605.649 — Introduction to Machine Learning

Programming Project #2

1 Project Description

The purpose of this assignment is to give you some hands-on experience implementing a nonparametric algorithm to perform classification and regression. Specifically, you will be implementing a k-nearest neighbor classifier and regressor. In this exploration, be very 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. One way of doing that is with the Value Difference Metric.

Specifically, let $f(\mathbf{x})$ be a feature of data instance $\mathbf{x}$, where $f(\mathbf{x}) \in \{v_1, \ldots, v_k\}$. Let $C_i = \#\{f(\mathbf{x}) = v_i\}$ and $C_{i,a} = \#\{f(\mathbf{x}) = v_i \wedge \text{class} = a\}$. We then define the "distance" between feature value v_i and feature value v_j as

$$\delta(v_i, v_j) = \sum_{a=1}^{\text{num classes}} \left| \frac{C_{i,a}}{C_i} - \frac{C_{j,a}}{C_j} \right|^p .$$

Then the distances between two examples $\mathbf{x}$ and $\mathbf{y}$ is

$$D_{\text{cat}}(\mathbf{x}, \mathbf{y}) = \left(\sum_{k=1}^{d} \delta(\mathbf{x}_k, \mathbf{y}_k) \right)^{1/p} .$$

In this project, you will also be implementing both the edited nearest-neighbor and condensed nearest-neighbor algorithms. The basic approach to edited nearest-neighbor starts with the complete training set, and the editing process can be summarized as follows:

1. Consider each data point individually.
2. For each data point, use its single nearest neighbor to make a prediction.
3. If the prediction is (your choice: correct or incorrect, but don't do both), mark the data point for deletion.
4. Stop editing once performance on the held out 20% starts to degrade.

The basic approach to condensed nearest-neighbor starts with an empty condensed training set, and the condensing process can be summarized as follows:

1. Add the first data point from the training set into the condensed set.
2. Consider the remaining data points in the training set individually.
3. For each data point, attempt to predict its value using the condensed set via 1-nn.
4. If the prediction is incorrect, add the data point to the condensed set. Otherwise, move on to the next data point.
5. Make multiple passes through the data in the training set that has not been added until the condensed set stops changing.

As a final comment, it's important to note that, for the regression data sets, a "correct" prediction is determined by whether or not the prediction falls within some ϵ of the ground truth value. You need to tune this ϵ. Note that the best way to consider possible values is by examining the range of target values possible in the training set and decide on an appropriate range of error thresholds to test from there.

Also in this project (and all future projects), the experimental design we will use 5×2 cross-validation. You should have written code to support this in Project #1. 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 fifth point for a given fold.

Let's talk about tuning. Basically, you will use the process you set up in Project #1. Suppose the entire data set has 1,000 examples. Pull out 20% (i.e. 200 examples) for tuning, testing various parameter settings via 5×2 cross validation with those 200 examples. After choosing the best parameters, use them with 5×2 cross validation using only the 80% (i.e. 800 examples). Report the results from the experiment by averaging over the five held-out folds from the 80%.

2 Data Sets

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

1. Breast Cancer [Classification]

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

   This breast cancer data set was obtained from the University of Wisconsin

2. Car Evaluation [Classification]

   `https://archive.ics.uci.edu/ml/datasets/Car+Evaluation`

   The data is on evaluations of car acceptability based on price, comfort, and technical specifications.

3. Congressional 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.

4. Abalone [Regression]

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

   Predicting the age of abalone from physical measurements.

5. 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.

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.

3 Project Requirements

For this project, the following steps are required:

- Download the six (6) data sets from the UCI Machine Learning repository. You can find this repository at `http://archive.ics.uci.edu/ml/`. Again, all data sets are also available in Canvas. All of the specific URLs are also provided above.
- 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 condensed k-nearest neighbor. See above with respect to tuning k and ϵ.
- For classification, employ a plurality vote to determine the class. For regression, apply a Gaussian (radial basis function) kernel to make your prediction. In particular, you should use the kernel $K(\mathbf{x}, \mathbf{x}_q) = \exp(-\gamma||\mathbf{x} - \mathbf{x}_q||_2)$, where $||\cdot||_2$ indicates the L2 norm (i.e. Euclidean distance in this case). You will need to tune the bandwidth $\gamma = 1/\sigma$.
- Run your algorithms on each of the data sets. These runs should be done with 5×2 cross-validation so you can compare your results statistically. You can use classification error or mean squared error (as appropriate) for your loss function.
- Write a very brief paper that incorporates the following elements, 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.
  1. Title and author name
  2. Problem statement, including hypothesis, projecting how you expect each algorithm to perform
  3. Brief description of your experimental approach, including any assumptions made with your algorithms
  4. Presentation of the results of your experiments
  5. A discussion of the behavior of your algorithms, combined with any conclusions you can draw
  6. Conclusion
  7. References (Only required if you use a resource other than the course content.)
- Submit your fully documented code, the outputs from running your programs, and your paper.
- 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.

4 Project Evaluation

Your grade will be broken down as follows:

- Code structure – 10%
- Code documentation/commenting – 10%
- Proper functioning of your code, as illustrated by a 5 minute video – 30%
- Summary paper – 50%