

Statistical Methods Supplement and R software tutorial: Gene Filtering with a Random Forest Predictor

Steve Horvath*, Tao Shi*, Ruty Mehrian Shai, Charlie Chen, Stan Nelson

* SH and TS jointly carried out the random forest analysis
Statistical Correspondence: shorvath@mednet.ucla.edu

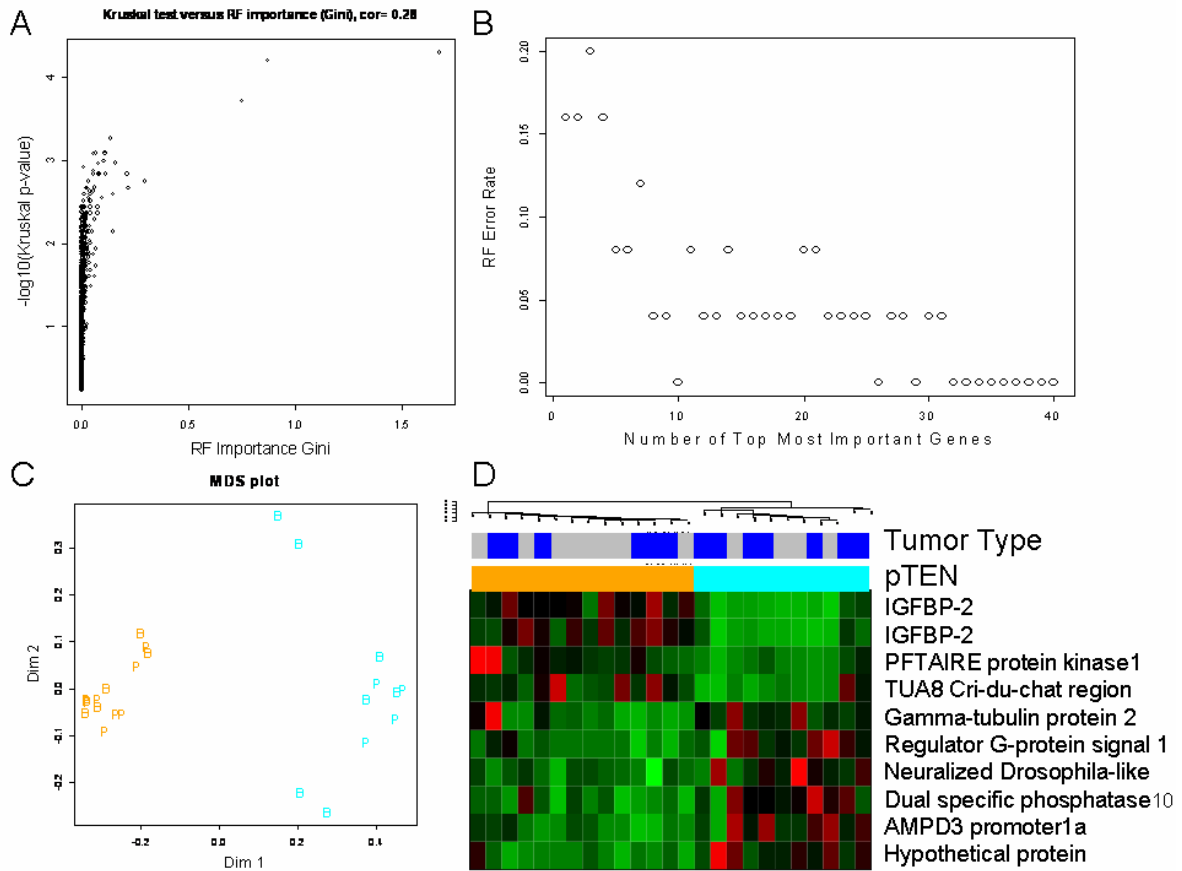
Here we provide additional information on the Materials and Methods for the random forest analysis of following article

Mehrian Shai R, Chen CD, Shi T, Horvath S, Nelson SF, Reichardt JKV, Sawyers CL (2007) IGFBP2 is a Biomarker for PTEN Status and PI3K/Akt Pathway Activation in Glioblastoma and Prostate Cancer. PNAS

The data and the statistical code ensures full reproducibility all of our findings. The supplementary text also serves as a tutorial for random forest analysis with the R software. Some familiarity with the R software (www.r-project.org) is desirable, but the document is fairly self-contained.

We use random forest predictors (Breiman 2001) to find genes that are associated with PTEN status in brain cancer (glioblastoma multiform) and prostate tumors. In our data, we find that 10 probesets are associated with PTEN status irrespective of the tissue origin. While our main analysis uses a random forest importance measure to implicate these 10 probesets, we show that they are also statistically significant according to a Kruskal Wallis test or a Student T-test. We use supervised hierarchical clustering and a classical multi-dimensional scaling plot to visualize the relationship between the microarrays (patients).

In particular, we focus on the results surrounding **Figure 1** which is reproduced here for convenience.



Legend: Molecular signature of the PTEN tumor suppressor gene. Microarrays were used to measure the expression profiles of 25 prostate cancer xenograft and glioblastoma samples, of which 11 have wild-type and 14 have mutated PTEN genes. To relate PTEN status to the gene expression data, we used random forest predictors and the Kruskal Wallis test of differential expression. A random forest importance measure was used to rank probesets according to their prognostic importance for PTEN status.

A. The random forest variable importance measure is highly correlated ($r=0.28$) with a conventional measure of differential expression (minus log 10 of the Kruskal Wallis test p-value). We find that our results are highly robust with respect to the gene screening method. **B.** The error rate of the random forest predictor as a function of the number of most important probesets. We find that the 10 most important probesets lead to an apparent error rate of 0. **C.** Classical multidimensional scaling plot for visualizing the dissimilarities between the microarray samples (based on the 10 most important probesets). Prostate cancer samples are labelled by "P" and glioblastoma by "B". PTEN mutated samples are colored in orange and wild-type in cyan (turquoise). This plot and the supervised hierarchical clustering plot (**D.**) show that the 10 most important probesets stratify the samples according to their PTEN status.

Review of random forest predictors

In the following, we will briefly review how to construct an RF predictor, how to arrive at error rate estimates, and how to estimate variable importance.

Here we adopt a review of random forest predictors from Breiman (2001).

An RF predictor is an ensemble of individual classification tree predictors (Breiman 2001). For each observation, each individual tree votes for one class and the forest predicts the class that has the plurality of votes. The user has to specify the number of randomly selected variables $mtry$ to be searched through for the best split at each node. The Gini index (Breiman et al 1984) is used as the splitting criterion. The largest tree possible tree is grown and is not pruned. The root node of each tree in the forest contains a bootstrap sample from the original data as the training set. The observations that are not in the training set, roughly 1/3 of the original data set, are referred to as out-of-bag (OOB) observations. One can arrive at OOB predictions as follows: for a case in the original data, predict the outcome by plurality vote involving only those trees that did not contain the case in their corresponding bootstrap sample. By contrasting these OOB predictions with the training set outcomes, one can arrive at an estimate of the prediction error rate, which is referred to as the OOB error rate.

Variable Importance

As part of the RF construction, several measures of variable importance can be defined. Here we focus on the node purity based variable importance measure, which is defined as follows. Recall that node splits are selected with the Gini index: at every split, one of the randomly chosen variables is used to form the split and there is a resulting decrease in the Gini index. The Gini based variable importance measure is defined as the sum of all decreases in the forest due to the given variable, normalized by the number of trees.

To facilitate a comparison, we also report the findings based on the variable importance measure defined by the mean decrease in the prediction accuracy.

Random forest dissimilarity

Another by-product of the RF construction is the RF dissimilarity measure.

We will briefly review how to use random forests to arrive at a dissimilarity measure (Breiman and Cutler 2002). Since an individual tree is un-pruned, the terminal nodes will contain only a small number of observations. The training data are run down each tree. If observations i and j both land in the same terminal node, the similarity between i and j is increased by one. At the end of the forest construction, the similarities are symmetrized and divided by the number of trees. The similarity between an observation and itself is set equal to one. The similarities between objects form a matrix, SIM , which is symmetric, positive definite, and each entry lies in the unit interval $[0,1]$. The RF dissimilarity is defined as $DISSIM(ij) = 1 - SIM(ij)$. The RF dissimilarity can be used as input of multi-dimensional scaling (MDS) or for clustering analyses.

Since a prediction outcome was used to create the RF dissimilarity, the resulting multidimensional scaling plot and hierarchical clustering tree should be considered as results of a supervised learning approach.

We briefly mention that one can also use random forest predictors in unsupervised learning (Breiman and Cutler 2002, Horvath and Shi 2006).

Classical multi-dimensional scaling plots

We used a multi-dimensional scaling plot to visualize the RF dissimilarities between the microarray samples (patients). Classical multidimensional scaling (MDS) is an unsupervised learning method, which takes the dissimilarities between tumor samples and returns a set of points in a lower dimensional space such that the distances between the points are approximately equivalent to the dissimilarities (Venables and Ripley, 1999). A review can be found in Cox and Cox (2001). Here we use 2 dimensional Euclidean space (the plane) which minimizes a "stress" function between the Euclidean distances and the reandom forest dissimilarity.

To be more specific, suppose we have a RF dissimilarity matrix (d_{ij}) between all pairs of n points. Classical MDS minimizes the "stress" function $\sum_{i \neq j} [d_{ij} - \tilde{d}_{ij}]^2$ where \tilde{d}_{ij} are the Euclidean distances between the points in the low (here 2) dimensional space.

Why use random forest predictors for gene screening?

Many prediction methods have been proposed for classifying tissues based on gene expression profiles; a comparison and review can be found in Fridlyand et al (2002). Here we use a random forest (RF) predictor by Breiman (2001) since it has many attractive properties for this application.

In general, a random forest predictor may be a good choice when most variables are noise and/or when the number of variables are much larger than the number of observations.

Since we expected that only very few genes are related to PTEN status, an RF predictor is a natural choice in this application.

Several authors have proposed to use random forests and its associated variable importance measure to screen for genes, e.g. (Breiman and Cutler 2002, Wu et al 2003, Gunther et al 2003, Izmirlian 2004, Man et al 2004, Alvarez et al 2005, Diaz et al 2006). Empirical studies suggest that random forest predictors are well suited for microarray data (Diaz et al 2006).

With relatively few samples other prediction methods, e.g. k-nearest neighbor or linear discriminant analysis require a variable selection step before fitting the predictor. For example, as part of the predictor construction, Fridlyand et al (2002) select genes on the basis of the ratio of between to within sums of squares before proceeding to fit the predictor. In contrast, RF predictors do not require such a gene filtering step and all genes can be used.

The choice of the RF parameter *mtry*

While the results of an RF analysis are highly robust with respect to the RF parameter *mtry* (the number of variables considered at each split), it is advisable to choose a high value of *mtry* if one expects that few genes are related to the outcome. A low value of *mtry* is appropriate when most variables are highly related to the outcome.

For each random forest fit, we chose 30000 (thirty thousand) trees and varied the number of random features *mtry* to assess the robustness of our findings. The default value of random features is the square root of the number of variables, i.e. the default value would have been *mtry*~75. Instead we found that large values, e.g. *mtry*=5000 leads to higher

prediction accuracy. This reflects the fact that relatively few genes are associated with PTEN status.

Random forest software

In this R tutorial, we use the randomForest library in R, which was created by Andy Liaw and Matthew Wiener based on original Fortran code by Leo Breiman and Adele Cutler. We also analyzed the data with the original software code.

Normalization and pre-processing of the gene expression data

For glioblastoma samples: Affymetrix U95av2 was used to interrogate 12533 probe sets encoding ~ 10,000 genes. For the prostate cancer xenografts, Hu6800 arrays were used interrogating 6,412 known genes (Affymetrix Santa Clara, CA). A list of U95av2-Hu6800 common genes was constructed to compare expression in all samples.

The .CEL files for all the microarray hybridizations (27 samples) generated by Affymetrix Microarray Suite Software were imported into the software dChip (Li and Wong, 2001). All arrays were normalized against the array with median overall intensity. We computed model based expression indices for each gene with PM/MM difference model and flooring the low expression values to 10th percentile of expressions called “A” in dChip. We restricted the analysis to the 10000 most variable (across all the 25 samples) genes to reduce the noise level. To avoid possible batch effects due to tissue or array type, we standardized the gene expression values within each tumor type; thus for a given gene, the expression levels had mean 0 and variance 1 within each tumor type. A few genes could not be standardized since the variance of their expression within a tissue type was 0. Thus, we ended up with a total of 9895 genes.

Since our R installation could not handle all 9895 probesets, we restricted the analysis to the 6422 probesets with Kruskal Wallis p-value <0.6, i.e. we removed probesets that do not even show the slightest hint of differential expression between the groups defined by the PTEN status. As one can see from Figure 1A below, the genes with low random forest importance have high (insignificant) Kruskal Wallis p-value. (Note that we plot minus log₁₀ of the p-value on the y-axis of Figure 1A). This result can be used to argue that no information is lost for the random forest analysis when restricting the analysis to probesets with Kruskal Wallis p-value <0.6.

But we want to emphasize that our results are identical to those that use all 9895 most varying genes (as one can verify with Breiman’s original Fortran code). In particular, the out of bag estimate of the test set error rate is unbiased.

Random forest error rates

We arrive at an out of bag error rate of 0.32, standard deviation=0.093 when using 6422 or all 9895 genes. Incidentally, a naive predictor, which would assign each observation as PTEN mutated, would have led to an error rate of 0.44%. The out of bag error rate is unimpressive, which may be due to one or more of the following reasons. First, relatively few genes appear to be associated with PTEN status, i.e PTEN status does have appear to have a broad effect on gene expression in our data. Second, we had only 25 samples in this study. Future studies with more microarray samples may lead to higher prediction accuracy.

The main interest of our study was to screen for genes related to PTEN status (as opposed to predicting PTEN status). In Figure 1B, we show that one can arrive at an apparent error rate of 0 if one restricts the analysis to the 10 most important genes. As discussed below, this apparent error rate is biased.

How many genes are important for PTEN status?

To identify which genes are associated with PTEN status, one could threshold a statistical significance levels (e.g. an uncorrected p-values, corrected p-values, false discovery rate, q-values etc). Since we had relatively few samples and PTEN status appears to affect few genes, the false discovery rate is unimpressive. However, we show below that IGF2BP2 is implicated by multiple gene screening methods (including the Kruskal Wallis test and the Student t-test). Unfortunately, the choice of a significance threshold will always be somewhat arbitrary. Prior biological knowledge on how many genes are related to a particular microarray sample trait, may guide the choice of a significance threshold.

It is worth emphasizing that a gene screening procedure is not a gene testing procedure. Any result of a gene screening procedure should be carefully validated in independent data sets.

Since the random forest analysis gene screening approach is based on a predictor and we expected very few genes to be related to PTEN status, we used the following heuristic for selecting a final list of genes:

Select as final gene list, the smallest number of most important genes (probesets) that minimize the apparent error rate.

In **Figure 1B**, we determine how the apparent error rate of the random forest predictor depends on the number of most important genes. Specifically, we fitted a random forest predictor to gene expression data comprised of the top 1,2,3,...,40 most important genes. Although, we define the error rate as the OOB error rate, this error rate is highly biased since the genes (variables) were pre-selected with the variable importance measure. As can be seen from Figure 1B, the apparent error rate becomes 0 on the subset of the 10 most important genes. We find empirically and in simulations (unpublished data) that this error rate based heuristic results in small gene lists, which may or may not be attractive in a given application. Since we expected that PTEN status affects very few genes, the heuristic was plausible in this application. We are very aware of the pitfalls of using heuristics in gene screening experiments, which is why we carefully validated our main finding (IGF2BP2 gene) in independent biological studies.

Alternative gene screening methods based on the Kruskal Wallis test or T-test

To find differentially expressed genes, most researchers make use of gene screening methods based on a univariate measure of differential expression, e.g. Student T-test, Kruskal Wallis test, Wilcoxon test, etc. To assess the robustness of our results with respect to the gene screening method, we compared our findings to those involving the Kruskal Wallis test and the Student T-test.

The Kruskal Wallis test is a non-parametric multi-group comparison test. When dealing with 2 groups, it is equivalent to the Wilcoxon rank sum test. Since both the Kruskal Wallis test and the Wilcoxon test are based on the variable ranks, the resulting tests are invariant to monotonic transformations of the gene expression profiles.

Since the node splits in the random forest are also based on the variable ranks (as opposed to the original values), RF importance measures are scale invariant as well.

Because of its scale invariance, we prefer the Kruskal Wallis test over the the Student T-test in this application. In Figure 1A, we show that the the random forest variable importance measure based on node purity (Gini index) is highly related with the Kruskal Wallis test in this application. Below we relate the random forest importance measures to both the Student T test p-value and the Kruskal Wallis p-value.

References

- Alvarez S, Diaz-Uriarte R, Osorio A, Barroso A, Melchor L, Paz MF, Honrado E, Rodriguez R, Urioste M, Valle L, Diez O, Cigudosa JC, Dopazo J, Esteller M, Benitez J (2005) A Predictor Based on the Somatic Genomic Changes of the BRCA1/BRCA2 Breast Cancer Tumors Identifies the Non-BRCA1/BRCA2 Tumors with BRCA1 Promoter Hypermethylation. *Clin Cancer Res.* 2005;11:1146–1153.
- Breiman L (1999). Random forests, random features. Technical Report 567, Department of Statistics, University of California, Berkeley, 1999
- Breiman, L (2001). "Random Forests". *Machine Learning* 45 (1), 5-32
- Breiman L and Cutler A (2002), "Manual On Setting Up, Using, And Understanding Random Forests V3.1", http://oz.berkeley.edu/users/breiman/Using_random_forests_V3.1.pdf.
- Breiman L, Friedman JH, Olshen RA, Stone CJ (1984) *Classification and Regression Trees*. Wadsworth International Group
- Cox TF, Cox MAA (2001) *Multidimensional Scaling*. Chapman & Hall.
- Diaz-Uriarte R and Alvarez de Andres S (2006) Gene selection and classification of microarray data using random forest. *BMC Bioinformatics.* 2006; 7: 3.
- Fridlyand J, Dudoit S, Speed TP (2002). Comparison of discrimination methods for the classification of tumors using gene expression data. *Journal of the American Statistical Association.* 97:77-87.
- Gunther EC, Stone DJ, Gerwien RW, Bento P, Heyes MP. Prediction of clinical drug efficacy by classification of drug-induced genomic expression profiles in vitro. *Proc Natl Acad Sci USA.* 2003;100:9608–9613.
- Tao Shi and Steve Horvath (2006) Unsupervised Learning with Random Forest Predictors. *Journal of Computational and Graphical Statistics.* Volume 15, Number 1, March 2006, pp. 118-138(21)

- Izmirlian G (2004) Application of the random forest classification algorithm to a SELDI-TOF proteomics study in the setting of a cancer prevention trial. *Ann NY Acad Sci.* 2004;1020:154–174.
- Kaufman, L. and Rousseeuw, P.J. (1990) *Finding Groups in Data: An Introduction to Cluster Analysis.* Wiley, Inc., New York
- Li C, Wong WH (2001) Model-based analysis of oligonucleotide arrays: Expression index computation and outlier detection, *Proc. Natl. Acad. Sci.* Vol. 98, 31-36.
- R Development Core Team. *R: A language and environment for statistical computing.* R Foundation for Statistical Computing, Vienna, Austria; 2004. [ISBN 3-900051-00-3].
- Venables, W.N. and Ripley, B.D. (1999) *Modern Applied Statistics With S-Plus.* Springer, New York
- Wu B, Abbott T, Fishman D, McMurray W, Mor G, Stone K, Ward D, Williams K, Zhao H. (2003) Comparison of statistical methods for classification of ovarian cancer using mass spectrometry data. *Bioinformatics.* 2003;19:1636–1643.

R software session

```
# Downloading the R software from http://www.R-project.org
# After installing R, you need to install several additional R library packages:
# For example to install randomForest, open R,
# go to menu "Packages\Install package(s) from CRAN",
# then choose randomForest. R will automatically install the package.
# When asked "Delete downloaded files (y/N)? ", answer "y".
# Do the same for some of the other libraries mentioned below. But note that
# several libraries are already present in the software so there is no need to re-install them.

# The results presented in the following used R version R 2.4.0
# and randomForest package 4.5-16.

# To get this tutorial and data files, go to the following webpage
# http://www.genetics.ucla.edu/labs/horvath/RandomForestScreening

# Unzip all the files into the same directory.
#Please copy and paste the following script into the R session.
#Text after "#" is a comment and is automatically ignored by R.

# Set the working directory of the R session by using the following command.
# Please adapt the following path. Note that we use / instead of \ in the path.
setwd("C:/Documents and Settings/shorvath/My
Documents/ADAG/TaoShi/pTENpaperNov2006")

# read in the R libraries
library(randomForest)
library(sma)

# functions to compute the variance and mean
var1=function(x) var(x, na.rm=T)
mean1=function(x) mean(x, na.rm=T)

# read in the following comma delimited text files.
ExpressionData <- read.csv("MicroarrayDataBrainProstateStandardized9895.csv",
header=T, row.names=1)
# The following contains the expression data (genes are columns, rows are samples)
# It is worth repeating that the data have been standardized within tumor type
# to remove a potential `batch` effect due to tissue type.
datExpr <- data.frame(t(ExpressionData[-c(1:2),]))
# the following file contains gene information
GeneInformation=read.csv("GeneSummary.csv",header=T)
XProbeSet=paste("X", GeneInformation$ProbeSet, sep="")

# Here we check whether the order of probesets in the gene information file
```

```

#lines up with the order of the probesets in the expression data.
table(names(datExpr)==XProbeSet)

TRUE
9895

# WARNING: If you don't get that all are true, then you are not reading in the data
#correctly. Please sort them so that gene information and gene expression data line up.

# the outcome y is defined as ptenstatus
y = as.factor(c(as.numeric(ExpressionData[2,])))
# this vector assigns colors according to the outcome vector
ycolor=ifelse(y==1,"orange","cyan")
# this vector is the tumor type: 1 for glioblastoma (brain tumor), 2 for prostate cancer
tumortype=as.numeric(as.character( ExpressionData[1,]))
table(tumortype)
tumortypelabel=ifelse(tumortype==1,"B", "P")
tumortypecolor=ifelse(tumortype==1,"blue", "grey")

# Differential expression between the groups defined by y
# calculate Kruskal test p value for each probe set
p.kruskal = apply(datExpr, 2, function(x) kruskal.test(x~y)$p.value )
# Student T test p-value
p.ttest= apply(datExpr, 2, function(x) t.test(x~y)$p.value )
ttest.statistic= apply(datExpr, 2, function(x) t.test(x~y)$statistic )
MeanExprPtenWildtype=apply(datExpr[y==2,],2,mean)
MeanExprPtenMutant=apply(datExpr[y==1,],2,mean)

# The following vectors will contain the 2 variable importance measures
# We focus the analysis on the importance measure based on node purity, which is
# defined as mean decrease in Gini index.
RFImportanceGini=rep(NA, dim(datExpr)[[2]] )
# To facilitate a comparison, we also compute the importance measure based on the mean
#decrease in accuracy
RFImportanceAccuracy=rep(NA, dim(datExpr)[[2]] )

```

```
#Since the R version of the random forest package cannot deal with a large number of
#genes, we restrict the analysis to a subset of the genes.
```

```
restrictGenes1 = p.kruskal < 0.6
RFImportanceGini[!restrictGenes1]=0
RFImportanceAccuracy[!restrictGenes1]=0
```

```
datExpr2=datExpr[,restrictGenes1]
dim(datExpr2)
```

```
25 6422
```

```
# Thus there are 6422 genes in the analysis. Our version of R can handle this size.
```

```
p.kruskal2=p.kruskal[restrictGenes1]
GeneInformation2=GeneInformation[restrictGenes1,]
p.ttest2= p.ttest[restrictGenes1]
ttest.statistic2= ttest.statistic[restrictGenes1]
MeanExprPtenWildtype2= MeanExprPtenWildtype[restrictGenes1]
MeanExprPtenMutant2= MeanExprPtenMutant[restrictGenes1]
```

Fitting the random forest predictor

Since only few genes are associated with PTEN status, we chose a high value of `mtry`. Our results regarding IGFP2 are highly robust with respect to `mtry` as the user can verify by choosing different values of `mtry`.

We fit a lot of trees (30000) so that the estimate of the importance measure becomes stable.

```
# To improve the reproducibility of our analysis, we set the seed of the random generator
set.seed(1)
```

```
RF1 = randomForest(factor(y)~., data=datExpr2, ntree=30000, importance=T,mtry=5000)
```

```
RF1
```

```
Call:
```

```
randomForest(formula = factor(y) ~ ., data = datExpr2, ntree = 30000,
importance = T, mtry = 5000)
```

```
      Type of random forest: classification
```

```
      Number of trees: 30000
```

```
No. of variables tried at each split: 5000
```

```
      OOB estimate of  error rate: 32%
```

```
Confusion matrix:
```

```
      1 2 class.error
```

```
1 11 3    0.2142857
```

```
2  5 6    0.4545455
```

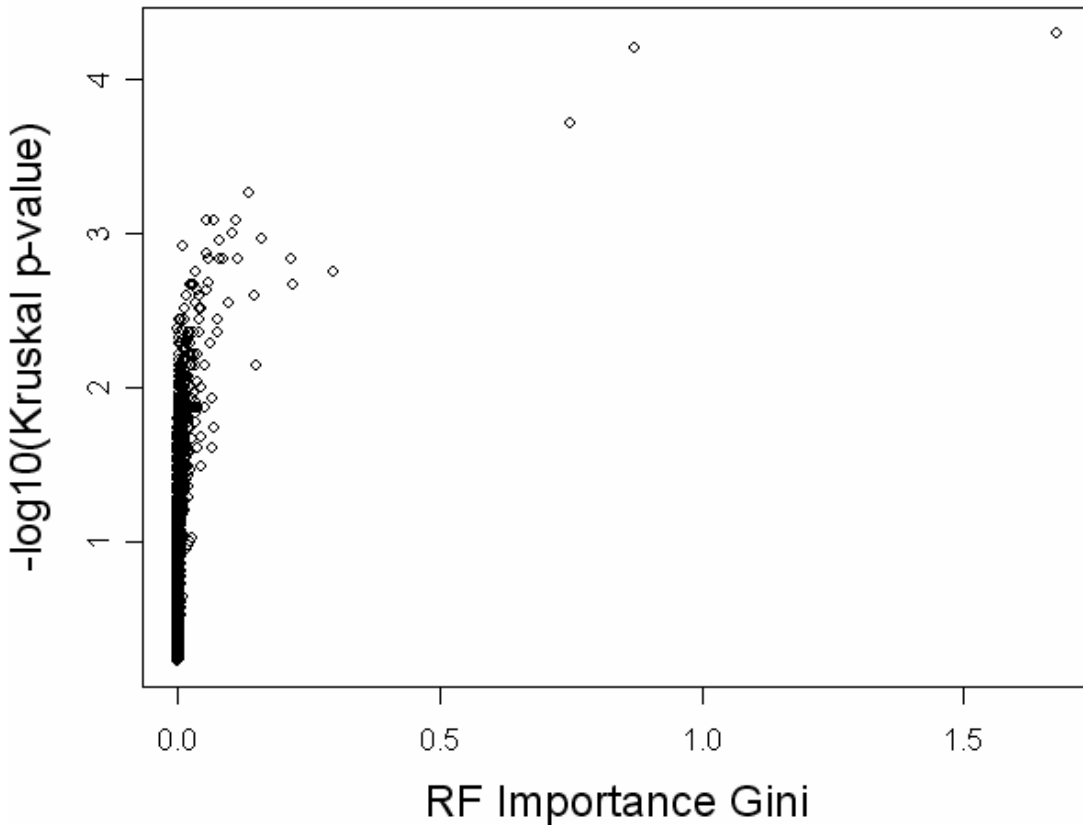
#Comment: This shows that the out-of bag error rate is 32 percent. Thus, the gene expression data do *not* contain a strong signal for predicting PTEN status. Only very few genes appear to be related to PTEN status.

The following plot shows that the random forest importance measure based on node purity (Gini index) is highly correlated with the p-value based on the Kruskal Wallis test. This is Figure 1A in our article.

```
# extract variable importance
var.imp = data.frame(importance(RF1))

par(mfrow=c(1,1))
x1= var.imp$MeanDecreaseGini
y1= -log10(p.kruskal[restrictGenes1])
plot(x1,y1,xlab="RF Importance Gini",ylab="-log10(Kruskal p-
value)",main=paste("Kruskal test versus RF importance (Gini),
cor=",signif(cor(x1,y1) ,2) ),cex.lab=1.5 )
```

Kruskal test versus RF importance (Gini), cor= 0.28

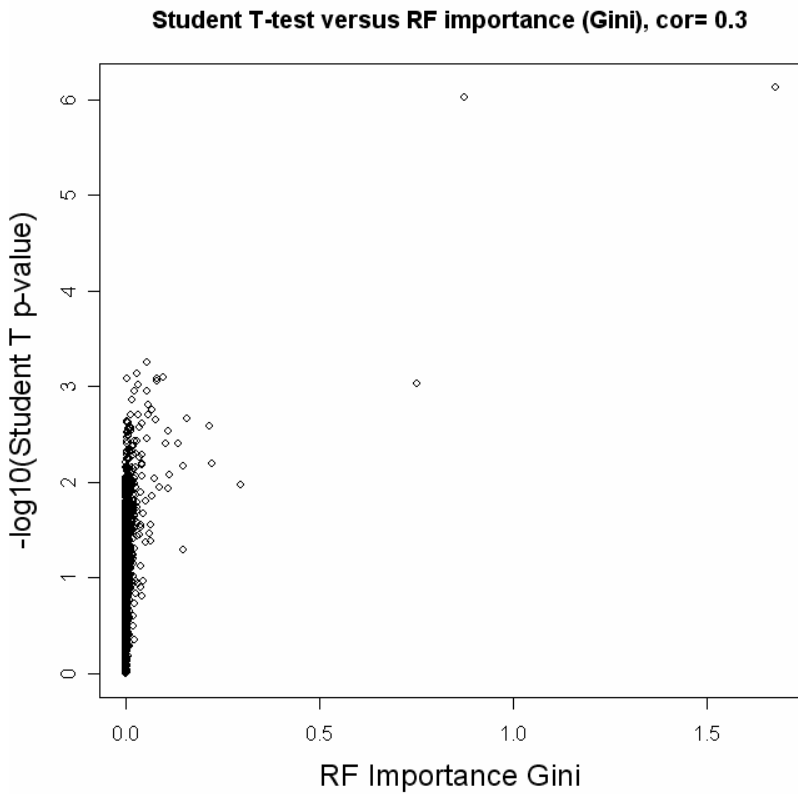


The above figure shows that when it comes to the top 2 probesets (in particular the #IGFBP2 gene) the two gene screening procedures agree.

```

par(mfrow=c(1,1))
x1= var.imp$MeanDecreaseGini
y1= -log10(p.ttest[restrictGenes1])
plot(x1,y1,xlab="RF Importance Gini",ylab="-log10(Student T p-
value)",main=paste("Student T-test versus RF importance (Gini),
cor=",signif(cor(x1,y1 ),2) ),cex.lab=1.5 )

```

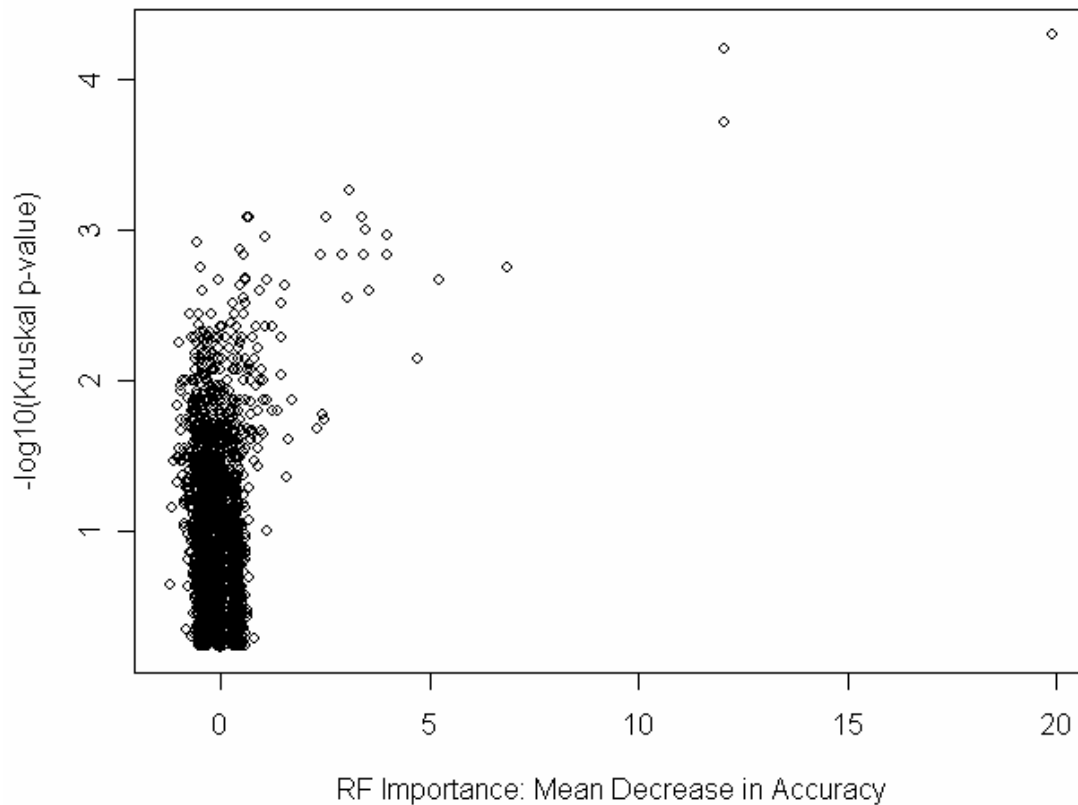


The above figure shows that when it comes to the top 2 probesets (in particular the #IGFBP2 gene) the two gene screening procedures agree.

The following plot shows the relationship between RF importance measure based on the mean decrease in accuracy and the Kruskal Wallis p-value. We find that this alternative RF importance measure is less related to the Kruskal Wallis test.

```
par(mfrow=c(1,1))
x1= var.imp$MeanDecreaseAccuracy
y1= -log10(p.kruskal[restrictGenes1])
plot(x1,y1,xlab="RF Importance: Mean Decrease in Accuracy",ylab="-log10(Kruskal p-
value)",main=paste("Kruskal test versus RF importance (Accuracy),
cor=",signif(cor(x1,y1) ,2) ) )
```

Kruskal test versus RF importance (Accuracy), cor= 0.14

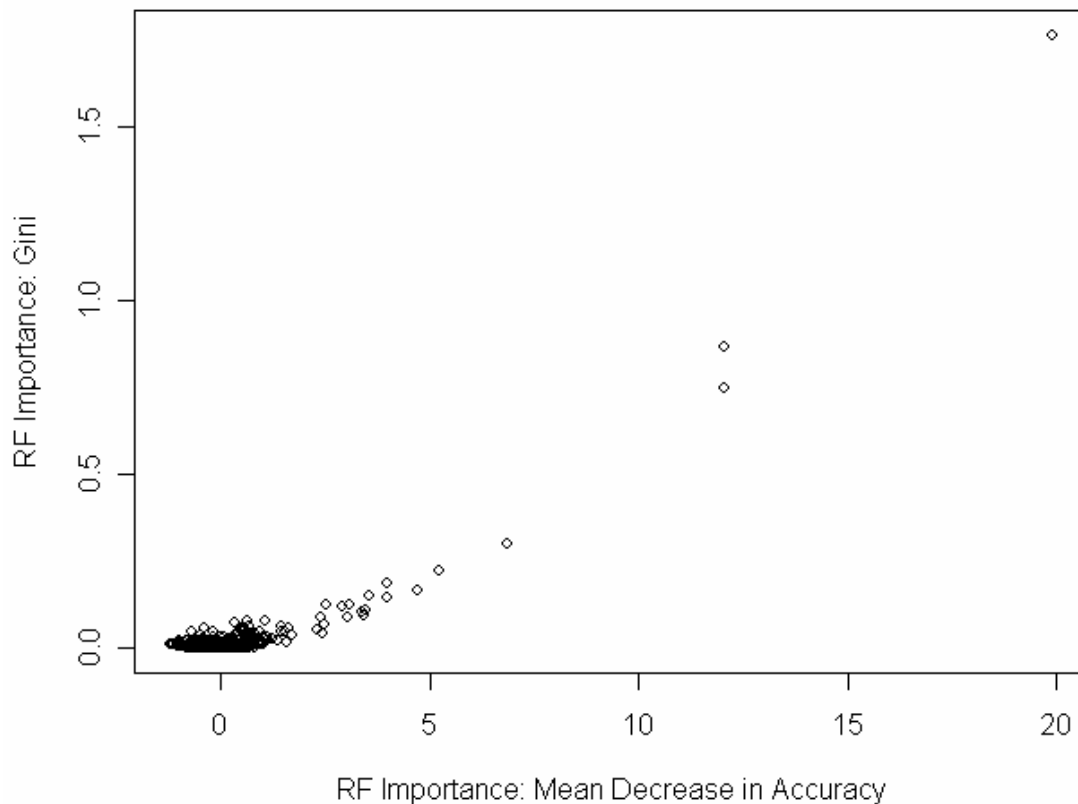


The above figure shows that when it comes to the top 2 probesets (in particular the #IGFBP2 gene) the two gene screening procedures agree.

The following plots compares the 2 different random forest importance measures. We find that both importance measures are highly correlated. But we prefer the Gini based importance measure in this analysis, since it is more closely related to the Kruskal Wallis test than the alternative measure.

```
par(mfrow=c(1,1))
x1= var.imp$MeanDecreaseAccuracy
y1= var.imp$MeanDecreaseGini
plot(x1,y1,xlab="RF Importance: Mean Decrease in Accuracy",ylab="RF Importance: Gini",main=paste("Comparing 2 RF importance measures, cor=",signif(cor(x1,y1),2) ) )
```

Comparing 2 RF importance measures, cor= 0.78



The above figure shows that when it comes to the top 2 probesets (in particular the #IGFBP2 gene) the two gene screening procedures agree.

Here we define the importance measure for all 9895 probesets

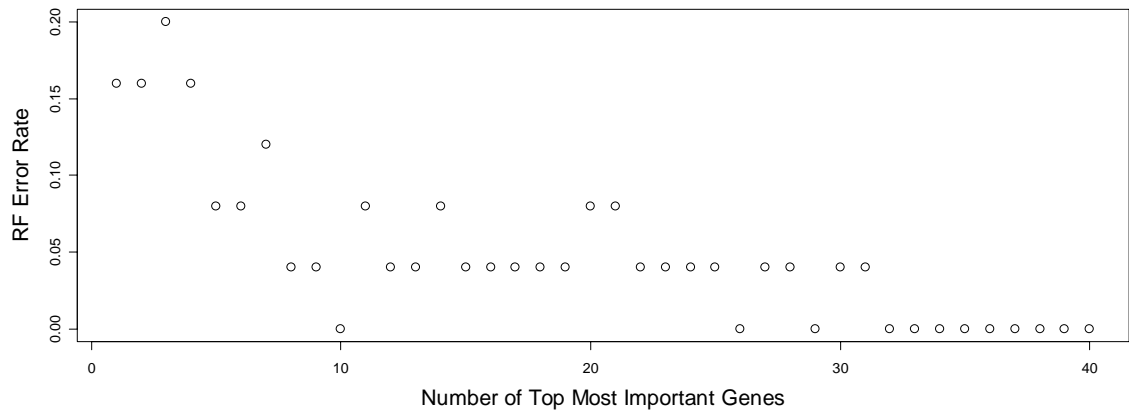
```
RFImportanceGini[restrictGenes1]=var.imp$MeanDecreaseGini
RFImportanceAccuracy[restrictGenes1]=var.imp$MeanDecreaseAccuracy
```

#The following plot depicts how the random forest error rate (y-axis) changes as a function of the number of most important probesets (x-axis).

```

set.seed(1)
NumberImportantGenes=c(1:40)
OOBError=rep(NA, length(NumberImportantGenes) )
for (i in c(1:length(NumberImportantGenes)) ) {
  topGenes= rank(-RFImportanceGini, ties.method="first" ) <= i
#Since the variables are highly correlated, we choose a small value for mtry (mtry=1).
#We choose 50000 trees.
  RF2 <- randomForest(y~., data=data.frame(datExpr[,topGenes]), ntree=50000,
  importance=F, mtry=1)
  OOBError[i]=1-sum(diag(table(RF2$predicted,y)))/length(y)
}
plot(NumberImportantGenes,OOBError ,cex=1.5,xlab="Number of Top Most
Important Genes", ylab="RF Error Rate",cex.lab=1.5 )

```



As mentioned above, we focus on the 10 most important probesets, which minimize the apparent #error rate. Below we show that these probesets are highly differentially expressed.

```
# Now we OUTPUT the importance measures etc.
datout=data.frame(GeneInformation[,1:2] , RFImportanceGini, RFImportanceAccuracy,
p.kruskal,p.ttest,ttest.statistic,MeanExprPtenWildtype,MeanExprPtenMutant)
write.table(datout, file="GeneSummary.csv", row.names=F,sep=",")
```

```
# Now we use the top 10 probesets to fit random forest
top10probes= rank(-var.imp$MeanDecreaseGini,ties.method="first") <= 10
set.seed(1)
mtry1=1
RF2 <- randomForest(y~., data=data.frame(datExpr2[,top10probes]),
ntree=30000, importance=T, mtry=mtry1, proximity=T)
```

RF2

```
Call:
randomForest(formula = y ~ ., data = data.frame(datExpr2[,
top10probes]),          ntree = 30000, importance = T, mtry = mtry1,
proximity = T)
```

```
      Type of random forest: classification
```

```
      Number of trees: 30000
```

```
No. of variables tried at each split: 1
```

```
      OOB estimate of  error rate: 0%
```

```
Confusion matrix:
```

	1	2	class.error
1	14	0	0
2	0	11	0

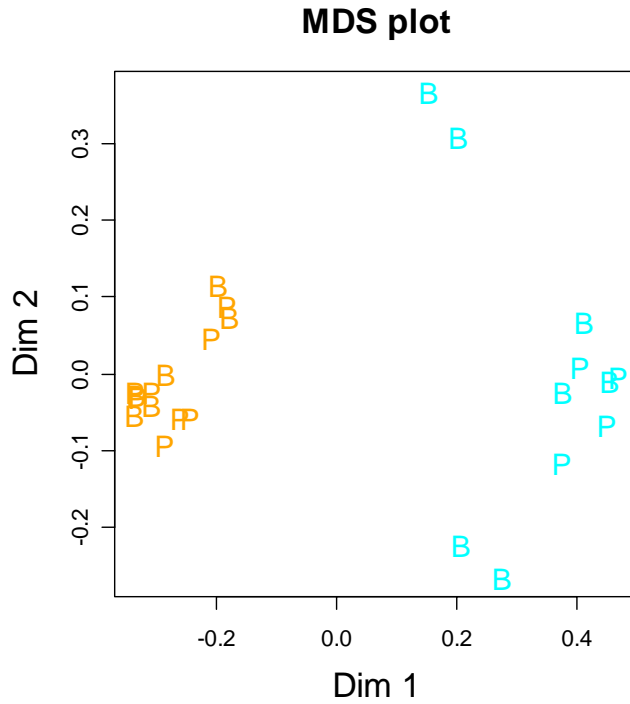
```
# Comment: As mentioned above, the apparent error rate estimate of 0 is highly biased.
```

#The following R function uses the random forest dissimilarity as input of multi-dimensional scaling.

Figure 1C

2-D MDS plot

```
MDSplot(RF2, y, pch=as.character(tumortypelabel), cex=1.3,palette=c("orange","cyan"),  
cex.lab=1.5,main="MDS plot",cex.main=1.5)
```

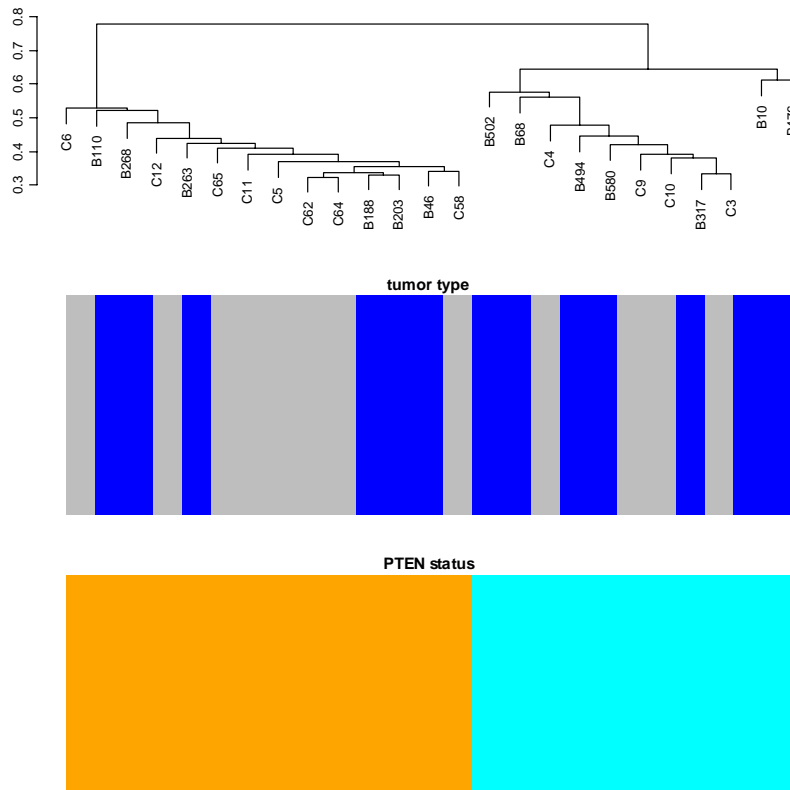


Hierarchical Clustering of the Samples (Figure 1D)

Using the 10 most important genes (according to the RF importance measure), we used supervised hierarchical clustering to cluster the genes. We used the random forest dissimilarity as input of average linkage hierarchical clustering. A review of clustering procedures can be found in Kaufman and Rousseuw (199) and Venables and Ripley (1999)

```
# The following R function hclustplot1 creates a barplot where the colors of the bars are sorted according to
# a hierarchical clustering tree (hclust object)
if (exists("hclustplot1")) rm(hclustplot1);
hclustplot1=function(hier1,couleur,title1="Colors sorted by hierarchical clustering") {
  if (length(hier1$Sorder) != length(couleur) ) {print("ERROR: length of color vector not compatible with no. of objects in the
  hierarchical tree");}
  if (length(hier1$Sorder) == length(couleur) ) {
    barplot(height=rep(1, length(couleur)), col= as.character(couleur[hier1$Sorder]),border=F, main=title1,space=0, axes=F)} }
```

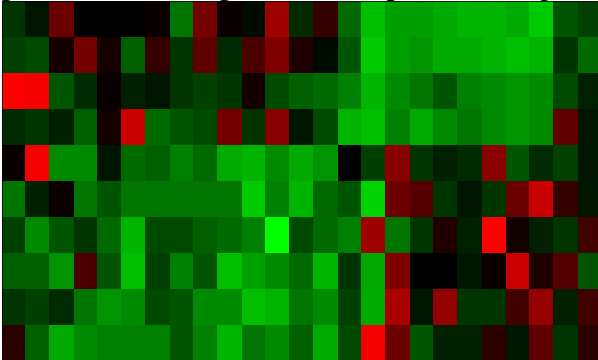
```
# Here we subset the data to the 10 most important genes
datExpr3=datExpr2[,top10probes]
GeneInformation3=GeneInformation2[top10probes,]
# This is the hierarchical clustering tree (dendrogram of the samples)
treeSamples=hclust( as.dist(1-RF2$proximity), method="average")
# Figure 1D.
par(mfrow=c(3,1),mar=c(2,2,2,2))
plot(treeSamples, ann=F)
hclustplot1(treeSamples, tumortypecolor, title1="tumor type")
hclustplot1(treeSamples, ycolor, title1="PTEN status")
```



```

# This computes the Student T-test statistic of differential expression
Tstatistic3=rep(NA, dim(datExpr3)[[2]] )
for (i in c(1:dim(datExpr3)[[2]] ) ) {
Tstatistic3[i]= t.test( datExpr3[,i]~y )$statistic
}
# The following is the heatmap plot that color codes gene expression.
# Rows correspond to genes sorted by the T-test statistic
# columns correspond to microarray samples sorted by the hierarchical tree above.
# Now we order the genes according to their Student t-test statistic (since this keeps track
of the sign of expression).
geneorder= order(-Tstatistic3)
par(mfrow=c(1,1), mar=c(2,2,2,2))
plot.mat( t(datExpr3[ treeSamples$order ,geneorder]))

```



#The 10 rows in this heatmap correspond to the following probesets

```

as.character(GeneInformation3$ProbeSet[geneorder])
[1] "40422_at" "1741_s_at" "36502_at" "36061_at" "39918_at"
"36575_at" "32717_at" "38555_at" "38463_s_at" "40026_g_at"

```

Here are the corresponding gene names

```

as.character(GeneInformation3$geneNames[geneorder])
[1] "insulin-like growth factor binding protein 2 (36kD)"
[2] " S37730 /FEATURE=cds /DEFINITION=S37712S4 insulin-like growth
factor binding protein-2 [human, placenta, Genomic, 1342 nt, segment 4
of 4] "
[3] "PFTAIRE protein kinase 1"
[4] "Cluster Incl. AF009314:Homo sapiens clone TUA8 Cri-du-chat region
mRNA /cds=UNKNOWN /gb=AF009314 /gi=2331117 /ug=Hs.49476 /len=1463"
[5] "gamma-tubulin complex protein 2"
[6] "regulator of G-protein signalling 1"
[7] "neuralized (Drosophila)-like"
[8] "dual specificity phosphatase 10"
[9] " Cluster Incl. U29926:Human AMP deaminase (AMPD3) gene, promoter
1a region /cds=(453,2777) /gb=U29926 /gi=1002661 /ug=Hs.83918
/len=4018 ]"
[10] "hypothetical protein"

```

We abbreviate the gene names in the above list as follows
 NamesTop10=c("IGFBP-2", "IGFBP-2", "PFTAIRE protein kinase 1",
 " TUA8 Cri-du-chat region", "Gamma-tubulin complex protein 2",
 "Regulator of G-protein signalling 1", "Neuralized (Drosophila)-like",
 "Dual specificity phosphatase 10", "AMPD3 promoter 1a" , "Hypothetical protein")

Here is some statistical information on the top 10 probesets

```
row.names(GeneInformation3)=""
signif(GeneInformation3[geneorder, -c(1,2)], 2)

RFImportanceGini RFImportanceAccuracy p.kruskal p.ttest ttest.statistic
MeanExprPtenWildtype MeanExprPtenMutant
1.70          20.0    5.1e-05 7.5e-07          6.9          -0.87          0.68
0.87          12.0    6.4e-05 9.4e-07          6.7          -0.86          0.68
0.13           3.1    5.5e-04 4.0e-03          3.3          -0.58          0.45
0.30           6.9    1.8e-03 1.1e-02          2.8          -0.54          0.42
0.15           4.6    7.3e-03 5.1e-02         -2.1           0.41         -0.33
0.15           2.8    2.6e-03 6.9e-03         -3.2           0.64         -0.50
0.22           4.8    2.2e-03 6.4e-03         -3.2           0.64         -0.50
0.21           4.4    1.5e-03 2.6e-03         -3.5           0.66         -0.52
0.16           4.0    1.1e-03 2.2e-03         -3.8           0.71         -0.56
0.75          12.0    2.0e-04 9.4e-04         -4.1           0.73         -0.58
```

Note that some of the mean expression values are negative. This stems from the fact that the gene expression data have been standardized within each tissue type (brain and prostate) to avoid a batch effect.

From this table, we find that the following genes are
over-expressed (up-regulated) in PTEN mutated samples

"IGFBP-2",
 "IGFBP-2",
 "PFTAIRE protein kinase 1",
 " TUA8 Cri-du-chat region",

The following genes are
under-expressed im PTEN mutated samples

"Gamma-tubulin complex protein 2",
 "Regulator of G-protein signalling 1",
 "Neuralized (Drosophila)-like",
 "Dual specificity phosphatase 10",
 "AMPD3 promoter 1a" ,
 "Hypothetical protein"

THE END