

Data Science with Python – Assignment #3

Assignment description

This assignment contains two tasks: (1) analysis of medical expenses data with linear regression, and (2) implementation of nearest neighbors in radius (NNR) classifier.

Task #1: linear regression analysis with medical insurance dataset

A dataset with families' medical expenses is attached to this task, as well as a jupyter notebook with questions. The task is to train a linear regression model for predicting the (continuous) value of medical expenses from other features. Explore the data and read the questions, implement your solution, and answer the questions. Multiple (valid) solutions exist, strive to make your code effective and elegant.

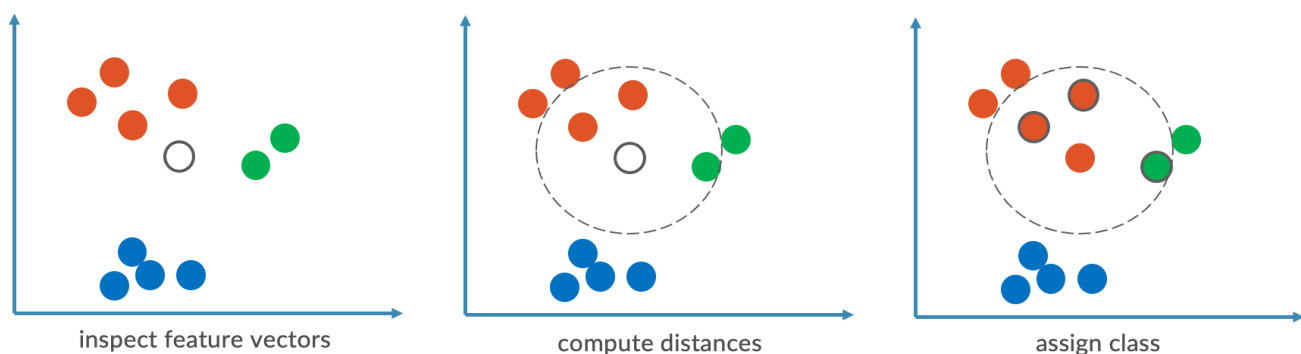
Comments:

- (1) Note that there are multiple (valid) ways to solve the same question.
- (2) Before solving the assignment, take few minutes to make sure you understand the data.
- (3) Before submission: (1) restart kernel (right click → restart kernel) and make sure your solution works as expected; (2) clear all outputs (right click → clear all outputs).

Task #2: implementation of the NNR classifier

A nearest neighbors in radius (NNR) classifier is a variation of the KNN classifier we saw in the lecture, where instead of looking for the majority vote within K nearest neighbors of an instance, it inspects all instance's neighbors in a given radius and assigns it a label according the majority vote in the radius.

Consider the below example: the new instance will be assigned the red label due to the majority of red instances in the dashed-line radius around it.



You are given two (independent) datasets, split into training, validation and test sets. The goal is to (1) find the optimal radius for a given dataset using the validation set, and (2) use this radius to assign labels on the test set, and (3) compute the accuracy (done for you in the main).

```
def classify_with_NNR(data_trn, data_vld, data_tst):  
    # implement this function  
    ...
```

Comments:

- (1) In the provided datasets: all columns but the last are features, the last column (class) is labels.
- (2) Your code should work seamlessly on any dataset of the given structure (and split into three parts): do not hardcode any dataset-specific details (e.g., file or column names, the number of classes). As a concrete example, the code should work on both datasets with the single change of names in the `config.json` file. The submission will be graded on different (unseen) dataset(s).
- (3) [Note that you are required to implement your own NNR version \(and not invoke the `RadiusNeighborsClassifier\(\)` classifier from `sklearn`\)](#). You can have a look at its implementation, but you may not find it very helpful. However, you can try it out on your data and compare the results.
- (4) Implementation obtaining roughly 96% and 58% accuracy on the two datasets: `students_loan` (two classes) and `body_performance` (four classes), respectively, can be considered a good achievement. However, the task will be graded on additional datasets (not limited to these two).
- (5) Make sure your code runtime doesn't exceed 5 minutes (invocation → results are printed).
- (6) The classifier should be implemented in PyCharm (similarly to assignment #1).

Submission

Submit a single zip file – `assignment3_XXXXXXXXXX_XXXXXXXXXX.zip`, where “XXXXXXXXXX” stands for a student id. Please specify two student ids (your and your partner's). It should include two files:

1. Your solution for task #1: `assignment3_task1.ipynb`
2. Your solution for task #2: a single `main.py` file with your solution to the task

Grading criteria include: correctness (the major part), code design, readability and documentation.

Good Luck!