CSE455/CSE552 – Machine Learning (Spring 2022)

Homework #1

Hand-in Policy: Via Teams. No late submissions will be accepted.
Collaboration Policy: No collaboration is permitted.
Grading: This homework will be graded on the scale 100.

Description: Experiments with KNN, SVM and DT on a classification data set (Bank Marketing Data - https://archive.ics.uci.edu/ml/datasets/Bank+Marketing) data.

This assignment expects you to write five different functions to test your solutions to the problem. You are to use the Python language. You will prepare a Jupyter Notebook (Google Colab) including your code and results.

- Part 1: Build a classifier based on KNN (K=4 for testing) using Euclidean distance.
  - You are expected to code the KNN classifier by yourself.
  - Report performance using an appropriate k-fold cross validation using confusion matrices on the given dataset.
  - Report the run time performance of your above tests.
- Part 2: Build a classifier based on KNN (K=4 for testing) using Manhattan distance.
  - You are expected to code the KNN classifier by yourself.
  - Report performance using an appropriate k-fold cross validation using confusion matrices on the given dataset.
  - Report the run time performance of your above tests.
- Part 3: Build a classifier based on linear SVM.
  - You may use an available implementation of SVM in Python.
  - Report performance using an appropriate k-fold cross validation using ROC curves and confusion matrices. Find the best threshold for the SVM output as described in the note by Fawcett.
  - Report the run time performance of your above tests.
- Part 4: Build a classifier based on polynomial SVM.
  - You may use an available implementation of SVM in Python.
  - Report performance using an appropriate k-fold cross validation using ROC curves and confusion matrices. Find the best threshold for the SVM output as described in the note by Fawcett.
  - Report the run time performance of your above tests.
- Part 5: Build a classifier based on DT (Decision Trees).
  - You may use an available implementation of DTs in Python.
  - Use different pruning strategies.
  - Report performance using an appropriate k-fold cross validation.
  - Report the run time performance of your above tests.
  - Write a function to convert one of your decision trees into a set of rules (i.e., extract the path to each leaf nodes).
- Part 6 (optional): Improve your search procedure in Part 1 and Part 2 using an advanced search algorithm such as kd-trees.

What to hand in: You are expected to hand in one of the following with your own comments and notes on each of the tests.

- **HW1_lastname_firstname_studentnumber_code.ipynb** (the Python notebook file containing the code and report output).

Your notebook should include something like the following:

Part 1:

Code:

Results:

Comments:

Part 2:

Code:

Results:

Comments:

Part 3:

Code:

Results:

Comments:

Part 4:

Code:

Results:

Comments:

Part 5:

Code:

Results:

Comments:

Part 6:

Code:

Results:

Comments: