|  |
| --- |
|  |
| **Predicting Gene Expression using Classification Models on TCGA Data** |
|  |

**Michael LeeJonathan Wolf**

Department of Computer Science Department of Computer Science

University of Washington University of Washington

Seattle, WA 98105 Seattle, WA 98105

*leem42@uw.edu* *wolfj4@cs.uw.edu*

**Abstract**

Gene expression refers to the process by which a functional gene is produced by the information from a gene. This process is ubiquitous in all organisms and understanding it better is all the more pertinent in a cancer data set. In this paper we predict gene expression for cancer afflicted patients using regression models.

**1 Introduction**

The Cancer Genome Atlas (TCGA) is a comprehensive study profiling thirty-three types of cancer. The goal of the study is to better understand the molecular basis for cancer. This study has generated over two petabytes of data publically available for researchers to analyze. This wealth of data has prompted many computational methods and made headway in finding features of cancer.

**1.1 Purpose**

The goal of this project is to predict gene expression for cancer afflicted patients. Making accurate predictions would provide potential for clinical application.

**2 Methods**

Our methods can be split into two areas of involvement. The first being data retrieval using SQL via Google BigQuery. The second area being our machine learning methods we chose to analyze our data.

**2.1 Data Retrieval**

We constructed our data tables by using SQL to combine elements of several tables on the Institute for Systems Biology (ISB) data set held in a Google BigQuerry table. More information on this data set can be found at source [1]. First, we selected 50 random genes and 500 patients from four cancer types. Then, we joined the clinical data for each patient with the bio-specimen information of each patient’s tumor type. Biospecimen data included the following: average percentage of lymphocytes infiltrated, the percentage of normal/tumor cells of the tumor, and others. The clinical data we used was the patient BMI, age at pathological diagnosis, and the stage of the cancer type. Of the features listed the one of principal interest was the nearest mutation for a specific patient relative to the gene in question. The rational can be seen in figure (1).

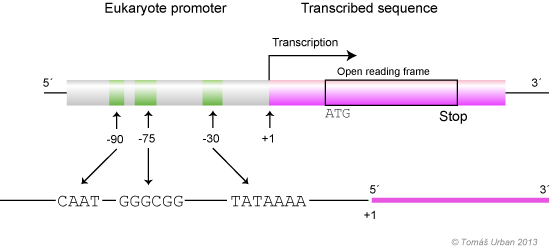


Figure 1: The promoter region is shown, examining proximal mutations is motivated by the many elements with which a mutation could alter transcription. **[1]**

**2.2 Machine Learning Algorithms**

Using a regression was a natural choice at the beginning of our study. The values of the patients were continuous and this model makes predictions accordingly. We chose to examine lasso regression and ridge regression. These three models were tried with variations of parameters for the penalty terms and intercept terms.

\*\*Include idea: we only split on feature once

The second model we used in our study was a decision tree. The motivation for employing this model lay in its capability to make broader predictions than a regression model. Broader meaning that we sought to classify a patient as having high gene expression or low gene expression. We chose the barrier between high expression and low expression as the median value because it represented a split between half the data points. The specific implementation we used was Iterative Dichotomiser 3 (ID3) with pseudo code provided at [2]. One alteration to this algorithm was using optimal thresholds for the deciding the split of each feature since our features were continuous. Our heuristic for this adaptation can be seen in figure (2).

sortedSplits = sorted(observationsfeaturesj)

sortedAltered = sortedSplits[1:]

sortedSplits = sortedSplits[0:len(sortedSplits -1)]

sortedSplits = set((sortedSplits + sortedAltered)/ 2.0)

best = 0

For split in sortedSplits

Compute classification error using split

best = min(best, split)

return best

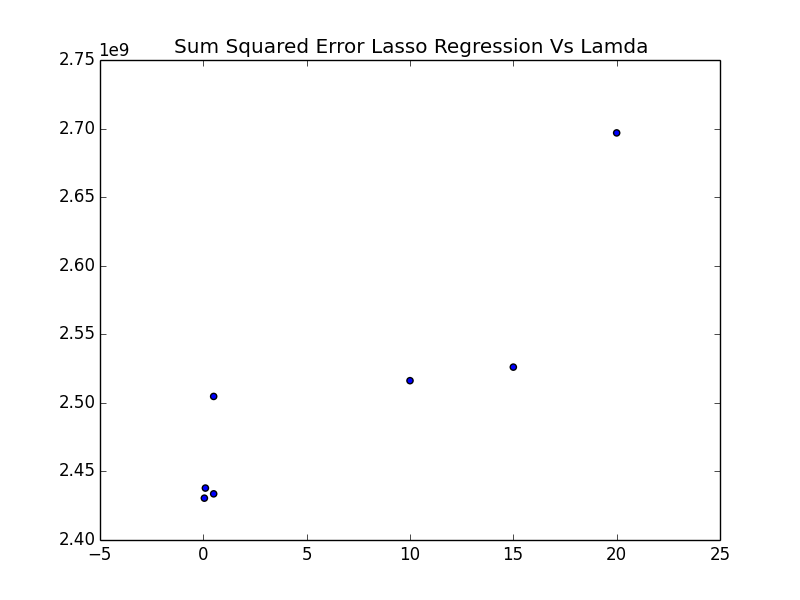
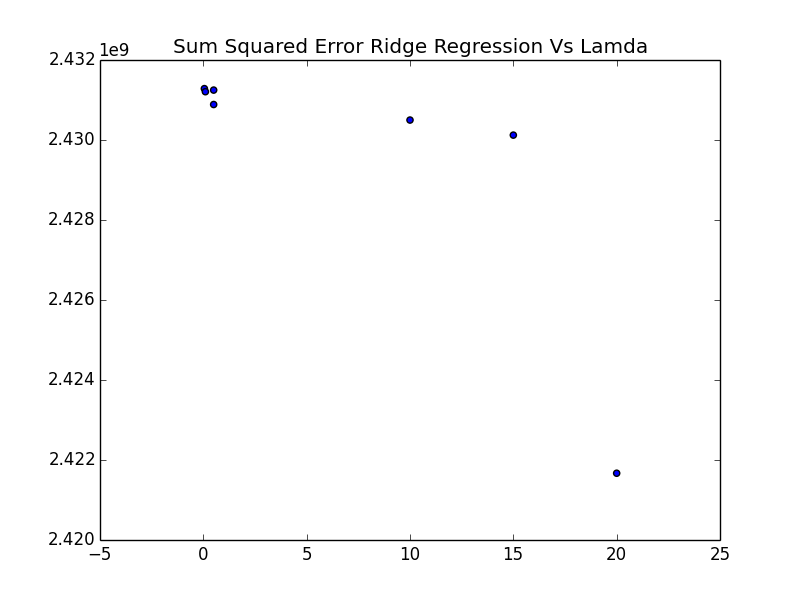
Figure 2: Optimal threshold heuristic for a feature *j*

The last model we considered was an AdaBoost framework. This model was chosen because it is another classifier that we could compare to the result of our decision tree model. Our framework used depth one decision trees for its base learners. We also normalized our weights for each data point after each iteration of the model. Further, to optimize the number of iterations trained over the data we used cross validation techniques. In particular, we implemented a 5-fold cross validation to pick the value of T such that it minimized error.

**3 Results**

**3.1 Regression Results**

Shown below is the results of our regression model and its sum squared error over the training data. The sum squared error for the Ridge model was 2.42 billion on average for multiple values of lambda. For comparison, the response for the 2300 testing data had a total sum squared value of 3.3 billion. We believe that the model performed poorly because of the complexity of the response variables. The variance in the response was high, it was 1083511 RPKM. The lasso model was comparable to the ridge model, although with slightly more error (~2% more error). We had expected this model to perform better. This is because we had reason to believe that some of the features would would have been eliminated by the algorithm (ie. set to zero).



[Figures 3,4: Sum Squared Error for Ridge Regression Versus Penalty Term Coefficients (left). Sum Squared Error for Lasso Regression Versus Penalty Term Coefficients (right)]

**3.2 Decision Tree Results**

**\*\*\*include the max-depth of the tree**

Our decision tree was able to perform with 2.5% classification error rate. Where the classification was the median split of the response variable, this was 238 RPKM in our data set. This boundary marked in real life our boundary of “high” expression versus “low” expression. The tree had a depth of thirtee

\*had multiple features, that we combined into one

\*then got way better performanc

\*this was implicityy a cap to the tree depth

**3.3 AdaBoost Results**

**4 Conclusion**

**5 Discussion**

**Acknowledgments**

**References**

[1]

http://isb-cancer-genomics-cloud.readthedocs.io/en/latest/sections/progapi/bigqueryGUI/WalkthroughOfGoogleBigQuery.html

[2] http://web2.mendelu.cz/af\_291\_projekty2/vseo/print.php?page=307&typ=html