Review of:

“*Genome-wide transcriptional analysis of T cell activation reveals differential gene expression associated with psoriasis”* [1]

Sam Brown, Wayne Kunze, Caleb Perry

EECS E6690, Spring 2018

# Introduction

The paper selected for review investigates the genetic mechanisms that cause activation of T cells, responsible for skin inflammation in sufferers of Psoriasis, a common autoimmune disease.

The paper discusses findings from T cell gene expression data collected from 24 individuals, 17 of which had Psoriasis. The dataset has expression rates for thousands of different genes, all of which are identified with markers. The paper concluded that two genes SPATS2L and KLF6 are most strongly associated with T cell activation in Psoriasis patients. The dataset itself was unique interesting due to the much larger number of features compared to amples. This fact plays an important role in much of the analysis conducted throughout this paper.

In addition to attempting to replicate the results of the paper, we chose to examine the data from a several new perspectives. We used Supervised learning methods such as decision trees to identify the genes indicating a patient has psoriasis. We also demonstrated how Support Vector Machines (SVM) and regression techniques can be used to train very accurate models. Since the dataset has many more components than data points, feature reduction was performed using Principle Component Analysis (PCA) as well as shrinkage and subset methods. Feature Reduction allowed us to reduce the dimensions of the data by several orders of magnitude. Lastly, we also used unsupervised learning techniques in which we left the response variable unused to see if clusters could be found indicating the presence of the disease.

# Data Set and Paper

The paper examines Gene expression profiles from in vitro activated T cells from 17 psoriasis patients and 7 control subjects. The data set developed for this study contains 47,222 transcripts for each sample cataloging the level of gene expression in the activated T-cells for each gene in each individual.

# Results

The paper, “Genome-wide transcriptional analysis of T cell activation reveals differential gene expression associated with psoriasis”, identifies several genes that significantly up-regulated or down-regulated in patients with psoriasis. These results were found by first pruning the data via removing all genes that were not expressed in at least 3 patients. Then the average expression rate of those who had the disease was computed and compared to the average expression rate of those who didn’t have the gene. This was used to produce tables of relative fold changes. Attempts to reproduce these results were made however results differ from the paper. Table 1 shows the 13 most upregulated gene indicators as a result of this analysis.

Table : List of most up-regulated genes

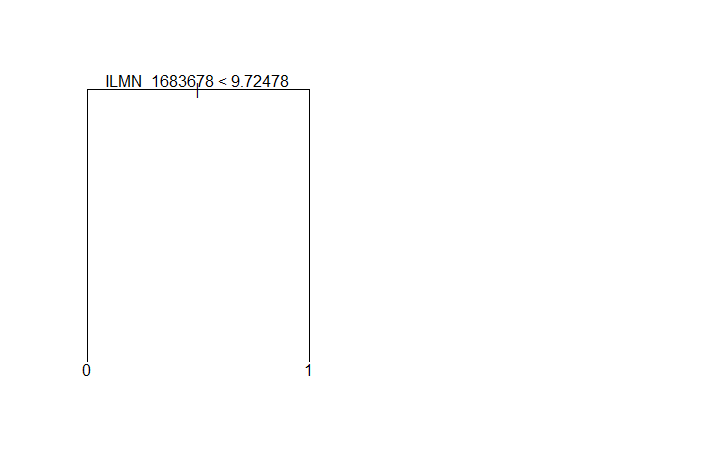
|  |
| --- |
| "ILMN\_2058782" "ILMN\_2305112" "ILMN\_1701789" "ILMN\_2410826" "ILMN\_1721113"  “ILMN\_1658247" "ILMN\_2054297" "ILMN\_2184373" "ILMN\_1739428" "ILMN\_1729749"  “ILMN\_1700967" "ILMN\_2347798" "ILMN\_1670134" |

The results from Table 1 differ from the results displayed in the paper. In the paper the principle gene discussed is SPATS2L, which the paper presented as being upregulated 1.37 fold in patients with psoriasis, however in our analysis, the gene was only upregulated 1.003101 fold. The cause of this discrepancy is unknown.

# New methods used

## Trees

We decided to utilize Random Forests to analyze the data. This is a process that can be computationally intensive. We focused on large forests (each forest with ten thousand trees) but we only considered the default number of factors for each tree.

Working with more than forty-eight thousand factors creates a unique challenge. This is more than RStudio can handle with the randomForest function without crashing. To address this problem, we had to choose a form of factor reduction. We chose to use the importance parameter from randomForest to take a subset of the “more important” factors from subsets of one thousand (or less) factors.

Once the “more important” factors have been identified, we used randomForest to create trees amongst these factors to identify the “most important” factors across the entire dataset.

We were surprised to note very limited overlap from Random Forests to other methods used by ourselves and the original paper. It’s not surprising that ILMN\_1683678 (SPATS2L) always makes the top slot (it has 100 percent accuracy in predicting the end result by itself, see Figure 2), but other factors not predicted elsewhere are also ranked very highly in terms of importance for Random Forests (see Figure 3).

Figure 1 SPATS2L Tree

Creating individual trees from these “most important” factors has fairly good accuracy. We only see 100% accuracy in the case of ILMN\_1683678, but other trees use two factors for 96% accuracy (for example, see Figures 4 and 5).

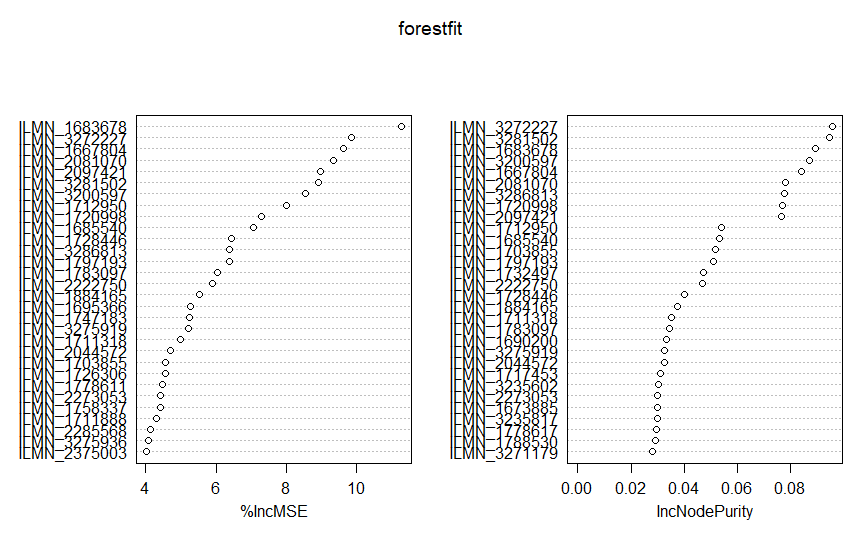


Figure 6 Relative Importance of "Most Important" Factors

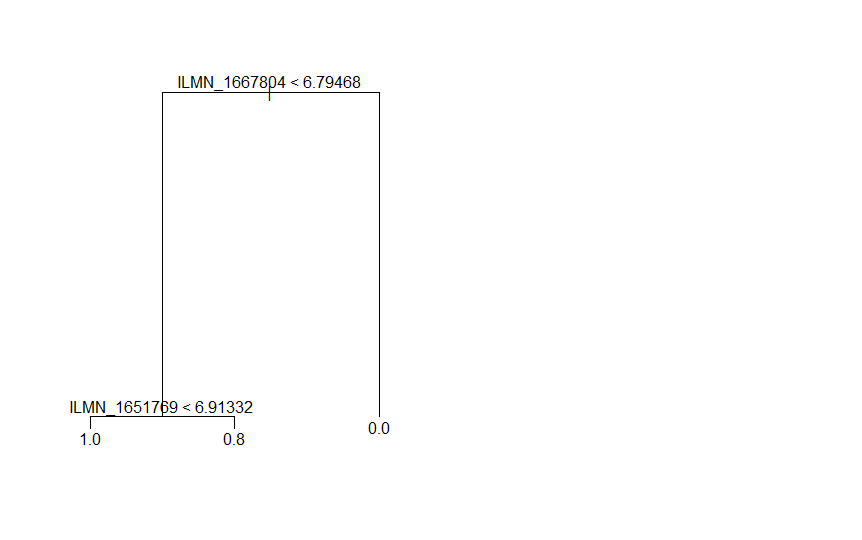
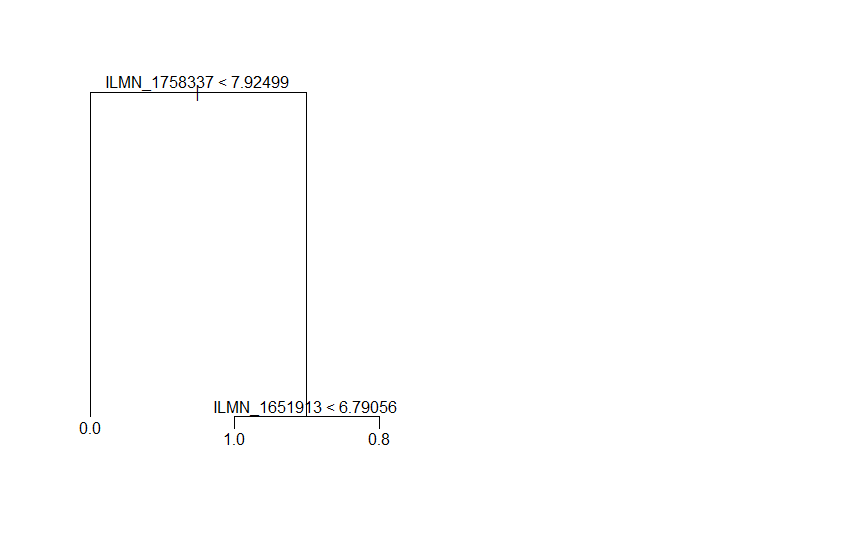


Figure 7 Sample Tree 1

Figure 8 Sample Tree 2

## Dimension Reduction

Traditional machine learning methods start with the assumptions that the model features are (mostly) independent and that there are more samples then there are features (n > p). In our data set there are 47222 features (genes) and only 24 samples (patients), p is three orders of magnitude larger than n. In addition, genes express in groups, therefore many of the features are not independent. Attempting to use traditional methods such as least squares linear regression would result in severe overfitting and a poor predictive model on the Psoriasis dataset.

In this situation methods to reduce the dimensionality (discarding of “unnecessary” features) can offer a more reasonably sized model that may still accurately perform classification.

This paper examines three different methods for reducing a high dimensional data set like Psoriasis: Subset selection, Shrinkage, and Principal Component Analysis. The objective is to perform feature reduction and then assess the effectiveness by comparing the reduced dataset to the list of highly expressed genes from the Psoriasis paper. If the method is effective, then a many of the identified genes should still be present in the resulting, reduced model.

## Subset Selection

Three general approaches to subset selection are commonly used in statistical learning, with speed versus optimality the primary tradeoff. The results of these techniques can be rated using either cross validation or indirect training error estimates, such as RSS or R2

Best Subset will provide an optimal model (based on either CV or indirect estimate) but at a significant processing cost. To reach the optimal model in the Psoriasis data set would require analysis of the almost inconceivably large 24722 potential models and is therefore not appropriate for our dataset.

Similarly, Backward Stepwise Selection, which would start from with the full, 47222 feature model, and works back removing the feature that provides the least value to the indirect estimate. Unfortunately, when the model has more features than samples, like ours, the full model will not fit and therefore cannot be used for our set.

That leaves Forward Stepwise Selection, which starts from the null model and adds the most useful feature at each stage. Forward selection requires inspection of on the order of 472222 potential models, not trivial but also not intractable. Because it starts with a model of no features it can be used when the full model has more predictors than samples.

*Step.r* performed forward stepwise reduction on the dataset using a logistic regression model (appropriate for binary classification) for fitting and produced reduced feature models based on the AIC criteria. The objective was to determine if a “good” reduced model would include the genes identified as important to the activation of T-cells in Psoriasis patients from Table 1 of the original paper.

Table : Forward Stepwise reduced model features

|  |  |  |  |
| --- | --- | --- | --- |
| Probe | Gene | Deviance | AIC |
| ILMN\_1683678 | SPATS2L | 0 | 4 |
| ILMN\_3286813 |  | 10.152 | 14.152 |
| ILMN\_3281502 |  | 11.93 | 15.93 |
| ILMN\_1735014 | KLF6 | 11.956 | 15.956 |
| ILMN\_1781285 | DUSP1 | 12.519 | 16.519 |
| ILMN\_1778617 | TAF9 | 12.753 | 16.753 |
| ILMN\_2321064 | BAX | 13.31 | 17.31 |
| ILMN\_3309534 |  | 13.765 | 17.765 |
| ILMN\_2143250 | FAR1 | 14.448 | 18.448 |
| ILMN\_3200597 |  | 14.904 | 18.904 |
| ILMN\_2119421 |  | 15.149 | 19.149 |
| ILMN\_1860954 |  | 15.579 | 19.579 |
| ILMN\_2285568 |  | 15.642 | 19.642 |
| ILMN\_2246956 | BCL2 | 15.764 | 19.764 |
| ILMN\_1731107 | CCDC92 | 15.922 | 19.922 |
| ILMN\_2397721 | GLB1 | 15.934 | 19.934 |

As Table 2 illustrates, nine of the important genes are selected in the first seventeen features Forward Stepwise identified as the most useful from a minimal AIC standpoint, including SPATS2L and KLF6—the newly identified genes from the Psoriasis paper. The remaining genes appear in the first 108 features selected by the algorithm, implying that a model reduced from ~47000 to ~100 features would still include the primary genetic predictors.

## Shrinkage

The two shrinkage methods attempt to fit the full model through least squares (and attempting to minimize RSS) but with an additive penalty for larger coefficient values.

Ridge regression uses the easier to compute *l*2 penalty but the cost of this is a model that shrinks the magnitude of each coefficient but does not remove them. Figure 9 is a plot of the ridge coefficients as a function of the shrinking penalty and demonstrated how the coefficients trend towards 0.

LASSO is a similar algorithm to Ridge but one that forces many coefficients to exactly zero—offering feature selection/reduction similar to Forward Stepwise. *Lasso.r* performs LASSO and Ridge on a logistic regression model fitted to the dataset across different shrinkage values. LASSO Models were selected manually for feature comparison to the Psoriasis paper list to determine if LASSO selected a similar set.

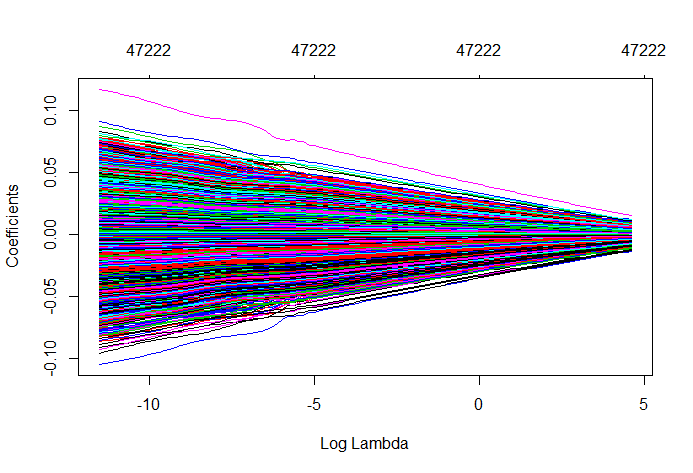


Figure 9: Ridge Regression

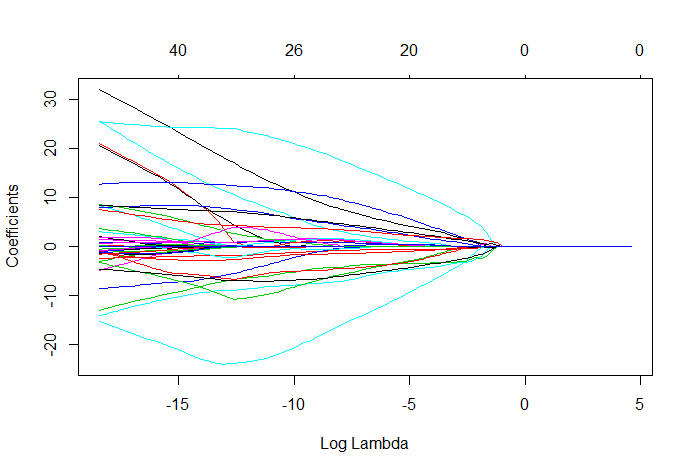


Figure 10: LASSO

Table : LASSO coefficients

|  |  |  |
| --- | --- | --- |
| Probe | Gene | Beta(j) |
| ILMN\_1651913 |  | 21.15 |
| ILMN\_1652431 |  | -3.07 |
| ILMN\_1652784 |  | 0.78 |
| ILMN\_1654942 |  | 3.06 |
| ILMN\_1655827 |  | 1.40 |
| ILMN\_1656052 |  | 20.58 |
| ILMN\_1656421 |  | 0.09 |
| ILMN\_1659378 |  | -0.52 |
| ILMN\_1661886 |  | -1.10 |
| ILMN\_1662807 |  | 8.34 |
| ILMN\_1663767 |  | -4.82 |
| ILMN\_1667804 |  | 2.02 |
| ILMN\_1670219 |  | 0.27 |
| ILMN\_1670385 |  | 3.67 |
| ILMN\_1671004 |  | 0.01 |
| ILMN\_1673885 |  | 25.67 |
| ILMN\_1679647 |  | -0.02 |
| ILMN\_1683036 |  | -1.34 |
| ILMN\_1683678 | SPATS2L | -2.43 |
| ILMN\_1685540 |  | 8.64 |
| ILMN\_1688749 |  | -8.63 |
| ILMN\_1694742 |  | -14.19 |
| ILMN\_1703855 |  | 32.11 |
| ILMN\_1712913 |  | -3.31 |
| ILMN\_1722916 |  | -15.38 |
| ILMN\_1735014 | KLF6 | -0.76 |

The variables selected by LASSO did include the two newly identified genes, SPATS2L and KLF6, but few of the other significantly up/down regulated genes listed. Our suspicion is the standardization of the expression levels masked some of the more strongly expressed genes by other associated genes. An interesting side note is the relatively moderate penalty required to force all the LASSO coefficients to zero. This is not surprising as the penalty is proportional to the sum of a function of the coefficients and with 47222 coefficients this summation will grow to a significant penalty quickly.

## Principal Component Analysis

The final reduction method explored is Principal Component Analysis. This method differs from the previous in that it doesn’t depend on a response to perform feature reduction. Instead it simply the “direction” along which the data is most varied. Subsequent components are found in the same manner with the added requirement that the direction of each new component much be orthogonal to the previous ones (on other words uncorrelated). The result is a reduced set of independent features each blended from the original data set. The downside of PCA versus subset or shrinkage methods is reduced visibility into the original feature set.

PCA.r performs the principal component analysis on the Psoriasis data set. As we are unable to compare the list of genes to that in the original paper the analysis how effectively the PCA based models predict the correct response. LOOCV validation was performed on models using the first 5, 10, 15 and 20 principal components fitted to a logistic regression model with the results displayed in Table 3.

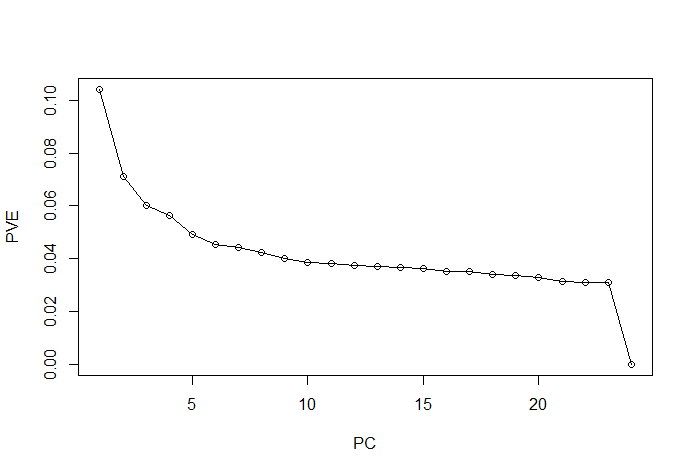


Figure 11: PVE

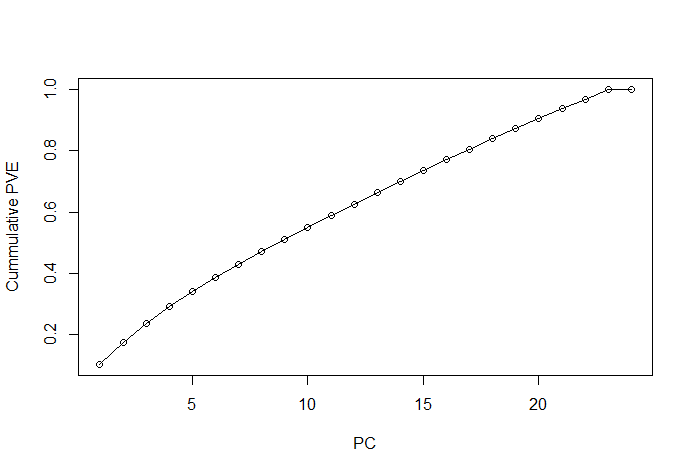
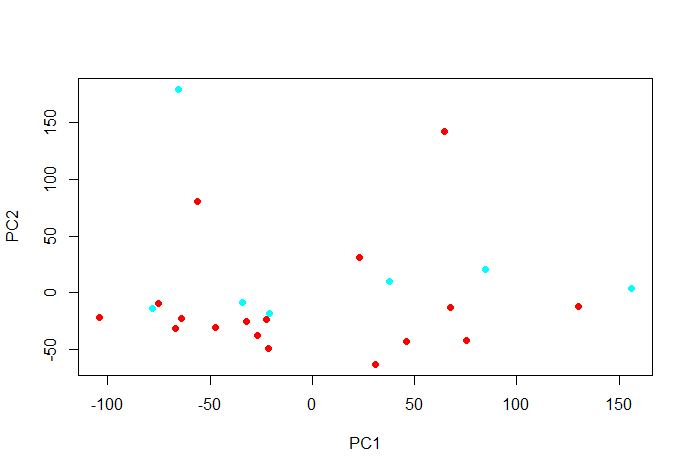


Figure 12: Visualizations of PCs (Red are patients, Blue are control samples)



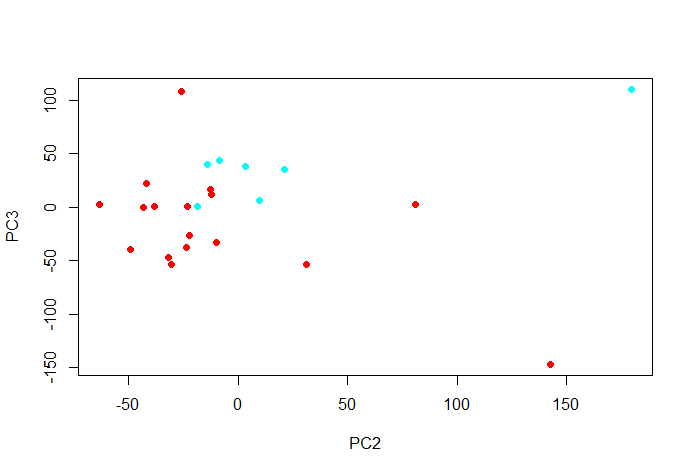


Table : Misclassification Rate for different sized PCA models

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  | All Samples | LOOCV | | | |
|  | Actual | 10 PCs | 5 PCs | 10 PCs | 15 PCs | 20 PCs |
| GSM1152973 | Yes | Yes | Yes | Yes | Yes | Yes |
| GSM1152974 | Yes | Yes | Yes | Yes | No | No |
| GSM1152975 | Yes | Yes | Yes | Yes | Yes | Yes |
| GSM1152976 | Yes | Yes | Yes | Yes | Yes | No |
| GSM1152977 | Yes | Yes | Yes | Yes | Yes | Yes |
| GSM1152978 | Yes | Yes | Yes | Yes | Yes | No |
| GSM1152979 | Yes | Yes | Yes | Yes | No | No |
| GSM1152980 | Yes | Yes | Yes | Yes | Yes | Yes |
| GSM1152981 | Yes | Yes | Yes | Yes | Yes | No |
| GSM1152982 | Yes | Yes | Yes | Yes | Yes | Yes |
| GSM1152983 | Yes | Yes | Yes | Yes | Yes | Yes |
| GSM1152984 | Yes | Yes | Yes | Yes | Yes | Yes |
| GSM1152985 | Yes | Yes | Yes | No | No | Yes |
| GSM1152986 | No | No | No | Yes | Yes | No |
| GSM1152987 | Yes | Yes | No | Yes | No | Yes |
| GSM1152988 | No | No | No | No | No | No |
| GSM1152989 | Yes | Yes | Yes | Yes | Yes | No |
| GSM1152990 | No | No | Yes | No | No | No |
| GSM1152991 | Yes | Yes | Yes | Yes | Yes | No |
| GSM1152992 | No | No | No | No | No | No |
| GSM1152993 | Yes | Yes | No | No | No | Yes |
| GSM1152994 | No | No | Yes | Yes | Yes | No |
| GSM1152995 | No | No | No | Yes | No | No |
| GSM1152996 | No | No | No | Yes | No | Yes |
| Error |  | 0 | 4 | 6 | 7 | 8 |
| Error Rate |  | 0.00% | 16.67% | 25.00% | 29.17% | 33.33% |

Figure 11 illustrates the “Proportion of Variance Explained” by each principal component. The first component only explains ~10% of the variance with 100% of the variance reached with 23 principal components. Figure 12 helps visualize the relationship between PCs, with the PC2/PC3 plot demonstrating good clustering between the two sample groups. Table 4 illustrates an interesting result: as more principal components were added to the model, the ability of the model (using LOOCV validation) to accurately predict the presence of Psoriasis worsened. Examination of the scree plot of PVE in Figure 11 provides a clue to this result. At approximately seven principal components there is a “knee” were the additional amount of variance explained levels out. From this point on the additional principal components add little new information to the model in exchange for greatly increased noise and unwanted flexibility.

## SVM

Another statistical learning method that was used was Support Vector Machines (SVM). Here many SVMs were trained on the data using a grid search with epsilon ranging from 0 to 1 in increments of 0.01, and the cost ranging from 22 to 29 in powers of 2. Each of these 700 different SVMs were tested using 10-fold cross validation to determine optimal choices for cost and epsilon. The Process was highly computationally expensive and took over 8 hours of computation time, however it yielded strong results. The best SVM utilized an epsilon value of 0 and a cost of 4. This SVM had a Root Mean Squared error of 0.0001073526, which is highly accurate. Figure 13 shows the error in each of the 24 patients. The data points were found to be completely separable. Figure 14 shows the results of the training. As can be seen larger values of epsilon produced higher error, with cost being somewhat independent of error. These results are to be expected, with data where the number of components from each sample far exceeds the number of samples. Since the data is perfectly separable, cost therefore has nearly no impact on the SVM.

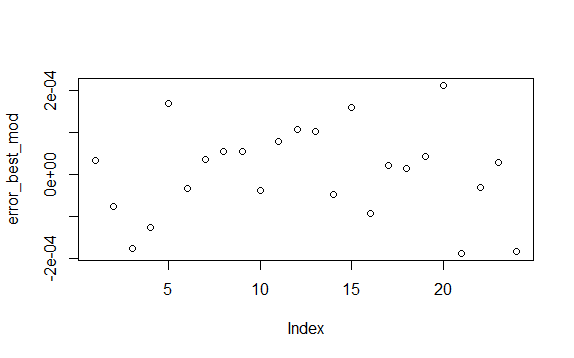


Figure 13: SVM error for each Patient

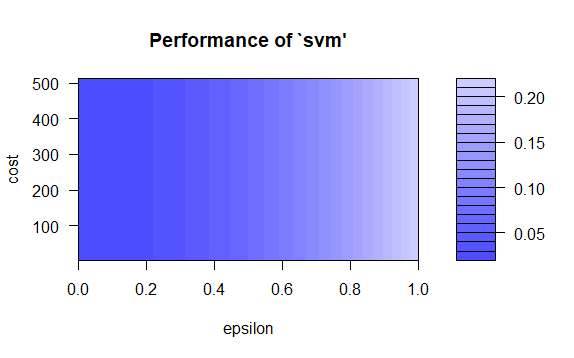


Figure 14: Performance of SVM using Grid search on Parameters

Unsupervised learning methods were also tried on the data. K-means was attempted to determine if clusters would form around the between the two classes without using the response variable. An important metric for K-means clustering is the distance. Figure 15 shows Euclidean distance between all points while Figure 16 shows the Manhattan distance. From these distance plots, one can see that some patients had more similar gene expression rates than others. K-means clustering was done using the Euclidian distance, with 50 random starts, taking the best of the 50. Attempting to split the data into 2 clusters produced clusters of size 9 and 15; however the patients in each cluster had a similar rate of the disease compared to the original cluster. PCA was done to make the data viewable and Figure 17 shows the 2 clusters. Expanding the search to look from 2 to 8 clusters, the presence of the disease did not seem to be hugely impactful. Using 6 or more clusters, the patents who did not have the disease appeared to be cluster more; however, the disease still did not seem to be a significant factor. Table 5 shows the results of clusters with various sizes. Here patients who did not have the disease are denoted as ‘1’, while those who did are ‘0’.

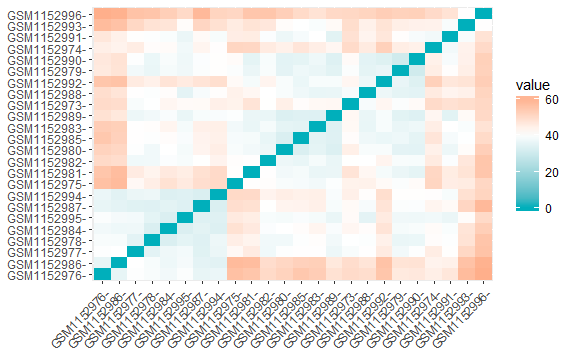


Figure 15: Euclidean Distance

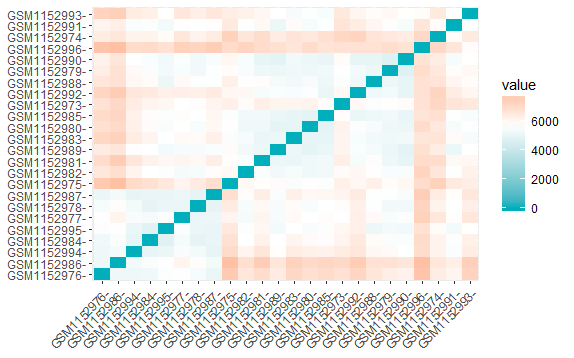


Figure 16: Manhattan Distance

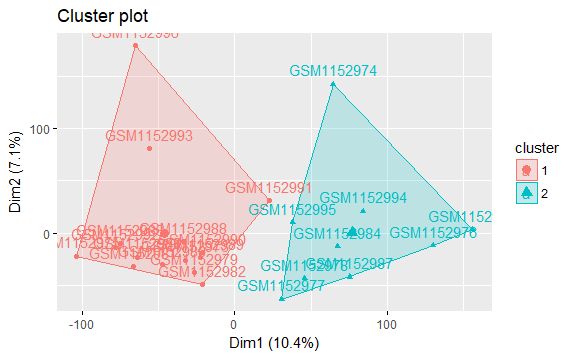


Figure : Clusters found using K-means

Table : results of Clusters with many different cluster sizes

|  |
| --- |
| [1] "Number of Cluster : 2"  Actual: 0 0 0 0 0 1 0 1 1[1] "Cluster: 1 mean: 0.333333333333333 "  Actual: 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1[1] "Cluster: 2 mean: 0.266666666666667 "  [1] "==========================="  [1] "Number of Cluster : 3"  Actual: 0 0 0 1[1] "Cluster: 1 mean: 0.25 "  Actual: 0 0 0 0 1 0 1 1[1] "Cluster: 2 mean: 0.375 "  Actual: 0 0 0 0 0 0 0 0 1 0 1 1[1] "Cluster: 3 mean: 0.25 "  [1] "==========================="  [1] "Number of Cluster : 4"  Actual: 0 0[1] "Cluster: 1 mean: 0 "  Actual: 0 1[1] "Cluster: 2 mean: 0.5 "  Actual: 0 0 0 0 0 0 0 0 1 0 1 1[1] "Cluster: 3 mean: 0.25 "  Actual: 0 0 0 0 1 0 1 1[1] "Cluster: 4 mean: 0.375 "  [1] "==========================="  [1] "Number of Cluster : 5"  Actual: 0 0 0 0 0 0 0[1] "Cluster: 1 mean: 0 "  Actual: 0 1[1] "Cluster: 2 mean: 0.5 "  Actual: 0 0 0 0 1 0 1 1[1] "Cluster: 3 mean: 0.375 "  Actual: 0 0[1] "Cluster: 4 mean: 0 "  Actual: 0 0 1 1 1[1] "Cluster: 5 mean: 0.6 "  [1] "==========================="  [1] "Number of Cluster : 6"  Actual: 0 1[1] "Cluster: 1 mean: 0.5 "  Actual: 0[1] "Cluster: 2 mean: 0 "  Actual: 0 0 0 0 1 0 1 1[1] "Cluster: 3 mean: 0.375 "  Actual: 0 1 1 1[1] "Cluster: 4 mean: 0.75 "  Actual: 0 0 0 0 0 0 0[1] "Cluster: 5 mean: 0 "  Actual: 0 0[1] "Cluster: 6 mean: 0 "  [1] "==========================="  [1] "Number of Cluster : 7"  Actual: 0 1[1] "Cluster: 1 mean: 0.5 "  Actual: 0 0 0 0 1 1[1] "Cluster: 2 mean: 0.333333333333333 "  Actual: 0 0[1] "Cluster: 3 mean: 0 "  Actual: 0 0 0 0 0 0 0[1] "Cluster: 4 mean: 0 "  Actual: 0 1 1 1[1] "Cluster: 5 mean: 0.75 "  Actual: 0 1[1] "Cluster: 6 mean: 0.5 "  Actual: 0[1] "Cluster: 7 mean: 0 "  [1] "==========================="  [1] "Number of Cluster : 8"  Actual: 0 0 0 0 1 1[1] "Cluster: 1 mean: 0.333333333333333 "  Actual: 0 1[1] "Cluster: 2 mean: 0.5 "  Actual: 0 1 1 1[1] "Cluster: 3 mean: 0.75 "  Actual: 0 1[1] "Cluster: 4 mean: 0.5 "  Actual: 0[1] "Cluster: 5 mean: 0 "  Actual: 0[1] "Cluster: 6 mean: 0 "  Actual: 0 0 0 0 0 0[1] "Cluster: 7 mean: 0 "  Actual: 0 0[1] "Cluster: 8 mean: 0 "  [1] "===========================" |

# Conclusions

**Comparisons to original paper findings**

**Difficulty with large feature/small sample data set**

**Cross-Validation**

**Future: SVM?**

# References

Hastie, Tibshirani, Friedman. *The Elements of Statistical Learning*. New York, New York: Springer, 2009. Ebook.

James, Witten, Hastie, Tibshirani. *An Introduction to Statistical Learning with Applications in R*. New York, New York: Springer, 2013. Ebook.

Meltzer, Sean Davis and Paul. "GEOquery: a bridge between the Gene Expression Omnibus (GEO) and BioConductor." *Bioinformatics* (2007): 1846 -- 1847.

Nuria Palau, Antonio Julia, Carlos Ferrandiz, Lluis Puig, Eduardo Fonsesa, Emilia Fernandez, Maria Lopez-Lasanta, Raul Tortosa, and Sara Marsal. "Genome-wide transcriptional analysis of T cell activation reveals differential gene expression associated with psoriasis." *BMC Genomics* (2013).

# Source Code

## ExamineData.R

# Install GEOquery library:

# source("http://bioconductor.org/biocLite.R")

# biocLite("GEOquery")

library(GEOquery)

library("preprocessCore")

citation("GEOquery")

# GSE

gse <- getGEO("GSE47598", GSEMatrix = FALSE)

gsmplatforms <- lapply(GSMList(gse),function(x) {Meta(x)$platform\_id})

head(gsmplatforms)

gsmlist = Filter(function(gsm) {Meta(gsm)$platform\_id=='GPL10558'},GSMList(gse))

PvalTable <- Table(gsmlist[[1]])

Columns(gsmlist[[1]])[1:5,]

probesets <- Table(GPLList(gse)[[1]])$ID

data.matrix <- do.call('cbind',lapply(gsmlist,function(x)

{tab <- Table(x)

mymatch <- match(probesets,tab$ID\_REF)

return(tab$VALUE[mymatch])

}))

data.matrix <- apply(data.matrix,2,function(x) {as.numeric(as.character(x))})

data.matrix[is.na(data.matrix)] = 0

#Data Processing mentioned in the Paper

#data.matrix = data.matrix[rowSums(data.matrix == 0) <= 3, ]

data.matrix <- log2(data.matrix)

data.matrix[is.na(data.matrix)] = 0

data.matrix = normalize.quantiles(data.matrix,copy=TRUE)

data.labels <- lapply(GSMList(gse),function(x) {Meta(x)$characteristics\_ch1}) == "group: Psoriasis Patient"

data.rowNames <- probesets

data.Pval <- rep(1, times = nrow(data.matrix))

for (ii in range(0, nrow(data.matrix))) {

data.Pval[ii] = PvalTable[PvalTable[, 1]== data.rowNames[ii], 3]

}

# Recreating results

#data.Filmatrix = data.matrix[data.Pval <= 0.05, ]

#data.Filmatrix = data.Filmatrix[rowSums(data.Filmatrix == 0) <= 3, ]

infected <- data.matrix[, data.labels]

ninfected <- data.matrix[, !data.labels]

meanInfected <- rowMeans(infected)

meanNinfected <- rowMeans(ninfected)

expressChange <- (meanInfected )/meanNinfected

#List of Upregulated genes

upregSort <- order(expressChange, decreasing = TRUE)

data.rowNames[upregSort[0:13]] #upregSort maps indexes to sorted indexes

expressChange[upregSort[0:13]]

#testing

leader <- which(data.rowNames =="ILMN\_1683678")[[1]]

expressChange[upregSort[leader]]

## Lasso.r

library(GEOquery)

library(glmnet)

library(lasso2)

library(illuminaHumanv4.db)

library(matrixStats)

rm(list=ls())

cat("\014")

#following the R script generated by GEO2R on the GSE47598 page

#fetch the data matrix from GEO. This contains a bunch of stuff including the expression for each gene/individual.

gse <- getGEO("GSE47598", GSEMatrix = TRUE)

#I believe this is formatting to make the gse list convertible later with the exprs function

if (length(gse) > 1) idx <- grep("GPL10558", attr(gse, "names")) else idx <- 1

gse <- gse[[idx]]

#show(gse)

#This is the classification vector (0 means patient, 1 is control)

y = c(0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,1,1)

#Convert the GSE list into a matrix (and traspose so rows are individuals (sample) and columns are genes (features).

#eset, which is the matrix we are interested in in huge so viewing it all takes forever in R. This just

# let me verify that the data matches the GSM files

eset <- t(exprs(gse))

show(eset[,1:5])

rm(gse)

#now do lasso on data set

grid =10^seq(2,-8, length=100)

lasso.geo = glmnet(as.matrix(eset), y, alpha=1, family="binomial", lambda=grid, standardize = TRUE)

plot(lasso.geo, xvar = 'lambda')

show(summary(lasso.geo))

which(coef(lasso.geo)[,100] != 0)

ind100 = which(coef(lasso.geo)[,100] != 0)

coefs100.lasso = coef(lasso.geo)[ind100[-1],100]

plot(coefs100.lasso)

which(coef(lasso.geo)[,50] != 0)

ind50 = which(coef(lasso.geo)[,50] != 0)

coefs50.lasso = coef(lasso.geo)[ind50[-1],50]

plot(coefs50.lasso)

ridge.geo = glmnet(as.matrix(eset), y, alpha=0, family="binomial", lambda=grid, standardize = TRUE)

plot(ridge.geo, xvar = 'lambda')

## PCA.R

library(GEOquery)

library(glmnet)

library(illuminaHumanv4.db)

library(matrixStats)

library(graphics)

library(boot)

rm(list=ls())

cat("\014")

#following the R script generated by GEO2R on the GSE47598 page

#fetch the data matrix from GEO. This contains a bunch of stuff including the expression for each gene/individual.

gse <- getGEO("GSE47598", GSEMatrix = TRUE)

#I believe this is formatting to make the gse list convertible later with the exprs function

if (length(gse) > 1) idx <- grep("GPL10558", attr(gse, "names")) else idx <- 1

gse <- gse[[idx]]

#Convert the GSE list into a matrix (and traspose so rows are individuals (sample) and columns are genes (features).

eset <- t(exprs(gse))

rm(gse)

#spats2l is 6963

#This is the classification vector (0 means patient, 1 is control)

y = c(0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,1,1)

eset.log <- log(eset)

pr.out = prcomp(eset.log, scale. = TRUE)

summary(pr.out)

PC1.sorted = sort(abs(pr.out$rotation[,"PC1"]),decreasing = TRUE)

which(names(PC1.sorted) == "ILMN\_1683678")

which(names(sort(abs(pr.out$rotation[,"PC1"]),decreasing = TRUE)) == "ILMN\_1683678")

Cols=function(vec) {

cols=rainbow(length(unique(vec)))

return(cols[as.numeric(as.factor((vec)))])

}

#sort(abs(pr.out$rotation[,"PC2"]),decreasing = TRUE)[1:50]

#sort(abs(pr.out$rotation[,"PC3"]),decreasing = TRUE)[1:50]

sum.pr = summary(pr.out)

plot(sum.pr$importance[1,], type = 'o', pch = 1, xlab = 'PC', ylab = 'Stdev')

plot(sum.pr$importance[2,], type = 'o', pch = 1, xlab = 'PC', ylab = 'PVE')

plot(sum.pr$importance[3,], type = 'o', pch = 1, xlab = 'PC', ylab = 'Cummulative PVE')

plot(pr.out$x[,c(1,2)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(1,3)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(1,4)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(1,5)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(1,6)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(1,7)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(1,8)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(1,9)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(2,3)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(2,4)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(2,5)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(2,6)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(2,7)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(2,8)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(2,9)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(3,4)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(3,5)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(3,6)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(3,7)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(3,8)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(3,9)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(4,5)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(4,6)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(4,7)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(4,8)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(4,9)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(5,6)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(5,7)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(5,8)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(5,9)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(6,7)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(6,8)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(6,9)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(7,8)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(7,9)],pch = 19, col=Cols(y))

plot(pr.out$x[,c(8,9)],pch = 19, col=Cols(y))

#now try to use PCs to predict

data.pca <- data.frame(y=y, eset %\*% pr.out$rotation[,1:10])

fit.pca <- glm(y ~.,data = data.pca, family = "binomial")

pred.pca <- predict(fit.pca, newdata = data.pca)

ypred <- (pred.pca < 0)

#LOOCV

ypred5.loocv = ypred

data.pca <- data.frame(y=y, eset %\*% pr.out$rotation[,1:5])

for(i in 1:24) {

fit.pca <- glm(y ~.,data = data.pca[-i,], family = "binomial")

pred.pca <- predict(fit.pca, newdata = data.pca[i,])

ypred5.loocv[i] <- (pred.pca < 0)

}

ypred10.loocv = ypred

data.pca <- data.frame(y=y, eset %\*% pr.out$rotation[,1:10])

for(i in 1:24) {

fit.pca <- glm(y ~.,data = data.pca[-i,], family = "binomial")

pred.pca <- predict(fit.pca, newdata = data.pca[i,])

ypred10.loocv[i] <- (pred.pca < 0)

}

ypred15.loocv = ypred

data.pca <- data.frame(y=y, eset %\*% pr.out$rotation[,1:15])

for(i in 1:24) {

fit.pca <- glm(y ~.,data = data.pca[-i,], family = "binomial")

pred.pca <- predict(fit.pca, newdata = data.pca[i,])

ypred15.loocv[i] <- (pred.pca < 0)

}

ypred20.loocv = ypred

data.pca <- data.frame(y=y, eset %\*% pr.out$rotation[,1:20])

for(i in 1:24) {

fit.pca <- glm(y ~.,data = data.pca[-i,], family = "binomial")

pred.pca <- predict(fit.pca, newdata = data.pca[i,])

ypred20.loocv[i] <- (pred.pca < 0)

}

show(ypred)

show(ypred5.loocv)

show(ypred10.loocv)

show(ypred15.loocv)

## show(ypred20.loocv)

## Step.r

library(GEOquery)

library(glmnet)

library(illuminaHumanv4.db)

library(matrixStats)

library(MASS)

rm(list=ls())

cat("\014")

#following the R script generated by GEO2R on the GSE47598 page

#fetch the data matrix from GEO. This contains a bunch of stuff including the expression for each gene/individual.

gse <- getGEO("GSE47598", GSEMatrix = TRUE)

#I believe this is formatting to make the gse list convertible later with the exprs function

if (length(gse) > 1) idx <- grep("GPL10558", attr(gse, "names")) else idx <- 1

gse <- gse[[idx]]

#show(gse)

#This is the classification vector (0 means patient, 1 is control)

y = c(0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,1,1)

#Convert the GSE list into a matrix (and traspose so rows are individuals (sample) and columns are genes (features).

#eset, which is the matrix we are interested in in huge so viewing it all takes forever in R. This just

# let me verify that the data matches the GSM files

eset <- t(exprs(gse))

show(eset[,1:5])

CoVar.geo = colSds(eset,cols = 1:ncol(eset))/colMeans(eset)

#take top 25% of genes with the most variation across samples as a first filter

filter.geo = which(CoVar.geo>quantile(CoVar.geo)[4])

eset.df <- as.data.frame(eset)

eset.filtered.df <- eset.df[,filter.geo]

yeset.filtered.df = (cbind(y,eset.filtered.df))

rm(gse)

rm(eset)

rm(eset.df)

rm(eset.filtered.df)

min.model <- glm(y ~ 1, family=binomial(link = 'logit'), data=yeset.filtered.df)

upper.model <- formula(glm(y~.,family = binomial(link = 'logit'), data = yeset.filtered.df))

fwd.geo <- step(min.model,scope = list(lower=min.model,upper=upper.model),direction = 'forward',steps = 10)

## SVM.R

library(GEOquery)

library(glmnet)

library(illuminaHumanv4.db)

library(matrixStats)

library(e1071)

#following the R script generated by GEO2R on the GSE47598 page

#fetch the data matrix from GEO. This contains a bunch of stuff including the expression for each gene/individual.

gse <- getGEO("GSE47598", GSEMatrix = TRUE)

#I believe this is formatting to make the gse list convertible later with the exprs function

if (length(gse) > 1) idx <- grep("GPL10558", attr(gse, "names")) else idx <- 1

gse <- gse[[idx]]

show(gse)

#This is the classification vector (0 means patient, 1 is control)

y = c(0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,1,1)

#Convert the GSE list into a matrix (and traspose so rows are individuals (sample) and columns are genes (features).

eset <- t(exprs(gse))

rm(gse)

#coefficient of varaition

CoVar.geo = colSds(eset,cols = 1:ncol(eset))/colMeans(eset)

spats2l <- which(colnames(eset)=="ILMN\_1683678")

klf6 <- which(colnames(eset)=="ILMN\_1735014")

sp140 <- which(colnames(eset)=="ILMN\_1703263")

rora <- which(colnames(eset)=="ILMN\_2322498")

#spats2l is 6963

#eset, which is the matrix we are interested in in huge so viewing it all takes forever in R. This just

# let me verify that the data matches the GSM files

show(eset[,c(spats2l,klf6,sp140,rora)])

show(CoVar.geo[,c(spats2l,klf6,sp140,rora)])

#train an SVM

x <- eset

x[is.na(x)]=0

train=data.frame(x,y)

my\_svm <- svm(y ~ x , train)

pred <- predict(my\_svm, train)

summary(my\_svm)

svm\_error <- sqrt(mean((y - pred)^2))

svm\_error

#Tuning the SVM (This takes a very long time)

yy <- array(data = y)

svm\_tune <- tune(svm, yy ~ x, data = train, ranges = list(epsilon = seq(0,1,0.01), cost = 2^(2:9)))

print(svm\_tune)

#best svm from the Tune

best\_mod <- svm\_tune$best.model

best\_mod\_pred <- predict(best\_mod, train)

error\_best\_mod <- train$y - best\_mod\_pred

best\_mod\_RMSE <- sqrt(mean(error\_best\_mod^2))

save(svm\_tune, file = "SVM\_data.RData")

## Kmeans.R

library(GEOquery)

library(glmnet)

library(illuminaHumanv4.db)

library(matrixStats)

library(e1071)

library(tidyverse) # data manipulation

library(cluster) # clustering algorithms

library(factoextra) # clustering algorithms & visualization

#following the R script generated by GEO2R on the GSE47598 page

#fetch the data matrix from GEO. This contains a bunch of stuff including the expression for each gene/individual.

gse <- getGEO("GSE47598", GSEMatrix = TRUE)

#I believe this is formatting to make the gse list convertible later with the exprs function

if (length(gse) > 1) idx <- grep("GPL10558", attr(gse, "names")) else idx <- 1

gse <- gse[[idx]]

show(gse)

#This is the classification vector (0 means patient, 1 is control)

y = c(0,0,0,0,0,0,0,0,0,0,0,0,0,1,0,1,0,1,0,1,0,1,1,1)

#Convert the GSE list into a matrix (and traspose so rows are individuals (sample) and columns are genes (features).

eset <- t(exprs(gse))

rm(gse)

#coefficient of varaition

CoVar.geo = colSds(eset,cols = 1:ncol(eset))/colMeans(eset)

spats2l <- which(colnames(eset)=="ILMN\_1683678")

klf6 <- which(colnames(eset)=="ILMN\_1735014")

sp140 <- which(colnames(eset)=="ILMN\_1703263")

rora <- which(colnames(eset)=="ILMN\_2322498")

#spats2l is 6963

#eset, which is the matrix we are interested in in huge so viewing it all takes forever in R. This just

# let me verify that the data matches the GSM files

show(eset[,c(spats2l,klf6,sp140,rora)])

show(CoVar.geo[,c(spats2l,klf6,sp140,rora)])

#Initialize

x <- eset

x[is.na(x)] = 0

#distance

distance\_Euclid <- get\_dist(x, method = "euclidean")

distance\_Manhattan <- get\_dist(x, method = "manhattan")

fviz\_dist(distance\_Euclid, gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07"))

fviz\_dist(distance\_Manhattan, gradient = list(low = "#00AFBB", mid = "white", high = "#FC4E07"))

#Kmeans

K2 <- kmeans(x, centers = 2, nstart = 100)

#notice that clusters were found of size 9 and 15

str(K2)

fviz\_cluster(K2, data = x)

#originally 29.17 percent of patients didn't have psoriasis

#in cluster 1, that is 33.33

mean(y[K2$cluster == 1])

#in cluster 2, that is 26.67

mean(y[K2$cluster == 2])

#Therefore this doesn't seem to be clustering on the disease

#supposing we looked at more than two clusters. Will any of them do better?

for (NumClusters in c(2, 3,4,5,6,7,8)){

Ktest <- kmeans(x, centers = NumClusters, nstart = 50)

print(paste("Number of Cluster : ", NumClusters))

for (ii in 1:NumClusters){

print(paste("Cluster: ",ii,

" mean: ", mean(y[Ktest$cluster == ii]),

cat(" Actual: ", y[Ktest$cluster == ii])))

}

print('===========================')

}

## Trees.R

library(ISLR)

library(tree)

#install.packages('randomForest')

library(randomForest)

#source("http://bioconductor.org/biocLite.R")

#biocLite("illuminaHumanv4.db")

library(illuminaHumanv4.db)

library(GEOquery)

citation("GEOquery")

gse <- getGEO("GSE47598", GSEMatrix = FALSE)

gsmplatforms <- lapply(GSMList(gse),function(x) {Meta(x)$platform\_id})

head(gsmplatforms)

gsmlist = Filter(function(gsm) {Meta(gsm)$platform\_id=='GPL10558'},GSMList(gse))

PvalTable <- Table(gsmlist[[1]])

Columns(gsmlist[[1]])[1:5,]

probesets <- Table(GPLList(gse)[[1]])$ID

data.matrix <- do.call('cbind',lapply(gsmlist,function(x)

{tab <- Table(x)

mymatch <- match(probesets,tab$ID\_REF)

return(tab$VALUE[mymatch])

}))

data.labels <- lapply(GSMList(gse),function(x) {Meta(x)$characteristics\_ch1}) == "group: Psoriasis Patient"

gse <- getGEO("GSE47598", GSEMatrix = TRUE)

if (length(gse) > 1) idx <- grep("GPL10558", attr(gse, "names")) else idx <- 1

gse <- gse[[idx]]

show(gse)

#Convert the GSE list into a matrix (and traspose so rows are individuals (sample) and columns are genes (features).

eset <- t(exprs(gse))

dataset3 <- data.frame(eset)

y <- data.labels

# Runs over subsets of data to find the most important factors

set.seed(1000)

importFactors.append = FALSE

fac.append = 0

for (i in 0:46){

start = 1+i\*1000

end = 1000+i\*1000

dataset7 <- dataset3[,start:end]

forestfit <- randomForest(y~., data=dataset7, importance=TRUE, ntree=10000)

varImpPlot(forestfit)

importTree <- importance(forestfit, type=1)

importTree2 <- importTree\*importTree

importFactors = apply(importTree2, 1, function(row) all(row > 10))

fac <- importTree[importFactors,]

fac <- fac[order(fac)]

symbols <- mapIds(illuminaHumanv4.db, names(fac), "SYMBOL","PROBEID")

symbols

fac.append = append(fac.append,fac)

importFactors.append = append(importFactors.append,importFactors)

}

dataset7 <- dataset3[,47001:47222]

forestfit <- randomForest(y~., data=dataset7, importance=TRUE, ntree=10000)

varImpPlot(forestfit)

importTree <- importance(forestfit, type=1)

importTree2 <- importTree\*importTree

importFactors = apply(importTree2, 1, function(row) all(row > 10))

fac <- importTree[importFactors,]

fac <- fac[order(fac)]

symbols <- mapIds(illuminaHumanv4.db, names(fac), "SYMBOL","PROBEID")

symbols

fac.append = append(fac.append,fac)

importFactors.append = append(importFactors.append,importFactors)

# cut off the extra factor added to initialize

importFactors.append = importFactors.append[2:length(importFactors.append)]

# find the best of the best

dataset8 <- dataset3[,importFactors.append]

forestfit <- randomForest(y~., data=dataset8, importance=TRUE, ntree=10000)

varImpPlot(forestfit)

importTree <- importance(forestfit, type=1)

importTree2 <- importTree\*importTree

importFactors = apply(importTree2, 1, function(row) all(row > 6))

fac <- importTree[importFactors,]

fac <- fac[order(fac)]

symbols <- mapIds(illuminaHumanv4.db, names(fac), "SYMBOL","PROBEID")

symbols

fac

# Referenced Tables from paper

Table : Significantly Upregulated genes in Psoriasis patients compared to healthy controls

|  |  |  |  |
| --- | --- | --- | --- |
| Probe | Gene | Fold Change | P value |
| |  | | --- | | ILMN\_1683678 | | ILMN\_1735014 | | ILMN\_1703263 | | ILMN\_2322498 | | ILMN\_2246956 | | ILMN\_2397721 | | ILMN\_1781285 | | ILMN\_2321064 | | ILMN\_1731107 | | ILMN\_1729749 | | ILMN\_2095660 | | ILMN\_1681301 | | ILMN\_1767470 | | ILMN\_1735979 | | ILMN\_1719543 | | ILMN\_2188333 | | ILMN\_1776723 | | ILMN\_1721626 | | ILMN\_1773742 | | ILMN\_1741003 | | ILMN\_1729374 | | ILMN\_1660368 | | ILMN\_2406410 | | ILMN\_2305112 | | ILMN\_2284998 | | |  | | --- | | *SPATS2L* | | *KLF6* | | *SP140* | | *RORA* | | *BCL2* | | *GLB1* | | *DUSP1* | | *BAX* | | *CCDC92* | | *HERC5* | | *TMEM156* | | *AIM2* | | *SCPEP1* | | *BCKDHA* | | *MAF* | | *CD69* | | *PHF11* | | *ARID5B* | | *DNAJB9* | | *ANXA5* | | *ETFB* | | *TRRAP* | | *RHBDD2* | | *CTH* | | *SP100* | | |  | | --- | | 1.37 | | 1.32 | | 1.38 | | 1.31 | | 1.23 | | 1.23 | | 1.21 | | 1.24 | | 1.33 | | 1.72 | | 1.29 | | 1.42 | | 1.26 | | 1.21 | | 1.36 | | 1.33 | | 1.24 | | 1.27 | | 1.23 | | 1.27 | | 1.23 | | 1.20 | | 1.39 | | 1.57 | | 1.27 | | |  | | --- | | 0.0009 | | 0.0012 | | 0.0025 | | 0.0041 | | 0.0062 | | 0.0062 | | 0.0071 | | 0.0096 | | 0.0136 | | 0.0267 | | 0.0453 | | 0.0464 | | 0.0469 | | 0.0639 | | 0.0658 | | 0.0677 | | 0.0683 | | 0.0788 | | 0.0805 | | 0.0896 | | 0.0921 | | 0.0922 | | 0.0925 | | 0.0944 | | 0.0944 | |

Table : significantly Downregulated genes in Psoriasis patients compared to healthy controls

|  |  |  |  |
| --- | --- | --- | --- |
| Probe | Gene | Fold Change | P value |
| |  | | --- | | ILMN\_3286813 | | ILMN\_3281502 | | ILMN\_1778617 | | ILMN\_1689294 | | ILMN\_2143250 | | ILMN\_1720114 | | ILMN\_2135175 | | ILMN\_1764163 | | ILMN\_1746148 | | ILMN\_1715401 | | ILMN\_2299072 | | ILMN\_2124802 | | ILMN\_1655827 | | ILMN\_2402936 | | ILMN\_3230435 | | ILMN\_1803799 | | ILMN\_1704873 | | |  | | --- | | *LOC391019* | | *LOC653375* | | *TAF9* | | *LOC85390* | | *FAR1* | | *GMNN* | | *SNORD36A* | | *LOC644330* | | *LRRC33* | | *MT1G* | | *CROP* | | *MT1H* | | *COPS2* | | *LOC440926* | | *LOC729086* | | *LOC649555* | | *TCEB1* | | |  | | --- | | -1.38 | | -1.31 | | -1.25 | | -1.20 | | -1.20 | | -1.20 | | -1.27 | | -1.20 | | -1.22 | | -1.82 | | -1.20 | | -1.64 | | -1.20 | | -1.22 | | -1.24 | | -1.21 | | -1.21 | | |  | | --- | | 0.0001 | | 0.0009 | | 0.0009 | | 0.0120 | | 0.0207 | | 0.0269 | | 0.0303 | | 0.0577 | | 0.0656 | | 0.0677 | | 0.0683 | | 0.0778 | | 0.0800 | | 0.0827 | | 0.0886 | | 0.0921 | | 0.0970 | |