Knowledge Discovery in Microarray Data

Rafiqul Islam

A Thesis
Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Data Mining
Department of Mathematics

Central Connecticut State University
New Britain, Connecticut
December 2004

Thesis Advisor

Dr. Daniel T. Larose
**Abstract:**

During the last decade, new technologies created a deluge of potential drug targets. Sifting through thousands of potential drug targets is a major industry bottleneck. Pharmaceutical companies can save billions of dollars by identifying most promising drug candidates at the earlier stages of pre-clinical development, and eliminating the rest.

Here we present a method of prioritizing potential drug targets based on their gene expression “signature”. Primary human pre-cursor neuronal cells were treated with three classes (antidepressant, antipsychotic, opioid receptor agonist) of psychoactive drugs for 24 hours. Microarray technology was used to capture expression of ~11,000 genes induced by these three categories of drugs. Clementine data mining software was used to build neural network and decision tree models that can categorize these three classes based on their gene expression profile. Then we tried to rank the predictions based on the confidence value generated by each model. These models can be used to classify and prioritize a portfolio of future novel drugs based on the gene expression induced by them.

In addition, a small set of genes relevant to all three target disease classes were identified based on neural network model sensitivity analysis and C5.0 ruleset. Biological experiments can be designed to better understand these genes, their relationship to the target classes and the mechanism of action for each drug class.

This paper is organized in two parts. The first part describes the biological techniques behind microarray technology and the second part describes how data mining techniques applied to microarray data. The data analysis part of this report is structured in CRISP-DM format, which is the cross-industry standard process for data mining.
<table>
<thead>
<tr>
<th>Table of contents</th>
<th>Page Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract</td>
<td>2</td>
</tr>
<tr>
<td>1.0 Molecular Biology</td>
<td>4</td>
</tr>
<tr>
<td>1.1 Microarray</td>
<td>4</td>
</tr>
<tr>
<td>1.2 Central Dogma of Molecular Biology</td>
<td>4</td>
</tr>
<tr>
<td>1.3 Graphical Overview of Microarray Process</td>
<td>6</td>
</tr>
<tr>
<td>2.0 Data Analysis</td>
<td>7</td>
</tr>
<tr>
<td>2.1 CRISP-DM Methodology</td>
<td>7</td>
</tr>
<tr>
<td>2.1.1 Business Understanding</td>
<td>8</td>
</tr>
<tr>
<td>2.1.2 Data Understanding</td>
<td>9</td>
</tr>
<tr>
<td>2.1.3 Data Preparation</td>
<td>9</td>
</tr>
<tr>
<td>A. Threshholding</td>
<td>10</td>
</tr>
<tr>
<td>B. Filtering</td>
<td>10</td>
</tr>
<tr>
<td>C. Feature section</td>
<td>11</td>
</tr>
<tr>
<td>C1. Signal to noise ratio method</td>
<td>12</td>
</tr>
<tr>
<td>C2. ANOVA method</td>
<td>13</td>
</tr>
<tr>
<td>2.1.4 Modeling and evaluation</td>
<td>16</td>
</tr>
<tr>
<td>a. Overfitting</td>
<td>16</td>
</tr>
<tr>
<td>b. Cross-validation</td>
<td>17</td>
</tr>
<tr>
<td>c. Boosting</td>
<td>17</td>
</tr>
<tr>
<td>d. Clustering</td>
<td>18</td>
</tr>
<tr>
<td>d1. Cluster Evaluation</td>
<td>19</td>
</tr>
<tr>
<td>e. Neural Network</td>
<td>22</td>
</tr>
<tr>
<td>e1. Wrapper Approach</td>
<td>24</td>
</tr>
<tr>
<td>e2. Neural Network Model Evaluation</td>
<td>25</td>
</tr>
<tr>
<td>f. Decision Tree</td>
<td>28</td>
</tr>
<tr>
<td>f1. C5.0 Model Evaluation</td>
<td>29</td>
</tr>
<tr>
<td>g. Model evaluation using out-group samples</td>
<td>31</td>
</tr>
<tr>
<td>h. Genes relevant to disease classes</td>
<td>32</td>
</tr>
<tr>
<td>3.0 Future directions and conclusions</td>
<td>33</td>
</tr>
<tr>
<td>4.0 References</td>
<td>34</td>
</tr>
<tr>
<td>5.0 Appendices: Clementine Scripts</td>
<td>36-42</td>
</tr>
<tr>
<td>6.0 Biographical Statement</td>
<td>43</td>
</tr>
</tbody>
</table>
1.0 Molecular Biology

1.1 Microarray

Microarrays are miniature arrays of gene fragments attached to glass slides or silicon chips. They are used to analyze expression of tens of thousands of genes in parallel. They expose differences in pattern of expression of genes to determine the underlying, molecular cause of disease states. Microarrays are a powerful technology not only for getting new fundamental insights into gene function but also for numerous application areas such as medicine, drug discovery and development, pharmacogenomics, toxicogenomics, and SNP (single nucleotide polymorphism) analysis.

1.2 Central Dogma of Molecular Biology

**Figure 1:** DNA acts as template to replicate itself, DNA is also transcribed into RNA, and RNA is translated into protein.¹

DNA is a double stranded polymer made up of individual chemical units called nucleotides – adenine (A), guanine (G), cytosine(C), thymine (T). DNA contains the master plan for living things. The entire DNA sequence of a living thing is called genome. Genomic information determines the form and function of an organism. The information content of the genome is carried hereditarily as deoxyribonucleic acid (DNA) from generation to generation.

Genes are the fundamental functional units of the genome. Genes encode proteins, which execute the genomic instructions. Proteins make up all structures and functions of cells and tissues. Genes control protein production. Gene expression is the process by which messenger RNA (mRNA) and eventually protein is synthesized from the DNA template of each gene.
There are two principle stages of protein production:

1. Transcription: A single stranded messenger RNA (mRNA) is synthesized from the DNA segment coding the gene
2. Translation: mRNA is used as a template to assemble a chain of amino acids to form the protein. Each mRNA encodes a unique protein

---

**Figure 2: Graphical representation of Gene Expression**

Gene expression is concerted, regulated manner under normal conditions. Levels of gene expression can be altered by a variety of factors such as changes in the environment and exposure of cells to drugs, germs, allergens. Aberrant expression results in diseases.

Each human cell can be viewed as a functional unit containing ~20,000 gene variables, whose expression level determines the functional state of the cell. For most genes, steady state mRNA levels approximate protein levels and thus quantitative expression monitoring at the mRNA level provides important clues as to function. mRNA can be viewed as a blue print containing the design of protein. Genomic analysis at the mRNA level can be used as a measure of gene expression. Specific changes in mRNA have been documented as a function of heat shock, drug treatment, and metabolic and disease state [2]. The hypothesis that many or all human diseases may be accompanied by specific changes in gene expression has generated much commercial interest in gene expression monitoring at the whole genome level. Microarray analysis is the primary tool for measuring gene expression at the whole genome level.
1.3 Microarray Process: A typical microarray experiment involves following six steps (figure 3). First RNA is extracted from cells and they are labeled with appropriate dye. Then the labeled RNA is applied to microarray chip or glass slide. The RNA is allowed to hybridize the gene fragments that are attached to the microarray surface. The excess RNA is washed out and an image of the microarray is generated. A scanner is used to read the image file and image analysis software is used to convert image file to a data file. Once the data file is generated, statistical and data mining tools can be used for analysis and visualization of the data.
2.0 Data Analysis:

2.1 Cross-Industry Standard Process for Data Mining (CRISP-DM)

The Cross-Industry Standard Process for Data Mining (CRISP-DM) is an industry neutral and software-platform neutral standard, non-proprietary process methodology for Data mining [4]. CRISP-DM divides each data mining project into six phases. Figure 4 illustrates six stages in a successful data mining process: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation and Deployment. The arrows represent common direction from one phase to another and indicate that the sequence of the phases is not necessarily linear. The outer circle indicates that the whole process is iterative. That is, data mining may continue after model deployment. The lessons learned from one data mining process can generate new hypotheses, which may be tested in subsequent data mining processes.

Figure 4: Graphical overview CRISP-DM

4
2.1.1 Business Understanding

The first phase of data mining involves understanding the business perspective of a project and formulating a data-mining problem with clearly defined objectives, based on the knowledge achieved from analyzing the business problem.

During last few years, pharmaceutical R & D expenditure increased significantly (from 15 billions in 1995 to 32 billion in 2002). But the number of approved drugs did not increase; in fact it fell from 47 approved drugs in 1996 to 23 approved drugs in 2002 (Figure 5).

![Figure 5: Number of approved drugs and expenditure in billion from 1995-2002](Image)

The pharmaceutical companies’ lack of productivity is not due to lack the of promising drug targets but due to the lack of tools for prioritizing drug targets in the early stages of development. Advances in genomics, proteomics, automation and high throughput screening technology have produced thousands of drug targets. In many cases, companies are generating more potential targets than they have the resources to investigate their efficacy. Building animal model to determine efficacy of thousands of drug targets will not only cost billions of dollars, it will take years. In addition, for some diseases like antidepressant there are no available animal models.

Today's drug development challenge is to accurately prioritize most promising drug candidates in the very early stage of development. Genomics has brought forth a wealth of new tools, technologies, and information that has enabled scientists to study
the human genome in a high throughput fashion in order to better understand disease. One such tool is Microarray technology. Microarray technology can be use to predict drug efficacy.

Gene expression profiles generated by microarray technology from cells in response to known drugs can be used as a “signature” to predict efficacy of a potential drug candidate. Thus the data-mining task would be to build classification models on gene expression data so that the models can predict the efficacy of potential drug candidates with relative accuracy. Once the drug candidates are classified, they should be ranked based on their classification prediction strength so that an organization can prioritize their potential drug portfolio and allocate their resources to the most promising drug candidate. When evaluating a classifier it would be more important to have better precision for top ranked classification/prediction than for the accuracy of whole dataset. In addition, a minimal set of genes should be identified that can support possible biological reasoning for a given classification/prediction or a set of gene that can be investigated for better understanding of the classification model and disease model.

2.1.2 Data understanding

As part of the experiment, primary human neuronal precursor cells were treated with 36 drugs and three control cultures were treated with Di-Methyl Sulfoxide (DMSO) only. Microarray technology was used to generate expression profiles for 10,992 genes in response to 36 drugs and 3 DMSO (control). These 36 drugs belong to three different classes, which represent target classes. They are: antidepressant (AD), antipsychotic (P), and opioid receptor antagonist (OP). Three DMSO treated samples do not belong to any of the three classes. So the data set contains 39 records, 10,992 fields and three target classes.

One major difficulty with microarray data is that they contain too few records (10 – 100 samples) and too many fields (10000-20000 genes). In statistics the term “Curse of dimensionality” is used to describe this problem. The curse of dimensionality refers to the exponential growth of hyper-volume as a function of dimensionality [6]. As hyper-volume grows bigger, running classification algorithms can become computationally expensive and may cause the classification algorithm to overfit the training data. In order to overcome (minimize) the effect of “curse of dimensionality” it is important to use a robust feature selection method, n fold cross-validation and to use multi-model approach like boosting in following data preparation and model building phases.

2.1.3 Data Preparation:

This phase includes all the processes necessary to convert the raw data to a final dataset that is ready for modeling. Key steps in microarray data preparation involve thresholding, filtering and feature selection.
A. Threshholding:

Microarray devices measure the level of gene expression based on image intensity. Due to the imperfect nature of hybridization, image analysis software returns raw expression values between 0 – 65,435. Several studies have shown that for Affymatrix platform, data below 100 and above 16000 is much less reproducible [7]. Image intensities below 100 and above 16,000 tend not to have linear correlation with actual expression values.

In order determine lower and upper threshold values, measurement error studies were performed on our proprietary microarray platform. The results showed that expression values below 50 and above 20,000 are less reproducible and did not correlate linearly to actual expression values. Therefore all values below 100 were set to 100 and all values above 20,000 were set to 20,000.

![Figure 6: An example of Microarray image file](image)

B. Filtering:

For the classification task, only genes that show variation in their gene expression across classes are useful for modeling. For example, if a gene shows expression values of 10, 12, and 14 across three classes, it will not be very useful in classifying the target classes. On the other hand, if a gene shows expression values of 10, 100, and 500 across three classes, it will have some discriminatory power and therefore it will be useful for classification modeling. To determine the biologically relevant level of variation gene
expression analysis was performed on housekeeping genes. Housekeeping genes are involved in basic functions of cell. They are absolutely necessary for cells to sustain. They are constitutively expressed in all condition in all tissues. These genes do not show any biologically significant variation in their gene expression when cells are exposed to drugs. One limitation of housekeeping genes is that they tend to be highly expressed and as a result they may not be representative of the genes in the experiment.

Based on the study of the housekeeping gene beta-actin, it was determined that a gene $i$ can be considered to have biologically insignificant level of expression variation if:

$$\frac{\text{Max}V(i)}{\text{Min}V(i)} \leq 3$$

AND

$$\text{Max}V(i) - \text{Min}V(i) \leq 500$$

Where $\text{Max}V(i) =$ maximum value of gene $i$ across all samples/record

And $\text{Min}V(i) =$ minimum value of gene $i$ across all samples/record

This filter would retain only the genes which variations are biologically significant. When the dataset of 10,992 genes were passed through this above filter, 2687 genes with biologically significant variation was discovered.

C. Feature Selection:

Microarray data analysis compared to other data such as CRM data or telecommunication data analysis, poses major difficulties for following two reasons. First, microarray data has large number of fields/genes compare to other kind of data. Human genome contains approximately 20,000–25,000 genes. Currently, technology is available to analyze all 20,000–25,000 genes in one chip. So the number of fields/genes in one microarray experiment can potentially be up to 25,000. Second microarray data contain very low number of records/samples typically around 20 to 100 for human data. The reason for such a low number of records is both cost and unavailability of human samples.

Therefore, it is important to reduce the number of fields before any predictive modeling can be done. Most learning algorithms such as neural networks, and decision trees look for non-linear combinations of features. Given too few records/samples and too many genes/fields these algorithms can easily find
many spurious combinations of genes that predicts outcome/class purely by chance.

During the last few years, several techniques for reducing the number of genes/fields have been proposed. All these techniques are collectively called feature selection. Some common feature selection techniques are: signal to noise ratio (S2N), analysis of variance (ANOVA), different variants of correlation scores, support vector machine (SVM), and chi-squared methods etc [9, 10, 11]. For this project, we chose to use the signal-to-noise-ratio method and the ANOVA method. The signal to noise method is intuitive, easy to understand and easy to explain. The ANOVA is most widely used and is well developed specifically for applying in microarray dataset [12, 13, 14].

C1. Signal to Noise ratio:

For a multi-class problem the signal to noise ratio can be defined as the following:

Signal to Noise ratio,

\[ S2N = \left| \frac{\mu_k - \mu_{-k}}{\sigma_k + \sigma_{-k}} \right| \]

Where,
\[ \mu_k = \text{mean of class } k \]
\[ \mu_{-k} = \text{mean of all class other than } k \]
\[ \sigma_k = \text{standard deviation of class } k \]
\[ \sigma_{-k} = \text{standard deviation for all classes other than } k \]

Once the S2N measure is calculated all genes are ordered based on their S2N value by class. It is hypothesized that fields/genes with greater absolute value of S2N will have greater discriminatory value across classes. In a two-class problem genes with high S2N value can be selected for modeling. But in a multi-class problem, as pointed out by Pomeroy et. al. [15], this technique may result in selection of genes that represent only one particular class and not representative of all classes. To avoid this problem genes were ordered by class based on S2N measure and then equal number of genes from each class was selected for modeling.

The signal to noise ratio is widely used by biologists because of its simplicity. But the drawback of S2N measure is that there is no way to estimate the significance value of S2N.
C2. ANOVA:

Analysis of variance (ANOVA) is used to test for differences between two or more means. ANOVA can be used to provide a significance test of the null hypothesis that three population (antidepressant, antipsychotic and opioid receptor agonist) means are equal. If the test is significant, then the null hypothesis can be rejected. The significance test involves the statistic F, which is the ratio of MSB to MSE: \( F = \frac{MSB}{MSE} \). The F-statistic is then used to calculate P-value and determine if the variation between means is significant. Using ANOVA in the case of microarray data requires some modifications in ANOVA like variance averaging, and false discovery rate (FDR).

Variance of replications within a group is called error variance. Error variance is estimated as the sum of the squared differences between the data and the corresponding group means. Error variance is usually used directly in ANOVA. But in case of microarray experiment, since the number of replication is very small, error variance can have destabilizing effect on F-statistics. Variance averaging in combination with some error model is an effective method to for stabilizing F-statistics.

Variance averaging is the method of averaging the error variance of genes with similar expression level. Genes were first ranked based on their average intensity. Then average error variance was estimated in a sliding window of 1000 genes. To avoid effect of genes with unusually high error variance, top 5% values of the error variance were discarded. Calculated average error variance was used in combination with the actual error variance calculated according to one of the following error models.

Error models attempt to get a better estimate for the true error variance with the data than the actual error variance estimated from data. One of the widely used error models is the Bayesian Error model proposed by Baldi and Long [13]. In the case of the Bayesian error model, error variance is the weighted average of the actual and averaged error variance calculated previously. It is an intermediate model between actual error variance, and averaged error variance.

Other models proposed are 1) maximum of averaged and actual error variance and 2) maximum of averaged and Bayesian error variances. The maximum of averaged and actual error variance is the most conservative; it reduces the number of false positives. The maximum of averaged and Bayesian error variance is an intermediate model between averaged error variance and maximum of averaged and actual error variance.

However none of these models is perfect. The Bayesian error model was used for this data set and genes with very high error variance were examined before accepting them as significant. A highly variable gene is defined as the genes with error variance at least 3 times higher than average.
Usually a P-value is used when determining if variation between means is statistically significant. In case of microarray experiments the null hypothesis for thousands of genes are calculated simultaneously. Due to the large number of genes, hypothesis testing based on P-value becomes meaningless. For example if the threshold for P value is set to 0.05 and there are 10,000 genes in the experiment, this will result in 500 false positive genes.

A possible solution to this problem was proposed by Bonferroni [16]. He proposes following correction to P-value.

\[
\text{Bonferroni correction} = P \times N
\]

Where P = P-value and N = number of genes

One drawback of Bonferroni correction is that it is too stringent, which results in the identification of few or no significant genes.

Since we can tolerate some small proportion of false positives, False Discovery Rate (FDR) proposed by Benjamini and Hochberg [14] can be used. FDR is an intermediate method between P-value and Bonferroni correction.

\[
FDR = \frac{P \times N}{k}
\]

Where P is P-value, N = total number of genes, and k = rank of a gene ordered in increasing P-value

Before ANOVA analysis, raw intensity values were converted to log scale. Following figures (7, 8, 9) show the distribution of over expressed and under-expressed genes by class pair. The X-axis shows log-intensity values of gene expression and the Y-axis shows log-ratio of expression values for two classes.

\[\text{Figure 7. Log-ratio vs. mean log-intensity for P/AD}\]
ANOVA in combination with Bayesian error model and FDR obtained 967 significant genes. They were ranked and were used as input for modeling. These models using ANOVA feature selection method did not perform as well as the signal to noise ratio method. So the results from models using genes selected by signal-to-noise method are reported in this project.

The possible reason for poor performance of ANOVA method is that it selects disproportionately very high number genes that differentiate between antipsychotic and opioid receptor agonist classes. ANOVA does not select enough genes that differentiate antidepressant class from other two classes. As a result, classification model does a very poor job on classifying antidepressant class. The classification performance becomes worse due to the fact that 20 out of 36 samples are from antidepressant group.
2.1.4. Modeling and Evaluation:

A model is an abstract representation of a real world process. Over the last few years, many different classification methods have been applied to microarray data. Two popular machine learning method for classification of disease and identification of small number of potentially relevant genes are: Artificial neural networks and decision tree. These two machine learning algorithm and two complementary techniques, cross-validation and boosting was examined for this project. Before building any model cluster analysis was used to gain better understanding of the data and to divide cases into natural groups. Cluster membership was used as input in the Neural Network model in an attempt to improve its accuracy.

a. Overfitting:

The critical issue in developing any model is generalization. A model should be able to perform nearly as well on unseen data as it does on the training data. Even after feature selection there are a large number of fields (967 genes for ANOVA the method and 2687 genes for the Signal-to-noise method) compare to the number of records (36) left in the dataset. Models built with so few records and so many fields will easily result in overfitting. Overfitting occurs when model start fitting not just the signal but also the noise in the training dataset. As a result, the model looses generality and performs very poorly on unseen data.

The following is a graphical example of overfitting (figure 10). Although the curve fits exactly through the training data, giving almost zero error for training data, the test data would be poorly classified.

![Graphical example of overfitting](image)

Figure 10: The curve shows a binary classifier-y is the output x is input and the black dots represent training data, and red dots represent test data. The curve passed almost exactly through black dots, which make the error in training data close to zero. When this curve is used to fit test data it will not perform nearly as well on test data because it overfits training data and lacks generalization.
The best way to avoid overfitting is to use lots of training data. Typically in most data mining applications when there is large number of records, data is partitioned exclusively into training set and test set. Models are built on the training set and the performance of a model is evaluated on the test set. But this method is very ineffective for microarray data due to the very low number of records. Partitioning a dataset with very low records will result in a training set with too few records and a test set with even fewer records that might be overly biased for one class and might not represent the training set. To overcome this problem \textit{n-fold cross validation} was used.

\textbf{b. Cross Validation:}

\textit{Cross-validation} works by dividing a data set into a number of subsets and then holding one subset each time as the test set and building the model on rest of the data set. Model performance is evaluated on the test set. In case of n-fold cross validation, the data set is divided into n subsets, and n different models are developed, each using n-1 data sets for training and the remaining data set for testing. The average accuracy of the models on those holdout samples is used as an estimate of the accuracy of the model. Leave one out cross validation is a special case of cross validation where \(n\) is equal to the number of cases/sample. According to Goutte (1997), cross validation works remarkably well in small data set compare to split-sample validation [17]

\textbf{c. Boosting:}

The \textit{Boosting} algorithm is used to improve accuracy and stabilize potentially unstable models such as decision trees. Decision tree models may be considered unstable because small changes in learning set may lead to larger changes in the classification performance. The boosting algorithm works by building several different models in sequence and then combining their separate predictions into one overall prediction by using a weighted voting scheme [18]. The first model is built in the usual way and then every subsequent model is built in a way that it focuses specifically on the records that were misclassified by the previous model. Cases for the training set are sampled with non-uniform sampling probabilities and with replacement. The general algorithm for boosting is governed by the following scheme (Figure: 11) as described by Dubitzky et.al. [19].

\textbf{Step 1:} Use current (index \(i\)) sampling probabilities, \(p_i\), and sample with replacement \(k\) instances from the learning set (black box labeled Test in the diagram).

\textbf{Step 2:} Build current classifier, \(f_i(x)\), based on the current training set.

\textbf{Step 3:} Establish current classifier's performance by testing it against the learning set (black box) and keep a record of correctly/incorrectly classified cases, and
Step 4: Update the sampling probabilities (feedback arrow in upper part of diagram) for each instance based on the classifier's performance and calculate a weight for the classifier also based on its performance (illustrated by the small box in the lower part of the diagram).

Step 5: Stop if number of classifiers is reached, otherwise go to Step 1.

**Figure 11.** Graphical depiction of Boosting. Thick arrows represent uniform sampling, without replacement; double arrows represent non-uniform sampling with replacement using sampling probabilities depicted by $p_i$.

**d. Clustering:**

Clustering is the most important unsupervised learning method. Clustering deals with finding the intrinsic grouping or structure in a set of unlabeled data.

There are many available algorithms to cluster sample data points based on nearness or similarity measures. These traditional clustering methods, like k-means clustering, are effective on small datasets but usually don’t scale up well in large datasets. These methods also do not have any built in criteria for determining the optimal number of clusters.

To overcome these problems, TwoStep clustering method found in Clementine was used [20]. TwoStep clustering methods employs BIRCH algorithm developed by
Zhang et. al. [21]. Unlike traditional cluster methods, two-step clustering can select an optimal number of clusters based on a statistical criterion. As the name implies, this algorithm utilizes following two-stage clustering approach. This two-stage method enables this algorithm to cluster large datasets very effectively.

**Stage 1: Pre-clustering the data:** At this stage a sequential clustering approach [22] is used to scan all records one by one and to decide based on a distance criterion if the current record should merge with the previously formed clusters or a new cluster should be started. This procedure is implemented using a modified version of cluster feature tree (CF) described by Zhang et. al [21]. The distance criterion used in this algorithm is an extension of model-based distance measure developed by Banfield and Raftery [23]. The original distance measure developed by Banfield and Raftery can only be used with continuous attributes. This measure was modified to handle both continuous and categorical variable.

**Stage 2: Grouping data into sub clusters:** In this step, the agglomerative hierarchical clustering method uses previously generated sub clusters as input and generates the desired number of clusters as output. The desired number of clusters can be an user input or it can be determined by auto-cluster algorithm built into the two-step cluster method. Since this algorithm requires only one data pass for the whole procedure, it does not require enormous machine resources.

**d1. Cluster Evaluation:**

After thresholding and filtering there were 2687 genes left in the dataset. Each samples was normalized to mean=0 and standard deviation =1 by subtracting from each gene expression the mean value of the sample and dividing it by standard deviation of the sample.

TwoStep clustering algorithm was applied on normalized data. Classes discovered by this algorithm were then compared with the known classes (Figure12).

<table>
<thead>
<tr>
<th>Value</th>
<th>Proportion</th>
<th>%</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>cluster 1</td>
<td></td>
<td>22.22</td>
<td>8</td>
</tr>
<tr>
<td>cluster 2</td>
<td></td>
<td>16.67</td>
<td>6</td>
</tr>
<tr>
<td>cluster 3</td>
<td></td>
<td>26.0</td>
<td>9</td>
</tr>
<tr>
<td>cluster 4</td>
<td></td>
<td>22.22</td>
<td>8</td>
</tr>
<tr>
<td>cluster 5</td>
<td></td>
<td>13.89</td>
<td>6</td>
</tr>
</tbody>
</table>

**Figure 12:** Distribution of Class_Label over clusters. AD= Antidpressant, P= Antipsychotic and P= Opioid receptor agonist
TwoStep clustering method generated 3 clusters automatically. Only one of them was pure (antidepressant) and other two were mixture of all 3 classes. In an attempt to achieve better separation of samples different number of clusters were tested. Five clusters achieved best separation of classes and subclasses. In this case, four out of the five clusters were pure. Clusters 1 and 4 only had data records from the antidepressant group. Cluster 2 only had records from the anti-psychotic group and cluster 5 only had records from the opioid receptor agonist group. Cluster 3 is the only cluster that contains records from more than one group. It had a total of 9 records. 4 of them were antidepressant, 2 of them were antipsychotic and 3 of them were opioid receptor agonist.

Each cluster was further analyzed based on the subclasses of each disease model (Figure 13). The dataset had samples from 4 subclasses of antidepressant (AD), two subclasses of antipsychotic (AP) and 4 subclasses of opioid receptor agonist. They are:

- **Antidepressant (AD):**
  - Tricyclic (10)
  - Selective serotonin reuptake inhibitor, SSRI (5)
  - Monoamine Oxidase Inhibitor, MAOI (3)
  - Atypical (2)

- **Antipsychotic (P):**
  - Classic (4)
  - Atypical (4)

- **Opioid receptor agonist:**
  - μ-OPR (3)
  - δ-OPR (2)
  - κ-OPR (2)
  - κ/μ-OPR (1)
• Cluster 1 had only samples from subclass tricyclic. 8 out of 10 tricyclic samples were in cluster 1.
• Cluster 4 had 5 SSRI, 2 tricyclic and 1 MAOI.
• Cluster 3 has 4 antidepressants (2 MAOI and 2 Atypical), 2 antipsychotic and 3 opioid agonist inhibitor.
• Cluster 2 and 5 does not have any antidepressants.

Selective serotonin reuptake inhibitors (SSRIs) are a new generation of antidepressants, which are most effective and have the least side effects. They most likely work on very specific pathways and least likely to induce non-specific pathways. So it is not surprising to see all the SSRIs are in the same cluster.

Tricyclics along with other subclasses block to varying degrees the reuptake of several neurotransmitters including serotonin. These subclasses share in some degree the same mechanism as SSRIs. It can be hypothesized from this clustering that 2 tricyclic and 1 MAOI that share the same cluster with SSRIs might have more serotonin uptake activity than other tricyclic or MAOIs.

Most antipsychotic (6 out of 8) and opioid agonist (5 out of 8) were in pure clusters 2 and 5 respectively. Remaining antipsychotic and opioid agonist shares cluster 3 with 4 other antidepressants (figure 14). This suggests that even though these drugs act on distinct drug targets within multiple neurotransmitter systems, they may share common biological pathways downstream from their intervention point.
Neural networks are a powerful data modeling tool that is able to capture and represent complex input/output relationships. Since neural networks can handle many interacting variables and capture non-linear relationships between input and output, it is widely used in the field of molecular biology for disease classification.

There are many different types of neural networks. One of the most widely used neural networks is the multi-layer perceptron (MLP). MLP networks, like other networks, map the input to the output using training data so that the model can then be used to produce the output when the desired output is unknown. MLP networks consist of an input layer, output layer and at least one hidden layer. Each layer is made up of one or more neurons and each neuron is linked to all the previous neurons by connections of varying weights. The input data (in this case gene expression microarray data) is fed into the input layer, which triggers a response to the hidden layer. The hidden layer is required to perform non-linear mapping. The hidden layer triggers a response to the output layer. Each neuron in the output layer represents a class.

Figure 14: Distribution of Cluster three

Figure 15: A graphical representation of a fully connected MLP. Three ($X_1$, $X_2$, $X_3$) inputs, one hidden layer and one output.24
The learning capability of a neural network is defined by the activation function of the neurons, and the network topology - the way the neurons are connected. The activation function of each neuron consists of two functions. First one is called the combination function, which merges all inputs, and their weights, into a single value. And the second one is called the transfer function, which transfers the output value of the combination function to the output neurons. The most common combination function is the weighted sum. Commonly used transfer functions are the linear, hyperbolic tangent, and sigmoid function.

![Figure 16: Activation function for one neuron](image)

The MLP and many other neural networks learn using an algorithm called backpropagation [19]. With backpropagation, the input data is presented to the neural network one case (Xi, C) at a time. Network then calculates and predicts the output for the case in question. The output of the neural network is compared to the desired output and an error is computed by backpropagation algorithm. This error is then fed back (backpropagated) to the neural network and used to adjust the weights in way that minimizes the error between predicted output and actual output. This process is known as "backpropagation learning". Neural networks may use other learning methods such as genetic algorithm [25].

Learning normally proceeds until a certain termination criterion chosen by the user is reached. A common criterion is the prediction accuracy threshold. The problem with this criterion is that if the value chosen is too low, the network may not be able to generalize well, and if too high the network may memorize or overfit the training data. In both cases, the expected performance on unseen test data will be low. Clementine offers a default mode for stopping. With this setting, the network will stop training when the network appears to have reached its optimally trained state.
There are 4 MLP algorithms available in Clementine. They are:

- **Quick.** This is the simplest MLP network. This method creates a network with one hidden layer. The number of neuron in hidden layer is determined by characteristics and quantity of the input data. This model is fastest and easiest to train.

- **Dynamic.** This method creates an simple initial topology, and as the training progresses it modifies the topology by adding and/or removing hidden units. The network continues to grow until it is determined that adding a neuron does not increase predictive power for a number of growing attempts.

- **Multiple.** This method creates more than one network each with different topologies. The number of hidden layer depends on the quantity and type of training data. All these networks are trained in a pseudo-parallel fashion and the network with lowest RMS error is presented as the final model at the end of training. Since multiple networks are trained in pseudo-parallel fashion, training for such network is extremely slow.

- **Prune.** This method creates a large initial network and as the training progresses, it removes (prunes) the weakest neurons in the hidden and input layers based on sensitivity analysis. This process continues until there has been no improvement. Like multiple, this method is usually slow.

All four MLP algorithms provide option for sensitivity analysis. Sensitivity analysis provides a measure of relative importance for each input to the network. It is helpful in evaluating the predictors and understanding the model.

**e1. Wrapper approach:**

In the data preparation section, the number of genes was substantially reduced from 10,992 to 2687 for the signal to ratio method. But still the number of genes is too high for Neural Network modeling. The wrapper approach suggested by Kohavi and John [28] was used to further reduce the number of genes and to identify the number of optimal genes for Neural Network modeling.

The wrapper approach suggests ranking 100-200 genes and building a model using increasing numbers of genes. For signal to noise ratio method, equal number of genes were selected from each class. For the initial run, numbers of genes are selected in a wide interval (for example 10, 20, 40, 60, 80, 100) and models were trained using 5 fold cross validation. A plot of error rate vs. genes per class is generated. The plot usually shows that error rate is initially higher for low number of genes and then the error rate goes relatively down when the number of genes reaches the optimal number and then the error rate starts to increase gradually as the number of genes becomes too large. This whole process is repeated by selecting the number of genes from the valley region of the curve in short intervals (for example: 5, 7, 12, 20). This process can be repeated several times until an optimal subset of genes is found.
2. Neural Network Model Evaluation:

The wrapper approach with 10 fold cross validation identified 21 genes (7 per class) that can classify 29 out of 36 gene expression profile in the category corresponding to the therapeutic application of the drug used to treat the cells (Avg. accuracy ~80%). Among 21 genes, four of them have relatively high importance measure. They are genes 6273, 5729, 7725, and 5525. The following graph (figure 17) shows genes per class and VS Error_avg for 10 fold cross validation. Error_avg was calculated by averaging classification errors over 10 runs.

![Figure 17: Neural Network output. Genes per class vs Error average (no cluster input)](image)

In an attempt to increase accuracy, cluster membership was used as input for the neural network. Neural network models were built with 3, 5, 7, and 9 genes per class along with cluster membership as inputs. Each model was run using 10 fold cross validation. The following graph (figure 18) shows Genes per class vs. error average.
Accuracy for each model was improved significantly. The model with 5 genes per class has better accuracy than the model with 7 genes per class. Five out of 36 samples were misclassified (average accuracy ~88.5 %). Four of them are antidepressant (Sample # 17, 18, 19, 20) and one anti-psychotic (22). Cluster membership has the highest relative importance value as expected. Sensitivity analysis shows that this model relies heavily on cluster membership. The relative importance measure for cluster membership is 0.42 and for all other 15 genes the value is between 0.065 and 0.03.

Using TwoStep cluster algorithm (as mentioned in section 2.1.4. d), 5 clusters were identified. Clusters 1, 2, 4 and 5 are completely separated by class. They contain class values of AD, P, AD and OP respectively. Cluster 3 is the only cluster containing all 3 classes. It contains 9 samples: 4 of them are AD (17, 18, 19, 20), 2 of them are P (22, 26) and 3 of them are OP (29, 30, 31). It was noted that all four misclassified samples (17, 18, 19, 22) belong to cluster three. The neural network model was able to classify correctly 5 samples out of 9 samples that belong to cluster 3.

The neural network method with cross validation makes prediction based on number of votes received by a given sample. The class value that receives most number of votes wins the prediction. Since it is a winner takes all scheme, it does not distinguish between the winners who won by large margin and small margin. For our purpose of prioritizing drug targets, we need a way to distinguish between “strong winner” and “weak winner”. Neural Network’s “Confidence Index” (CI) can be used for this purpose. Confidence value/ index is defined as the number of training records for which the model or submodel makes a correct prediction divided by the number of training records for which the model or submodel makes any prediction. The CI
provides a continuous output from 0 to 1, 0 being the weakest prediction and 1 being the strongest prediction. The comparison of CI with prediction class indicates that the accuracy of the prediction scales closely with the magnitude of the confidence value. All the predictions above 0.30 CI value are correct. All 5 misclassified samples have confidence index <= 0.26. Sample # 26, which belongs to cluster 3 passed with low confidence value (0.25). A cutoff value of 0.30 for confidence Index would increase the precision for each class to 100%. Thus CI can be used to prioritize correct predictions and also reduce false positive.

Table 1: Neural Network output in ascending order of confidence index

<table>
<thead>
<tr>
<th>Sample #</th>
<th>Sample ID</th>
<th>Class Label</th>
<th>Predicted Class</th>
<th>Confidence Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>22</td>
<td>Colzapine</td>
<td>P</td>
<td>OP</td>
<td>0.15</td>
</tr>
<tr>
<td>19</td>
<td>Bupropion</td>
<td>AD</td>
<td>P</td>
<td>0.17</td>
</tr>
<tr>
<td>20</td>
<td>Trazadone</td>
<td>AD</td>
<td>P</td>
<td>0.22</td>
</tr>
<tr>
<td>17</td>
<td>Phenylzine</td>
<td>AD</td>
<td>P</td>
<td>0.23</td>
</tr>
<tr>
<td>26</td>
<td>Trifluperazine</td>
<td>P</td>
<td>P</td>
<td>0.25</td>
</tr>
<tr>
<td>18</td>
<td>Iproniazid</td>
<td>AD</td>
<td>P</td>
<td>0.26</td>
</tr>
<tr>
<td>25</td>
<td>Chlorpromazine</td>
<td>P</td>
<td>P</td>
<td>0.38</td>
</tr>
<tr>
<td>24</td>
<td>Loxapine</td>
<td>P</td>
<td>P</td>
<td>0.42</td>
</tr>
<tr>
<td>29</td>
<td>Enkephalin</td>
<td>OP</td>
<td>OP</td>
<td>0.44</td>
</tr>
<tr>
<td>23</td>
<td>Haloperidol</td>
<td>P</td>
<td>P</td>
<td>0.48</td>
</tr>
<tr>
<td>27</td>
<td>Triflupromazine</td>
<td>P</td>
<td>P</td>
<td>0.53</td>
</tr>
<tr>
<td>31</td>
<td>Dynorphin</td>
<td>OP</td>
<td>OP</td>
<td>0.57</td>
</tr>
<tr>
<td>28</td>
<td>Pimozide</td>
<td>P</td>
<td>P</td>
<td>0.60</td>
</tr>
<tr>
<td>30</td>
<td>BW373U86</td>
<td>OP</td>
<td>OP</td>
<td>0.63</td>
</tr>
<tr>
<td>21</td>
<td>Risperidone</td>
<td>P</td>
<td>P</td>
<td>0.64</td>
</tr>
<tr>
<td>34</td>
<td>Endomorphin</td>
<td>OP</td>
<td>OP</td>
<td>0.70</td>
</tr>
<tr>
<td>35</td>
<td>DALDA</td>
<td>OP</td>
<td>OP</td>
<td>0.70</td>
</tr>
<tr>
<td>4</td>
<td>Desipramine</td>
<td>AD</td>
<td>AD</td>
<td>0.71</td>
</tr>
<tr>
<td>5</td>
<td>Doxepin</td>
<td>AD</td>
<td>AD</td>
<td>0.77</td>
</tr>
<tr>
<td>2</td>
<td>Amoxepine</td>
<td>AD</td>
<td>AD</td>
<td>0.79</td>
</tr>
<tr>
<td>32</td>
<td>U50488</td>
<td>OP</td>
<td>OP</td>
<td>0.80</td>
</tr>
<tr>
<td>33</td>
<td>U62066</td>
<td>OP</td>
<td>OP</td>
<td>0.82</td>
</tr>
<tr>
<td>36</td>
<td>DAMGO</td>
<td>OP</td>
<td>OP</td>
<td>0.82</td>
</tr>
<tr>
<td>6</td>
<td>Imipramine</td>
<td>AD</td>
<td>AD</td>
<td>0.82</td>
</tr>
<tr>
<td>14</td>
<td>Paroxetine</td>
<td>AD</td>
<td>AD</td>
<td>0.83</td>
</tr>
<tr>
<td>8</td>
<td>Nortryptiline</td>
<td>AD</td>
<td>AD</td>
<td>0.83</td>
</tr>
<tr>
<td>10</td>
<td>Trimipramine</td>
<td>AD</td>
<td>AD</td>
<td>0.84</td>
</tr>
<tr>
<td>3</td>
<td>Clomipramine</td>
<td>AD</td>
<td>AD</td>
<td>0.86</td>
</tr>
<tr>
<td>7</td>
<td>Maprotiline</td>
<td>AD</td>
<td>AD</td>
<td>0.87</td>
</tr>
<tr>
<td>12</td>
<td>Fluoxetine</td>
<td>AD</td>
<td>AD</td>
<td>0.87</td>
</tr>
<tr>
<td>16</td>
<td>Tranylcypromine</td>
<td>AD</td>
<td>AD</td>
<td>0.88</td>
</tr>
<tr>
<td>15</td>
<td>Sertraline</td>
<td>AD</td>
<td>AD</td>
<td>0.89</td>
</tr>
<tr>
<td>1</td>
<td>Amitriptyline</td>
<td>AD</td>
<td>AD</td>
<td>0.89</td>
</tr>
<tr>
<td>11</td>
<td>Citalopram</td>
<td>AD</td>
<td>AD</td>
<td>0.90</td>
</tr>
<tr>
<td>13</td>
<td>Fluvoxamine</td>
<td>AD</td>
<td>AD</td>
<td>0.91</td>
</tr>
<tr>
<td>9</td>
<td>Protriptyline</td>
<td>AD</td>
<td>AD</td>
<td>0.92</td>
</tr>
</tbody>
</table>
f. Decision Trees:

Decision tree is a technique of representing a set rule in tree structure that lead to a class value. A decision tree generates model that are both predictive and descriptive.

The basic algorithm for decision tree follows a top down, divide and conquers strategy. First an attribute/ field is selected based on some statistical test on how well an attribute alone classifies the training samples and is placed as a root node. This root node at the top of the tree represents all observations. Based on a logical test on the attribute, data is partitioned into sub-groups. Then a single branch is formed for each outcome of the test and subset of training records satisfying that outcome move to the corresponding child node. This process is repeated recursively for each child node. If at any instances, all records at a node have the same classification, that part of the tree stops developing. Alternatively, tree stops developing based on a termination rule.

Clementine’s C5.0 decision tree algorithm is based on a decision tree software developed by J. Ross Quinlan [29]. The C5.0 algorithm uses a combination of information gain and gain ratio criterion, based on information theory to partition training data. The gain ration represents the proportion of information generated by dividing the training set in the parent node, into each of the outcomes of the field at the parent node that is useful. The portioning stops when node is pure or no more splits can be found. This algorithm uses a pruning method called pessimistic pruning. Pessimistic pruning uses statistical methods to calculate the error rate associated with each node and adjusts the tree to reflect bias. Then it pessimistically increases errors observed at each node using statistical measurements to encourage pruning. Advantages of this pruning method over other pruning method are that it builds only one tree and provides a more reliable tree when data is scarce.

Winnowing attributes mode in C5.0 allows the algorithm to pre-select a subset of the attributes that will be used to construct the decision tree or the ruleset. It is very important in microarray data mining application since microarray dataset contains thousands of attributes/ genes and only tens of samples/ cases. The C5.0 algorithm can automatically winnow/ discard genes that appear to be only marginally relevant to the classification task before a classifier is constructed. The winnowing attributes lead to smaller classifiers and higher predictive accuracy. It also reduces the time required to generate tree/ rule sets.
Clementine’s C5.0 node was used to build decision tree models on microarray dataset. On initial run, C5.0 algorithm with leave one out cross-validation achieved 88.9% accuracy or 32 out of 36 samples were classified correctly. Misclassified samples were 18, 19, 25, and 29. The confidence index values were 0.64, 0.64, 0.75 and 0.86 respectively. Samples 18, 19 and 29 belong to cluster 3. Samples 18 and 19 also failed in neural network model. Samples 25 and 29 passed in neural network model with low confidence index.

In an attempt to increase accuracy, decision tree model was rerun using 10-fold boosting. This time 33 out of 36 samples were classified correctly (91.2% accuracy). Samples misclassified were 26, 27, and 28. Like neural network models, decision tree model (C5.0) also generates confidence value as an estimate of the accuracy of a prediction and confidence value is defined as the number of training records for which the model or a specific rule or decision tree branch makes a correct prediction divided by the number of training records for which the model or submodel makes any prediction.

The confidence index values for misclassified samples 26, 27 and 28 were 0.54, 0.77, and 0.54 respectively. The comparison of CI with prediction class indicates that the accuracy of the prediction scales closely with the magnitude of the confidence value. For example: 97% of the predictions above CI value of 0.54 are correct and 100% of the predictions above 0.77 CI value are correct. A cutoff value of 0.55 for confidence index would increase the precision for classification to 97%. Thus the CI generated by C5.0 can be used to prioritize correct predictions and also reduce false positive.
<table>
<thead>
<tr>
<th>Sample #</th>
<th>Sample ID</th>
<th>Class_Label</th>
<th>Predicted Class</th>
<th>Confidence Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>26</td>
<td>Trifluperazine</td>
<td>P</td>
<td>AD</td>
<td>0.538</td>
</tr>
<tr>
<td>28</td>
<td>pimozide</td>
<td>P</td>
<td>AD</td>
<td>0.538</td>
</tr>
<tr>
<td>21</td>
<td>Risperidone</td>
<td>P</td>
<td>P</td>
<td>0.698</td>
</tr>
<tr>
<td>22</td>
<td>Colzapine</td>
<td>P</td>
<td>P</td>
<td>0.698</td>
</tr>
<tr>
<td>25</td>
<td>Chlorpromazine</td>
<td>P</td>
<td>P</td>
<td>0.701</td>
</tr>
<tr>
<td>27</td>
<td>Triflupromazine</td>
<td>P</td>
<td>OP</td>
<td>0.773</td>
</tr>
<tr>
<td>29</td>
<td>Enkephalin</td>
<td>OP</td>
<td>OP</td>
<td>0.773</td>
</tr>
<tr>
<td>31</td>
<td>Dynorphin</td>
<td>OP</td>
<td>OP</td>
<td>0.773</td>
</tr>
<tr>
<td>18</td>
<td>Iproniazid</td>
<td>AD</td>
<td>AD</td>
<td>0.791</td>
</tr>
<tr>
<td>19</td>
<td>Bupropion</td>
<td>AD</td>
<td>AD</td>
<td>0.791</td>
</tr>
<tr>
<td>1</td>
<td>Amitriptyline</td>
<td>AD</td>
<td>AD</td>
<td>0.811</td>
</tr>
<tr>
<td>8</td>
<td>Nortryptiline</td>
<td>AD</td>
<td>AD</td>
<td>0.811</td>
</tr>
<tr>
<td>2</td>
<td>Amoxepine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>3</td>
<td>Clomipramine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>4</td>
<td>Desipramine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>5</td>
<td>Doxepin</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>6</td>
<td>Imipramine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>7</td>
<td>Maprotiline</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>9</td>
<td>Protriptyline</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>10</td>
<td>Trimipramine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>11</td>
<td>Citalopram</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>12</td>
<td>Fluoxetine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>13</td>
<td>Fluvoxamine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>14</td>
<td>Paroxetine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>15</td>
<td>Sertraline</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>16</td>
<td>Tranylcypromine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>17</td>
<td>Phenylzine</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>20</td>
<td>Trazadone</td>
<td>AD</td>
<td>AD</td>
<td>1.000</td>
</tr>
<tr>
<td>23</td>
<td>Haloperidol</td>
<td>P</td>
<td>P</td>
<td>1.000</td>
</tr>
<tr>
<td>24</td>
<td>Loxapine</td>
<td>P</td>
<td>P</td>
<td>1.000</td>
</tr>
<tr>
<td>30</td>
<td>BW373U86</td>
<td>OP</td>
<td>OP</td>
<td>1.000</td>
</tr>
<tr>
<td>32</td>
<td>U50488</td>
<td>OP</td>
<td>OP</td>
<td>1.000</td>
</tr>
<tr>
<td>33</td>
<td>U62066</td>
<td>OP</td>
<td>OP</td>
<td>1.000</td>
</tr>
<tr>
<td>34</td>
<td>Endomorphin</td>
<td>OP</td>
<td>OP</td>
<td>1.000</td>
</tr>
<tr>
<td>35</td>
<td>DALDA</td>
<td>OP</td>
<td>OP</td>
<td>1.000</td>
</tr>
<tr>
<td>36</td>
<td>DAMGO</td>
<td>OP</td>
<td>OP</td>
<td>1.000</td>
</tr>
</tbody>
</table>
g. Model Validation using out-group sample:

A good model for the purpose of our target prioritization task should be able to distinguish between in-group samples (AD, P, OP) and out of group samples even though out-group samples were not used to build models. There were 3 samples that were treated with DMSO, which does not belong to any class of drug treatment used here. The profile generated by cells in response to DMSO was passed through neural network and C.50 described above. The neural network model classified them as AD, AD, OP with confidence index of 0.33, 0.36, and 0.41. All 36 samples were plotted (figure 18) as samples on x axis and confidence value on y axis. The highest value among three DMSO samples (0.41) falls around 80th percentile. A cut off value of 0.41 for ranking would achieve 100% precision for classification and 80% sensitivity.

![Figure 19: Sample vs. Neural network confidence Value](image)

The C5.0 decision tree model classified three DMSO treated samples as AD, P, AD with confidence value of 0.811, 0.773 and 0.773 respectively. All 36 samples were plotted (figure 19) as samples on x-axis and confidence value on y-axis. The highest value among three DMSO samples (0.81) falls around 70th percentile. A cut off value of 0.81 for ranking would achieve 100% precision for classification and 70% sensitivity.

![Figure 20: Sample vs. C5.0 confidence value](image)
h. Genes relevant to disease classes:

In addition to classifying three disease classes and ranking samples based on their prediction strength, one of our goals was to identify a subset of genes that are relevant to the three target disease classes.

The sensitivity analysis generated by neural network model was used to identify genes that are relevant to three disease classes. Out of 15 genes that were used for neural network modeling, 9 of them showed relatively strong association (sensitivity >= 0.04).

The C5.0 algorithm like other sequential classifier by nature identifies smallest set of genes with strong inputs. In an attempt to generate larger number of genes with prediction power, C5.0 model was run sequentially eight times (until the average accuracy of the model fell below 80%) and each time excluding the genes used for prediction in the previous run. All eight runs generated 28 genes that are relevant to three disease models. Table 3 shows complete list of genes by family, 9 from neural network and 28 from decision tree (C5.0) model and 3 of them were common in both models.

<table>
<thead>
<tr>
<th>Gene Identity</th>
<th>Gene Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Microsomal glutathione S-transferase(GST)</td>
<td>1</td>
</tr>
<tr>
<td>LDHA</td>
<td>1</td>
</tr>
<tr>
<td>GPCR</td>
<td>9</td>
</tr>
<tr>
<td>Survivin</td>
<td>1</td>
</tr>
<tr>
<td>Colipase precursor</td>
<td>1</td>
</tr>
<tr>
<td>CRGN X1</td>
<td>1</td>
</tr>
<tr>
<td>DEOXYCYTIDINE KINASE</td>
<td>1</td>
</tr>
<tr>
<td>Erythroblast membrane associated protein</td>
<td>5</td>
</tr>
<tr>
<td>Homo sapiens transcription factor</td>
<td>1</td>
</tr>
<tr>
<td>CG small GTPase (RAB5C) mRNA</td>
<td>1</td>
</tr>
<tr>
<td>Splicing factor</td>
<td>12</td>
</tr>
</tbody>
</table>

*Table 3: Genes relevant to classification of antidepressant, antipsychotic and opioid agonist diseases.*

Biological experiments can be designed to confirm association of these genes to the target classes. It should be noted that these genes does not reflect any cause-effect relationship with disease. It only indicates that drugs used in this project manipulate disease pathways in a way that these genes are upregulated or down regulated significantly. Biological experiments like RTQ-PCR, western blot can be used to confirm the outcome of neural network and decision tree models. These gene-disease associations can be used to better understand the mechanisms of action of the drugs from all three classes.
3.0 Future directions and conclusions:

In this project it was demonstrated that microarray technology in combination with state of the art data mining algorithm could be used to classify and rank novel drug targets. The classification accuracy for neural network model and decision tree (C5.0) model was 80% and 91% respectively. When cluster information was used as input in neural network model classification accuracy increased to 88.5%. It was also demonstrated that confidence value generated by each model could effectively be used for prioritizing a portfolio of novel drug targets. Finally, a list of genes strongly related to classification task was generated. Further biological experiments on these genes may provide better understanding of mechanism of action for three target classes of drugs.

During last few years several methods were presented for analyzing microarray data. Most of these methods primarily focus on identifying biomarkers, and predicting clinical efficacy of a drug target [9,12,15,30, 31,32]. Very few methods attempted to rank drug targets based on their prediction strength. The methods presented here successfully rank novel drug targets in addition to classifying them. The early prioritization of drug targets can significantly increase productivity of drug discovery process.

In this paper we attempted to use signal-to-noise and ANOVA feature selection methods. But the genes selected by ANOVA had very poor predictive power compare to signal-to-noise ratio method. One promising idea for future extension would to optimize ANOVA method and also explore other feature selection methods like SVM, discriminant analysis, correlation score methods.

TwoStep cluster analysis performed very well in capturing natural segmentation in data. It was even able to capture subclasses for antidepressant class. There were not enough examples of other antidepressant subclass or antipsychotic and opioid agonist subclass. More subclass representation would most likely result in better segmentation of sample space. More subclass representation for each class should be added for future work. Since neural network model heavily relies on cluster membership, better clustering would also result in a better performance by neural network model.

C5.0 algorithm performed very well in classifying samples. But drawback of this method is that it relies on too few genes. It does not perform as well as neural network for ranking classification. We generated a list of predictive genes by running C5.0 model sequentially and removing strong predictors from previous run. Another promising future work would be to run C5.0 model until model performance falls below certain threshold and then they can be combined for final classification. It can be hypothesized that model would be more stable because it would rely on more genes. But the challenge would be to improve accuracy and to obtain better separation
of confidence value between false positives and true positives without overfitting the data.

From biological point of view, for future work the cells could be treated with drugs longer than 24 hours. Longer exposure of cells to these drugs might be able exaggerate the differences among their gene expression profile which would be easier for classification algorithm to differentiate.

4.0 REFERENCES

[12] Pavlidis, P., Using ANOVA for gene selection from microarray studies of the nervous system, Department of Biomedical Informatics and Columbia Genome Center, www.sciencedirect.com, Columbia University, New York, NY 10032, USA


5.0 Appendices:

1. Clementine Script for Thresholding

# Project Global Parameters
# set vDIR = "thesis/cns/"
set vDIR = "thesis/cns_tmp/"
set vPROJ = "cns"

# create gene description file
set PROJ_gene_description.full_filename = vDIR >< vPROJ >< 
"_gene_description.tab"
execute PROJ_gene_description

# thresholded data file, genes in rows
set PROJ_thres_gr.full_filename = vDIR >< vPROJ >< "_thres_gr.dat"
execute PROJ_thres_gr

# transposed data file, genes in columns
set PROJ_thres_TRgc.full_filename = vDIR >< vPROJ >< "_thres_gc.dat"
set PROJ_thres_TRgc.output_to = "Transposed"

# rename table during execution
var t
set t = get node PROJ_thres_TRgc
set vTable = vPROJ >< "_thres_TRgc"
rename ^t as ^vTable
execute ^t
rename ^t as PROJ_thres_TRgc
2. Clementine Script for Filtering

# Project Parameters
set vPROJ = "cns"
# set vDIR = "thesis/cns/"
set vDIR = "thesis/cns_tmp/"
set vMAX_MIN_RATIO = 3
set vMAX_MIN_DIFF = 500

# set tmp file name
set TMP_thres_minmax.full_filename = vDIR >< vPROJ >< "_thres_minmax.tmp"
execute TMP_thres_minmax

# Report on the effect of the Fole Filter
execute Genes_remaining

# use tmp file name
set TMP_thres_minmax_f2.full_filename = vDIR >< vPROJ >< "_thres_minmax.tmp"

# output file name
set PROJ_thff_gr.full_filename = vDIR >< vPROJ >< "_thff_gr.dat"
execute PROJ_thff_gr

# transposed file name
set PROJ_thff_TRgc.full_filename = vDIR >< vPROJ >< "_thff_gc.dat"
set PROJ_thff_TRgc.output_to = "Transposed"

# rename table during execution
var t
set t = get node PROJ_thff_TRgc
set vTable = vPROJ >< "_thff_TRgc"
rename ^t as ^vTable
execute ^t
rename ^t as "PROJ_thff_TRgc"
3a. Clementine Script for signal-to-noise calculation

Project Global values
# set vDIR = "thesis/cns/
set vDIR = "thesis/cns_tmp/
set vPROJ = "cns"
# minimum and maximum class number
set vMIN_CLASS = 0
set vMAX_CLASS = 2

# set the class file & input file
# set '*PROJ_class'.full_filename = vDIR >< vPROJ >< "_class.dat"
# set '*PROJ_thff_gc'.full_filename = vDIR >< vPROJ >< "_thff_gc.dat"

# set output file stem
var outstem
set outstem = vDIR >< vPROJ >< "_s2n_"

# go over all classes
for vClass from vMIN_CLASS to vMAX_CLASS
    # User feedback
    set vQname = "Stats for Class " >< vClass
    rename :qualitynode as ^vQname

    # set output file for each class
    set PROJ_s2n_C.full_filename = outstem >< vClass >< ".dat"
    execute 'PROJ_s2n_C'
endfor

3b. Clementine Script for signal-to-noise calculation

# project parameters
# set vDIR = "thesis/cns/
set vDIR = "thesis/cns_tmp/
set vCATDIR = "thesis/cns/
set vPROJ = "cns"
set vMIN_CLASS = 0
set vMAX_CLASS = 2
set vTOP_GENES_CLASS = 200

# number of all genes selected
set vAllgenes = vTOP_GENES_CLASS * (vMAX_CLASS - vMIN_CLASS + 1)

# User feedback
var vAname
set vAname = "select " >< vTOP_GENES_CLASS >< " top genes per class"
rename :analysis as ^vAname

# input files
# set '*PROJ_thff_gr'.full_filename = vDIR >< vPROJ >< "_thff_gr.dat"

# temp file
set TMP_top_all_gr.full_filename = vDIR >< vPROJ >< "_top_all_gr.tmp"
set TMP_top_all_gr.writemode = "Overwrite"

# select top genes for each class
for vClass from vMIN_CLASS to vMAX_CLASS
  set vQname = "Processing Class " >< vClass
  rename :qualitynode as ^vQname

  # range of input files from mclass_p1
  # set PROJ_s2n_C_f1.full_filename = vDIR >< vPROJ >< "_s2n_" >< vClass >< ".dat"
  set PROJ_s2n_C_f1.full_filename = vCATDIR >< vPROJ >< "_s2n_" >< vClass >< ".dat"

  # save the merged expression values to a file
  execute TMP_top_all_gr
  set TMP_top_all_gr.write_mode = "Append"
endfor

# set the temp source node
set TMP_top_all_gr_s.full_filename = vDIR >< vPROJ >< "_top_all_gr.tmp"

# Temporarily rename transpose table during execution
var t
set t = get node PROJ_topNNg_TRgc

set vTable = vPROJ >< "_top" >< vAllgenes >< "g_TRgc"
rename ^t as ^vTable
set ^t.full_filename = vDIR >< vPROJ >< "_top" >< vAllgenes >< "g_gc.dat"
set ^t.output_to = "Transposed"
execute ^vTable
rename ^t as "PROJ_topNNg_TRgc"

execute 'Gene,Class by S2N'
4. Clementine Script for Neural Network Modeling

# This script tests accuracy of neural net models by selecting a different number of
genets from each class, building a model using the selected number of genes, and
testing this model using cross-validation.
# At the end, average error rate is reported and graphed.
# Cross-validation works by split data into vXVAL_FOLDS group by tags;
# then loop through tags building model with all but one and testing
# on the one not used for training.
# During run, the unconnected quality node changes name to 'Genes per class'

# input parameters
# set vDIR = "thesis/cns/"
set vDIR = "thesis/cns_tmp/"
set vPROJ = "cns"
# number of all genes in the input data
set vALL_GENES = 300
set vNUM_CLASSES = 3

#set vTITLE = "Selecting top genes sorted by S2N, regardless of class"
set vTITLE = "Selecting same # of top genes from each class"
# user-modifiable -- number of folds for cross-validation
set vXVAL_FOLDS = 10
# rename the analysis node to display the number of folds
var vAn_name
set vAn_name = vXVAL_FOLDS >< " folds"
rename :analysisnode as ^vAn_name

# Temp file for xval results
set :outputfile.full_filename = vDIR >< vPROJ >< " _net_xval.tmp"
set :outputfile.write_mode = "Overwrite"
set TMP_xval_all.full_filename = vDIR >< vPROJ >< " _net_xval.tmp"

# Outer loop, iterate through different numbers of genes
var vgenes vQname
for vgenes in 9 15 21 30
  if (vgenes <= vALL_GENES)
    set vSel_genes = vgenes
  else
    set vSel_genes = vALL_GENES
  endif
  # User feedback
  set vQname = 'using' >< vSel_genes >< " genes of " >< vALL_GENES
rename :qualitynode as ^vQname

# inner loop -- set the correct number of genes as "IN"
var vField_no vField vGene_no
set vField_no = 0
for vField in_fields_at type
  set vField_no = vField_no + 1

# subtract two because the first two fields are not genes
set vGene_no = vField_no - 2

# initialize all genes to none
if (vGene_no > 0)
  set type.direction.^vField = "NONE"
endif

# set first vgenes to IN
if (vGene_no > 0 and vGene_no <= vSel_genes)
  set type.direction.^vField = "IN"
endif
endfor

# Perform cross-validation for the selected number of fields
clear generated palette

# Inner cross-validation loop
var vXcount vModel
set vXcount = 1
for vTag from 1 to vXVAL_FOLDS
  set vModel = "Genes " >< vSel_genes >< ", Net " >< vXcount >< " of " >< vXVAL_FOLDS
  set :trainnet.netname = vModel
  execute :trainnet
  insert model ^vModel connected between 'select vTag' and ErrorRate
  execute :outputfile
  delete last model
  delete generated ^vModel
  set :outputfile.write_mode = "Append"
  set vXcount = vXcount + 1
endfor
endfor

execute :report
execute :plot
5. Clementine Script for C5.0 Modeling

# Project globals
set vDIR = "thesis/cns/"
set vDIR = "thesis/cns_tmp/"
set vPROJ = "cns"
set vALL_GENES = 300

# User selectable fold for Xvalidation
set vXFOLD = 36

# Input file
set '*PROJ_top_all_gc'.full_filename = vDIR >< vPROJ >< "_top" >< vALL_GENES >< "g_gc.dat"

set Class_Label:c50.xval_fold = vXFOLD
clear generated palette

execute Class_Label:c50

delete :c50rule

insert model Class_Label connected between :type and filter
connect :c50rule to analysis

execute analysis

execute Predicted
6.0 Biographical Statement:

Rafiqul Islam came in the United States as a student from Bangladesh in 1992. He earned his B.S. degree in Biology from Mercy College, New York. He worked as a Molecular Biologist for six years at New Haven based biotech company, CuraGen Corporation. He is currently working as a Research Scientist at Huntingdon Life Sciences’ Princeton Research Center in New Jersey.