605.449 — Introduction to Machine Learning

Programming Project #2

Due: June 20, 2019

The purpose of this assignment is to give you an introduction to unsupervised learning by implementing feature selection with a clustering algorithm. Specifically, you will implement StepwiseForwardSelection, or SFS (introduced in Module 03). For extra credit, you may also implement GeneticAlgorithmSelection, or GAS (also introduced in Module 03). Since these are wrapper methods, you will need to test these algorithms using another algorithm as well. For that, you will use the Naïve Bayes algorithm you implemented in Project 1. You will then choose to implement one of the following two clustering algorithms — k-Means and HAC — both of which were introduced in Module 04. For extra credit, you may implement both. To evaluate HAC, grow the tree until you get k clusters. You should evaluate these algorithms for $k =$ the number of classes in the data set. Specifically, the Silhouette Coefficient should be used for your evaluation.

For kicks, you will also look at the classes that fall in each of the clusters to determine how well clustering works as a classifier. In other words, for each cluster, determine the predominant class and assign that class to the cluster. Then calculate the accuracy of the clustering-based classifier.

The silhouette coefficient is calculated as follows.

1. For data point $\mathbf{x}_i$ in cluster $\mathbf{c}_j$, calculate the average distance to all other objects in $\mathbf{c}_j$.

$$a_i = \frac{1}{m_i} \sum_{\mathbf{x}_k \in \mathbf{c}_j} \delta(\mathbf{x}_i, \mathbf{x}_k).$$

2. For data point $\mathbf{x}_i$ and any cluster $\mathbf{c}_j$ that does not contain $\mathbf{x}_i$, find the average distance to all of the points in that other cluster. Then return the minimum such value over all of the clusters. For this, denote the cluster containing $\mathbf{x}_i$ as $\mathbf{c}(\mathbf{x}_i)$.

$$b_i = \min_{\mathbf{c}_j \in \mathbf{C} \setminus \{\mathbf{c}(\mathbf{x}_i)\}} \left\{ \frac{1}{m_k} \sum_{\mathbf{x}_k \in \mathbf{c}_j} \delta(\mathbf{x}_i, \mathbf{x}_j) \right\}.$$

3. For data point $\mathbf{x}_i$ calculate the silhouette coefficient.

$$s_i = \frac{b_i - a_i}{\max\{a_i, b_i\}}.$$

4. To evaluate the overall clustering of the data, find the average silhouette for the data set.

$$sil(\mathbf{C}) = \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x}_i \in \mathcal{D}} s_i.$$

This provides a measure between -1 and $+1$ where a negative value indicates the average distance to points within the cluster is greater than the minimum average distance to point in other clusters. This indicates the clusters are not well separated. Ideally, we would like all silhouette coefficients to be positive ($a_i < b_i$), and we would like a_i to be as close to zero as possible. When this occurs, the coefficient approaches 1.0.

For this assignment, you will use three datasets that you will download from the UCI Machine Learning Repository, namely:

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

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

2. Iris — `https://archive.ics.uci.edu/ml/datasets/Iris`

   The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant.

3. Spambase — `https://archive.ics.uci.edu/ml/datasets/Spambase`

   This collection of spam e-mails came from a postmaster and individuals who had filed spam. This is a two-class problem with a large number of attributes and a large number of instances.

As with the prior assignment, some of the data sets 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.

Our ability to handle large data sets is becoming more and more important, and one of the data sets has been selected to get you thinking about how to handle such data. Specifically, for the Spambase data set, some attention should be given to optimizing the feature selection process. One approach is to apply the feature selection on a subset of the data, rather than the full data set. While there is some risk associated with missing cluster boundaries this way, it can greatly speed up the feature selection process. Note that you still need to apply the final set of features for clustering the entire data set.

For this project, the following steps are required:

- Download the three (3) data sets from the UCI Machine Learning repository. You can find this repository at `http://archive.ics.uci.edu/ml/`. All of the specific URLs are also provided above.
- Pre-process each data set as necessary to handle missing data.
- Implement SFS with the loss function defined above.
- Implement either k-means or HAC.
- Extra Credit:
  - For up to 15 additional points, implement both k-means and HAC.
  - For up to 15 additional points, implement GAS for feature selection, in addition to SFS.
  - Run experiments on all relevant combinations of algorithms.
- Run your algorithms on each of the data sets.
  1. Do feature selection using Naïve Bayes for the classifier.
  2. Take the results of feature selection and cluster the data.
  3. Select k clusters (where $k =$ the number of classes).
  4. Evaluate the clusters using the Silhoutte Coefficient.
  5. Label each cluster according to the predominant class of the data in the cluster. If more than one cluster has the same predominant class, apply a reasonable tie-breaker.
  6. Compare the classification accuracy of Naïve Bayes to your cluster-based classifiers.
- Write a very brief paper that incorporates the following elements, summarizing the results of your experiments. You should also output the summary statistics on clustering performance.
  1. Title and author name
  2. A brief, one paragraph abstract summarizing the results of the experiments
  3. Problem statement, including hypothesis, projecting how you expect each algorithm to perform
  4. Brief description of algorithms implemented
  5. Brief description of your experimental approach
  6. Presentation of the results of your experiments
  7. A discussion of the behavior of your algorithms, combined with any conclusions you can draw
  8. Summary

9. References (you should have at least one reference related to each of the algorithms implemented, a reference to the data sources, and any other references you consider to be relevant)

- Submit your fully documented code, the outputs from running your programs, and your paper. 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%