CSE455/CSE552 – Machine Learning (Spring 2024)

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 in a classification (Audit - https://archive.ics.uci.edu/ml/datasets/Audit+Data) and regression (Bike Sharing – https://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset) problems.

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

- Part 1: Build a classifier based on KNN (K=3 for testing) using Euclidean distance.
  - You are expected to code the KNN classifier (including the distance function).
  - 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 regressor based on KNN (K=3 for testing) using Manhattan distance.
  - You are expected to code the KNN classifier (including the distance function).
  - Report performance using an appropriate k-fold cross validation on the given dataset.
  - Report the run time performance of your above tests.
- Part 3: Build a classifier based on the linear SVM.
  - You may use any 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 regressor based on the linear SVM.
  - You may use an available implementation of SVM in Python.
  - Report performance using an appropriate k-fold cross validation.
  - Report the run time performance of your above tests.
- Part 5: Build a classifier based on the radial basis function SVM.
  - You may use any 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 6: Build a classifier based on DT (Decision Trees).
  - You may use any available implementation of DTs in Python.
  - Experiment with two different pruning strategies (explain what you use).
  - Report performance using an appropriate k-fold cross validation.

    - 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 7: Build a regressor based on DT (Decision Trees).
    - You may use an available implementation of DTs in Python.
    - Report performance using an appropriate k-fold cross validation.
    - Write a function to convert one of your decision trees into a set of rules (i.e., extract the path to each leaf nodes).

For each solution above, report the performance of training and testing for a single fold of the 6-fold validation case.

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.zip (or rar)** including the Python notebook file containing the code and report output and its pdf version. Your zip file may contain other code or intermediate files necessary to run your solutions.

Your notebook should include something like the following: (shown for the first two parts only)

Part 1:

Code:

Results:

Comments:

Part 2:

Code:

Results:

Comments: