MECE 788: Applied Com. Int. for Eng. Assignment 2

By: M. Nazarahari, H. Rouhani

Winter 2021

Assignment 2

- Assignment 2 has three sections:
  - Feature Selection: Classical Methods
  - Dimension Reduction
  - Feature Selection as a Multi-Objective Optimization Problem (Solved With NSGA-II)

- For each section, sample codes will be provided for you, and you need to modify the codes to obtain the required results.

- You need to submit the finalized codes as well as a short report including the required results.
  - You don’t need to include the finalized codes in your report. Only include the qualitative answers or parameter investigation results.

- See the posted videos for more help with your assignments.

Part 1: Feature Selection, Classical Methods

- In this section, you are going to use classical feature selection techniques (in particular, Sequential Forward/Backward Search and Best Individual d Features) to reduce the dimension of a given dataset while achieving an acceptable classification accuracy.

- A feature selection problem has three main parts:
  1. **A dataset containing observations (input, output) pairs.** Each observation has a number of features (In our case: MATLAB “Body Fat Estimation” dataset and MATLAB “Breast Cancer” dataset; read more about them in MATLAB Help).
  2. **A search technique** which evaluates the accuracy of a classifier for one or a number of features toward finding the most informative features (In our case: “Sequential Forward/Backward Search” or “Best Individual d Features”).
  3. **A machine learning technique (classifier)** which uses the selected features to perform training and then return some performance metric for training/testing data (In our case: An “Artificial Neural Network”).

Part 1: Feature Selection, Best Individual d Features

- Given code named “FeatureSelection” contains the required code for loading a dataset (Section 1 of the code), defining the objective function: Artificial Neural Network (ANN) performance, and feature selection techniques.

- In particular, Section 2 of the code uses MATLAB function “sequentialfs” to perform Sequential Forward/Backward Search. When you run this section (after loading data in Section 1), you will see the process of training and testing an ANN and finally a list of selected features.
  - See MATLAB Help for more information about “sequentialfs”.

- Section 3 of “FeatureSelection” code implements another feature selection technique which uses Laplacian Scores using MATLAB function “fsulaplacian”.
  - We did not discuss this technique in our lectures. It is provided here just for learning purposes.
  - See MATLAB Help for more information about “fsulaplacian”.

Part 1: Feature Selection, Classical Methods

- **Question 1** (Section 4 of "FeatureSelection" code)
  - (code) Complete function named "BestInd_d_Features_Incomplete" to select features using "Best Individual d Features" technique.
  - Follow the procedure for "Best Individual d Features" in lecture notes as well as the provided comments in the function "BestInd_d_Features_Incomplete".
  - Inputs of "BestInd_d_Features_Incomplete":
    - **d**: Number of features that we want to select.
    - **Objective**: A function handle for our classifer. Executing "Objective" with one or a set of features will return machine learning error as a scalar.
    - **Data**: A structure containing inputs, outputs, number of features.
  - **Hint:** Use "Objective" with one input to record the performance (error between actual and predicted outputs) of the ANN. The input to "Objective" must be a binary vector (contains 0s and 1s) with same number of elements as number of features. If you want to evaluate the performance for the j-th feature, the j-th element of the binary vector must be 1 and the others must be 0.

Part 2: Dimension Reduction, Principal Component Analysis (PCA)

- Section 5 of “FeatureSelection” code uses PCA to perform dimension reduction by mapping raw features to a new space. In PCA, in contrast to feature selection techniques, you don’t need evaluate a machine learning technique performance to select the most informative features. This makes PCA more computationally efficient.

- Run Section 5 to obtain the mapped data, so later, you can compare the performance of PCA with other feature selection/dimension reduction techniques using the same ANN.

Part 2: Dimension Reduction, Kernel Principal Component Analysis (KPCA)

- MATLAB does not contain a function to perform KPCA. Thus, in Section 6 of "FeatureSelection" code, we aim to implement the KPCA using "Kernel_PCA_Incomplete".
- "Kernel_PCA_Incomplete" function contains all the operations required for performing KPCA. In fact, if you call this function with "Linear" or "Gaussian" kernel, you will get the mapped data as the output.
  - In lecture notes, we introduced 10 kernel functions (provided in the next page as well). However, only two of them ("Linear" and "Gaussian") are already implemented in "Kernel_PCA_Incomplete".

- **Question 2:** (Section 6 of "FeatureSelection" code)
  - (code) Implement the other 6 kernel functions such that the "Kernel_PCA_Incomplete" can be called with other kernel functions to obtain the mapped features.

Section 2: KPCA, Kernel Functions

① linear : $\phi(\underline{x}_i, \underline{x}_j) = (\underline{x}_i^T \underline{x}_j)^p$

② polynomial : $\phi(\underline{x}_i, \underline{x}_j) = (1 + \underline{x}_i^T \underline{x}_j)^p$

③ Gaussian $\phi(\underline{x}_i, \underline{x}_j) = \exp\left(-\frac{1}{2\sigma^2} \|\underline{x}_i - \underline{x}_j\|^2\right)$

④ Hyperbolic tangent : $\phi(\underline{x}_i, \underline{x}_j) = \tanh(\beta_0 + \beta_1 \underline{x}_i^T \underline{x}_j)$

⑤ Exponential : $\phi(\underline{x}_i, \underline{x}_j) = \exp\left(-\frac{1}{2\sigma^2} \|\underline{x}_i - \underline{x}_j\|\right)$

⑥ Laplacian : $\phi(\underline{x}_i, \underline{x}_j) = \exp\left(-\frac{1}{\sigma} \|\underline{x}_i - \underline{x}_j\|\right)$

⑦ Rational Quadratic : $\phi(\underline{x}_i, \underline{x}_j) = 1 - \frac{\|\underline{x}_i - \underline{x}_j\|^2}{\|\underline{x}_i - \underline{x}_j\|^2 + c}$

⑧ Multiquadric : $\phi(\underline{x}_i, \underline{x}_j) = \sqrt{\|\underline{x}_i - \underline{x}_j\|^2 + c}$

⑨ Power : $\phi(\underline{x}_i, \underline{x}_j) = -\|\underline{x}_i - \underline{x}_j\|^d$

⑩ log : $\phi(\underline{x}_i, \underline{x}_j) = -\log(\|\underline{x}_i - \underline{x}_j\|^d + 1)$

Parameter Kernel Function

Parts 1 and 2: Classification Error Comparison

- Section 7 of "FeatureSelection" code records the classification error (for an ANN) using the selected features (for feature selection methods) or mapped data obtained by dimension reduction techniques.

- As the same ANN is being used to evaluate errors, comparing errors can show which method is most powerful in discarding useless information.

- **Question 3:** (Section 7 of "FeatureSelection" code)
  - (report) Record the classification errors for "Body Fat" dataset in the following table and discuss the your findings:

Number of Features	d=1	d=2	d=3	d=4	d=5
Sequential Forward/Backward Search					
Laplacian Scores					
Best Individual d Features					
Principal Component Analysis					
Kernel PCA (Linear Kernel)					
Kernel PCA (Another Kernel, your choice)					
Kernel PCA (Another Kernel, your choice)					

Parts 1 and 2: Classification Error Comparison

- **Question 3, Continue:** (Section 7 of "FeatureSelection" code)
  - For Sequential Forward/Backward Search (Section 2), you cannot control the number of selected features. Thus, evaluate this method by setting value of 5 and 10 for parameter 'cv' in "sequentialfs" function and record the error in the proper column.
  - For other techniques, parameter 'd' identifies the number of selected features after feature selection or dimension reduction. Thus, to obtain the result for each column, set the value of parameter 'd' to the desired value (1 to 5).
  - For KPCA, use "Linear" (with parameter = 1) and two other kernels (your choice); therefore, your table would have three rows for three KPCAs. For the two selected kernels, you might need to try a few parameter values to achieve acceptable performance.

Part 3: Evolutionary Feature Selection with NSGA-II

- In this last section, we define the feature selection problem as a multi-objective problem and try to minimize them with NSGA-II.
  - Objective 1: Classification error for ANN → e
  - Objective 2: Number of selected features → d

- Provided code “EvoFeatSelec_NSGA_II_Incomplete” contains all the required functions to solve the above problem using NSGA-II.
  - This code operates similar to the NSGA-II code provided before. See the related lecture notes and online video for NSGA-II to remember the details, if needed.
  - Objectives are defined in “FeatureSelectionObjective” function.
  - Solution Coding: a binary vector (1s and 0s) with length equal to number of features. 1 means a feature is selected. For example, for a three feature problem, solution [0 1 0] shows that only the second feature is selected.
  - As solutions are coded as binary strings, we need to use “Binary Crossover” and “Binary Mutation”, in contrast to continuous problems we dealt before.

Part 3: Evolutionary Feature Selection with NSGA-II

- **Question 4:**
  - (code) Implement (1) Double Point Crossover and (2) Uniform Crossover in "BinaryCrossover_Incomplete" function. See lecture notes of "Genetic Algorithm" for details of these crossover operators.
    Hint: Single Point Crossover has been implemented as an example for you.

  - (code) Implement Binary Mutation in "Mutation_Incomplete" function. See lecture notes of "Genetic Algorithm" for details of this mutation operators.

  - (report) After implementing crossover and mutation operators, you can run the "EvoFeatSelec_NSGA_II_Incomplete" code (Use the provided default parameters). The results contain the best set of features for various number of features ("d") and associated classification errors. Compare the performance of evolutionary feature selection with classical methods using the table you obtained in **Question 3** and discuss the reports.