in the command line e.g. “python -W

ignore project_285129.py". In windows console it is needed to activate Anaconda base before (command for Anaconda with Python 3 default path: "c:\ProgramData\Anaconda3\Scripts\activate base" and the same commands as for Anaconda console). After that help message appears and whole code with default variables written in help message is executed. To change any of those variables it is needed to close the figure with chart and rewrite line "python -W ignore project_285129.py" with the list of described variables, space and value in the end for chosen ones e.g. "python -W ignore project_285129.py -l 500 -i 700". Two last parameters are Boolean type so writing them in the end lets them to be executed.

3.Results

Use of sklearn packages allows system to learn by itself which variables are useful. Comparing received result with the given chart in the instruction it is needed to say that there are differences.

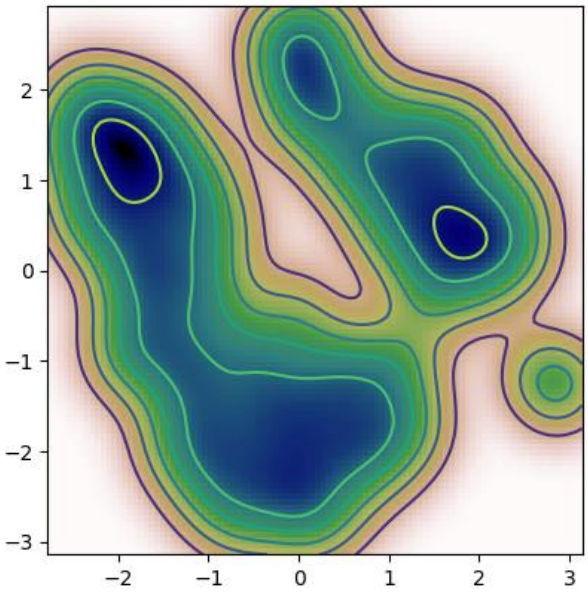

Fig.1. High-dimensional tensor projection onto a low-dimensional space with parameters sampling=100, learning rate=200, iterations number=1000, minimum gradient norm=1e-5

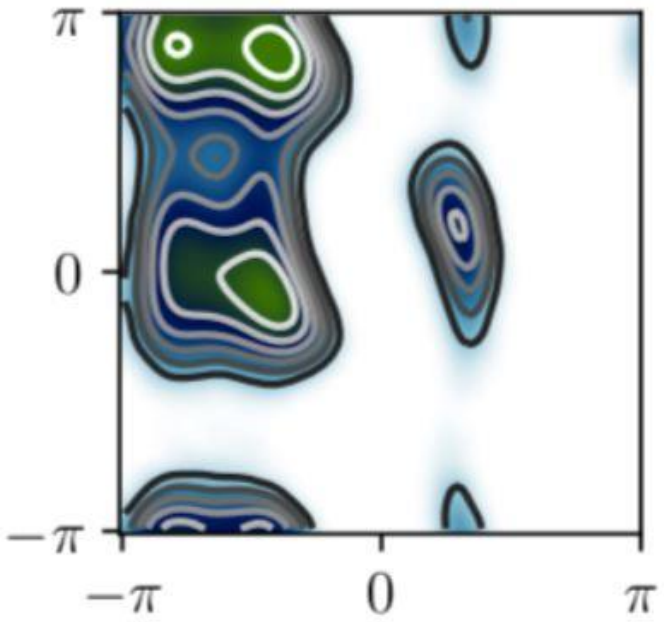

Fig.2. High-dimensional tensor projection onto a low-dimensional space given in the instruction

Algorithm learns by itself and does not know which components are useful so significant differences can be noticed. Without the knowledge about the system it is hard to get similar results. Nevertheless, machine learning becomes more and more popular tool for more efficiency, productivity, reliability and cost reduction but in that case it is better to trust experienced scientist as a supervisor to machine learning to get more preferable result.