Assignment No. 4
EECS 658
Introduction to Machine Learning
Due: 11:59 PM, Tuesday, October 22, 2024
Submit deliverables in a single zip file to Canvas
Files in other formats (e.g., .tar) will not be graded
Name of the zip file: FirstnameLastname_Assignment4 (with your first and last name)
Name of the Assignment folder within the zip file: FirstnameLastname_Assignment4

Deliverables:

1. Copy of Rubric4.docx with your name and ID filled out (do not submit a PDF)
2. Python source code for CompareFeatureSelectionMethods
3. Screen print showing the successful execution of CompareFeatureSelectionMethods. (Copy and paste the output from the Python console screen to a Word document and PDF it).
4. For Part 2, using the PoV formula and the values from the eigenvalue matrix, show that the program calculated the PoV correctly. (see "Deliverable 4 (PoV) Example" on Canvas).
5. Answers to the following questions for CompareFeatureSelectionMethods:
    a. Based on accuracy which dimensionality reduction method, PCA, simulate annealing, or the genetic algorithm worked the best?
    b. For each of the two other methods, explain why you think it did not perform as well as the best one.
    c. Did the best dimensionality reduction method produce a better accuracy than using none (i.e. the results of Part 1)? Explain possible reasons why it did or did not.
    d. Did Part 2 produce the same set of best features as Part 3? Explain possible reasons why it did or did not.
    e. Did Part 2 produce the same set of best features as Part 4? Explain possible reasons why it did or did not.
    f. Did Part 3 produce the same set of best features as Part 4? Explain possible reasons why it did or did not.

Assignment:

- In this assignment, you will use 2-fold cross-validation of the iris data set using the Decision Tree machine learning model.
- This assignment has four parts.
- In each part (except the first one) you will use different dimensionality reduction methods on the iris data set.
- For each of the parts, the Python program should display (with a label showing the Part number):
    - Confusion matrix
    - Accuracy metric
    - List of features used to obtain the final confusion matrix and accuracy metric.
- Name the program CompareFeatureSelectionMethods

- Part 1:
    - Use the original 4 features: sepal-length, sepal-width, petal-length, and petal-width.
- Part 2:
    - Refer to the “Python Example” in the “PCA Feature Transformation” lecture slides.
    - Use PCA to transform the original 4 features (i.e., sepal-length, sepal-width, petal-length, petal-width) into 4 new features (z_1, z_2, z_3, and z_4).
    - Display the eigenvalues and eigenvectors matrices.
    - Select a subset of the transformed features, so that PoV > 0.90.
    - Display the PoV
    - Use the selected subset of transformed features to calculate the confusion matrix and accuracy metric.
- Part 3:
    - Use simulated annealing to select the best set of features from the 4 original features (i.e., sepal-length, sepal-width, petal-length, petal-width) plus the 4 transformed features (z_1, z_2, z_3, and z_4) from Part 2 (for a total of 8 features).
    - Set the iterations = 100
    - Perturb with randomly selected 1 or 2 parameters (because 1-5% of 8 is < 1)
    - c in Pr[accept] = 1
    - Use restart value (x) of 10
    - Print out for each iteration:
        - Subset of features
        - Accuracy
        - Pr[accept]
        - Random Uniform
        - Status: Improved, Accepted, Discarded, or Restart
- Part 4:
    - Use the genetic algorithm we discussed in class to select the best set of features from the 4 original features plus the 4 transformed features from Part 2 (for a total of 8 features).
    - For the initial population use the following sets of features:
        - z_1, sepal-length, sepal-width, petal-length, petal-width
        - z_1, z_2, sepal-width, petal-length, petal-width
        - z_1, z_2, z_3, sepal-width, petal-length
        - z_1, z_2, z_3, z_4, sepal-width
        - z_1, z_2, z_3, z_4, sepal-length
    - Run the algorithm for 50 generations
    - At the end of each generation, print out the features and the accuracy for the 5 best sets of features and the generation number.

Rubric for Program Comments		
Exceeds Expectations (90-100%)	**Meets Expectations (80-89%)**	**Unsatisfactory (0-79%)**
Software is adequately commented with prologue comments, comments summarizing major blocks of code, and comments on every line.	Prologue comments are present but missing some items or some major blocks of code are not commented or there are inadequate comments on each line.	Prologue comments are missing all together or there are no comments on major blocks of code or there are very few comments on each line.

Adequate Prologue Comments:

- Name of program contained in the file (e.g., EECS 658 Assignment 1)
- Brief description of the program, e.g.,
  - Check versions of Python & create ML "Hello World!" program
- Inputs (e.g., none, for a function, it would be the parameters passed to it)
- Output, e.g.,
  - Prints out the versions of Python, scipy, numpy, pandas, and sklearn
  - Prints out "Hello World!"
  - Prints out the overall accuracy of the classifier.
  - Prints out the confusion matrix.
  - Prints out the P, R, and F1 score for each of the 3 varieties of iris.
- All collaborators
- Other sources for the code ChatGPT, stackOverflow, etc.
- Author's full name
- Creation date: The date you first create the file, i.e., the date you write this comment

Adequate comments summarizing major blocks of code and comments on every line:

- Provide comments that explain what each line of code is doing.
- You may comment each line of code (e.g., using //) and/or provide a multi-line comment (e.g., using /* and */) that explains what a group of lines does.
- Multi-line comments should be detailed enough that it is clear what each line of code is doing.
- Each block of code must indicate whether you authored the code, you obtained it from one of the sources listed in the prolog, or one of your collaborators authored the code, or if it was a combination of all of these.

Collaboration and other sources for code:

- When you collaborate with other students or use other sources for the code (e.g., ChatGPT, stackOverflow):
  - Your comments must be significantly different from your collaborators.
  - More scrutiny will be applied to grading your comments in particular explaining the code "in your own words", not the source's comments (e.g., ChatGPT's comments).

- Failure to identify collaborators or other sources of code will not only result in a 0 on the assignment but will be considered an act of Academic Misconduct.
- Students who violate conduct policies will be subject to severe penalties, up through and including dismissal from the School of Engineering.