CSCI 447 — Machine Learning

Project #2

Assigned: September 19, 2022
Project Due: October 12, 2022

Introduction

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.

For the regression data sets, to stratify do the following. First, sort the data based on the response (i.e., target) value in the data. From the sorted data, break the data into groups of ten consecutive examples. Construct ten data sets where you draw the first item from each group for the first data set, the second item for the second data set, and so on. Basically, this process corresponds to taking 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. As before, you should apply loss functions such as 0/1-loss, precision, recall, or F1 to evaluate your algorithms.

Let's talk again about tuning. Start by extracting 10% of the data (at random) 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 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. To be a bit more explicit, your data set will be partitioned like this

$$[F1 : 9\%, \ldots, F10 : 9\%; \text{Tune} : 10\%].$$

Data Sets

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:

- Breast Cancer [Classification]

  `https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29`

  This breast cancer databases was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg.

- Glass [Classification]

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

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

- Soybean (small) [Classification]

  `https://archive.ics.uci.edu/ml/datasets/Soybean+%28Small%29`

  A small subset of the original soybean database.

- Abalone [Regression]

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

  Predicting the age of abalone from physical measurements.

- Computer Hardware [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. The gives you a chance to see how well you can replicate the results with these two models.

- 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.

Issues

Some of the data sets have missing attribute (i.e., feature) 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, it can be sampled according to the conditional probability of the values occurring, given the underlying class for that example, or it can just be the mean or median value of the feature. The choice is yours, but be sure to document your choice.

Requirements

Programming

Your assignment consists of the following steps:

- 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.
- Pre-process the data to ensure you are working with complete examples (i.e., no missing attribute values).
- 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.
- 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.
- Implement k-means clustering and use the cluster centroids as a reduced data set for k-NN.
- 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.
- Develop a hypothesis focusing on final performance of each of the chosen algorithms for each of the various problems.

- 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 edited nearest neighbor rather than tuning. Your experimental design should apply 10-fold cross validation.

Paper

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. Your paper should contain the following elements:

- Title and author name
- Problem statement, including hypothesis
- Description of your experimental approach and software design
- Presentation of the results of your experiments
- A discussion of the behavior of your algorithms, combined with any conclusions you can draw relative to your hypothesis
- Summary
- References (Only required if you use a resource other than the course content.)

Video

Create a video demonstrating the functioning of your code. This video should focus on behavior and not on walking through the code. You need to show input, data structure, and output. 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 ten 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 and its prediction.
- Demonstrate an example of a point being regressed using k-nn. Show the neighbors returned as well as the point being predicted and its prediction.
- Demonstrate an example being edited out of the training set using edited nearest neighbor.
- Demonstrate a data point being associated with a cluster while performing k-means clustering.
- Show the average performance across the ten folds for each of k-nn, ENN, and k-means k-NN on a classification data set.
- Show the average performance across the ten folds for each of k-nn, ENN, and k-means k-NN on a regression data set.

Group Work

All projects are completed in teams of two or three people. Team dynamics can be difficult in that there can be an unfair balance of work completed. As an attempt to avoid this issue, the following group work requirements are put in place.

- Submit a short document that identifies the tasks completed and percent level of effort for each member of the team in accordance with the following items. This document shall be included as part of your code submission. The level of effort will be used to determine weighting of the project grade. Specifically, if a team member does less than 40% (for a two-person team) or 25% (for a three-person team), their grade will be adjusted downward to reflect their level of effort.

- There are three major components to each project—code, paper, and video. A division of labor where one person is responsible for code, one person for the paper, and one person for the video is *not allowed*. Everyone on the team is required to contribute to all three of the components.

- The coding can be subdivided based on the individual coding requirements (e.g., data preparation, algorithms, experimental framework, analysis code etc.); however, each member of the team must implement at least one algorithm.

- The paper should be subdivided based on paper section. It is not sufficient for a member of the team to write the entire paper or just to proofread the paper. The goal is for each member of the team to contribute equal content to the final paper.

- The video should be recorded by a single person; however, all must contribute to the development of the video (e.g., script, setup for a particular item to be shown, editing of the final video). For the video, a different person should do the recording from project to project.

Submission

Submit your fully documented code with the outputs from running your programs, your video, and your paper. Your submission should include the following components and should be submitted as a "group" submission (i.e., only one submission for the entire group).

- Zip your source code files into a single zip file. Do not use alternative or archive formats like tar, gz, rar, etc. Be sure to include the team member contribution report with your code. Submit to the "P2 Code" portion of the assignment. The code must be well structured and fully commented. Code is worth 30% of the overall project grade.

- Submit a PDF of your report to the "P2 Paper" portion of the assignment. Brightspace is set up to accept PDF files only. Your report will be checked using the TurnItIn plagiarism checking system. The paper must be formatted per JMLR formatting requirements. All math should be included using the math editing capabilities of your document processing tool. Figures needs to be high quality. Do not use screenshots.

- Submit either your video or a text file containing a link to your video in the "P2 Video" portion of the assignment. Note that a file is required to be submitted, so if you are using a streaming service, remember to submit that text file. Your video is limited to five minutes maximum. Anything that appears in the video beyond five minutes will be ignored. Use the video the satisfy the requirements above, rather than "walking through" the code.