Feature Selection with Clustering Algorithms (CS605.449 – Project 2)

By Max Robinson

Abstract

In this experiment the performance of two wrapper feature selection algorithms, Stepwise Forward Selection (SFS) and a Genetic Algorithm (GA), were compared using two different clustering algorithms. The clustering algorithms used were K-Means and Hierarchical Agglomerative Clustering. These algorithms were run on three different data sets of differing size and feature length. The two clustering algorithms were also compared to each other on each data set. The result of the experiment showed that the SFS algorithm typically performed worse on the smaller data set, though it produced fewer selected features. In addition, the results showed that again on smaller data sets, K-Means typically performed better than HAC, while on large data sets, HAC out performed K-Means.

Problem

The problem that being investigated in this experiment is how well two different feature selection algorithms, Stepwise Forward Selection (SFS) and a Genetic Algorithm (GA), perform when using different clustering algorithms to determine the fitness of the selected features. The clustering algorithms used are K-Means, and Hierarchical Agglomerative clustering (HAC). The data sets used for selecting features and clustering on as follows: Glass Identification, Iris Variety, and Spam E-Mail Database.

For this experiment, I hypothesize that the SFS algorithm will select fewer features but have lower overall Fisher Scores for each clustering algorithm than the GA. In addition, I expect that the K-Means clustering Algorithm will have a higher Fisher Score on each dataset than HAC on their respective selected feature sets, due to the fairly simplistic nature of most of the data sets, and the large number of zero elements in the Spam database set.

SFS I expect to produce smaller selected feature sets because of the greediness of the algorithm. The algorithm starts from a scratch and builds up a feature set from there. This lends itself to stopping early, or finding a local maxim earlier on in its search for features. The GA starts with randomly selected features and modifies the set from there. On data sets with larger numbers of features, this increases the likelihood for a feature set to be larger.

Algorithm Implementation

Feature Selection

Stepwise Forward Selection

Stepwise Forward Selection (SFS) is a greedy search algorithm that is used to find which combination of features produce the best "fitness" or score given some scoring mechanism. The algorithms beings with

an empty set of features and then iteratively adds features to the selected feature set based on if the score being optimized has increased given the addition of a new feature.

This algorithm does a linear search through each feature that has not been selected. For each feature, it is added to the features that have already been selected and then evaluates how good that new set of features is. After all possible new features have been added, if the overall best score has increased the feature responsible for the increase in score is added to the selected features. If the score does not improve, the algorithm returns the features that were found

The evaluation process in this experiment is actually running one of the two clustering algorithms and then calculating the Fisher score on the clusters that have been created. This makes the SFS algorithm what is called a wrapper feature selector since it wraps around another learning algorithm to optimize these features.

Genetic Algorithm

The Genetic Algorithm (GA) does not implement a greedy search like SFS. GA is instead a local search algorithm that is meant to bypass some of the restrictions that greedy searches can have and to have a better chance of finding better local maxima or minima than some greedy search algorithms.

GA works by creating a "gene" that represents the data that you are searching across. In this case, which features to select for a given data set. A population of genes is then randomly initialized. Each gene is evaluated and given a fitness score. These fitness scores are then used to probabilistically select the which genes from the population will be selected for "reproduction".

Reproduction is where two genes are selected and then with some probability, the genes are mixed together using a crossover mechanism. Crossover selects a point in the gene to then swap parts of the two genes. For example, I have a gene [0,0,1,1] and [1,1,0,0]. If the crossover point is in between position one and two, the genes would then the resulting genes would look like [0,0,0,0] and [1,1,1,1].

Selecting genes for reproduction continues until an entire new population of the same size has been created. For this implementation, a tournament style selection was implemented to select genes for reproduction for performance reasons. Tournament selection works by selecting a constant number of parents with uniform randomness from the population, and then selecting the best gene from those randomly selected.

For this experiment, a population size of 25 was chosen with the number of generations for the GA to run being 20. The crossover rate was .8 and the mutation rate was .6.

Clustering

K-Means

The K-Means algorithm is a partitioning clustering algorithm that is limited to partitioning convex regions. The algorithm also assumes that the data being clustered came from some Gaussian distribution. K-Means uses Expectation Maximization as the underlying mechanism by which it clusters data together. K-Means works by selecting k random means in the data and then slowly adjusts those means based on how far away it is from the points that are closest to that mean.

The algorithm starts with the k random means. It then finds all of the data points that are closets to each mean. Each data point is then assigned to a cluster based on which mean it is closest too. Once all of the

data has been assigned a cluster, the means are recalculated which is the, maximization step in expectation maximization. This continues until the means stop changing. For this experiment the means had to stop changing by .00001 to conclude the adjustments. These are then the means or centroids of the clusters and also define the clusters.

Hierarchical Agglomerative Clustering

The Hierarchical Agglomerative Clustering (HAC) algorithm is an unsupervised, bottom up, polythetic clustering algorithm. HAC is bottom up because of how it starts clustering points together. HAC starts by treating each individual data point as a cluster. From there, it finds the closest two points in the data and places those two points into a cluster. This cluster then "replaces" those two points in the data and the distance from the cluster to every other point in the data is then calculated. For this experiment, single linkage HAC was implemented. Single linkage HAC calculates the new distance from the cluster to every other point by using the minimum distance from either of the points that were combined to the other data point ("10.2 – Example"). One new cluster is created each iteration and the clustering stops once k clusters have been found, where k is the desired number of clusters.

Scoring

Fisher Score

For this experiment, the Fisher Score was used, which is a variation of Fisher's LDA score, to describe and put a metric on how well a set of data is clustered. The Fisher Score is defined as follows:

$$L = \frac{w^T(\sum_{i=1}^{k}(m_i - m)(m_i - m)^T)w}{kw^T\left(\sum_{i=1}^{k}\sum_i \quad\right)w}$$

What this equation does, is it uses a vector w and multiplies it by the matrix that is created by finding the squared difference between a mean of means vector m and every other mean vector m_i. This distills the numerator down into a scalar value that describes the between class scatter, taking into account the features that were used, w. This is how well separated the classes are from one another. The denominator describes the within class scatter by finding the sum of the covariance matrices for each cluster. The within class scatter describes how tightly clustered the data is within a cluster. If the data is not clustered very tightly together in a class, then the overall score is penalized.

Experimental Approach

The approach for this experiment was two fold. The first part was done through data preprocessing, and the second part was done through how the experiments were run. The metric used for comparison of the clustering algorithms and thus which features to select is the Fisher Score, where larger is better.

Data Pre-processing

The data pre-processing for these data sets were minimal. There were no missing values for any items in the data sets and as such little was needed to be done to the data set other than ensure the class label was the last element in each data point.

Feature Selection and Clustering

To get results that can be compared to each other, each feature selection algorithm was run with each clustering algorithm on each set of data. For instance, SFS was used with K Means on the Iris data set. This produced results where each feature selection algorithm selected two sets of features for each data

set, one for each clustering algorithm that was used. The output from the algorithms was which features were selected, the data points in each cluster, and the Fisher score for each the selected set of features. For each data set, the number of clusters was determined by the number of classes in the data set.

For the smaller datasets, iris and glass, each experiment that was run where the feature selection algorithm used all of the data to pass on to the clustering algorithm to receive a clustering or "train" the clustering algorithms. Either a set of means or a set of clustered points was returned from KMeans and HAC respectively. The returned "models" were then used to evaluate the clusters using the entire set of data to evaluate the model. This evaluation is the score returned by the Fisher Score.

For the largest data set, the spam e-mail database, a fraction of the data was used to "train" the clustering algorithms during feature selection. When evaluating each feature, based on this learned model, the entire set of data was used. This was done by taking each of the models, transforming them into a vector of means in the case of HAC, and then clustering the rest of the data based on these means. This provided a full clustering of all of the data points for the largest data set for evaluation. The Fisher Score requires means of clusters to be calculated any ways in order to score the clusters, so HAC would have its means computed from the clusters as a result. For each experiment using the larger dataset, a tenth of the data was used for training using SFS and hundredth of the data was used for training GA, and all of the data was used for evaluation.

The reason for this was for performance. With a very large data set such as the Spam E-mail Database, it became very difficult to use all of the data for selecting features. It simply took exorbitant amounts of time to get all of the data through the algorithm.

Results

After the running the experiment as described in the above sections, the resulting Fisher score for each data set with each combination of feature selection and clustering algorithm is shown in Table 1. The Feature Overlap column is calculated and displayed as the Union of the feature sets and the difference of the feature sets. Ex, Union and Difference.

Fisher Score Table	Feature Selection Algorithm	Number of Clusters	KMeans	Features Selected KMeans	HAC	Features Selected HAC
Iris	SFS	3	13.23388	2, 3	3.96399	3
	GA	3	13.27508	0, 2, 3	6.98400	1, 2, 3
Glass	SFS	6	128.62518	6, 9	86.01941	6, 9
	GA	6	42705.66580	0, 1, 2, 3, 5, 6, 7, 9	73452.44472	0, 2, 3, 5, 7, 8, 9
Spambase	SFS	2	4486.60050	1, 46	7683.64095	19, 46
	GA	2	885.29659	0, 3, 6, 7, 9, 12, 14, 16, 17, 19, 21, 29, 30, 32, 33, 37, 43, 47, 49, 50	1308.52097	0, 1, 2, 3, 4, 5, 9, 12, 13, 14, 15, 18, 25, 27, 31, 37, 38, 39, 41, 42, 44, 50

Table 1. Fisher Score for Selected Features per Feature Selection Algorithm and Clustering Algorithm

Behavior

The behavior of each feature selection algorithm in terms of the number of features selected was fairly predictable. Since SFS is a greedy algorithm that starts with no selected features, it can be fairly easy for it to hit a local maxima early on and miss later permutations that might provide better scores but uses a feature that did not initially provide a high score, or it could not find a large permutation where the score only increased by having each feature.
The GA also performed as expected. More features where selected on every data set for GA than SFS. On the smaller data sets the GA also had a much higher Fisher Score with each respective clustering algorithm than SFS did. This most likely came from the ability for the GA to find permutations of features where the combination of all of the features yielded a better insight into the data than the features could individually or in another combination. This would also explain why there is a larger gap in fisher score in between SFS and GA on the glass data set, than the on the iris data set, since the Glass data set has more features.

On the largest data set, the tables turned and SFS did significantly better than the GA did. The SFS had high fisher scores compared to the GA on both clustering algorithms. This seemed a little strange given the trend of the first two data sets. While it might be the case the SFS algorithm would find better features in an overly large data set both in feature length and in number of data points, I believe it has more to do with how the experiment was run on that particular data set.

For the Spam Email Database data set, a training set of data was used to build the clusters for each feature on, and then the evaluation of those clusters were done using the entire data set, per feature selection. For SFS, it used a tenth of the data for training / feature selection. For GA it used only a hundredth of the data for training / feature selection. While the difference in data used for each experiment was necessary for run time constraints, I believe that it skews the results. As a result it seems that there was an error in the experiment method. To re-evaluate to see if this trend is correct, re-running this experiment where both algorithms had access to the same amount of data for feature selection would prove more useful, given there is enough time to run the experiment.

When comparing the clustering algorithms to each other, the results were a little surprising. K-Means did perform better where the number of features selected were small, and the number of data points were relatively few. This covers mostly the SFS algorithm for Glass and Iris data sets and the GA algorithm on the Iris data. These are rows 1-3 in Table 1. For larger feature sets, and for more data points, K-Means falls behind HAC. HAC scored better in these scenarios. I believe the reason for this is due to HAC's ability to not react quite so strongly to noise. K-Means can be influenced by noise, and in larger feature spaces it is possible to have much more noise per feature.

Summary

Overall, the feature selection algorithms performed closely to what was hypothesized. SFS yielded fewer features selected, but at the cost of a lower Fisher Score. The GA had usually higher Fisher Scores, but at the cost of more features being selected thus increasing the complexity and often run time, though only shown anecdotally. When comparing the clustering algorithms, the K-Means algorithm did well with a small feature space and fewer data points but was over shadowed by HAC on more complex feature selections and data size.

References

Algorithms

HAC

"10.2 - Example: Agglomerative Hierarchical Clustering." 10.2 - Example: Agglomerative Hierarchical Clustering | STAT 555, Pennsylvania State University, 2017, onlinecourses.science.psu.edu/stat555/node/86.

Data Sources

Glass — https://archive.ics.uci.edu/ml/datasets/Glass+Identification
Iris — https://archive.ics.uci.edu/ml/datasets/Iris
Spambase — https://archive.ics.uci.edu/ml/datasets/Spambase