IDENTIFYING GENE EXPRESSION PATTERNS ASSOCIATED WITH INCREASED SURVIVAL TIME ACROSS CANCERS

1 BACKGROUND

MOTIVATION

Aside from predicting drug efficacy, genetic biomarkers also function as important tools for cancer treatment. For example, by examining gene expression differences in tumors from different breast cancer patients and comparing their survival, it may be possible to learn what genetic differences lead to better breast cancer prognosis. However, performing this analysis *across* different types of cancers will hopefully identify genetic patterns that contribute most to cancer patient survivorship in general. If multiple cancers can be shown to function similarly at a genetic level, then it may be possible to repurpose a drug therapy developed for one cancer to another.

Unfortunately, this is a complex task because humans have over 20,000 genes. Simultaneously examining expression in these many genes for thousands of patients proves to be quite challenging. Therefore, we will utilize machine-learning methods to handle this complexity.

STATE OF THE ART

To date, researchers have tried using weighted co-clustering methods to identify cancer subtypes by assigning weights to genes that are differentially expressed across tumors (Figure 1) [1]. However, consensus clustering has been inconclusive in predicting survival outcomes and hence there is a need to develop new methods to determine the relationship between differential expression of genes and increased survival times across cancer patients.

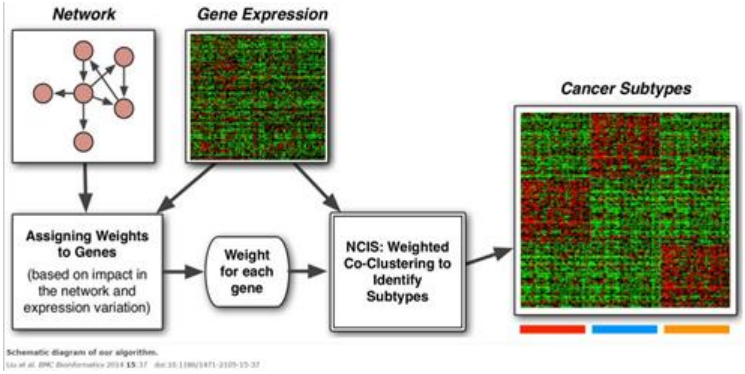

Figure 1: Cluster of cluster algorithm used in previous gene expression studies across cancer types.

Approach Overview

In this project, we apply supervised machine learning methods to publicly available cancer gene expression data to identify genes expression patterns that are associated with increased survival across multiple cancers (if such patterns exist). Our algorithms include Least Absolute Shrinkage and Selection Operator (LASSO) and Classification and Regression Tree (CART) compared to a standard method, logistic regression. In our analysis, we will create a meta-dataset of gene expression data from breast, ovarian, and uterine cancer tumors paired to corresponding clinical data with survival outcome. Our goal is to identify a subset of genes that are associated with increased survival time across multiple cancers (Figure 2)

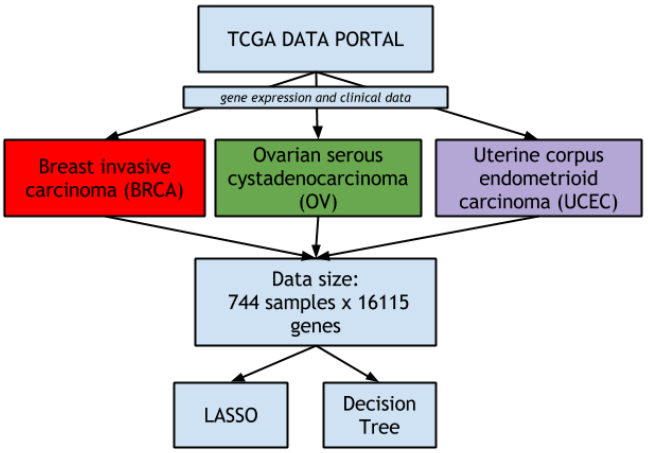

Figure 2: Schematic design for our study data.

2 Methods

Data Collection

We downloaded breast, ovarian, and uterine cancer gene expression and clinical data from The Cancer Genome Atlas (TCGA). TCGA is a large database aimed at cataloging genetic mutations that are responsible for a variety of cancers by sequencing tumors and measuring gene expression from patients in both tumor and normal tissue [2]. It involves multiple research institutions and is managed by the National Cancer Institute (NCI) and the National Human Genome Research Institute (NHGRI), which are part of the National Institutes of Health (NIH).

Data Preprocessing

Training examples downloaded from TCGA were combined to create a meta-dataset of gene expression values paired to corresponding clinical data. Our total dataset is comprised of 1,455 tumor samples and expression data from 16,115 genes (Figure 2). Only samples that had gene expression data available were selected for our analysis. Furthermore, any genes whose

expression values were "NA" were removed from the feature list. Gene expression values were normalized to a scale of -1 to 1.

Clinical information for each patient was determined by looking at time-to-event (survival analysis). Because different cancers can have widely different expected survival rates, we chose to define survival by examining each patient's time-to-event in relation to the median survival time for the cancer type. In order to most equally divide the groups, survival time cutoff was set to the median survival time for each cancer multiplied by 1.5. If the patient survived longer than the median survival time, their outcome was denoted +1 for better survival. Otherwise, the patient's outcome was denoted -1. Patients who were lost in follow-up before the cutoff time were not included in our analysis because we were unable to ascertain their vital status in relation to the cutoff time. In the combined dataset, 387 patients survived longer than the cutoff survival time, and 357 patients did not survive past the cutoff survival time. The median survival times, and corresponding survival cutoff criteria, for each cancer type are listed in Table 1.

Table 1: Median survival times for each cancer were multiplied by 1.5 to determine the cutoff criteria that equally divided patients in each cancer type.

Cancer Type	Median Survival Time	Survival Cutoff (Days)
Breast	1,448	2,172
Ovarian	1,075.5	1,613.25
Uterine	456	684

DATA SUBSETS

For algorithm implementation, the dataset was randomly split into two clusters: 2/3 used for training, 1/3 saved for final testing.

To build the initial models, we used an even smaller subset of the original dataset to shorten run time and produce preliminary results for the milestone. Genes were filtered and selected by median absolute deviation; the value required being greater than .1. Selecting the subset this way increased the likelihood of choosing informative genes, because those features have more variation in expression values. This selection resulted in a set of 2,550 genes. Furthermore, the data subset was limited to a random sample of 50 patients.

LASSO Theory and Implementation

LASSO (Least Absolute Shrinkage and Selection Operator) is the primary method we have explored for this project. LASSO aims to build a model using only the most informative variables [4]. It reduces many of the feature coefficients to 0, which ultimately removes redundant features [5]. In this way, we will be able to select a subset of features which pinpoint the most informative genes for predicting increased survival time across cancers.

Our methodology for implementing LASSO follows closely the algorithms described in *Stochastic Methods for l1-regularized Loss Minimization* [6]. For the ith sample, (i=1, …, n), let $x^{(i)} = [x^1, \ldots, x^m]$, where $x^{(i)} \varepsilon [-1,1]$, be the *m* x 1 gene expression profile vector and $y^{(i)} = [y_1, \ldots, y_m]$, where $y \varepsilon \{-1,+1\}$, be the *m* x 1 survival data vector. Let $\theta^T = [\theta_1, \ldots, \theta_n]^T$ be the *n* x 1 vector of quantitative weights assigned to each feature. Thus, our problem takes the following form:

$$\min_{\theta \varepsilon \boldsymbol{R}^d} \frac{1}{m} \sum_{i=1}^{m} [L(< \theta^T, x^{(i)} >), y^{(i)}] + \lambda ||\theta||_1$$

In our problem, $\lambda > 0$ is the regularization parameter and will be optimized using cross validation. $L: \boldsymbol{R}^d * Y \rightarrow [0, \infty)$ is a non-negative loss function. We will use stochastic coordinate descent, which is considered to be a good method for large scale loss minimization [6].
The error loss function for using LASSO is as follows:

$$e(\theta) = \sum_{i=1}^{m} \ln[1 + \exp(-y^{(i)} \theta^T x^{(i)})]$$

The logistic error function, which we will minimize over θ_j, is:

$$\left(\frac{\partial E}{\partial \theta_j}\right) = \sum_{i}^{m} -\frac{y^{(i)}}{e^{y^{(i)} \theta^T x} + 1} x_j^{(i)}$$

Decision Tree Theory and Implementation

We chose to implement a decision tree for our second algorithm in this project. The most appropriate method to implement with our decision tree for this project was CART (Classification and Regression Trees). Specifically, we used the classification component in terms of patients surviving well or poorly for the binary outcome.

Classification trees are built by categorizing examples via a splitting rule, which divides the data into two subgroups for further analysis. The examples are split at a parental node to maximize homogeneity within each child node. The splitting criteria are determined by the impurity function. For our algorithm, we chose to implement the Gini index as the impurity function. It tends to work well for noisy data and it is an appropriate choice given that the classification label is categorical [7]. The Gini Splitting Rule is shown below, where *k, l* = index of the class:

$$i(t) = \sum_{k \neq l} p(k|t)p(l|t)$$

At each node, the following equation is maximized to determine the right value upon which to split to minimize N, where j = 1,…,m to be the number of examples:

$$\arg\max_{x_j \leq x_j^R} [i(t_p) - P_l i(t_l) - P_r i(t_r)]$$

In order to choose the right-sized tree, it is necessary to prune. To prune a decision tree, we analyze the cost-complexity function, which weights the accuracy of the tree against the complexity of the tree in the following equation:

$$R_\alpha(T) = R(T) + \alpha(\tilde{T})$$

Over-fitting is a common symptom of decision trees; the tree that has the highest accuracy on the training data may perform poorly on subsequent data. Because generalizability is a vital component for our project, it will be necessary to prune the maximal tree and select the pruned tree that performs most consistently.

LOGISTIC REGRESSION

For a baseline method of comparison, we implemented logistic regression. Logistic regression is a probabilistic method for classification, which is often used to determine the binary outcome of a categorical random variable. This method provides us with a reference point for our project, because it allows us to determine how well we will be able to classify patients according to survival. However, given the number of features being considered in this project, it is unlikely that logistic regression will be able to generalize across datasets. Although the method is not particularly well suited to identify a small subset of predictive genes, it will be able to show us the magnitude of information each gene contributes to the overall outcome. Additionally, it allows for comparison of the misclassification rates between our other implemented models.

The logistic function is defined as follows, where θ is a vector of weights assigned to each feature, $x^{(i)}$:

$$y^{(i)} = \frac{1}{1 + e^{\theta^T x}}$$

3 RESULTS AND CONCLUSIONS

LASSO

LASSO was implemented in Matlab. The primary purpose of utilizing LASSO for this project was because it works to keep most coefficient values around zero while letting only the most informative features' coefficients grow. To visually depict this, a snapshot of the coefficient values for all of the features was taken at every iteration. These values are shown plotted in Figure 3. Each line represents a certain feature's coefficient through the training process. It is easy to see that most coefficients are held close to zero, while others increase over time.

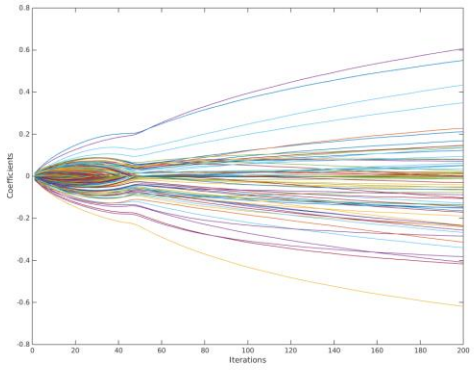

Figure 3: Coefficient values in LASSO over many iterations.

Validation results of LASSO are shown in Figure 4. The blue line represents the average misclassification rates for the ten-fold cross validation across different levels of lambda, which ranged from 1e-6 to 0.5. The red line shows the test data misclassification. The step size (α) used for coordinate descent is a scalar value set to 2.

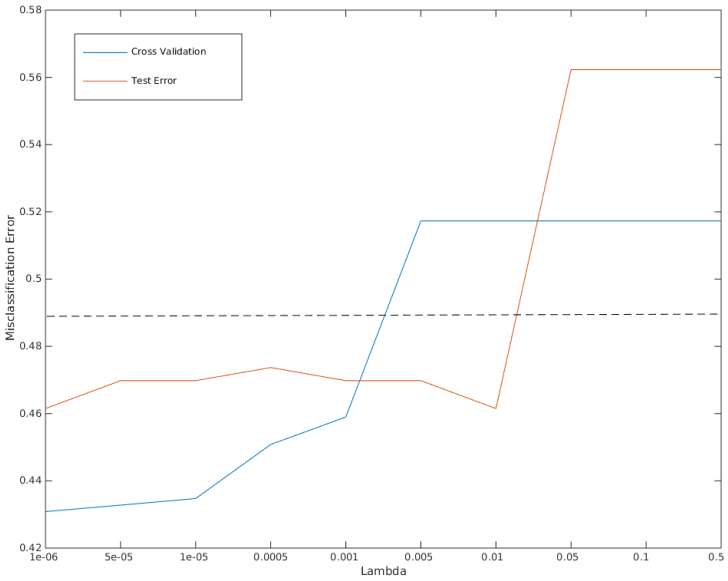

Figure 4: Cross validation results on the training data across values of lambda shown in blue. Test data misclassification shown in red.

In our implementation of LASSO, we chose to limit the number of informative genes (coefficients allowed to deviate from zero) to 1,000. This subset of 1,000 most informative genes was then further analyzed to determine the magnitude of information for each gene. In order to better understand how the informative features contributed to survival, we plotted the 1,000 genes according to fold change in coefficient (Figure 5). Genes that are most differentially expressed between patients surviving well and poorly have a higher likelihood of directly contributing to the overall outcome.

We can see in this graph (Figure 5) that even though this subset consists of the most informative genes according to LASSO, most of the genes are still not highly differentiable; in fact, many of the genes still cluster around zero. This may indicate that we could have limited our subset of genes in LASSO to smaller than 1,000 genes. Additionally, this graph allowed us to select an even smaller number of genes to investigate as to their known relation to cancer therapy. Three such genes of interest are shown in Table 2, below. Other genes, such as MLEC (Malectin) and SLC7A4 (solute carrier family 7, member 4: transport protein) have not

previously been implicated in breast, ovarian, or uterine cancer but may be important in further cancer investigation and drug therapy repurposing.

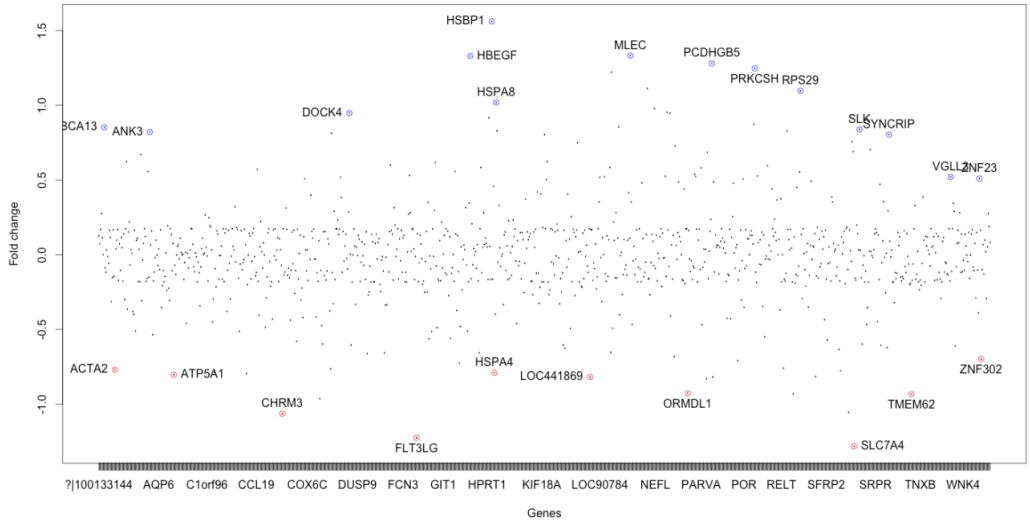

Figure 5: Fold change values in gene expression for 1000 top genes from LASSO.

Table 2: Interesting genes pinpointed from LASSO that are differentially expressed in patients surviving well and poorly

Gene	Description	Cancer Relation
HSPB1	Heat shock protein	Currently being investigated as a drug target
HBEGF	heparin-binding EGF-like growth factor	Role in cellular proliferation
PCDHGB5	protocadherin gamma subfamily B, 5	Cell adhesion protein

DECISION TREE

The maximal tree created is shown in Figure 6. It consists of 36 nodes which correspond to the 36 most informative genes used in this decision tree. The maximal tree was able to correctly classify 99% of the data. Each gene and its corresponding importance level is shown in Table 3. Genes which are bold in the table were also identified in the top 1,000 genes from LASSO. The importance is determined by the nodes' ability to create homogeneity in its respective child nodes.

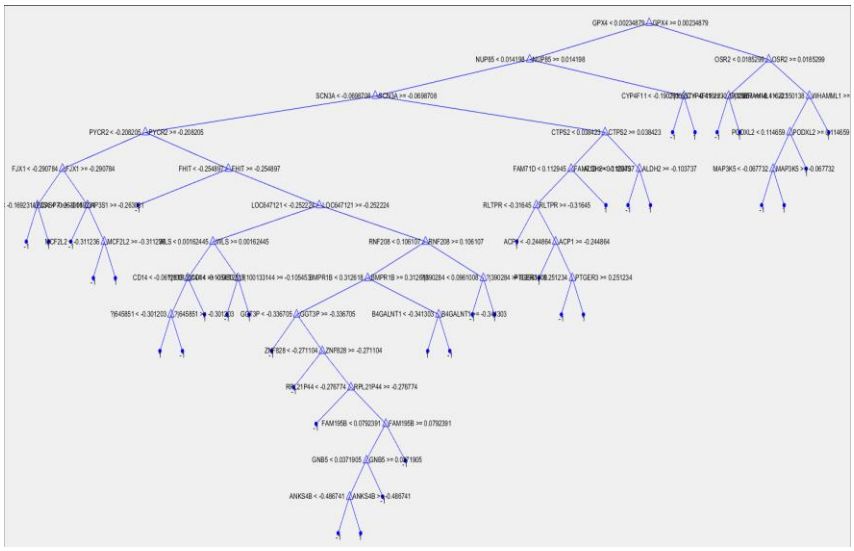

Figure 6: The maximal decision tree.

Table 3: Genes identified in maximal tree.

Gene	Importance
FHIT	100.0%
SCN3A	95.3%
GPX4	91.4%
RNF208	87.2%
NUP85	79.2%
WLS	75.9%
PYCR2	73.9%
OSR2	68.1%
BMPR1B	67.6%
WHAMML1	65.8%
LOC647121*	63.0%
GGT3P	62.0%
CTPS2	60.9%
FJX1	57.9%
PODXL2	55.5%
ZNF828	53.5%
AP3S1	50.2%
FAM71D	47.7%
RPL21P44*	47.2%
ALDH2	44.0%
CASP7	44.0%
MCF2L2	43.4%
CYP4F11	41.1%
CD14*	40.6%
RLTPR	40.4%
MAP3K5	40.2%
FAM195B	39.6%
B4GALNT1	37.6%
GNB5	30.7%
PTGER3	30.0%
ACP1	28.4%
ANKS4B	19.8%
10357	15.1%
645851	15.0%
390284	14.5%
10013144*	13.9%

Pruning the maximal tree was the second step in implementing the decision tree methodology. In order to prune the tree to the correct level, we performed ten-fold cross-validation on each pruned tree. This is shown in Figure 7. The misclassification rate (often referred to as "resubstitution") for the training data is also shown in this graph. As expected, the misclassification rate is inversely related to tree complexity; a smaller tree yields higher misclassification. This step was necessary, however, in order to produce a more generalizable tree due to the fact that decision tress tend to over-fit training data. Over-fitting in the maximal tree held true in our scenario – the

test data was misclassified 87%. Additionally, we compared test data misclassification results for each pruned tree.

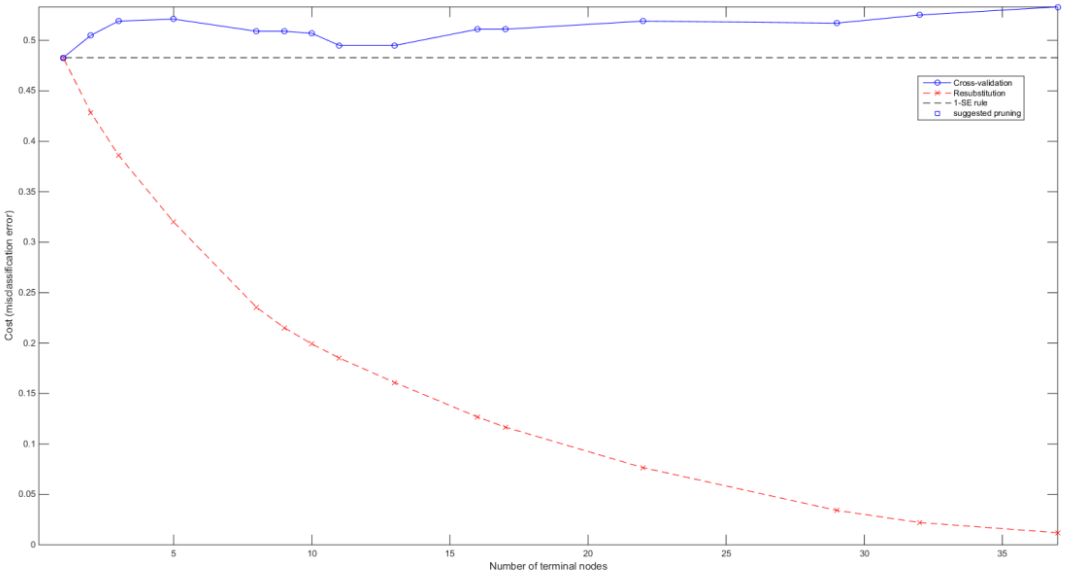

Figure 7: Cross-validation on prune level.

Apparent from Figure 7, the cross-validation results tended to not change drastically with tree complexity. However, the test data misclassification greatly varied. The best pruned tree (prune level = 13) was the tree which only contained one node (corresponding to the decision made by the very first gene on the top of the tree. This tree is shown in Figure 8. Still, the misclassification rate on the test data for this tree was 68%.

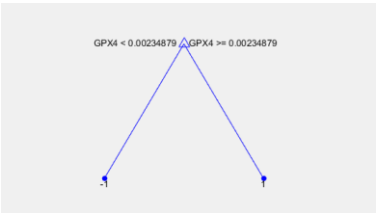

Figure 8: The best pruned tree. (Prune level =13)

While the pruned tree performed considerably better than the maximal tree, both trees are essentially non-informative in terms of classification. However, this is not to say that the decision tree does not provide contextual value. The most informative gene, FHIT, is a protein coding gene which has previously been implicated in breast and ovarian cancer studies. While our decision tree suffered greatly from over-fitting, it was still able to identify genes which could contribute to cancer survivorship. A likely explanation for this outcome is that the decision tree is modeling several paths to better survival. In order to remedy this problem in future work,

random forest should be implemented. In terms of our project goals, the genes which are consistently used in the trees through the forest should be analyzed for biological significance.

Logistic Regression

A logistic regression model was trained and tested for baseline comparison to our other models. Unfortunately, the model misclassified 47.69% of the training data and 48.18% of the testing data (these results are slightly different than what was shown in the poster presentation due to the discovery of a bug in the logistic regression).

4 Discussion

Cross Comparison Between Models

LASSO and logistic regression are naturally very similar, in that they are both based on maximizing the probability of the logistic function. The different methods used (e.g. coordinate descent for LASSO vs. gradient ascent for ordinary logistic regression), do not change that optimization goal. Therefore, one would expect that the results on the training data would be extremely close, which is indeed what we observed. Despite these similarities, there are compelling reasons to use LASSO. 1) The model is sparser. Theoretically, this should mean that the genes it uses are the most informative and least vulnerable to noise in the data. Therefore, the model should generalize better than what is found by ordinary logistic regression; 2) The sparse model itself is helpful in determining which features of the data are worth investigating further (although something of the sort can be accomplished simply by looking at the coefficient values from logistic regression and ranking the features by the size of the coefficients).

Compared to the decision tree, both LASSO and logistic regression performed measurably better. The decision tree, despite its poor classification, should not be immediately discredited. Even though the decision tree was non-informative in terms of correctly classifying patients, it was still able to identify some known genes which are associated with cancer and impact survivorship. Thus, this algorithm promises potential in future studies which could utilize random forests to help with the over-fitting problem. However, for this project, we decided to

only proceed with the best results (genes in the LASSO subset) for external validation of our methodology.

External Validation

To externally validate the results of our machine learning algorithms, we performed a gene enrichment analysis using *GOrilla*. Given a list of genes, enrichment analyses examine the genes' annotations to determine functions and processes which are either over- or under-represented within the set. This analysis is performed taking into account the background (sample) frequency and assigning a p-value to gauge significance of a particular finding [3]. *GOrilla* was chosen for our gene enrichment analysis because it is best used for ranked gene lists [8]. Our ranked gene list from LASSO as input in order to output a graphical representation of likely molecular functions and cell processes being differentially expressed between cancer patients surviving well and poorly in our data analysis. Figures 9 and 10 show output of the gene enrichment analysis. The darker color boxes correspond to more statistically significant results. Many of these results validate our algorithm as a method of identifying biologically importance genes related to cross-cancer survivorship.

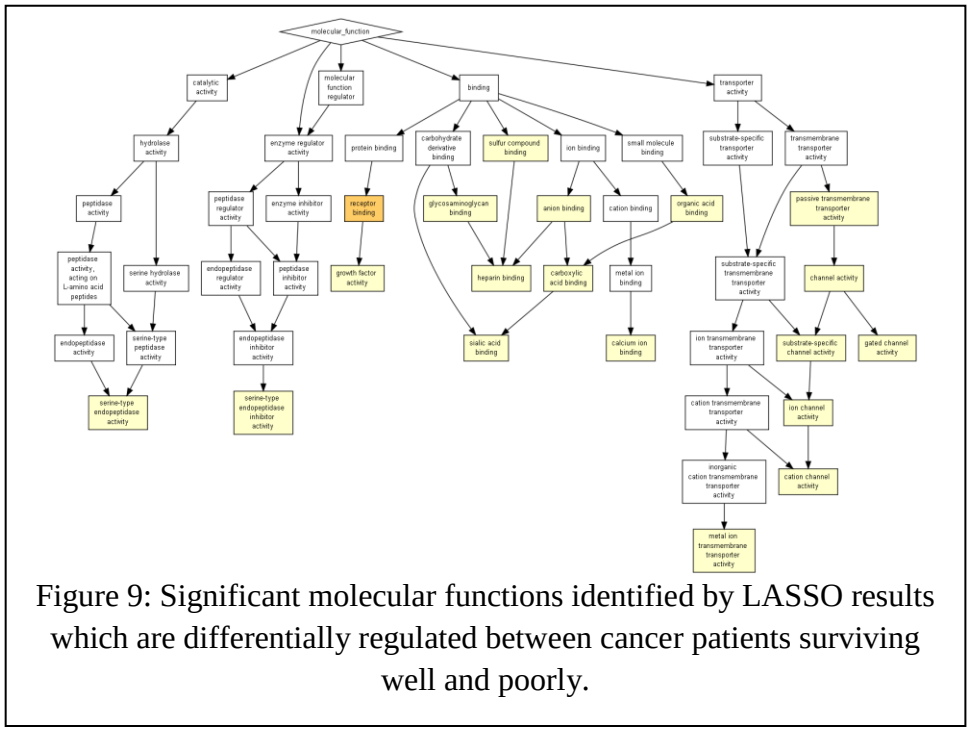

Figure 9: Significant molecular functions identified by LASSO results which are differentially regulated between cancer patients surviving well and poorly.

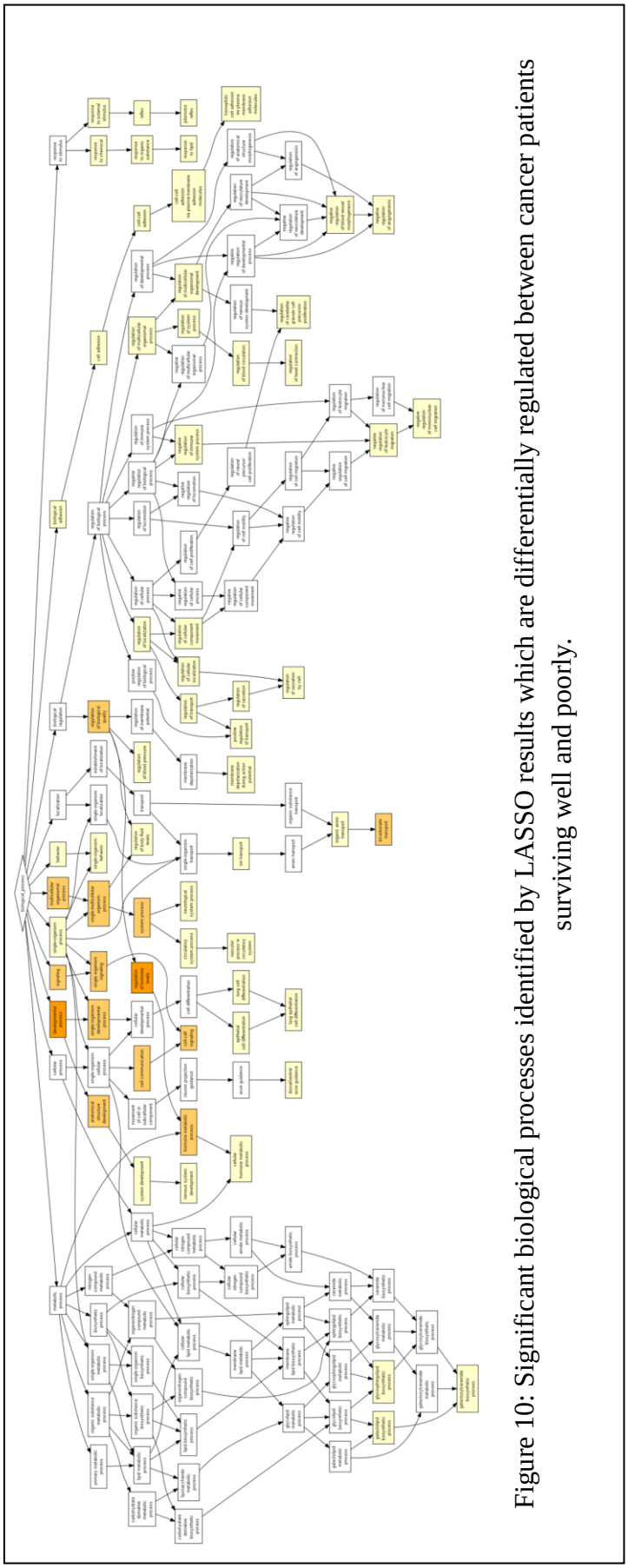

Figure 10: Significant biological processes identified by LASSO results which are differentially regulated between cancer patients surviving well and poorly.

5 REFERENCES

1. Cancer Genome Atlas Research Network, Weinstein, J.N., Collisson, E.A., Mills, G.B., Shaw, K.R., Ozenberger, B.A., Ellrott, K., Shmulevich, I., Sander, C. & Stuart, J.M. (2013), The Cancer Genome Atlas Pan-Cancer analysis project. *Nature genetics*, **45**(10), 1113-1120.
2. Cancer Genome Atlas Network (2008). Comprehensive genomic characterization defines human glioblastoma genes and core pathways. *Nature*, **455**(7216), 1061-1068.
3. The Gene Ontology Consortium. (2005). Gene ontology: tool for the unification of biology. *Nature Genetics*,**25**(1):25-9.
4. Liang, Y., Liu, C., Luan, X.-Z., Leung, K.-S., Chan, T.-M., Xu, Z.-B., and Zhang, H. (2013). Sparse logistic regression with a L1/2 penalty for gene selection in cancer classification. *BMC Bioinformatics*, **14**(1), 198.
5. Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, **58**(1), 267-288.
6. Shalev-Shwartz, S., Tewari A. (2011). Stochastic Methods for l1-regularized Loss Minimization. Journal of Machine Learning Research, **12**, 1865-1892.
7. Timofeev, R. (2004). *Classification and Regression Trees (CART) Theory and Applications*. Center of Applied Statistics and Economics.
8. Eden, E., Navon, R., Steinfeld, I., Lipson, D., Yakhini, Z. (2009). *GOrilla*: a tool for discovery and visualization of enriched GO terms in ranked gene lists. *BMC Bioinformatics*, **10**:48.

6 IMPLEMENTATION

LASSO: contains our LASSO implementation, entirely coded by our group, based on Shalev-Shwartz and Tewari (2010). We also include functions: lasso_pred for prediction, lasso_error for error calculation and lasso_cross_validation_error for cross validation.

CART: original code from A. Padoan, "Decision Trees and Predictive Models with cross-validation and ROC analysis plot" modified by us for our project. Modifications include changing data input type and format, output files, validation scheme, etc.

C4.5: Because the CART code calls to Matlab's built-in tree functionality, we aimed to write our own entire decision tree code, based on Quinlin's C4.5 algorithm. We ran into trouble generating the tree graphs, although the algorithm itself is complete. It was basically a matter of time that we were not fully able to complete this code in place of the CART code. There is also a function included that we wrote to extract the nodes from the resulting tree, in preparation for plotting it.

LOGISTIC: This is essentially just the logistic regression code that we developed for the homework.