Assignment

2226768Y

19 March 2016

Introduction

Using microarray data to find link between gene exression and phenotype is . Finding a key set of genes where changes in expression indicate a the presance or absense of a condition. such as disease state or drug response.
Chronic Myloid Leukemia is a disease that is sucsessfully treated by the drug . Unfortunately some patients do not respond and others become resitant to the dtug after a long period of sucessful treatment. Being able to predict the non-responders early gives the clinicianc the oppotunity to try differant options early .
In this assignment I attempt to use two supervised learning classifiers, logistic regression and random forest, to produce models that can predict drug response from micro array data.

The Data.

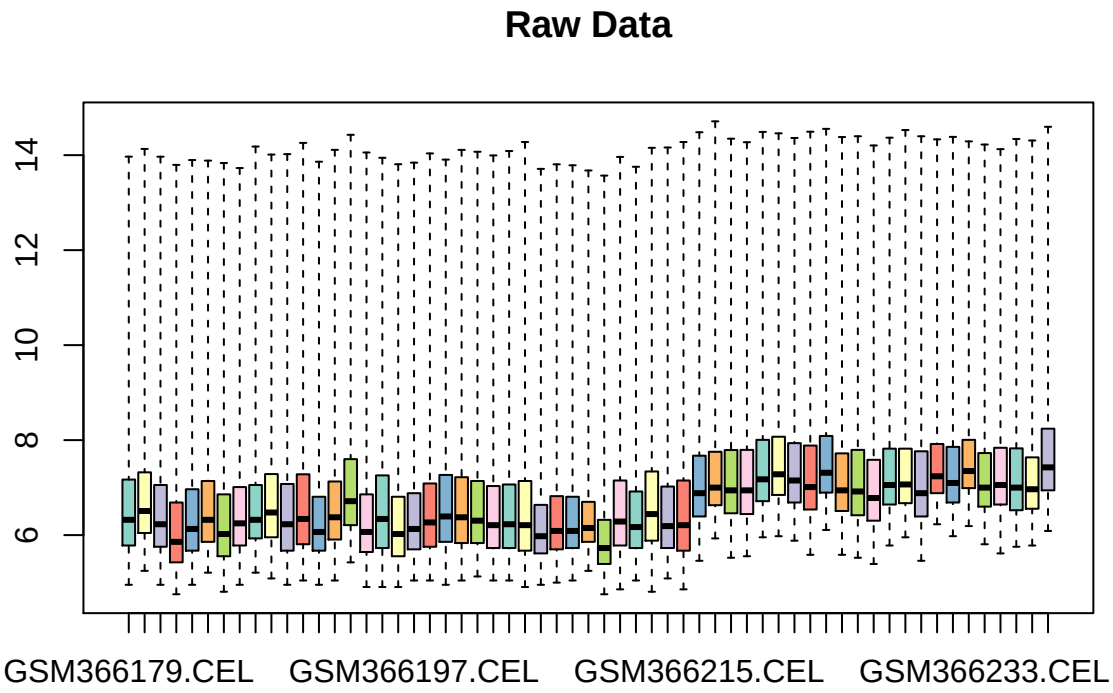
The dataset GSE14671 was downloaded from GEO at the NCBI websiteis. It consists 59 samples of microarray data each of 54675 probes, the metadata includes information on drug response. The sets were collected and processed at differant times and the differance can be seen in the raw data.

```
boxplot(data, main="Raw Data", col=brewer.pal(8,"Set3"))
```

```
## Warning: replacing previous import by 'utils::tail' when loading
## 'hgu133plus2cdf'
```

```
## Warning: replacing previous import by 'utils::head' when loading
## 'hgu133plus2cdf'
```

```
##
```




```
#dat is an eset
ggboxplot <-function ( dat, title){
  library(reshape2)
  plotData <- as.data.frame(t(exprs(dat)))
  plotData$sample <- row.names(plotData)
  plotData.m <- melt(plotData, id.vars = "sample")


  g<-  ggplot(plotData.m, aes(x=sample,y=value, fill= sample )) +
  geom_boxplot() +guides(fill=FALSE)+ ggtitle(title)

return(g)
}
```

Function to plot boxplots of expression sets using ggplot

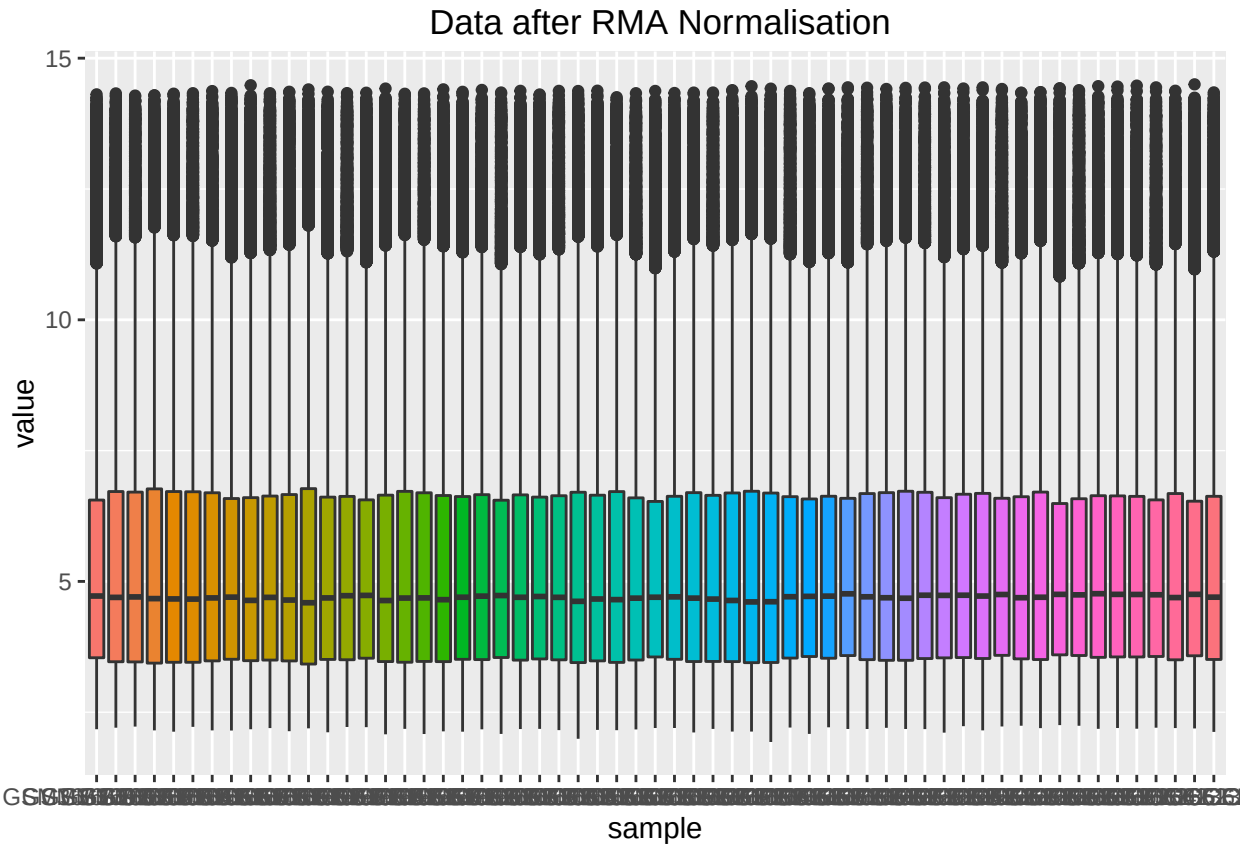
Normalisation of the data

The data was normalised using the RMA routine in Affy. Robust Multi-Array Average, is a 3 pre-processing method. Background correction is applied to remove local artefacts and "noise" . Then the data is normalised across the samples to remove array effects, this means measurements from different arrays are comparable . Finally the data is summarised to combine probe intensities across arrays to give final expression values . (log2 values). The results can be seen in the boxplot below .

```
eset <- rma(data)
```

```
## Background correcting
## Normalizing
## Calculating Expression
```

```
ggboxplot (eset, " Data after RMA Normalisation")
```



Read in the phenotype data and extract columns needed and merge eith the expression set.

```
phenoTypes <- read.table("E-GEOD-14671.sdrf.txt",header = TRUE, sep= "\t",stringsAsFactors = F,row.names

pheno <- phenoTypes
# change sample names to same as data
sampleN <- paste0(t(as.data.frame(strsplit(rownames(pheno)," ")))[,1],".CEL")
```

Extract columns needed response and training set

```
response <- factor(pheno$Comment.Sample_source_name., labels= c("NR","R"))
responseAll <- response# to try to get aroud knitting problem later on
tset<- data.frame(strsplit(pheno$Comment.Sample_description., " "),stringsAsFactors = FALSE)
tset <- paste(tset[1,],tset[2,],tset[3,] ,sep ="_")
trainingSet <- factor(tset, labels = c( "TS_R","TS_NR" ,"VS_NR","VS_R"))
pheno<- data.frame(response,trainingSet,row.names = sampleN)
```

merge with pheno data fom eset so in correct order

```
pd<- pData(eset)
pheno <- merge(pd, pheno, by=0)
rownames(pheno) <- pheno[,'Row.names']

pheno.metadata <- data.frame(labelDescription=colnames(pheno) )
pheno.annotatedDF <- new("AnnotatedDataFrame",data=pheno, varMetadata=pheno.metadata)
phenoData(eset) <- pheno.annotatedDF
```

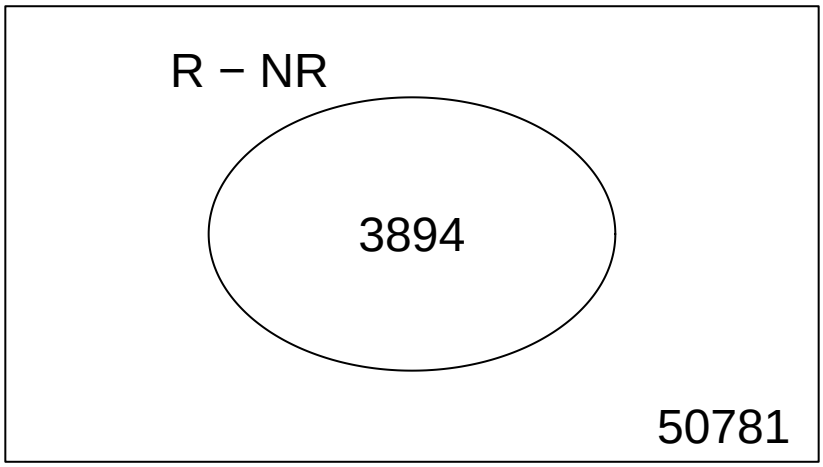
Re-order the eset so samples grouped by response.

```
cindex <-rownames( pData(eset)[order(pData(eset)$response),,drop =FALSE])
eset <- eset[,cindex]
```

Data Exploration

Limma was used to calculate the differential expression.
create design array for all data where CML subset ==CP.

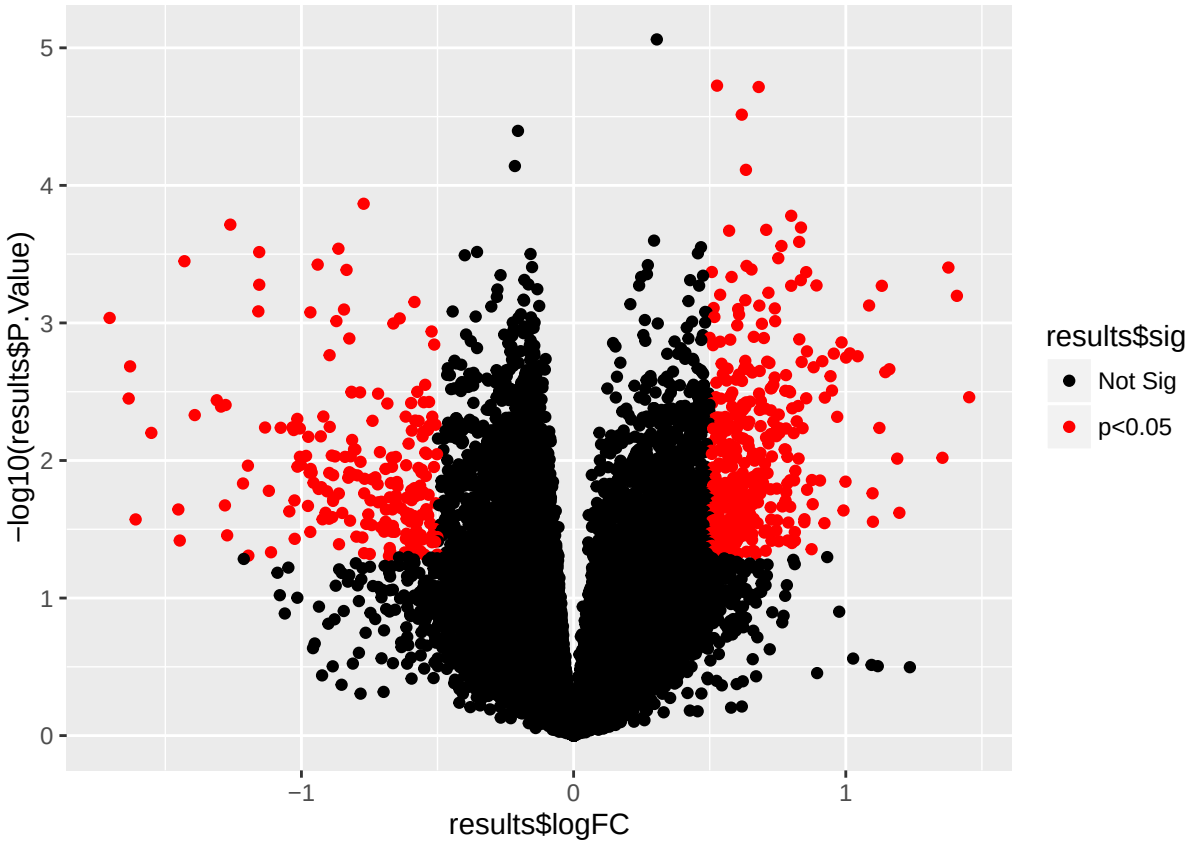
```
design <- model.matrix(~0 + eset@phenoData$response )
colnames(design) <- c("R","NR")
contrast.matrix <- makeContrasts(R-NR,levels=design)

fit <- lmFit(eset,design)
fit2 <- contrasts.fit(fit, contrast.matrix)
fit2 <- eBayes(fit2)

res = decideTests(fit2,adjust.method = "none",p.value= .05 )
vennDiagram(res, sub= "DE genes, with no multi-test correction")
```



Volcano plot of results

The red genes have a p-value of less than 0.05 and a log2(fold-change) of greater than 0.5

```
results = topTable(fit2,number = Inf)
results = mutate(results, sig=ifelse((results$P.Value<0.05 & abs(results$logFC) > 0.5), "p<0.05", "Not S
p = ggplot(results, aes(results$logFC, -log10(results$P.Value))) +
  geom_point(aes(col=results$sig)) +
  scale_color_manual(values=c( "black","red"))
p
```



The p-values are high so when a multitest correction was applied no genes pass the test. Only 51 genes have a fold change of greater than 2 and p value of less than 0.05. 'sum(results$sig == "p<0.05")' genes have a log2fold change = 0.5, (actual fold change of 1.4 and p<0.05).

Predictive Modeling

Aims

I aimed to see how much accuracy of prediction can be gained by using a more complex algorithm.
I took the following steps.

1. Geneset Selection
   - By statistical significance.
   - By difference in expression.

- By correlation.

2. Splitting data into training and validation set.
   By simple sampling, and with holding this data from training the model.
3. Modeling
   Logistic regression is used to determine how much variance in a binary dependent variable is explained by a set of independent variables. Logistic Regression does not make the same assumptions as linear regression such as linearity and homoscedasticity bet does have a few key assumption that may not hold

for gene expression data is that the data to be independent. The logit model was used that assumes that there is a log distribution of the probability of the event , “non responder”.

The Random Forrest Classifier is an ensemble method that generate a multitude of decision trees in ored to predict classes It has very few assumptions but is hard to interpret and therefor refine.

4. Testing
   Run the model on the withheld test data to estimate its accuracy.

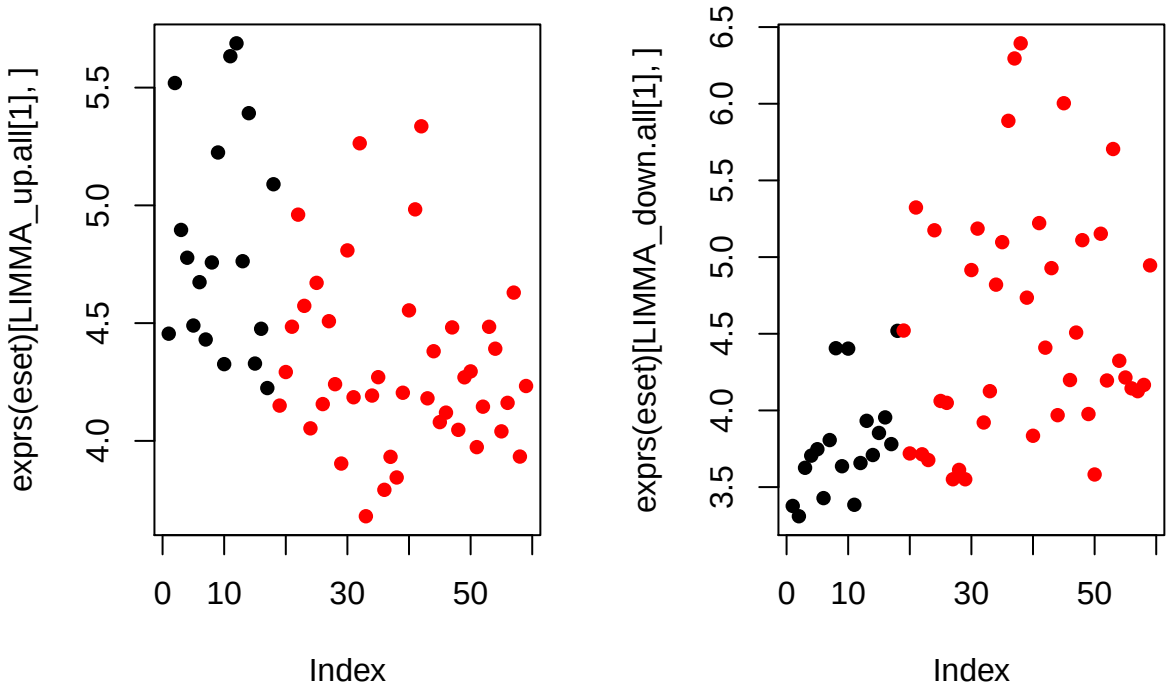
5. Evaluation.
   Calculate the accuracy, False discovery rate , pprecission and recall from the prediction resuts.

Feature Selection

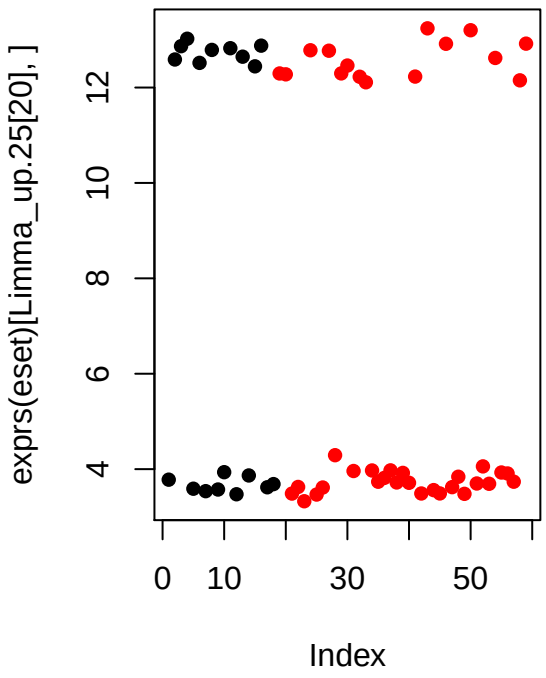
To reduce the dimesionality curse of high feature number to low sample number differant ways were used in attempt to pick genes that might be relavent to the response. This also reduces the computational complexity of the problem for use on a PC.
1.Limma was used to produce a list of genes that showed a statistaclly significant fold change .
2.Average Differance between responers and non- responders were calculated and then ranked on absolute value. It is assumed that DE is an affect of drug treatment.
3.Correlation between the change in gene expression and respone was calcvulated and ranked on absolute value. It is assumed that the expresion of mportant genes will correlate with response.

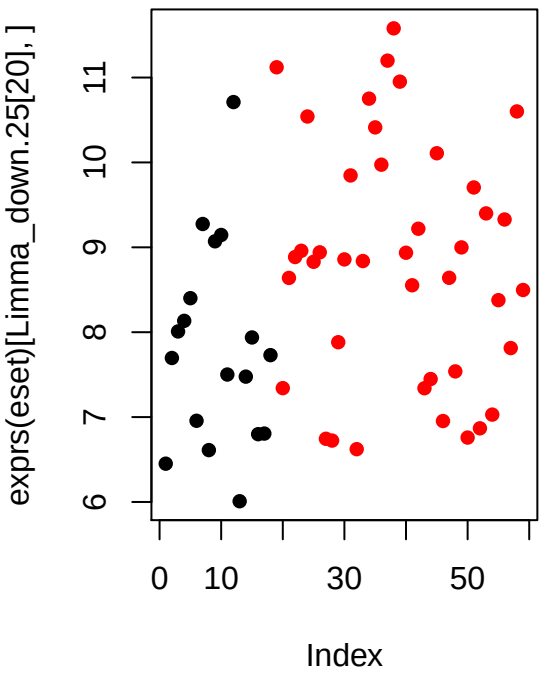
Differentially expressed genes from Limma

```
tt <- topTable( fit2, adjust="none",n=Inf, sort.by="none" )
all.results.sorted <- tt[ order(abs(tt$P.Value)), ]

q.threshold <- 0.05
fc.threshold <- 0.5
LIMMA_up.all <- rownames( subset( all.results.sorted,
adj.P.Val<q.threshold & logFC>fc.threshold ) )
LIMMA_down.all <- rownames( subset( all.results.sorted,
adj.P.Val<q.threshold & logFC< (0-fc.threshold ) ))
Limma.all <- c(LIMMA_down.all,LIMMA_up.all)


par( mfrow=c(1,2) )
plot( exprs(eset)[LIMMA_up.all[1],],pch=16, col=as.factor(pData(eset)$response))
plot( exprs(eset)[LIMMA_down.all[1],],pch=16, col=as.factor(pData(eset)$response))
```

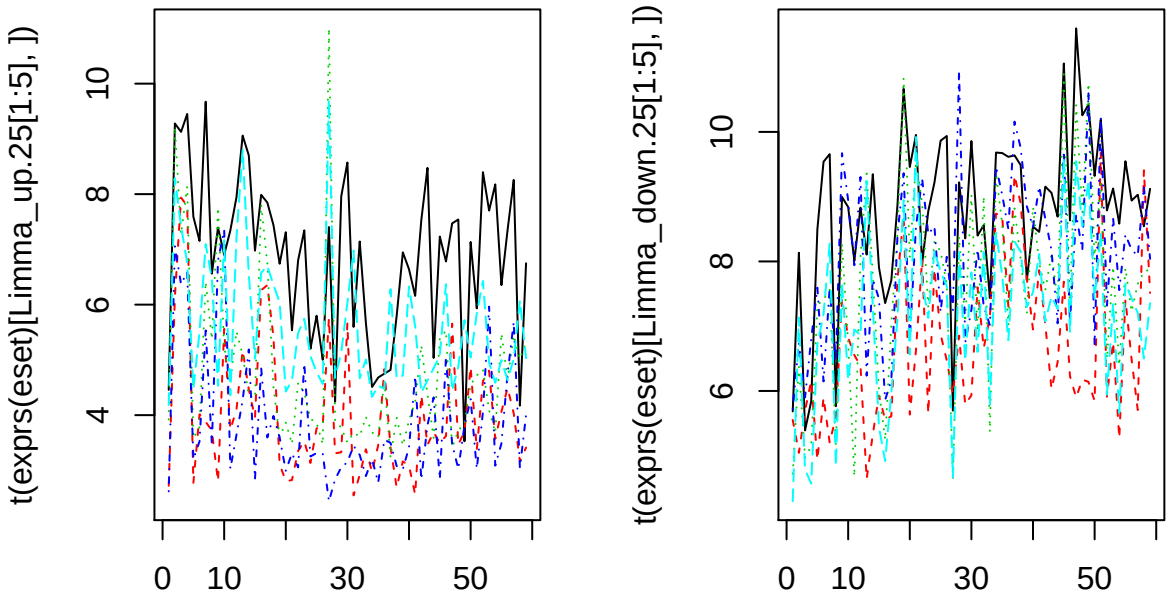



```
n.genes <- 25
Limma_up.25 <-rownames (subset(all.results.sorted, logFC>1 ))
Limma_down.25 <- rownames(subset(all.results.sorted, logFC< -1 )[1:n.genes,])
Limma50<- c(Limma_up.25,Limma_down.25)

plot( exprs(eset)[Limma_up.25[20],],pch=16, col=as.factor(pData(eset)$response))
plot( exprs(eset)[Limma_down.25[20],],pch=16, col=as.factor(pData(eset)$response))
```





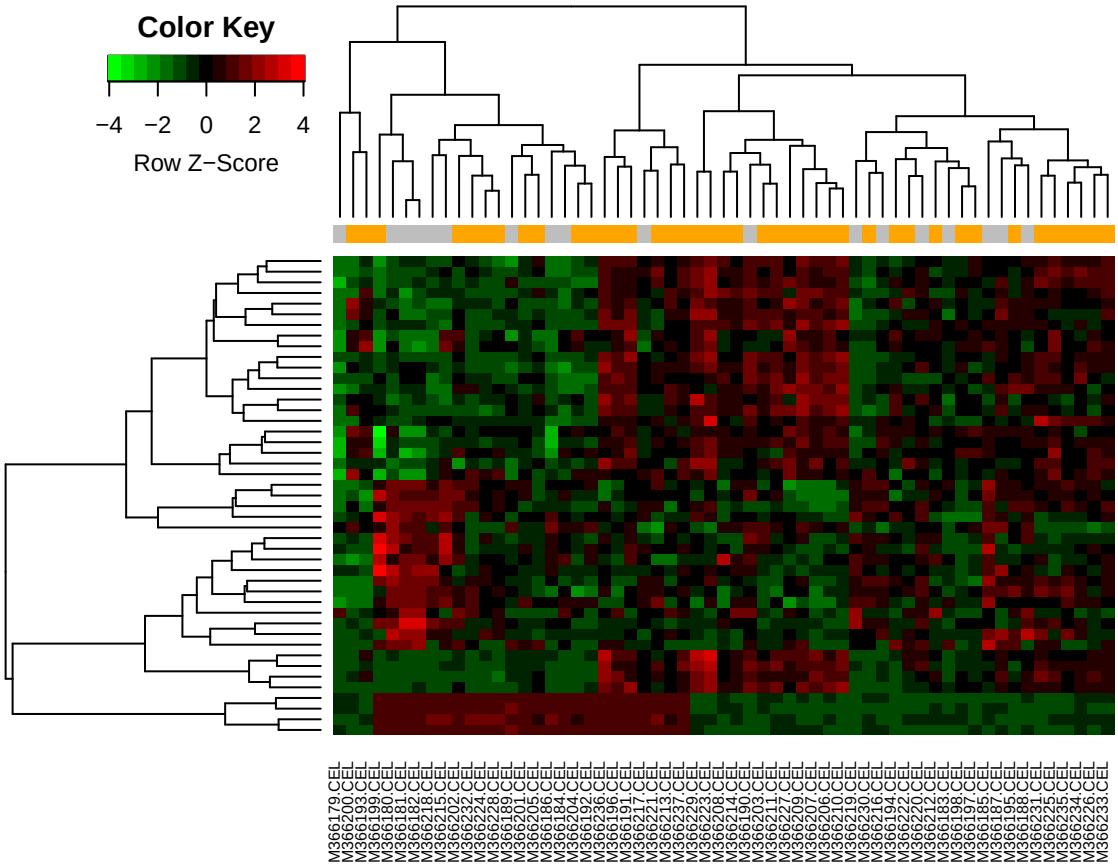

```
matplot (t(exprs(eset)[Limma_up.25[1:5],]),type = "l")
matplot (t(exprs(eset)[Limma_down.25[1:5],]),type = "l")
```



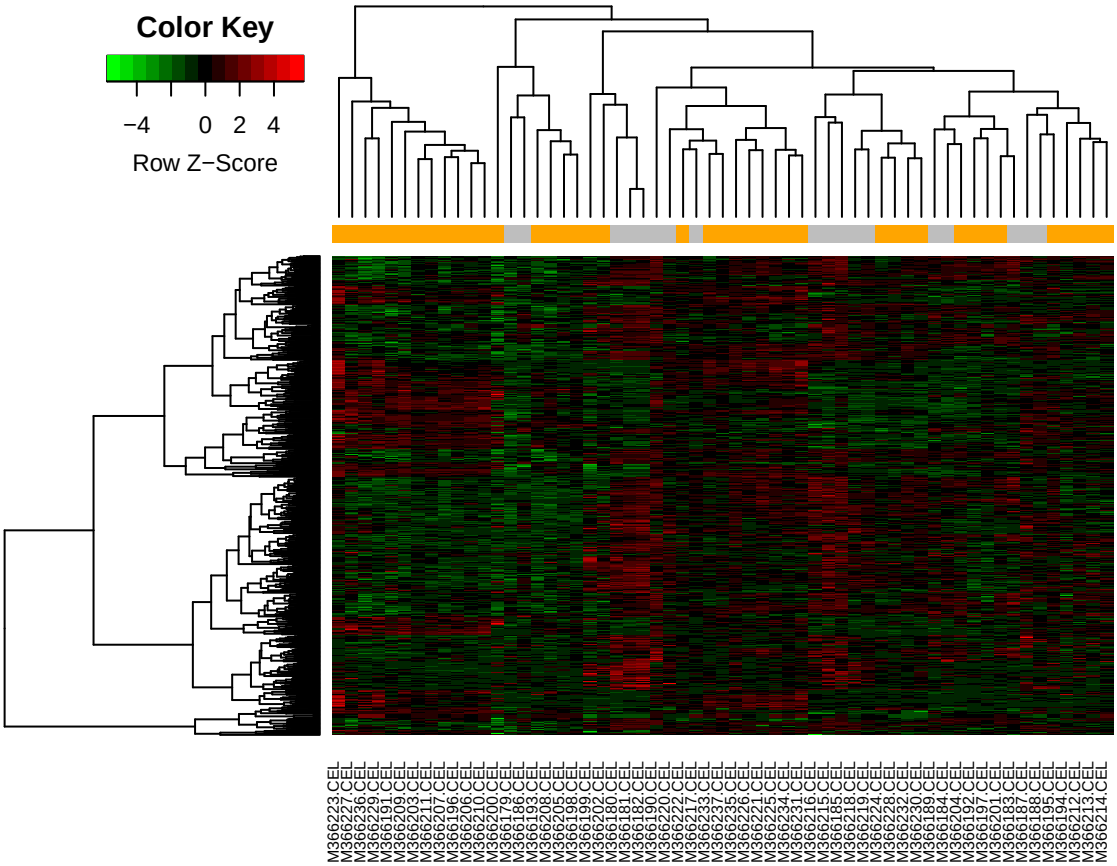
Summary of the mean fold change. All up regulated genes is: 0.6620217
Most significant up reg genes: 1.1679126 All down regulated genes: -0.7836567
Most significant down-reg genes:-1.2319727

Heat map of most significant genes.

```
labels = paste0(pData(eset)$sample,"_",pData(eset)$response)
column.classification <- rep.int( "grey", ncol(eset))
column.classification[ pData(eset)$response=="R" ]<- "orange"


heatmap.2(exprs(eset)[Limma50,],scale="row",trace="none",density.info="none",col=greenred,labRow=NA,Col
```




```
heatmap.2(exprs(eset)[Limma.all,],scale="row",trace="none",density.info="none",
col=greenred,labRow=NA,ColSideColors=column.classification,key=T)
```



Mean Difference in Expression

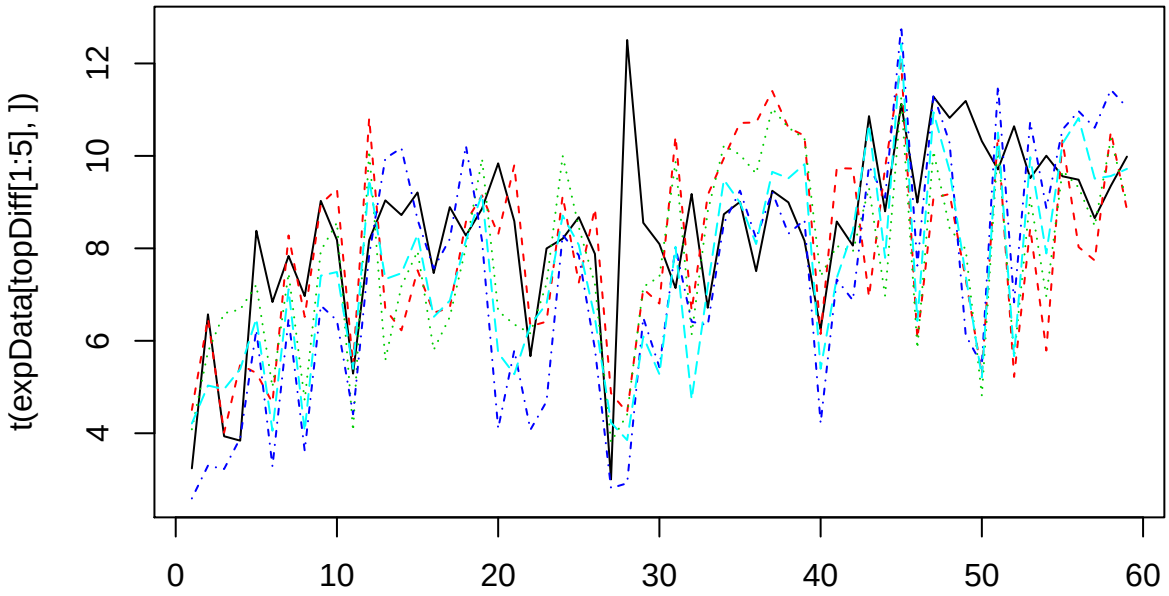
Using the difference between mean expression values for the responders and non-responders. This takes no account in the variance so that the DE of many genes may not be statistically significant.

```
expData <- exprs(eset)
responder <-  (eset@phenoData$response == "R")
nonResponder <- (eset@phenoData$response != "R")
 # avN <-  mean(exData[1,normal])
  #diff <- abs (avD-avN)
  diff  <- vector(, length = length(expData[,1]))
  varG <- vector(, nrow(expData))
  for (i in 1: length(expData[,1])){
    avNR<- mean(expData[i,nonResponder])
    avR <-  mean(expData[i,responder])
    varG[i] <- var(expData[i,])
    diff[i] <- abs (avNR-avR)
  }

rnames <- rownames(expData)
diffData <- data.frame (diff,row.names = rnames)

topDiff <-head (diffData[order(-diffData$diff), , drop = FALSE],n=50)
topDiff1000 <-head (diffData[order(-diffData$diff), , drop = FALSE],n=1000)
topDiff <- rownames (topDiff)
```

```
topDiff1000 <- rownames (topDiff1000)
```

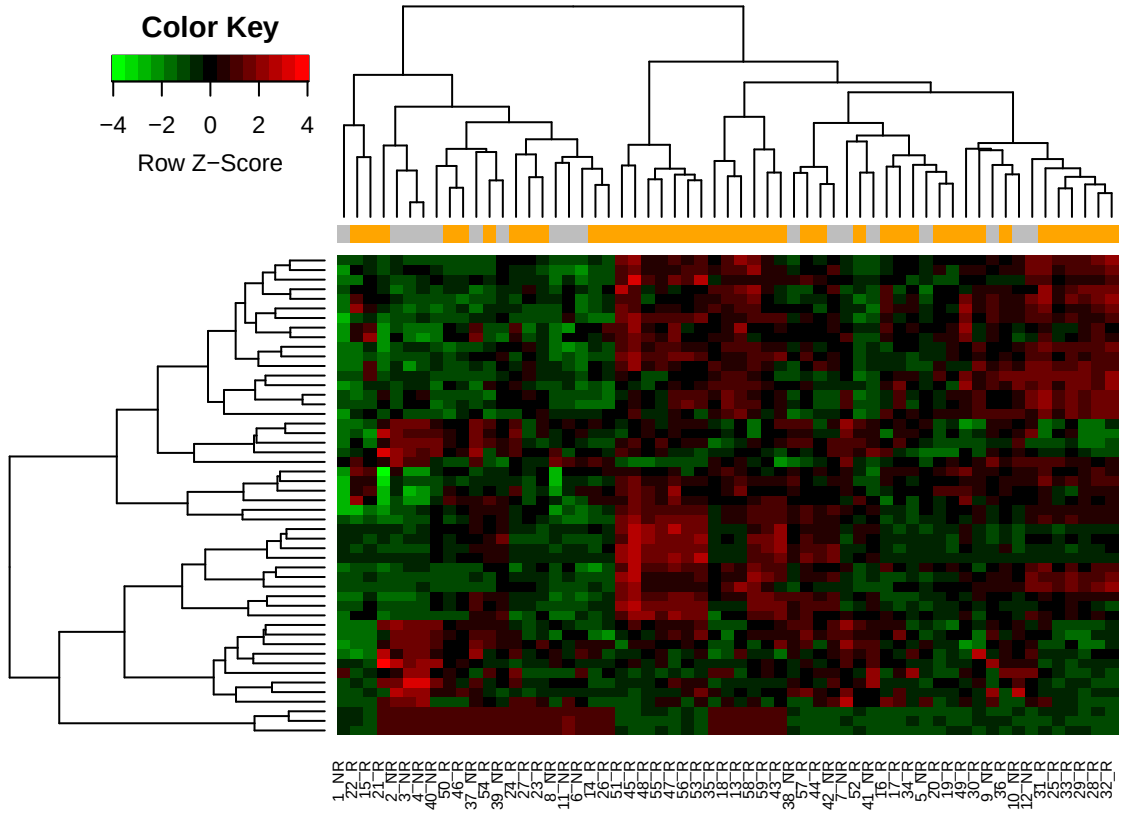
Plot of expression againt sample showing how even for the 5 genes with highest differance in mean the change is very small compared to the variance.

```
matplot(t(expData [topDiff[1:5],]), type = "l")
```



Heatmap of the 1000 genes showing the greatest differance in mean expression.

```
heatmap.2(expData[topDiff,],scale="row",trace="none",density.info="none",col=greenred,
 labCol = labels,labRow=NA,ColSideColors=column.classification,key=T)
```

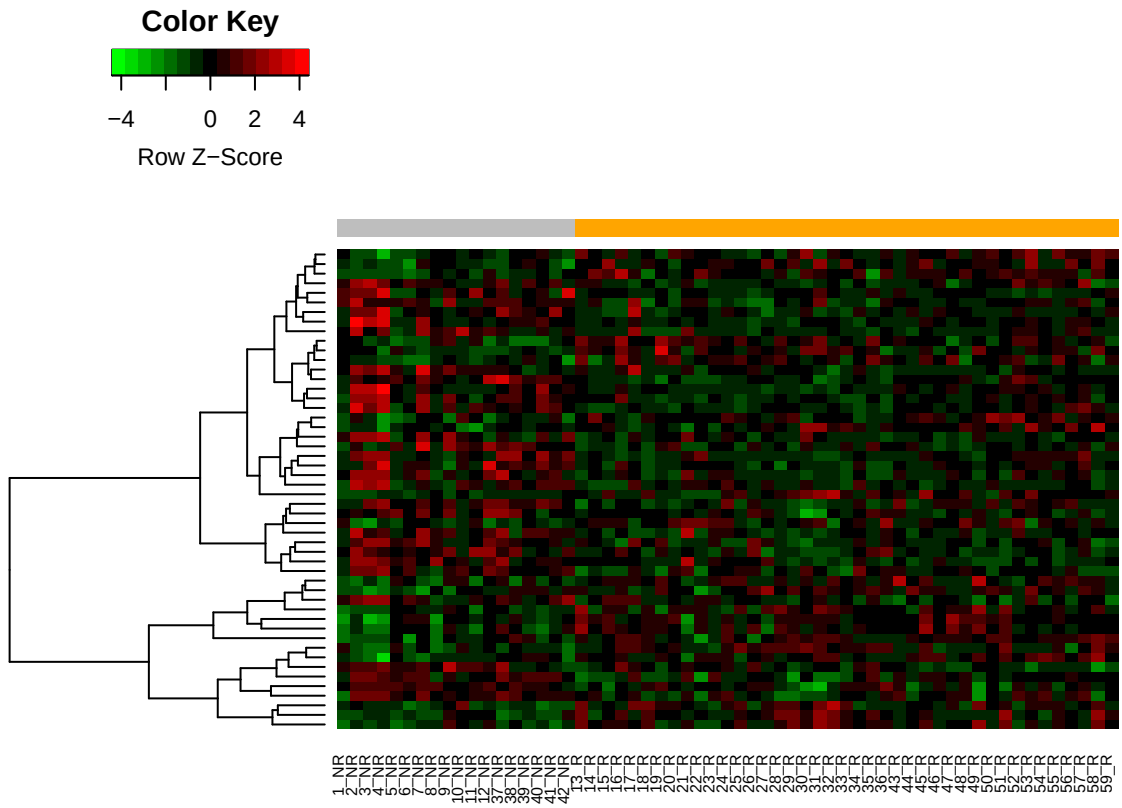


Correlation

Choosing the genes that are most correlated to the respone (either positive or negative).

```
resp <- as.numeric(pData(eset)$response)
expData <- exprs(eset)

c<-  cor (t(expData),resp)
#m[order(m[,1]),]
cc<- c[order(-abs(c[,1])),  ]
c50 <- cc[1:50]
c1000 <- cc[1:1000]
corrGenes <- names(c50)
corr1000Genes <- names(c1000)
heatmap.2(expData[corrGenes,],
          scale="row",trace="none",density.info="none",col=greenred,Colv = FALSE,
          labCol = labels,labRow=NA,ColSideColors=column.classification,key=T)
```

```
## Warning in heatmap.2(expData[corrGenes, ], scale = "row", trace = "none", :
## Discrepancy: Colv is FALSE, while dendrogram is `both'. Omitting column
## dendogram.
```

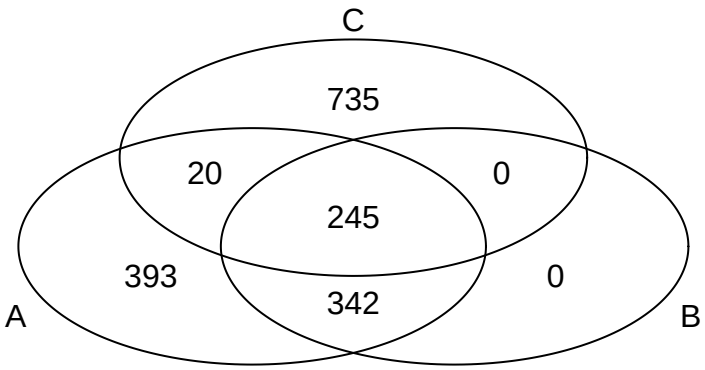


Comparison between differant genesets

Now have 3 differant sets of genes generated in differant ways, limma,corr and topDiff. These have been used to produce differant size genesets , 1000 for random forest (except limma that only had 587 genes statistically differant with a fold change of over 1.4 (log2(0.5).with a subset of the top 50 to use with logistic regression . These were reduced more to see the effects on the results.

Venn Diagram showing the seperation of genesets A= top average differce, B= Differance from Limma, C= Genes most correlated to reponse.

```
venn(list(topDiff1000,Limma.all,corr1000Genes))
```



Training and Test Data

To estimate the predictive ability of the models produced, we need to test the model on data not used in training the model. Splitting the data into training and test samples is the easiest and most simplistic way to achieve this. This may well affect the potential of the model by reducing the amount of training data which is already small compared to the number of features being searched . A more sophisticated approach is leave-one-out cross-validation which is used by the Random Forest algorithm.

The data was split into training set and a test set in a proportion of 7:3 resulting in 41 (R:NR 28:13) samples in the training set and 18 (R:NR 13:5) in the test set.

```
d <- t(exprs(eset))
set.seed(1)
trainSet<-sample_frac(as.data.frame(d), 0.7)
sid<-(rownames(trainSet)) # because rownames() returns character
testSet<- as.data.frame(d[!(rownames(d) %in% sid),])
trainResponse <- pData(eset)[rownames(trainSet),]$response
testResponse <- pData(eset)[rownames(testSet),]$response
colnames(testSet) <- paste0("X",colnames(testSet))
allData <- data.frame(t(expData))
```

Modeling

Logistic Regression Modeling

1.For correlated genes

```
genelist <- corrGenes
response <- trainResponse
d = data.frame(trainSet[,genelist])


vars = paste0('response ~ X', paste(genelist, collapse=' + X'))
LRcorr.model = glm(as.formula(vars), data=d, family=binomial(link='logit'))

reality <- pData(eset)$response
Reality_pred <- data.frame(reality, row.names = colnames(exprs(eset)))
corrPredA <- predict(LRcorr.model,allData , type= "response")

corrPredAll <- factor( (corrPredA > 0.5),labels = c("NR","R"))
RPtemp <- merge(Reality_pred,corrPredAll, by =0, all.x=TRUE)
Reality_pred$corrA <- RPtemp$y

corrPredT <- predict(LRcorr.model,testSet , type= "response")
corrPredTest <- factor( (corrPredT > 0.5),labels = c("NR","R"))
RPtemp <- merge(Reality_pred,corrPredTest, by =0, all.x=TRUE)
Reality_pred$corrT <- RPtemp$y
```

2.For Limma genes.

```
genelist <- Limma50
d = data.frame(trainSet[,genelist])


vars = paste0('response ~ X', paste(genelist, collapse=' + X'))
LRLimma.model = glm(as.formula(vars), data=d, family=binomial(link='logit'))

pred <-   predict(LRLimma.model,allData, type= "response")
pred <- factor( (pred > 0.5),labels = c("NR","R"))
RPtemp <- merge(Reality_pred,pred, by =0, all.x=TRUE)
Reality_pred$LGLimmaA <-RPtemp$y

pred <-   predict(LRLimma.model,testSet, type= "response")
pred <- factor( (pred > 0.5),labels = c("NR","R"))
RPtemp <- merge(Reality_pred,pred, by =0, all.x=TRUE)
Reality_pred$LGLimmaT <-RPtemp$y
```

3.For MeanDiff genes.

```
genelist <- topDiff
d = data.frame(trainSet[,genelist])


vars = paste0('response ~ X', paste(genelist, collapse=' + X'))
LRdiff.model = glm(as.formula(vars), data=d, family=binomial(link='logit'))

pred <-   predict(LRdiff.model, allData, type= "response")
pred <- factor( (pred > 0.5),levels = c(FALSE,TRUE),labels = c("NR","R"))
RPtemp <- merge(Reality_pred,pred, by =0, all.x=TRUE)
Reality_pred$meandiffA <-RPtemp$y

pred <-   predict(LRdiff.model, testSet, type= "response")
pred <- factor( (pred > 0.5),levels = c(FALSE,TRUE),labels = c("NR","R"))
RPtemp <- merge(Reality_pred,pred, by =0, all.x=TRUE)
Reality_pred$meandiffT <-RPtemp$y
```

4.First attempt to refine the model by using only the most important genes in new model.

```
genelist <- corrGenes
response <- trainResponse
d = data.frame(trainSet[,genelist])

anovaModel <-anova(LRcorr.model,test="Chisq")
head(anovaModel)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: response
##
## Terms added sequentially (first to last)
##
##
##               Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                             40     51.221
## X220979_s_at   1  10.4474        39     40.773 0.001228 **
## X40612_at      1   9.6075        38     31.166 0.001938 **
## X228411_at     1   4.6239        37     26.542 0.031529 *
## X1553572_a_at  1   6.5402        36     20.002 0.010546 *
## X239849_at     1   3.8441        35     16.158 0.049922 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# pick the significant genes
glmGenes <- row.names(anovaModel[anovaModel$`Pr(>Chi)`<.05, ])
glmGenes <- glmGenes [2:5]

newVars <-  paste0('response ~ ', paste(glmGenes, collapse=' +'))
```

```r
newModel = glm(as.formula(newVars), data=d, family=binomial(link='logit'))
summary(newModel)
```

```
##
## Call:
## glm(formula = as.formula(newVars), family = binomial(link = "logit"),
##     data = d)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.05087  -0.04957   0.04913   0.40788   1.75973
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -47.13940   38.29697  -1.231   0.2184
## X220979_s_at   -8.78028    3.57597  -2.455   0.0141 *
## X40612_at      -2.09600    1.36816  -1.532   0.1255
## X228411_at     -0.01987    1.09878  -0.018   0.9856
## X1553572_a_at  13.95733    6.88461   2.027   0.0426 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 51.221  on 40  degrees of freedom
## Residual deviance: 20.002  on 36  degrees of freedom
## AIC: 30.002
##
## Number of Fisher Scoring iterations: 7
```

```r
anovaModel <-anova(newModel,test="Chisq")
head(anovaModel)
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: response
##
## Terms added sequentially (first to last)
##
##
##               Df Deviance Resid. Df Resid. Dev Pr(>Chi)
## NULL                             40     51.221
## X220979_s_at   1  10.4474        39     40.773 0.001228 **
## X40612_at      1   9.6075        38     31.166 0.001938 **
## X228411_at     1   4.6239        37     26.542 0.031529 *
## X1553572_a_at  1   6.5402        36     20.002 0.010546 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
newPred <- predict(newModel, allData, type= "response")
newPred <-factor( (newPred > 0.5),labels = c("NR","R"))
Reality_pred$LGcorrImpA <- newPred

pred <-   predict(newModel, testSet, type= "response")
pred <- factor( (pred > 0.5),levels = c(FALSE,TRUE),labels = c("NR","R"))
RPtemp <- merge(Reality_pred,pred, by =0, all.x=TRUE)
Reality_pred$LGcorrImpT <-RPtemp$y
```

Functions to calculate the accuracy,FDR,precision and recall from the confusion matrix.

```
accFDR <- function (TP,FP,TN,FN){
    Acc <- (TN+TP)/(TP+TN+FP+FN)
    Pre <- TP/(TP+FP)
    Recall <- TP/(TP+FN)
    FDR <- FP/(FP+TN)
    resList <- list(c(Acc,FDR,Pre,Recall,TP,FP,TN,FN))
    return (resList)
}
```

Function Calculate Accuracy,FDR, recall and precision from TP, FP, FN, TN for each model

```
confuse <- function(rp) {
  res <- list("")
  resultList <- list ("")
  predNames <- colnames(rp)
  for (x in 1:(ncol(rp)-1)){
      predName <- predNames[x+1]
      realPred <- rp[, c("reality" ,predName)]
     # final[complete.cases(final),]
      colnames (realPred) <- c("reality","prediction")
      realPred <-realPred[complete.cases(realPred),]
      TP<-FP<-FN<-TN <-0
      for (i in 1:nrow(realPred)){
        if (realPred$reality[i]== "NR" && realPred$prediction[i]=="NR")
          TP =TP +1
        else if (realPred$reality[i]== "NR" && realPred$prediction[i]== "R")
          FN = FN +1
        else if (realPred$reality[i]== "R" && realPred$prediction[i]== "NR")
          FP = FP +1
        else if(realPred$reality[i]== "R" && realPred$prediction[i]=="R")
          TN = TN +1
      }

    resAcc <- accFDR(TP,FP,TN,FN)
    resultList <- append(resultList,resAcc)
  }
   return(resultList)
}
```

Randomn Forest

1.Correlated geneset

```
response <- pData(eset)$response
subsetGenes <- corr1000Genes
trainSubset <- t(expData[subsetGenes,])

fitRF.corr <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)
confusion <- fitRF.corr$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults <- data.frame(accRes[[1]])
colnames(RFresults) = ("RFcorrA")
```

top 1000 correlated geneset

```
subsetGenes <- corrGenes
trainSubset <- t(expData[subsetGenes,])
fitRF.corr <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)
confusion <- fitRF.corr$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults$RFcorr50 <-  accRes[[1]]
```

Top 50 correlated geneset

```
subsetGenes <- corrGenes[1:6]
trainSubset <- t(expData[subsetGenes,])
fitRF.corr <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)
confusion <- fitRF.corr$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults$RFcorr6 <-  accRes[[1]]
```

Top 6 correlated genes

2.Limma Geneset

```
subsetGenes <- Limma.all
trainSubset <- t(expData[subsetGenes,])

# Fitting model
fitRF.limma <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)
```

```
confusion <- fitRF.limma$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults$RFlimmaAll <-  accRes[[1]]
```

Limma all geneset (567 genes)

```
subsetGenes <- Limma50
trainSubset <- t(expData[subsetGenes,])

# Fitting model
fitRF.limma <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)

confusion <- fitRF.limma$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults$RFlimma50 <-  accRes[[1]]
```

Limma 50 geneset

```
subsetGenes <- Limma50[1:10]
trainSubset <- t(expData[subsetGenes,])

# Fitting model
fitRF.limma <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)

confusion <- fitRF.limma$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults$RFlimma10 <-  accRes[[1]]
```

Limma 10 geneset

3.Mean Difference Geneset

```
subsetGenes <- topDiff1000
trainSubset <- t(expData[subsetGenes,])
fitRF.meanD <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)
confusion <- fitRF.meanD$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults$RFtopD1000 <-  accRes[[1]]
```

top 1000 geneset

```
subsetGenes <- topDiff1000[1:50]
trainSubset <- t(expData[subsetGenes,])
fitRF.meanD <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)
confusion <- fitRF.meanD$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults$RFtopD50 <-  accRes[[1]]
```

top 20 geneset

```
subsetGenes <- topDiff[1:10]
trainSubset <- t(expData[subsetGenes,])
fitRF.meanD <- randomForest(x= trainSubset, y=response ,importance=TRUE ,ntree=5001, proximity=TRUE)
confusion <- fitRF.meanD$confusion
accRes<- accFDR (confusion[1,1],confusion[2,1],confusion[2,2],confusion[1,2])
RFresults$RFtopD10 <-  accRes[[1]]
```

Top 10 of mean differance geneset

Results

```
LGResults <- confuse (Reality_pred)
resNames <- colnames(Reality_pred)
LGtable <- data.frame(LGResults[[2]])
for (i in 3:(length(LGResults)))
   LGtable <- cbind(LGtable,LGResults[[i]])
rownames(LGtable) <- c(  "Accuracy","FDR", "Precision", "Recall","TP","FP","TN","FN"  )
colnames(LGtable) <- resNames[2:length(resNames)]
LGtable <- t(LGtable)
```

Logistic Regression

```
kable (LGtable, digits = 2)
```

	Accuracy	FDR	Precision	Recall	TP	FP	TN	FN
corrA	0.69	0.27	0.50	0.61	11	11	30	7
corrT	0.44	0.55	0.33	0.43	3	6	5	4
LGLimmaA	0.78	0.17	0.63	0.67	12	7	34	6
LGLimmaT	0.72	0.18	0.67	0.57	4	2	9	3
meandiffA	0.76	0.17	0.61	0.61	11	7	34	7
meandiffT	0.67	0.18	0.60	0.43	3	2	9	4
LGcorrImpA	0.90	0.05	0.88	0.78	14	2	39	4
LGcorrImpT	0.83	0.00	1.00	0.57	4	0	11	3

Random Forest

```
rownames(RFresults) <- c(  "Accuracy","FDR", "Precision", "Recall","TP","FP","TN","FN"  )
RFresults <- t(RFresults)
kable (RFresults, digits = 2)
```

	Accuracy	FDR	Precision	Recall	TP	FP	TN	FN
RFcorrA	0.80	0.02	0.88	0.39	7	1	40	11
RFcorr50	0.86	0.02	0.92	0.61	11	1	40	7
RFcorr6	0.88	0.05	0.87	0.72	13	2	39	5
RFlimmaAll	0.75	0.07	0.67	0.33	6	3	38	12
RFlimma50	0.80	0.07	0.75	0.50	9	3	38	9
RFlimma10	0.76	0.10	0.67	0.44	8	4	37	10
RFtopD1000	0.73	0.07	0.62	0.28	5	3	38	13
RFtopD50	0.76	0.12	0.64	0.50	9	5	36	9
RFtopD10	0.73	0.17	0.56	0.50	9	7	34	9

Discussion

All the plots of expression by sample show how noisey the genes are compared to the fold change. This together with the lack of sample clustering in the heatmaps meant the expectations of building a reasonably accurate predictive model were quite low.

The LR models were all built just using the training subset of data, but predictions were made on both the test data *.T and on all the data* A to try and get improve the accuracy of the accuracy/recall data. (This is maybe completely invalid). The accuracy of all the models is very poor and the recall is very low so these would be useless as predictors of drug response where it is important to pick up all the non-responders. In this scenario it is important to have a very low FN number , to make sure all the non-responders are identified at the expense of falsely identifying responders (FP).

The improvements made by picking only the 4 most important genes is very dramatic with accuracy rising (for just the test set) from 0.44 to 0.83 and the recall going from 0.43 to 0.57 (FN drops by 1 from 4 to 3).

The results from the RF models show generally a higher accuracy but low recall which mmeans an unacceptable level of FN. Using 50 rather than 1000 (587 for Limma) improves accuracy but then when reducing this to 10 genes it drops. The use of 6 genes for the correlation genest was chosen by trial and error to get the lowest FN number.

In future attempts to improve the models we should first check if the most important genes in each geneset overlap within each LG model and how many of the genes are in common. It may also be interesting to pick genesets by an unsupervised clustering method to compare the results.

Tuning the models, this was the first step in developing a model ad it can be seen by the improvement in LG for the correlation data the improvement. It would be interesting to see what further improvements were possible and what would happen if you combined the most important genes from each geneset.

Tuning the random forest model is more complicated because we do not get given a list of most important genes because different variables are important in different regions of the search. This means tuning is not a straight forward process but RF models are known to be greatly enhanced by removing the unimportant variables.

In conclusion it looks like both methods have the potential to produce an accurate predictor,although the RF models were more accurate overall it was at the expense of low recall which is more important in this

case. Further investigation of the refinement of both models is needed to attempt to improve the recall. Unfortunately I did not have time to refine the random forest models other than by trial and error. More data should be used to verify any model produced.

References

[1] Shannon K. McWeeney, Lucy C. Pemberton, Marc M. Loriaux, Kristina Vartanian, Stephanie G. Willis, Gregory Yochum, Beth Wilmot, Yaron Tur- paz, Raji Pillai, Brian J. Druker, Jennifer L. Snead, Mary MacPartlin, Stephen G. O'Brien, Junia V. Melo, Thoralf Lange, Christina A. Harring- ton, and Michael W. N. Deininger. A gene expression signature of cd34+ cells to predict major cytogenetic response in chronic-phase chronic myeloid leukemia patients treated with imatinib. Blood, 115(2):315{325, 2010.

Predictions for LG modeling

```
kable(Reality_pred)
```

	reality	corrA	corrT	LGLimmaA	LGLimmaT	meandiffA	meandiffT	LGcorrImpA	LGcorr
GSM366179.CEL	NR	NR	NA	NR	NA	NR	NA	R	NA
GSM366180.CEL	NR	NR	NR	R	R	NR	NR	NR	NR
GSM366181.CEL	NR	NR	NA	NR	NA	NR	NA	NR	NA
GSM366182.CEL	NR	NR	NA	NR	NA	NR	NA	NR	NA
GSM366183.CEL	NR	NR	NA	NR	NA	NR	NA	NR	NA
GSM366184.CEL	NR	NR	NR	NR	NR	R	R	NR	NR
GSM366185.CEL	NR	R	R	NR	NR	R	R	NR	NR
GSM366186.CEL	NR	R	R	NR	NR	NR	NR	NR	NR
GSM366187.CEL	NR	NR	NA	NR	NA	NR	NA	NR	NA
GSM366188.CEL	NR	NR	NA	NR	NA	NR	NA	R	NA
GSM366189.CEL	NR	NR	NA	NR	NA	NR	NA	NR	NA
GSM366190.CEL	NR	NR	NA	NR	NA	NR	NA	NR	NA
GSM366215.CEL	NR	R	NA	R	NA	R	NA	NR	NA
GSM366216.CEL	NR	R	R	NR	NR	NR	NR	NR	R
GSM366217.CEL	NR	R	NA	R	NA	R	NA	R	NA
GSM366218.CEL	NR	R	NA	R	NA	R	NA	R	NA
GSM366219.CEL	NR	NR	NR	R	R	R	R	NR	R
GSM366220.CEL	NR	R	R	R	R	R	R	NR	R
GSM366191.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366192.CEL	R	NR	NR	R	R	R	R	R	R
GSM366193.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366194.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366195.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366196.CEL	R	NR	NR	R	R	R	R	R	R
GSM366197.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366198.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366199.CEL	R	R	NA	R	NA	R	NA	NR	NA
GSM366200.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366201.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366202.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366203.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366204.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366205.CEL	R	R	NA	R	NA	R	NA	R	NA

	reality	corrA	corrT	LGLimmaA	LGLimmaT	meandiffA	meandiffT	LGcorrImpA	LGcorr
GSM366206.CEL	R	R	R	R	R	NR	NR	R	R
GSM366207.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366208.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366209.CEL	R	NR	NA	NR	NA	NR	NA	R	NA
GSM366210.CEL	R	NR	NA	NR	NA	NR	NA	R	NA
GSM366211.CEL	R	R	R	R	R	R	R	R	R
GSM366212.CEL	R	NR	NA	NR	NA	NR	NA	R	NA
GSM366213.CEL	R	NR	NA	NR	NA	NR	NA	R	NA
GSM366214.CEL	R	NR	NA	NR	NA	NR	NA	R	NA
GSM366221.CEL	R	R	R	NR	NR	NR	NR	R	R
GSM366222.CEL	R	R	R	R	R	R	R	R	R
GSM366223.CEL	R	R	R	R	R	R	R	R	R
GSM366224.CEL	R	NR	NR	NR	NR	R	R	R	R
GSM366225.CEL	R	NR	NR	R	R	R	R	R	R
GSM366226.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366227.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366228.CEL	R	NR	NR	R	R	R	R	R	R
GSM366229.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366230.CEL	R	R	NA	R	NA	R	NA	NR	NA
GSM366231.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366232.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366233.CEL	R	NR	NR	R	R	R	R	R	R
GSM366234.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366235.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366236.CEL	R	R	NA	R	NA	R	NA	R	NA
GSM366237.CEL	R	R	NA	R	NA	R	NA	R	NA