**Introduction**

A human genome typically contains close to 25000 genes. The genome includes both the [genes](http://en.wikipedia.org/wiki/Gene) and the [non-coding sequences](http://en.wikipedia.org/wiki/Non-coding_DNA) of the DNA. Microarray technology evolved from [Southern blotting](http://en.wikipedia.org/wiki/Southern_blotting), where fragmented DNA is attached to a [substrate](http://en.wikipedia.org/wiki/Substrate_%28biochemistry%29) and then probed with a known gene or fragment. The goal of a microarray experiment is to simultaneously measure the expression levels of thousands of genes in a collection of cells.

Microarray is a powerful tool for genome analysis. It gives the global view of the genome analysis in a single experiment. Data analysis in the Microarray is a vital part as this part influences the final result. Each microarray study comprises of multiple microarray experiments that would give tens of thousands of data points. It is hard to analyze such large volumes of data. In general, greater the volume of data, more chances arises for erroneous results. Handling such large volumes of data requires high end computational infrastructures and programs that can handle multiple data formats.

Selection of relevant genes for sample classification is a common task in most gene expression studies, where researchers try to identify the smallest possible set of genes that can still achieve good predictive performance. This is crucial for designing future experiments and for diagnostic purposes in clinical practice.

Many gene selection approaches use univariate (gene-by-gene) rankings of gene relevance and arbitrary thresholds to select the number of genes, can only be applied to two-class problems, and use gene selection ranking criteria unrelated to the classification algorithm. In contrast, classification algorithms such as recursive feature elimination SVM and random forest are well suited for microarray data: they show excellent performance even when most predictive variables are noise, can be used when the number of variables is much larger than the number of observations and in problems involving more than two classes [1]. It is important to determine which algorithm is best suited for selecting the best set of genes from a gene expression dataset.

The main goal of our project was to compare the performance of different feature selection algorithms based on the criteria of minimum number of genes selected and maximum classification accuracy.

There are three different feature selection schemes as shown in Figure 1. that are generally used for microarray data analysis. The Filter method uses some statistical criteria such as p-value from t-test to select relevant genes and these genes are then used to build the model. The wrapper method such as binary Particle Swarm Optimization uses different combinations of subsets of features and selects the best model. The embedded methods have built-in variable selection for instance lasso that uses a penalty function and gives a value of greater than 0 if a feature is relevant, and 0 otherwise.

We focused on comparing the performance of F-test, T-test, Kruskal test, Elastic net, Lasso, rfe-SVM, Random Forest, and Binary PSO in this study.

Figure1. Feature selection schemes

Algorithms and Implementation

**1) Pattern Selection based on classification** (implemented in R):

PSO Algorithm:

Basic idea: Particles x[i][j] are 2D arrays where index i is the particle and index j is the feature. If x[i][j]=1, then the feature j is selected, otherwise it is not selected.

Initializaiton

1)Initially every particle xi has p randomly selected features where p<d

2) After time step 1, the features of a particle are set to 0, or 1 depending on its velocity. If the S measure <rand() then feature is not selected, otherwise it is selected.

3)

A new feature selection algorithm based on a consensus scoring methodology was implemented. The algorithm is divided into three steps. The first two steps use the CMA package implemented in R, and the last step uses a particle swarm optimization based search strategy to find the optimal set of features that would allow maximum classification accuracy.

The user uploaded dataset is first randomly divided into distinct training and test sets based on the splitting percentage provided by the user (default: 60% train, 40%test).

In the **first step** of the feature selection process, a ranked list of features is generated by performing a t.test on the training dataset. Only the top X number of features (default 150; user defined) are retained for further analysis.

The **second** step uses one or more feature selection algorithms selected by the user to determine a consensus set of best features. The user can select t.test, f.test, kruskal.test, recursive feature elimination, random forest, Welch.test, Wilcox.test, lasso, elasticnet, and boosting. For every selected algorithm, the number of features is iteratively increased from 1 to X (default 150) in increments of delta (default 5) till there is no improvement in the 10fold cross validation accuracy for I number of iterations (default 10). This gives m subsets of top ranked features where m is the number of selected feature selection algorithms. A consensus score of a feature is defined as the number of subsets that include that feature, and its value ranges from 0 to m.



In the **final** step, the consensus subset of features selected by at least M\_min (default=100%) algorithms is then used as an input for the PSO algorithm. The PSO algorithm is designed to avoid particles from getting stuck in local optima. The search criteria (local or global search) depends on the rank of a particle. In other words, the best particle will have a narrow or local search space while the worst particle will have a much broader or a global search space controlled by the inertia variable in the velocity update equation of the PSO algorithm.

**2) Pattern Selection based on regression** (implemented in R):

A new feature selection algorithm that uses PSO based stochastic search to select optimal set of features and a SVM-regresssion method for evaluating the fitness or the mean squared error of the selected features. The user uploaded dataset is first randomly divided into distinct training and test sets based on the splitting percentage provided by the user (default: 60% train, 40%test).

The PSO algorithm is designed to avoid particles from getting stuck in local optima. The search criteria (local or global search) depends on the rank of a particle. In other words, the best particle will have a narrow or local search space while the worst particle will have a much broader or a global search space controlled by the inertia variable in the velocity update equation of the PSO algorithm.

3) Heterogeneous population:

* Split the training set into 10 sets of 1000 bootstrapped samples.
* Perform cross-validation on the nth set and use the remaining 9 sets for validation
* Get the best set of features for each set
* Get the consensus features that are selected in each set

**Datasets**

We used three publicly available gene-expression datasets for our study. The three datasets varied in their level of complexity. The NCI60 dataset had 60 training samples belonging to 9 classes, but there was no test data, and we could not split the dataset due to very few samples. The SRBCT dataset and Leukemia dataset had both training and test data.

All datasets had number of genes >> number of samples. Table 1 summarizes the details of the three datasets.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Dataset** | **Description** | **Number of Samples** | **Number of genes** | **Classes** | **Source of dataset** |
| NCI60 –RNA HU6800 | Gene expression data of 9 cell lines obtained from 60 cancer patients. **Already processed and filtered.** | Train: 60 | 6112 | 9 | Staunton et al. **Chemosensitivity Prediction by Transcriptional Profiling** Proc Natl Acad Sci USA 2001 Sep 11;98(19):10787-92 <http://discover.nci.nih.gov/cellminer/loadDownload.do> |
| Leukemia | Acute myeloid leukemia (AML) and acute lymphoblastic leukemia (ALL) gene expression data. **Processed, but not filtered.** | Train: 38  Test: 34 | 7130 | 2 | Golub et al. **Molecular Classification of Cancer:Class Discovery and Class Prediction by gene expression profiling** *Science* 15 October 1999: Vol. 286. no. 5439, pp. 531 - 537 <http://www.broad.mit.edu/cgi-bin/cancer/publications/pub_paper.cgi?mode=view&paper_id=43> |
| Small round blue cell tumors (SRBCT) | Gene expression data of 4 tumors (neuroblastoma, rhabdomyosarcoma, Hodgkin lymphoma, and the Ewing family tumor | Train: 63  Test: 20 | 2098 | 4 | Khan et al. **Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks**  <http://stat.ethz.ch/%7Edettling/bagboost.html> |

Table1. Summary of the three datasets used in our study

**Pre-processing**

The NCI60 dataset and SRBCT dataset were already processed and available for download from the source listed in Table 1.

The leukemia dataset was processed according to the guidelines in the Golub et al. paper. We first performed thresholding by using 16000 and 100 as the upper limit and lower limit for the intensity values in the dataset. The dataset was then filtered by excluding the genes that varied by less than 5-fold and 500 units across training cell lines. Finally, the dataset was transformed to log (base10).

This reduced the number of genes to 3051 for the Leukemia dataset.

**Implementation and R packages used**

We used the Classification for Microarrays (CMA) package - a comprehensive Bioconductor package for supervised classification with high dimensional data [2]. The package has built-in functions for performing feature selection using F-test, T-test, Elasticnet, Lasso, rfeSVM, Kruskal test, and Random Forest. We performed feature selection and cross-validation using the knn classifier from CMA package. The function has in-built feature to tune the value of k to be used.

**Usage:** http://www.bioconductor.org/packages/2.3/bioc/manuals/CMA/man/CMA.pdf

Step 1) Generate learningsets for training

set.seed(111)

lset <- GenerateLearningsets(class labels, method = c("LOOCV", "CV", "MCCV", "bootstrap"), fold, strat =TRUE)

Step 2). Perform gene selection

selttest <- GeneSelection(training feature set , class labels, learningsets = lset, method,=c("t.test", “kruskal.test”, “rf”, “elasticnet”, “lasso”, “rfesvm”,. Or “f.test”), scheme=c(“one-vs-all”, “pairwise”, or “multiclass”)

The multiclass scheme was used only for kruskal.test and f.test

Step 3) Cross-validation

knn1 <- classification(golubX, golubY, learningsets = lset, genesel = selttest,

tuneres = tunek, nbgene = 20, classifier = knnCMA)

The gene selection was performed for n=1 to 150, where n=number of genes to be selected. The training and testing datasets were modified according to the genes selected and then the testing error was calculated.

We used Leave-one-out cross-validation and a tuned knn classifier to train our model. The model was evaluated on a test dataset using knn classifier.

We implemented Binary Particle Swarm Optimization algorithm in R based on the following paper:

Cheng et al. **A two-stage feature selection method for gene expression data** Journal of Integrative Biology Volume 13 Number 2 2009

**Results and Analysis**

**Leukemia Dataset**

We determined the affect of number of genes using knn on the cross-validation accuracy for all the algorithms except binary PSO. The binary PSO selects global optimal minimum number of genes that give maximum accuracy so it was not considered in this experiment. We used leave one out cross-validation method. A lot of variation in the performance of the 7 algorithms was observed for the first 100 genes in all the three datasets. Table 2 and Figure 2 show the results for Leukemia dataset.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Num genes** | **T-test** | **F-test** | **Kruskal** | **Elasticnet** | **Lasso** | **rfeSVM** | **Random Forest** |
| 1 | 92.1 | 92.1 | 94.7 | 89.5 | 86.8 | 92,1 | 76.3 |
| 3 | 84.2 | 84.2 | 100 | 92.1 | 94.7 | 86.8 | 100 |
| 5 | 94.7 | 94.7 | 100 | 94.7 | 97.4 | 92.1 | 100 |
| 10 | 94.7 | 94.7 | 97.4 | 94.7 | 94.7 | 92.1 | 100 |
| 15 | 94.7 | 94.7 | 97.4 | 94.7 | 94.7 | 92.1 | 97.4 |
| 25 | 97.4 | 97.4 | 100 | 94.7 | 94.7 | 92.1 | 97l.4 |
| 50 | 97.4 | 97.4 | 97.4 | 94.7 | 94.7 | 97.4 | 94.7 |
| 100 | 97.4 | 97.4 | 100 | 94.7 | 94.7 | 100 | 97.4 |
| 150 | 97.4 | 97.4 | 100 | 94.7 | 94.7 | 98.4 | 97.4 |

Table2. Leave one out cross-validation results of Leukemia training data

Figure 2. Affect of number of genes on cross-validation accuracy for Leukemia training data using Leave-one-out cross validation

There are many possible models that give 100% cross-validation accuracy. The possible best models are: n=3, 5, 10, 25, and 100. These models gave a cross-validation accuracy of 100%. We evaluated the performance of the selected models on the test dataset with 34 samples.

**Test results for Leukemia Dataset:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Feature Selection Algorithm** | **Accuracy (%) using the top n=3 genes** | **Accuracy (%) using the top n=5 genes** | **Accuracy (%) using the top n=10 genes** | **Accuracy (%) using the top n=25 genes** | **Accuracy (%) using the top n=100 genes** |
| Elasticnet | 58.83 | 58.83 | 58.83 | 41.2 | 41.2 |
| Ttest | 58.83 | 58.83 | 58.83 | 64.7 | 41.2 |
| rfe-svm | 58.83 | 58.83 | 58.83 | 85.3 | 58.8 |
| Random Forest | 58.83 | 58.83 | 58.83 | 61.8 | 38.3 |
| F-test | 58.83 | 58.83 | 58.83 | 70 | 41.2 |
| Kruskal | 58.83 | 58.83 | 58.83 | 58.83 | 41.2 |
| Lasso | 58.83 | 58.83 | 61.8 | 41.2 | 41.2 |
| PSO |  |  |  |  |  |

Table 3. Classification accuracy of the Leukemia test dataset using the best models based on CV

The best classification accuracy on Leukemia test dataset was 85.3% achieved by rfe-SVM algorithm using the top 25 genes. The binary-PSO algorithm selected 4 genes and gave a classification accuracy of 58.83% on the test dataset. All the other algorithms that gave high cross-validation accuracy performed poorly on the test dataset. This shows that most of the algorithms over-fit to the training dataset.

**SRBCT Dataset**

We used leave-one-out cross-validation method to train our model. There was variation in the performance of the 7 algorithms for the top 25 selected genes. Table 4 and Figure 3 show the comparison of cross-validation performance for SRBCT train dataset.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Num genes** | **T-test** | **F-test** | **Kruskal** | **Elasticnet** | **Lasso** | **rfeSVM** | **Random Forest** |
| 1 | 85.7 | 63.5 | 39.7 | 87.3 | 90.5 | 79.4 | 60.3 |
| 3 | 100 | 77.8 | 82.5 | 87.3 | 90.5 | 96.8 | 100 |
| 5 | 100 | 81 | 79.4 | 87.3 | 90.5 | 96.8 | 100 |
| 10 | 100 | 98.4 | 93.7 | 87.3 | 90.5 | 100 | 100 |
| 15 | 100 | 98.4 | 96.8 | 87.3 | 90.5 | 100 | 100 |
| 20 | 100 | 100 | 98.4 | 87.3 | 90.5 | 100 | 100 |
| 25 | 100 | 100 | 98.4 | 87.3 | 90.5 | 100 | 100 |
| 50 | 100 | 100 | 100 | 87.3 | 90.5 | 98.4 | 100 |
| 100 | 98.4 | 100 | 100 | 87.3 | 90.5 | 100 | 98.4 |
| 150 | 98.4 | 100 | 100 | 87.3 | 90.5 | 98.4 | 98.4 |
| 200 | 98.4 | 100 | 100 | 87.3 | 90.5 | 98.4 | 98.4 |

Table 4. Leave one out cross-validation results of SRBCT training data



Figure 4. Affect of number of genes on cross-validation accuracy for SRBCT training data

There are many possible models that give 100% cross-validation accuracy. We evaluated the performance of the models with n=3 to n=100 on the SRBCT test dataset with 20 samples.

**Test results for SRBCT Dataset:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Feature Selection Algorithm** | **Accuracy (%) using the top n=3 genes** | **Accuracy (%) using the top n=5 genes** | **Accuracy (%) using the top n=10 genes** | **Accuracy (%) using the top n=15 genes** | **Accuracy (%) using the top n=25 genes** | **Accuracy (%) using the top n=50 genes** |
| Elasticnet | 50 | 60 | 55 | 60 | 60 | 65 |
| Ttest | 60 | 50 | 50 | 65 | 60 | 60 |
| rfe-svm | 40 | 35 | 55 | 70 | 50 | 50 |
| Random Forest | 50 | 45 | 60 | 75 | 55 | 70 |
| F-test | 50 | 50 | 70 | 68 | 55 | 55 |
| Kruskal | 50 | 50 | 40 | 60 | 60 | 65 |
| Lasso | 70 | 75 | 50 | 60 | 60 | 65 |

Table 5. Classification accuracy of the SRBCT test dataset using the best models based on CV

The best classification accuracy on Leukemia test dataset was 85.3% achieved by rfe-SVM algorithm using the top 25 genes. The binary-PSO algorithm selected 4 genes and gave a classification accuracy of 58.83% on the test dataset. All the other algorithms that gave high cross-validation accuracy performed poorly on the test dataset. This shows that most of the algorithms over-fit to the training dataset.

**NCI60 Dataset**

For the NCI60, we only had the training dataset with 60 samples. This is a very hard dataset to analyze as the 60 samples are classified into 9 cancer classes. As shown in Figure5, the variation was very high for the top 25 genes, but even after that variation was observed up to top 200 genes. None of the algorithms gave a siginificantly good performance.

Figure 5 and Table 6 show the comparison of cross-validation performance for SRBCT train dataset.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Num genes** | **T-test** | **F-test** | **Kruskal** | **rfeSVM** | **Random Forest** |
| 1 | 50 | 33.3 | 18.3 | 45 | 46.7 |
| 3 | 48.3 | 23.3 | 26.7 | 61.7 | 61.7 |
| 5 | 48.3 | 25 | 38.3 | 66.7 | 56.7 |
| 10 | 55 | 28.3 | 35 | 60 | 61.7 |
| 15 | 51.7 | 33.3 | 45 | 65 | 56.7 |
| 20 | 65 | 60 | 65 | 65 | 65 |
| 25 | 63.3 | 56.7 | 61.7 | 68.3 | 63.3 |
| 50 | 66.7 | 58.3 | 68.3 | 66.7 | 68.3 |
| 100 | 65 | 60 | 65 | 65 | 65 |
| 150 | 65 | 58.3 | 66.7 | 63.3 | 63.3 |
| 200 | 50 | 33.3 | 18.3 | 45 | 46.7 |

Table 6. Leave one out cross-validation results of NCI60 training data



Figure 6. Affect of number of genes on cross-validation accuracy for NCI60 training data

The binary PSO selected 11 genes with a CV accuracy of 58.8%. So, the best CV performance for the NCI60 dataset was given by rfe-SVM of 68.3% using the top 25 genes.

**Results Summary**

|  |  |
| --- | --- |
| **Dataset** | **Best algorithm and model** |
| Leukemia | Best algorithm: rfe-SVM  Best model: N=25 genes  Test Accuracy= 85.3% |
| SRBCT | Best algorithm: Lasso  Best model: N=5 genes  Test Accuracy= 75% |
| NCI60 | Best algorithm: rfe-SVM  Best model: N=25 genes  CV Accuracy= 68.3% |

**Conclusion**

It can be concluded that the embedded feature selection algorithms perform better than the filter and wrapper methods. Rfe-SVM performed best on the Leukemia and NCI60 dataset, but Lasso gave better performance on the SRBCT dataset. Thus, the performance of a feature selection algorithm depends on the dataset being used, the number of samples in the train and test dataset as well as the number of classes.

Most feature selection algorithms (mainly the filter and wrapper methods) generate models that over-fit to the training data, and thus generalize poorly. Recursive Feature Elimination- SVM method and Lasso achieves better performance because they minimize over-fitting by penalizing the irrelevant features, or removing them at every iteration.

More samples and better classification algorithms are required for building robust models for large-class datasets like NCI60. A comparison of different classification algorithms should be considered for building the trained model.

**References**

1) Guyon et al. **An introduction to variable and feature selection** Journal of Machine Learning Research 2003 (3): 1157-1182

2) Staunton et al. **Chemosensitivity Prediction by Transcriptional Profiling** Proc Natl Acad Sci USA 2001 Sep 11;98(19):10787-92 http://discover.nci.nih.gov/cellminer/loadDownload.do

3) Golub et al. **Molecular Classification of Cancer: Class Discovery and Class Prediction by gene expression profiling** S*cience* 15 October 1999: Vol. 286. no. 5439, pp. 531 - 537  
http://www.broad.mit.edu/cgi-in/cancer/publications/pub\_paper.cgi?mode=view&paper\_id=43

4) Khan et al. **Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks** Nature Medicine June 2001: Volume 7 no. 6, pp 673-679

<http://stat.ethz.ch/%7Edettling/bagboost.html>

5) Slawski et al. Classification for Microarrays **–** a comprehensive Bioconductor package for supervised classification with high dimensional data BMC Bioinformatics 2008 Vol. 9 pp439

6) Cheng et al. **A two-stage feature selection method for gene expression data** Journal of Integrative Biology Volume 13 Number 2 2009