CSCI 447 — Machine Learning

Project #2

Assigned: September 11, 2020
Project Due: October 5, 2020

This assignment requires you to implement several instance-based learning algorithms to perform classification and regression on several data sets from the UCI Machine Learning Repository. In addition to implementing and testing the algorithms, you will be writing a research paper describing the results of your experiments. Be careful with how the attributes are handled. Nearest neighbor methods work best with numeric attributes, so some care will need to be taken to handle categorical (i.e., discrete) attributes.

In this project, we will continue to use 10-fold cross-validation. Ultimately, you would like the same number of examples to be in each class in each of the ten partitions. This is called "stratified" cross-validation. For example, if you have a data set of 100 points where 1/3 of the data is in one class and 2/3 of the data is in another class, you will create ten partitions of 10 examples each. Then for each of these partitions, 1/3 of the examples (around 3 or 4 points) should be from the one class, and the remaining points should be in the other class.

Note that stratification is not expected for the regression data sets. You should be sure to sample uniformly across all of the response values (i.e., targets) when creating your folds. One approach for doing that (that's not particularly random) is to sort the data on the response variable and take every tenth point for a given fold, offset by fold $\# - 1$.

With ten-fold cross-validation, you will run ten experiments where you train on nine of the partitions (so 90% of the data) and test on the remaining partition (10% of the data). You will rotate through the partitions so that each one serves as a test set exactly once. Then you will average the performance on these ten test-set partitions when you report the results.

Let's talk again about tuning. Extract 10% of the data to be used for tuning. For your training set, test against this 10% with different parameter values and pick the best model. Then apply the model that goes with those tuned values against your test set. To be specific, since you are doing 10-fold cross-validation, you first take out 10% for tuning. Then from the remaining 90% split into ten folds of 9% of the data each. For each of the five experiments, combine nine of the folds, holding out the tenth as your test set. Train on the nine folds while tuning with the 10%. Take the result and evaluate generalization ability on the held-out fold. Repeat this process ten times for each of the folds but using the same 10% for tuning.

For this assignment, you will use three classification datasets and three regression data sets that you will download from the UCI Machine Learning Repository, namely:

1. Glass [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Glass+Identification`

   The study of classification of types of glass was motivated by criminological investigation.

2. Image Segmentation [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Image+Segmentation`

   The instances were drawn randomly from a database of 7 outdoor images. The images were hand segmented to create a classification for every pixel.

3. Vote [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records`

   This data set includes votes for each of the U.S. House of Representatives Congressmen on the 16 key votes identified by the Congressional Quarterly Almanac. Don't forget that "?" does not mean the value is missing.

4. Abalone [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Abalone`

   Predicting the age of abalone from physical measurements.

5. Computer Hardware (Machine) [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Computer+Hardware`

   The estimated relative performance values were estimated by the authors using a linear regression method. That results is indicated in the data set as "ERP." **Do not use this field as a feature in your model!** Be sure to use PRP as the target response variable. The gives you a chance to see how well you can replicate the results with these two models.

6. Forest Fires [Regression]

   `https://archive.ics.uci.edu/ml/datasets/Forest+Fires`

   This is a difficult regression task, where the aim is to predict the burned area of forest fires, in the northeast region of Portugal, by using meteorological and other data .

Some of the data sets may have missing attribute values. When this occurs in low numbers, you may simply edit the corresponding values out of the data sets. For more occurrences, you should do some kind of "data imputation" where, basically, you generate a value of some kind. This can be purely random, or it can be sampled according to the conditional probability of the values occurring, given the underlying class for that example. The choice is yours, but be sure to document your choice.

Your assignment consists of the following steps:

1. Download the six (6) data sets from the UCI Machine Learning repository. You can find this repository at `http://archive.ics.uci.edu/ml/`. The data sets are also available in Brightspace.

2. Pre-process the data to ensure you are working with complete examples (i.e., no missing attribute values).

3. Implement k-nearest neighbor and be prepared to find the best k value for your experiments. You must tune k and explain in your report how you did the tuning.

4. Implement edited k-nearest neighbor. See above with respect to tuning k. On the regression problems, you should define an error threshold ϵ to determine if a prediction is correct or not. This ϵ will need to be tuned.

5. Implement condensed k-nearest neighbor. See above with respect to tuning k and ϵ.

6. Implement k-means clustering and use the cluster centroids as a reduced data set for k-NN.

7. Implement Partitioning Around Medoids for k-medoids clustering and use the medoids as a reduced data set for k-NN. Note that the k for k-medoids is different than the k for k-NN.

8. For classification, employ a plurality vote to determine the class. For regression, apply a Gaussian (radial basis function) kernel to make your prediction. You will need to tune the bandwidth σ for the Gaussian kernel.

9. Develop a hypothesis focusing on final performance of each of the chosen algorithms for each of the various problems.

10. Test each of the k-NN algorithms using at least five different values for k. When clustering, set k to equal the number of points returned from both edited nearest neighbor and condensed neighbor.

11. Write a very brief paper summarizing the results of your experiments. Your paper is required to be at least 5 pages and no more than 10 pages using the JMLR format You can find templates for this format at `http://www.jmlr.org/format/format.html`. The format is also available within Overleaf. Make sure you explain the experimental setup, the tuning process, and the final parameters used for each algorithm.

12. Your paper should contain the following elements:

    (a) Title and author name

(b) Problem statement, including hypothesis

(c) Description of your experimental approach

(d) Presentation of the results of your experiments

(e) A discussion of the behavior of your algorithms, combined with any conclusions you can draw relative to your hypothesis

(f) Summary

(g) References (Only required if you use a resource other than the course content.)

13. For the video, the following constitute minimal requirements that must be satisfied:

- The video is to be no longer than 5 minutes long.
- The video should be provided in mp4 format. Alternatively, it can be uploaded to a streaming service such as YouTube with a link provided.
- Fast forwarding is permitted through long computational cycles. Fast forwarding is *not permitted* whenever there is a voice-over or when results are being presented.
- Be sure to provide verbal commentary or explanation on all of the elements you are demonstrating.
- Show your data being split into five folds for one of the data sets.
- Demonstrate the calculation of your distance function.
- Demonstrate the calculation of your kernel function.
- Demonstrate an example of a point being classified using k-nn. Show the neighbors returned as well as the point being classified.
- Demonstrate an example of a point being regressed using k-nn. Show the neighbors returned as well as the point being predicted.
- Demonstrate an example being edited out of the training set using edited nearest neighbor.
- Demonstrate an example being added to the training set using condensed nearest neighbor.
- Show the average performance across the ten folds for each of k-nn, ENN, and CNN on a classification data set.
- Show the average performance across the ten folds for each of k-nn, ENN, and CNN on a regression data set.

14. Submit your fully documented code with the outputs from running your programs, your video, and your paper.