A Comparison of Different Clustering Methods on Breast Cancer Tumor Data

Ruchi Asthana, Joseph Zoller

May 11, 2017

Abstract

According to the New England Medical Journal, the over-diagnosis of breast cancer via mammography is "larger than is generally recognized", resulting in only a 1% overall absolute mortality benefit for women screened. This problem arises from the fact that we have developed very good methods for detecting tumors (based on size), but there is currently no effective method of predicting which tumors will become dangerous (requiring an aggressive approach to treatment) and which tumors will remain benign (requiring a more cautious approach to treatment). This project will be conducted to address this fundamental over-diagnosis problem by finding a better method of determining if a tumor is malignant and likely to be harmful to the host. Using phylogenetic trees constructed from Fluorescence in situ Hybridization (FISH) data, we will first find a set of tree statistics to distinguish malignant metastatic tumor phenotypes from benign ones. Then we will compare different classification and clustering methods to determine how effective the given feature vector is in distinguishing between metastatic and non-metastatic tumor phenotypes.

Introduction

In randomized control trials, for women between the ages of 40 and 74 years, screening with mammography has been associated with a 15% relative reduction in mortality from breast cancer. However, the absolute mortality benefit for women screened annually is only 1% overall (National Institutes of Health). This raises uncertainty about the benefit of mammography in the present day. Additionally, it presents a research topic with much urgency, as unnecessary mammography can result in a set of potential harms including: (1) an over-diagnosis resulting in treatment of insignificant cancers, (2) false positives with additional testing and anxiety, and (3) radiation-induced breast cancer. Such problems arise because we have developed very good methods for detecting tumors, but there is currently no effective method of predicting which tumors will become dangerous (requiring an aggressive approach to treatment) and which tumors will remain benign (requiring a more cautious approach to treatment). This fundamental problem can also be extended to other cancer types. This project will be conducted to address the fundamental over-diagnosis problem by finding a better method of determining if a tumor is malignant and likely to be harmful to the host.

The approach used within this investigation will try to address one of the main problems with cancer genome studies—tumor heterogeneity. Tumors are enormously heterogeneous, meaning that they vary greatly from patient to patient, as well as from cell to cell in a single patient. This heterogeneity makes it difficult to separate random noise from meaningful signals that can help us predict future behavior of a tumor. Today, many research groups are attempting different strategies for extracting predictions of future tumor behavior from different kinds of genomic data. Unfortunately, no such methods are yet reliably effective. The Schwartz lab is currently taking a specific approach, largely based on trying to explicitly reconstruct the process of evolution in single tumors, in order to identify features of the mutational process that are predictive of how a tumor will evolve in the future. This work falls into a special subfield of computational biology called ”tumor phylogenetics.” To the present day, this lab has developed a series of algorithms for building single-tumor evolutionary trees, also known as tumor phylogenies. They have also shown that one can identify features of these phylogenies, through statistical and machine learning methods, that can be used to predict cancer progression. The intent of our work will be to improve existing prediction methods by finding more predictive properties, or combinations of properties, that can help reliably predict a patient's prognosis from their tumor data.

The project design will be composed of three methods: (1) constructing phylogenetic trees from Fluorescence in situ Hybridization (FISH) data, (2) developing a set of tree statistics to interpret the phylogenetic tree, and (3) applying these tree statistics, along with different clustering methods learned in class, to determine if they can distinguish between metastatic and non-metastatic tumor phenotypes.

Methods

Building Trees From FISH Data Sets

Fluorescence in Situ Hybridization (FISH) serves as a powerful tool for identifying locations of a cloned DNA sequence on metaphase chromosomes. This methods is effective in mapping specific genes and portions of genes to genetic diseases and abnormalities. FISH is particularly valuable for cancer studies, because it provides a way of tracking gain or loss of genetic material, which is the primary way by which tumors evolve. Fluorescently labeling small pieces of DNA in single cells extracted from a tumor allows us to count the pieces of DNA, and identify when cells have gained or lost copies, relative to a non-cancerous cell. This data can be used to construct phylogenetic trees, which describe how a set of cells might have evolved from a common, healthy ancestor. FISH data from breast tissue was obtained from the Schwartz lab. We will run FISH data through existing software for evolutionary tree building, developed by Dr. Salim Chowdhury. This will produce trees similar to those in figure 1 below. Tree statistics will then be determined to analyze the gain and loss mutations outlined by this tree.

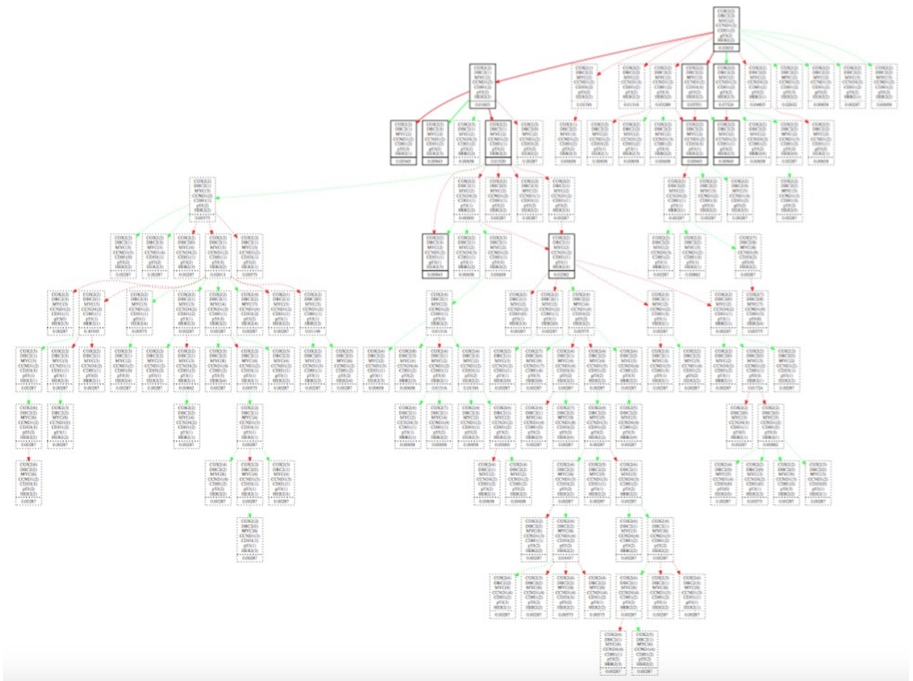


Figure 1: Phylogenetic Tree produced from FISH program.

Developing Tree Statistics

Once phylogenetic trees were generated, FISH code was modified to extract information about the gain and loss in expression for each of the following 7 genes: COX2, DBC2, MYC, CCND1, CDH1, p53, and HER2. Expression data is represented by the number of copies of each gene in cells of the tumor sample. This data was added to a feature vector to represent each sample, and this vector was then used in clustering analysis.

Clustering Methods

- K-means clustering

  For our first method, we used k-means clustering. Given a specific distance metric, this method finds exactly k clusters in which each data point belongs to the cluster with the nearest mean, where the k different means represent the most "typical" point in each cluster. The algorithm works by initializing values for the k means (using any of various methods), and then runs an iterative minimization algorithm, where the objective function is the within-cluster sum of squares (WCSS), i.e. the within-cluster variance. The initialization can be done at random, but the default method used in Matlab is called the "k-means++ algorithm". When using k-means, it is important to determine how many clusters you want, i.e. the value of k. We set k to values in the range of $\left[2, \lceil\sqrt{n}\rceil\right]$ based off results from deSouto et al.[1] In our case, since $n = 18$, so we used $k = 2, 3, 4, 5$.

  In our analysis, we applied the k-means clustering to the gain/loss rate data pairs for each of the 7 genes, and we used all 18 samples for each of the clusterings.

- Hierarchical Clustering

  For our second method, we used hierarchical clustering. The basis of this clustering method is constructing a hierarchical linkage tree that portrays the hierarchy of similarity between data points. In order to construct such a tree, one needs both a distance metric and a linkage method; we used correlation as our distance metric and WPGMA as our linkage method. The WPGMA (Weighted Pair Group Method with Arithmetic Mean) algorithm constructs a hierarchical linkage tree, starting with a pairwise-distance matrix, as follows: at each iteration, the two nearest clusters, say I and J, are joined together to make a new cluster, say K. Then, the new cluster's distance to every other cluster is calculated as the arithmetic mean of the distances between the other cluster and each of the parent clusters:

$$\mathrm{dist}(L, K) := \frac{\mathrm{dist}(L, I) + \mathrm{dist}(L, J)}{2}.$$

  The algorithm terminates once there is only one overall cluster. The linkage tree is then constructed by connecting clusters that were joined during an individual step in the algorithm, and the heights of the branches of the tree are half the distances between immediately-adjacent clusters.

  In our analysis, we applied the hierarchical clustering to our entire dataset, using our feature vector for each sample as a sequence of values for the clustering.

- Spectral clustering

  For our third and final method, we used spectral clustering. This method of clustering can be more easily thought of as a two-part algorithm: (1) constructing (an adjacency matrix for) an undirected similarity graph from the data, and (2) computing the Laplace embedding from the similarity graph. First, one needs to construct a similarity graph, either weighted or unweighted, using the data, and this can be done in many different ways. For our project, we constructed two different similarity graphs: (i) An unweighted, undirected similarity graph that is the k-nearest neighbors graph, with a specified value of k (we used $k = 4$), and (ii) a weighted, undirected similarity graph, where edge weights are computed as 1 minus the correlation between feature vectors.

Second, we computed the Laplace embedding using the similarity graph. Laplace embedding is a method of embedding a given data set into a new space by mapping each data point to the eigenvectors of the graph Laplacian at that data point. The motivation for this embedding is that we would like the data to be injected into a space where Euclidean distances indicate similarity between data points, and the eigenspace of the graph Laplacian does this task well. Finally, we will explain what the graph Laplacian is. Given a similarity matrix (or weight matrix) W, the unnormalized graph Laplacian matrix L is defined as

$$L = D - W,$$

where D is the diagonal matrix given by

$$d_i = \sum_{j=1}^{n} w_{i,j}.$$

Finally, if $\{\psi_k\}_{k=1}^{m}$ are the eigenvectors of L, then the Laplace embedding of a given data point X_i is given by $(\psi_1(i), \psi_2(i), \psi_3(i), \ldots, \psi_m(i))$. It is important to note that, due to the nature of the graph Laplacian, the first eigenvector ψ_1 is always a constant vector of 1's (up to scaling), and so it can be ignored in our analyses. Thus, when we say "significant" eigenvectors, we mean all the other eigenvectors, i.e. $\psi_2, \psi_3, \ldots, \psi_m$.

In the paper by von Luxburg[2], after they have the Laplace embedding in hand, they cluster the data points using k-means clustering, and only using the first k eigenvectors. However, one is not restricted to k-means clustering, as one can perform any kind of clustering that uses the Euclidean distance as a distance metric. As a consequence, we decided to cluster all of our Laplace embeddings with hierarchical clustering.

In our analysis, we computed the Laplace embedding of our entire dataset, using our feature vector for each sample as a sequence of values for the clustering, and using two different similarity graphs (as specified above).

Comparison Analysis

For a comparison analysis we compared the result of our clustering to their actual phenotype; this was done visually. The phenotype of a cluster was determined by the majority phenotype present in the cluster. In the case where there were ties, the phenotype was chosen arbitrarily. It should be noted that "fair" clustering is one that clusters values with greater than 50% accuracy, because such clustering is likely to be non-random.

Implementation Details

Building Trees From FISH Data Sets

1. Obtain .txt files that contain expression for each of the genes probed
2. Run the .txt files in the FISH program
3. Add print statements to FISH code to get gene expression for each of the genes probed. These genes include: COX2, DBC2, MYC, CCND1, CDH1, p53, and HER2
4. Write code to export these values to an excel spreadsheet
5. Then import this data into MATLAB for clustering analysis.

Clustering Methods

We implemented all of the cluster methods using Matlab.

- K-means Clustering

  For this clustering method, we iterated through the gain/loss data for each gene separately, and we applied k-means clustering for each of these genes. Thus, each k-means clustering was run using 2 data points (gain proportion and loss proportion) from each of the 18 samples, and we implemented this clustering for various values of k. We implemented the k-means clustering by using the built-in function `kmeans` in Matlab, with the default distance metric and the default cluster initialization algorithm. When plotting the clustering results for each gene, we used shapes to indicate the different clusters, and colors to indicate whether the corresponding sample was from a primary (green) or metastatic (red) tumor, and used those plots as our final output.

- Hierarchical Clustering

  For this clustering method, we considered the collective gain/loss data as a sequence of values, for every sample. First, we constructed the hierarchical linkage tree by using the built-in function `linkage` in Matlab, with the `"weighted"` linkage method and `"correlation"` distance metric. Second, we clustered the data using the tree by writing a simple function `get_maxcluster_cutoff` that found the maximum tree height cutoff that would generate at most k_0 clusters. For our analysis, we set $k_0 = 5$. We plotted the clustering via the dendrogram, with the branches colored based on cluster, and used that plot as our final output.

- Spectral Clustering

  For this clustering method, we again considered the collective gain/loss data as a sequence of values, for every sample. We defined and derived a few functions to perform this clustering method:

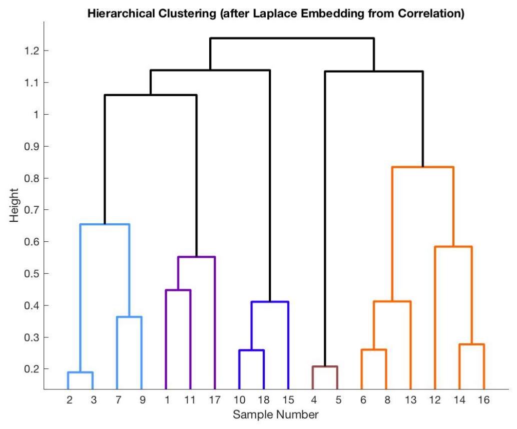
- `similarity_matrix_knn` - this constructs (an adjacency matrix for) an unweighted, undirected similarity graph, that is the k-nearest neighbors graph with a specified value of k. We did so by using the built-in function `knnsearch` with $k = 4$, and then making the graph undirected by making any present directed edges into edges.
- `similarity_matrix_corr` - this constructs (an adjacency matrix for) a weighted, undirected similarity graph with weights calculated as 1 minus the correlation between feature vectors. We did so by using the built-in function `pdist` with `"correlation"` as the distance metric.
- `lap_embed_knn`, `lap_embed_corr` - using the corresponding similarity graphs, these calculate the eigenvectors of the graph Laplacian at each data point, sorted such that the corresponding eigenvalues are decreasing in magnitude.

We then plotted the first three significant eigenvectors (i.e. ψ_2, ψ_3, and ψ_4) against each other using a 3D scatterplot, and judged which two of the three eigenvectors best clustered the data. We then plotted those two eigenvectors against each other in an ordinary 2D scatterplot, and used that plot as one of our visual outputs.
For the clustering itself, we applied hierarchical clustering to both of the embeddings, with the `"ward"` linkage method and `"euclidean"` distance metric. We set $k_0 = 5$ as we did in the ordinary hierarchical clustering above.

Results and Conclusions

From this project we learned that comparative studies for different clustering data is imperative and highly valuable. There is a significant difference between the various clustering methods, especially when they are applied to high dimensional data, like breast cancer tumor samples. Given heterogeneous tumor data from primary and metastatic breast cancer samples, we were able to derive the following conclusions. First, k-means clustering performed with the least accuracy. This can be observed visually in figure 4 and figure 5, where shape represents cluster assignment, while color represents tumor phenotype. On average accuracy of tumor samples was around 50%. Since these results mirror accuracy scores of randomly assigned phenotypes, we concluded that no significant relationships could be derived from the K-means clustering. Second, spectral clustering via hierarchical clustering was effective in separating the different phenotypes. As shown in figure 2, on average only 3 phenotypes were misidentified, leading to an accuracy score of 83%.

However, this metric was outperformed by our final clustering method: hierarchical clustering with the correlation metric. As shown in figure 3 this method correctly identified all but one sample, resulting with an accuracy score of 95%. We hypothesized that hierarchical clustering performed the best because it was able to use correlation values (instead of a euclidean metric) to compare the tumor phenotypes, which makes sense when the data being clustered is a sequence of values rather than a point in some high-dimensional space. In a biological sense, this is interesting because it shows metastasis can be achieved through various combinations of gain and loss mutations. Thus, while it is hard for a human to be able to predict which combinations of gain and loss mutations will result in a metastatic phenotype,



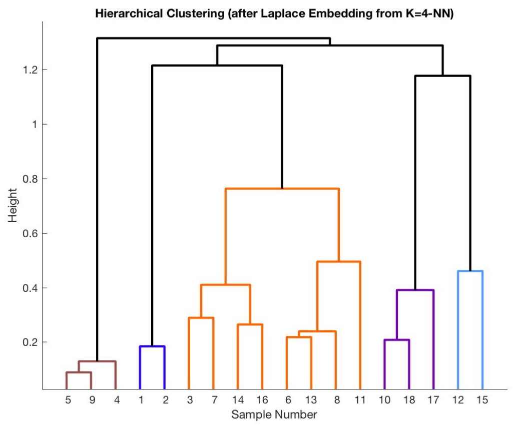


Figure 2: Results of hierarchical clustering after Laplace embedding. Samples 1-9 had primary phenotype and samples 10-18 had metastatic phenotype.

hierarchical clustering has the potential to be effective in reducing high-dimensional space into a meaningful low dimensional space in which similar phenotypes are close to each other.

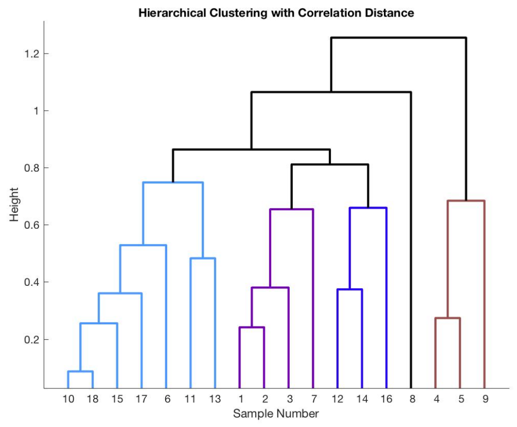


Figure 3: Results of ordinary hierarchical clustering. Samples 1-9 had primary phenotype and samples 10-18 had metastatic phenotype.

References

[1] De Souto, et al. "Clustering cancer gene expression data: a comparative study". In: *BMC Bioinformatics* 9.497 (2008).

[2] Von Luxburg, U. "A tutorial on spectral clustering". In: *Statistics and Computing* 17.4 (2007), pp. 395–416.

Appendix

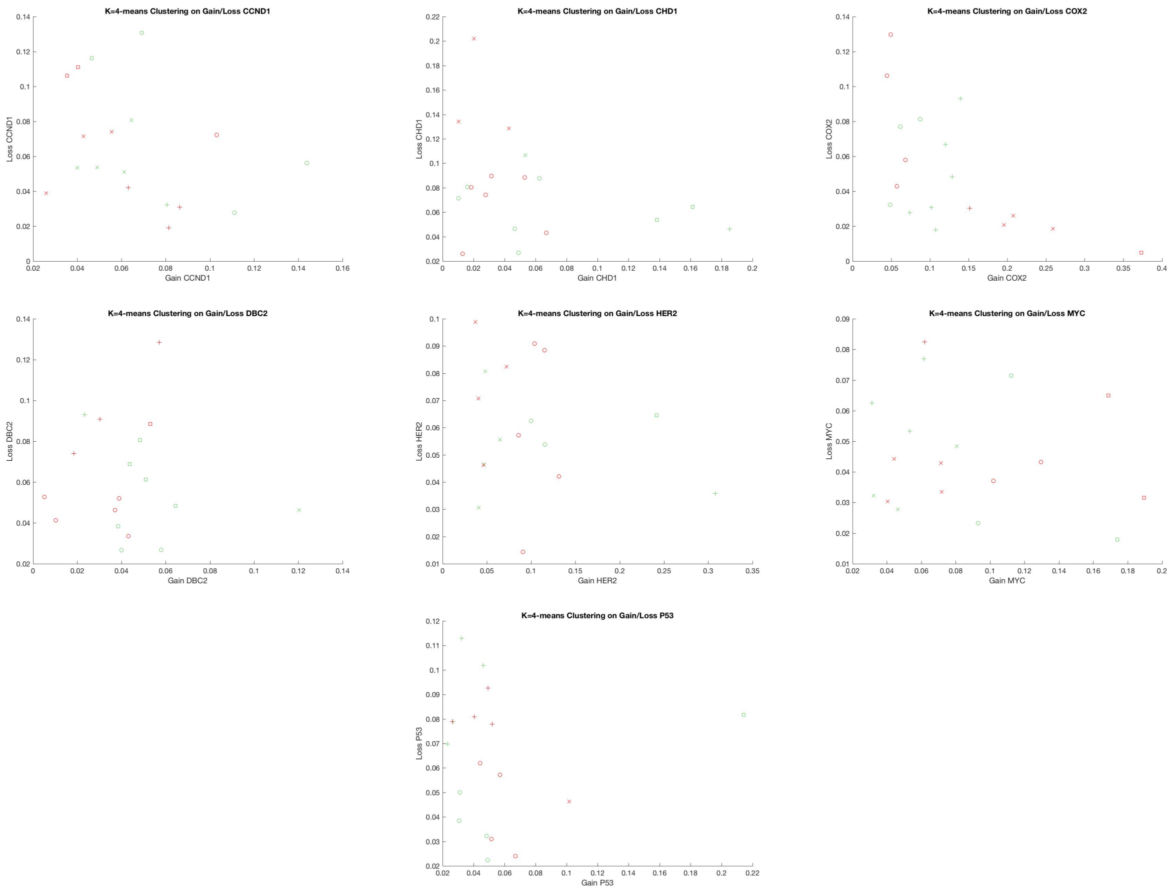


Figure 4: Results of k-means clustering for k = 4. Green indicates primary phenotype and red indicates metastatic phenotype.

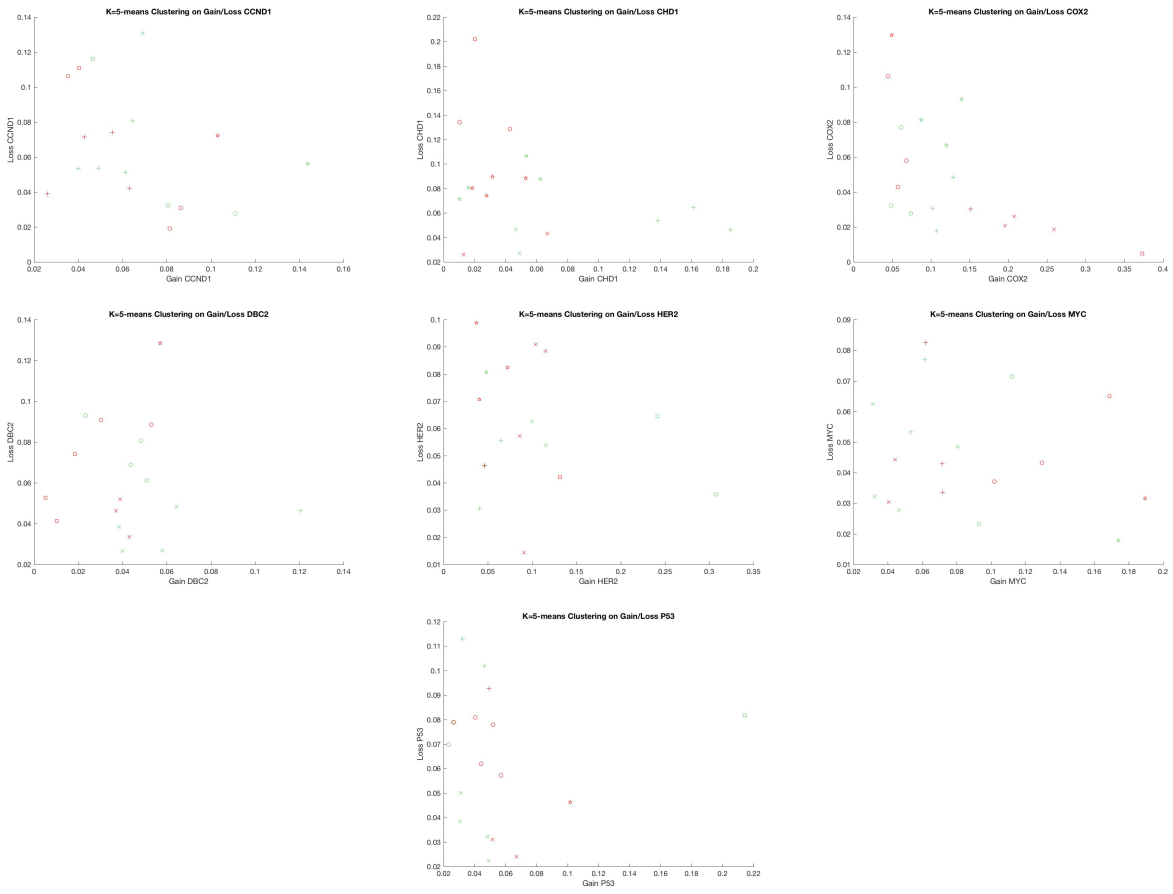


Figure 5: Results of k-means clustering for k = 5. Green indicates primary phenotype and red indicates metastatic phenotype.

```
function cutoff = get_maxcluster_cutoff(Z, k)
    cutoff = Z(end-k+2,3)-eps;
end

function S = similarity_matrix_knn(X, k)
    sz = size(X);
    n1 = sz(1);
    S = double(zeros(n1));
    IDX = knnsearch(X, X, 'K', k);
    for i=1:n1
        for j=1:n1
            sij = (sum(IDX(i,:) == j) > 0);
            sji = (sum(IDX(j,:) == i) > 0);
            if (sij || sji) S(i,j) = 1; else S(i,j) = 0; end;
        end
    end
end

function [LapX,Lambda] = lap_embed_knn(X, k, m)
    S = similarity_matrix_knn(X, k);
    n = length(S);
    d = double(zeros(1,n));
    for i=1:n
        d(i) = sum(S(i,:));
    end
    P = double(zeros(n));
    for i=1:n
        for j=1:n
            P(i,j) = S(i,j)/d(i);
        end
    end
    [V,Lam] = eig(P);
    Lambda = diag(Lam);
    [list,I] = sort(abs(Lambda),'descend');
    Lambda = Lambda(I);
    Lambda = Lambda.';
    V = V(:, I);

    LapX = V(:,1:min(m,n));
    Lambda = Lambda(1:min(m,n));
end

function S = similarity_matrix_corr(X)
    S = squareform(pdist(X,'correlation'));
end
```

```
function [LapX,Lambda] = lap_embed_corr(X, m)
    S = similarity_matrix_corr(X);
    n = length(S);
    d = double(zeros(1,n));
    for i=1:n
        d(i) = sum(S(i,:));
    end
    P = double(zeros(n));
    for i=1:n
        for j=1:n
            P(i,j) = S(i,j)/d(i);
        end
    end
    [V,Lam] = eig(P);
    Lambda = diag(Lam);
    [list,I] = sort(abs(Lambda),'descend');
    Lambda = Lambda(I);
    Lambda = Lambda.';
    V = V(:, I);

    LapX = V(:,1:min(m,n));
    Lambda = Lambda(1:min(m,n));
end
```