Using Gene Expression as a Means to Predict Response in Potato Data

By Mia Parenteau
Dalhousie University

May 22, 2018

Introduction

Phenotypic expressions of traits in organisms are determined by a combination of genetic predisposition and environmental factors. Hereditary predisposition enhances likelihood of certain genes to be expressed while environmental factors can trigger genes to replicate. The purpose of the current research is to examine potato data and determine if gene expression measured during the growth period can predict certain quantitative phenotypic features observed after harvesting. The features that will be used for the response are total yield, specific gravity and total nitrogen uptake (N uptake).

We want to know if using gene expression data will improve predictions in addition to the already established predictor of controlling the amount of fertilizer added. Ultimately the goal is to find genes that can predict yield, total N uptake and specific gravity before harvesting. This information could be utilized in many ways such as gauging the amount of fertilizer that should be added during the growth season. A high yield prediction early in the growth cycle could mean less need for the addition of fertilizer, this is beneficial because overuse of fertilizer leads to toxins in the local groundwater. Also adding fertilizer delays the formation of tubers until after harvesting occurs meaning that although tubers collected may be larger, they will be immature which can affect quality during food production. The dataset used for the analysis also contains variations in the experimental design across different sites and so the current analysis also attempts to find predictions that are able to generalize results across various sites and cultivars.

A total of 63 genes were considered during gene expression measurement of leaf samples. Therefore, another objective is to reduce the number of genes that can be used to predict the response. This is important from a practical perspective since gene expression sampling is laborious and expensive; minimizing the number of genes to sample will increase efficiency. It is also interesting to compare the different genes selected between response variables as it could provide insight into the nature of the underlying mechanisms of each response.

There are a number of ways to perform feature selection. The current analysis compares two different methods of variable selection and model assessment: Lasso regression and Random Forest regression. Lasso feature selection is a method that penalizes linear regression coefficients, truncating small coefficients to zero while random forest is a non-linear, non-parametric method that uses bootstrapped training samples to build decision trees. In the current analysis we will be using Lasso as a feature selection tool and then fitting the reduced predictors onto a linear regression model. Random forest will also be used to find a reduced model and then fit the reduced set with a random forest regression. Both models will be compared to the full set of predictors to assess if reducing predictors eliminated any noise.

Experimental Design

Leaves were sampled for gene expression at different time points. Depending on the site, leaves were sampled either once, twice or three times. All response variables were calculated at the end of the analysis and therefore different time points for leaf sampling have the same response values. For the purposes of this analysis, only gene expression data collected at the first time point for each site will be used. This corresponds to any gene sampling that took place between 42 and 50 days after planting.

Data was collected at various different sites in four different provinces across Canada. Five cultivars were used and various different levels of fertilizer were added depending on the site. The cultivars used were Russet Burbank, Jemseg, Shepody, Atlantic and Classic Russet. Different cultivars were used at different sites, as outlined in the table below.

	Atlantic	Classic Russet	Jemseg	Russet Burbank	Shepody
Charlottetown 2014	0	0	0	20	0
Fredericton GE 2012	20	0	0	19	19
Fredericton MAT 2014	0	0	12	12	12
Fredericton PK 2014	0	0	0	8	4
Off-Carberry 2014	0	0	0	12	0
On-Carberry 2014	0	0	0	12	0
Peribonka 2014	0	12	0	0	0

Table 1. Cultivars used by the different sites. Note that Classic Russet is confounded with Peribonka.

The Fredericton sites are marked three different ways due to different experimental conditions and years. Different types of fertilizer were added depending on site however for the purposes of the current analysis we will treat all fertilizer as the same. Fertilizer was also added at different time intervals depending on site but only the total amount added was used to simplify the analysis. Amount of fertilizer factors 106 and 120 were combined as well as the factors 180 and 200 since they are similar enough in value and did not show any significant differences in the response. The following tables outline the distribution of total fertilizer added with regards to site and cultivar.

	0	60	106/120	180/200	240
Charlottetown 2014	4	4	4	4	4
Fredericton GE 2012	11	12	12	12	11
Fredericton MAT 2014	12	0	12	12	0
Fredericton PK 2014	7	0	0	5	0
Off-Carberry 2014	4	4	0	4	0
On-Carberry 2014	4	4	0	4	0
Peribonka 2014	4	4	0	4	0

Table 2. Total fertilizer added by site.

	0	60	106/120	180/200	240
Atlantic	4	4	4	4	4
Classic Russet	4	4	0	4	0
Jemseg	4	0	4	4	0
Russet Burbank	25	16	12	23	7
Shepody	9	4	8	10	4

Table 3. Total fertilizer added for the different cultivars.

These variables (site, cultivar and total fertilizer added) were treated as factors so that the effects based on these variables can be controlled for in the regression model.

Exploratory Analysis of Predictive Factors

The following are exploratory plots of the factors against total yield. This should give some idea about how exactly these factors could have a predictive effect on the responses.

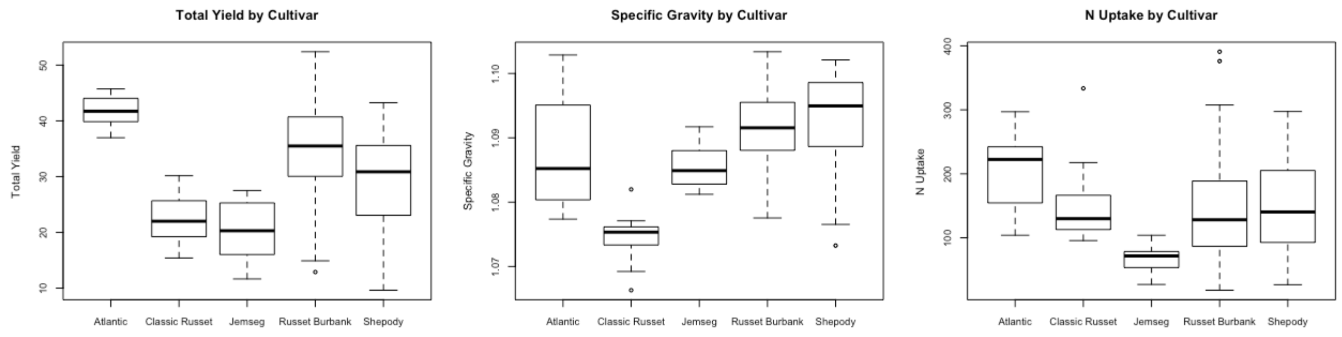

Figure 1. Boxplots showing the distribution of total yield, specific gravity, and N uptake for each cultivar.

It appears as though the response varies depending on cultivar but in different ways depending on the response variable. The total yield for the cultivar Atlantic has a reliably high yield compared to other cultivars, this pattern is not found for specific gravity and N uptake.

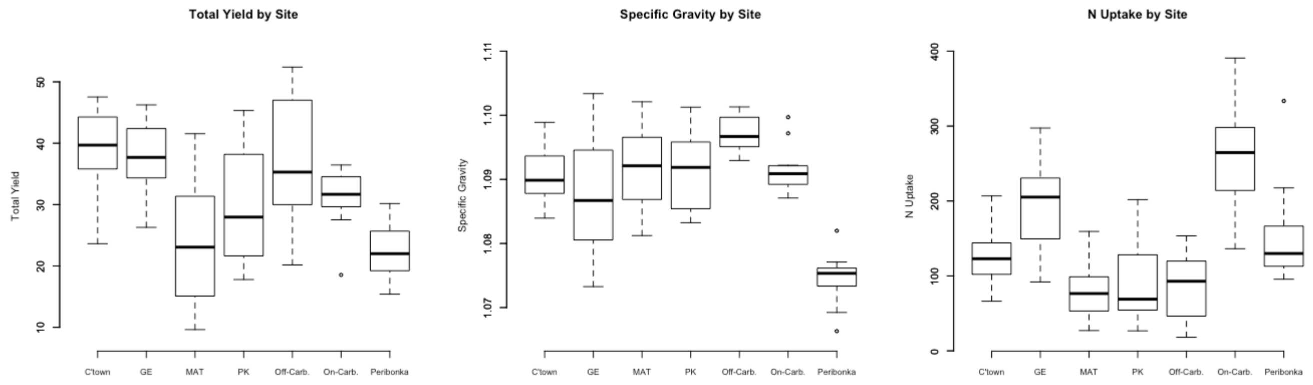

Figure 2. Response variables plotted against site.

There also seems to be some variation in the response depending on site. It is important to remember that there is a confounding response with the cultivar classic russet and the Peribonka site.

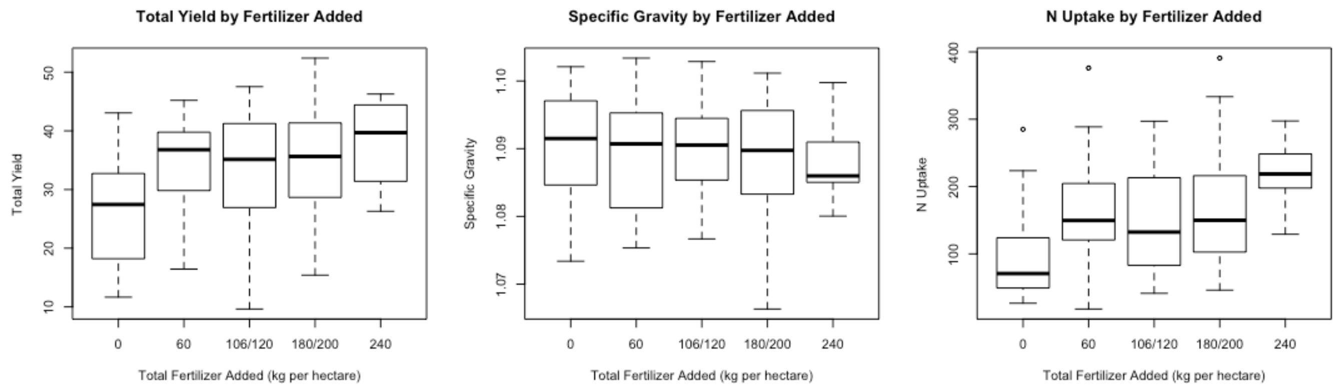

Figure 3. Fertilizer added by response.

Here we see a possible trend in the response as more fertilizer is added. There also appears to be less variability for the 240 kg/ha condition. The effect is minimal for the specific gravity response compared to the variations in N uptake and total yield. All three sets of plots show that there is some variation between the different site and cultivars that could be controlled for, as we discussed earlier.

Methods

Setup

The natural logarithm was shown to be an effective method in normalizing the gene data and so the transformed data was used in the analysis. Outliers were deleted as they were suspected to be data recording errors. R packages glmnet and randomForest were used for the analysis. Total Yield was measured as the weight of the tubers in kilograms per hectare. Specific gravity is the weight of the tubers in air divided by the difference between the weight of the tubers in air and the weight of the tubers in water. To measure total nitrogen uptake, vines, stolons and roots of the plant were washed and dried, then total concentration was determined by combustion.

Overview of Statistical Concepts

a) Lasso

Lasso, which stands for least absolute shrinkage and selection operator, is a regularization method for regression. Coefficients for the model are calculated using the following formula (ESL).

$$\hat{\beta}^{lasso} = argmin\left\{\frac{1}{2}\sum_{i=1}^{N}(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda\sum_{j=1}^{p}|\beta_j|\right\} = RSS + \lambda\sum_{j=1}^{p}|\beta_j|$$

The penalty term $\lambda\sum_{j=1}^{p}|\beta_j|$ truncates small coefficients to zero, as regulated by the tuning parameter λ; larger values for λ result in more conservative regularization. In the current analysis λ was chosen through 10-fold cross validation, as is convention.

b) Random Forest

Random Forest is a non-linear, non-parametric method that can be used for either regression or classification. It involves repeated sampling of training datasets (bootstrapping) as defined through the following algorithm (ISLR):

For bootstrap training samples 1 to B, grow a regression tree T_b through the following procedure:

i. Consider a random subset m of the p predictors.
ii. Find the strongest variable and split the response into two nodes.
iii. Repeat for the next most important variable on each of the resulting nodes.
iv. Stop until a certain specified node size is reached.

At the end of the algorithm, the output a set of B trees $\{T_b\}$. It should be noted that for regression, it is convention to consider p/3 predictor for the subset m.
The final model is an average of all trees from each bootstrap

$$\hat{f}_{rf}^{B}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$$

This can be useful if data is non-linear or non-parametric and often leads to a higher predictive accuracy but loses some interpretability in the final result. If relationships in the data are linear, random forest is not necessarily ideal.

c) Cross-validation

In an attempt to strengthen the results of the feature selection, 10-fold cross validation was used. Data was separated into 10 random, even chunks (i.e. 90% training and 10% testing). One chunk would be used as testing data and the remaining nine chunks were used as training data. This was repeated 10 times for each chunk, i.e. until all 10 separate chunks were used as testing data. This method was used to validate both the Lasso and Random Forest feature selection models. Both methods used the same training and testing sets in each fold. The analysis was repeated ten times for each test fold and final test error was averaged over each fold as defined by the following formula (ISLR).

$$CV_{10} = \frac{1}{10}\sum_{i=1}^{10} MSE_i$$

Model Assessment

Five different models were assessed, each for a linear model and a non-linear random forest model. The mean squared error was used to assess the fit because it can be used for both linear and non-linear models. Adjusted R-squared values were also calculated for the linear models as to have another indicator of the effectiveness of the model. The following are the models that were assessed:

a) Full model: includes all genes and factor variables.
b) Factors-only model: a model without the gene data (i.e. only site, cultivar and total amount of fertilizer). This was done as a way to compare the contribution that gene expression predictors made to other models.
c) Reduced within each fold model: a reduced set of variables were chosen for this model from the 100 bootstrap samples within each fold. For the random forest models the top 20 highest predictors were chosen.
d) 10 times chosen Lasso & 10 times chosen random forest: after all cross-validation was completed, the model that contained only variables that were selected in all

10 folds was calculated, giving a set of predictors from the Lasso and a set of predictors from random forest methods. This was done because it translates to a more realistic approach as ideally only a certain selection of genes would be used as a prediction tool, as was previously discussed.

Feature Selection Procedure

a) Lasso

Consider a set training and testing data. Using the glmnet package in R, ten-fold cross validation was used to find the best lambda based on a model using data from the training set. This chosen lambda was then used to tune the training model and select out the coefficients that were not equal to zero. This was repeated 100 times on bootstrapped samples of the training data and a count of how many times each coefficient was included was calculated. It should be noted that 1000 iterations would be ideal but was too computationally expensive with current resources.

A reduced model was selected from the coefficients that were included more than 50% of the time. In other words, the reduced model includes only the genes that were selected from the bootstrapped training samples more than 50 times. The reduced model was then fitted to a linear model and MSE between predicted test response and actual test response was calculated and compared to a linear model fitted with all predictors (reduced within each fold).

After all folds of the cross-validation were completed, predictors that were included in every fold were then selected to be in another reduced model (10 times chosen Lasso). This model was applied across all folds of the cross-validation and MSE was calculated. These results were then compared to MSE from a linear model including all predictors and a linear model that only includes site, cultivar and total fertilizer added.

b) Random Forest

The random forest model was fitted to the training data, using 1000 trees and 20 variables considered at each split. The top 20 most important variables were then selected for the reduced model. This was determined by selecting the largest values of percentage mean squared error decrease. The full and reduced model were used to predict values for the test data and the MSE was calculated and compared for both as well as a model that only contained site, cultivar and total fertilizer added. Another model was used that only included predictors found in all 10 folds, similarly to the Lasso mentioned above. The reduced predictor model considered a subset of m=6 predictors and the model that does not contain gene data only considers m=1, as is consistent with the p/3 subset convention that was mentioned above.

Results

Predictors Chosen through Feature Selection

The following is a table of the predictors that were selected in each bootstrap more than 50% of the time for all 10 folds during Lasso selection.

Total Yield	Specific Gravity	N Uptake
Cultivar	Cultivar	Cultivar
Site	Site	Site
Total N	Total N	Total N
St.ATrfA	St.ATrfA	St.AOX
St.CatTR	St.EPI	St.Apase
St.CGL	St.GluAse	St.ATrfA
St.CysPI1	St.Nod	St.CLH
St.FT	St.P109A	St.FT
St.LOB38B	St.PLD	St.LOB38B
St.MSRB	St.ProD	St.MtN21
St.Nod		St.NT
St.P109A		St.PEPT
St.PDX		
St.PLD		
St.PolyAP		
St.ProD		
St.RPK		

Table 4. Genes and factors selected for the 10 times chosen Lasso model.

Note that the factor variables (site, cultivar and total N) and the gene St. ATrfA were selected in each model. Below are the predictors that were among the 20 most important for all ten folds of the random forest.

Total Yield	Specific Gravity	N Uptake
Cultivar	Cultivar	Site
Site	Site	TotalN
Total N	St_NT2	St_NT2
St.TRDX	St.CatTR	St.Apase
St.Unk2	St.DUF506A	St.GR3
St.CWP	St.FT	St.ProH
St.DUF506A	St.GluAse	St.SulfT2C
St.Nod	St.MSF5A	St.Unk1
St.Unk4	St.SulfT	St.Xyl
St.Xyl	St.Xyl	

Table 5. Predictor variables used in the 10 times chosen random forest model for each response variable.

Model Assessment

The following tables summarize the MSE calculated for the ten models in each of the response variables as well as the adjusted R-squared value for the linear models.

Total Yield	Full Model	Factors only	Reduced within each fold	10 times chosen Lasso	10 times chosen Random Forest
Linear model	58.950	34.015	36.600	29.994	32.311
Random Forest	37.093	31.224	38.633	39.550	34.212

Table 6. MSE for total yield. The 10 times chosen Lasso linear model was the most successful with an MSE of 29.994

Specific Gravity	Full Model	Factors only	Reduced within each fold	10 times chosen Lasso	10 times chosen Random Forest
Linear Model	6.7328×10^{-5}	4.7524×10^{-5}	4.1137×10^{-5}	3.9527×10^{-5}	3.8259×10^{-5}
Random Forest	3.8352×10^{-5}	3.9707×10^{-5}	3.9492×10^{-5}	3.7682×10^{-5}	3.3520×10^{-5}

Table 7. MSE for specific gravity.

N Uptake	Full Model	Factors only	Reduced within each fold	10 times chosen Lasso	10 times chosen Random Forest
Linear Model	2596.246	1848.019	1570.721	1562.44	1543.521
Random Forest	1735.094	1768.837	1905.381	1750.924	1548.272

Table 8. MSE for N uptake. The predictors found from the 10 times chosen random forest model executed on a linear model yielded the smallest MSE.

The units of the MSE between the response variables are vastly different and therefore the MSE can only be compared between models within one response. Adjusted R-squared is a unit-less measurement of the variability explained by a given model. This was used to compare linear models between responses.

	Full Model	Factors only	Reduced within each fold	10 times chosen Lasso	10 times chosen Random Forest
Total Yield	0.721	0.669	0.748	0.739	0.686
Specific Gravity	0.526	0.408	0.554	0.499	0.444
N Uptake	0.815	0.760	0.786	0.772	0.775

Table 9. Adjusted R-squared for the linear models. Specific gravity shows consistently lower scores compared to the other responses.

Conclusion

Total Yield

For the linear model, including all predictors resulted in the largest mean squared error. The best fit for the linear model was to use only the genes that were selected in all folds during cross validation, which yielded an MSE of 29.994. For the random forest model, the reduced within each fold model actually performed slightly worse than the model containing all predictors. The random forest model that only contained the factors performed the best with an MSE of 31.22. Random forest variable selection did not tend to change the error rate between the full and reduced model very much in general compared to the Lasso models. Both models performed better using their own 10 times chosen variables. Adjusted R-squared values indicate that 66.9% of the variability has been explained by the factor variables. The genes however do add some explanation for additional variability in all the models where they were included. The reduced within each fold model had the highest percentage, with added genes explaining an extra 7.9% of the variability.

Specific Gravity

The small MSE values are due to the small differences in the response values, which is simply the nature of the measurement. The 10 times chosen random forest reduced predictor set used on a random forest model was the most successful with an MSE of 3.3520 x 10^{-5}. The full linear model had the poorest fit by far. Random forest modeling outperformed the linear model in all categories. Factor variables explained 40.8% of the variability according to the adjusted R-squared value, and the reduced within each fold model fared best with a 14.6% increased in the explained variability. In terms of factors total N was not included in the 10 times chosen random forest model, indicating that amount of fertilizer added was less important to the model compared to the other responses, which was consistent with the observations made on the exploratory plots.

N Uptake

The reduced model as determined by the 10 times random forest while using a linear model indicated the lowest MSE with a value of 1543.521. The full model had the highest R-squared of all models across all responses, with a value of 0.815. However, the MSE score for this was by far the largest when compared to other N uptake models. Factors were indicated to explain 76% of the variability and adding genes in all cases gave a slight improvement to the model. The random forest variable selection did not indicate cultivar as a predictor that needed to be included in the model.

General Conclusions

There is not a strong connection between predictors chosen in the two models. It is not clear from the current analysis which genes included during variable selection were most significant, especially for the Lasso model. In other words since we used the cutoff of 50% genes selected from the bootstrap, we did not assess which genes that were above the cutoff contributed the most times. The first 20 variables chosen for the random forest analysis was also an arbitrary cutoff. Both methods could benefit from further analysis on how many significant predictors are necessary.

Overall, results indicate that the most significant predictors were by far the factor predictors (i.e. site, cultivar and total N added) for both the MSE and adjusted R-squared calculations. Although gene expression predictors improved scores in most cases, it was only marginally so. Site was the only predictor that was selected amongst all responses in both 10 times chosen random forest and lasso models, indicating a strong site effect. This means for the current study, it is not realistic to use only genes as a predictive measure for the response without including/controlling for site, cultivar and total fertilizer added. Further studies with a more uniform experimental design where these factors are controlled for could be beneficial in providing further insight on the predictive abilities of the gene expression. Error rates between both models in each response are similar. This indicates that the data is following a linear model since the non-linear model did little to improve the fit.

Although the genes do not appear to significantly contribute to predictive accuracy, it is possible to analyze selected genes from the reduced models from a qualitative perspective. Genes that were selected from the 10 times chosen lasso and random forest models indicate that there is some relation between these gene expressions and the response. Even though the relation is not powerful enough to yield an accurate prediction from a statistical perspective, these results do offer some scientific insight as to which genes are responsible for regulating certain characteristics in the plant. For example, the St.FT is responsible for regulating when tubers will begin to form. Therefore the expression of gene is related to the size of the tuber and therefore it makes sense that it would be a significant gene for the total yield response. In conclusion, further research can be done to further tune reduced models with more validity.

References

Introduction to Statistical Learning: with Applications in R

G. James, D. Witten, T. Hastie and R. Tibshirani (2013).

The Elements of Statistical Learning: Data Mining, Inference, and Prediction

T. Hastie, R. Tibshirani and J. Friedman (2008).

RStudio: Integrated Development for R.

RStudio Team (2015).

Special thanks to Dr. H. Gu and Dr. H. Tai.

Appendix

R code

The following code is provided for the total yield response variable. The same code was applied to other response variables with slight modifications where necessary.

~~~
```{r setup}
rm(list=ls())
setwd("/Users/miaparenteau/Desktop/HONOURS")
library(plyr)
library(glmnet)
library(randomForest)
set.seed(679)

# import the raw dataset
raw.dat<-read.csv("Sorted and Compiled.csv",header = TRUE)

# eliminate unnecessary columns
dat<-raw.dat[,-c(1,4,5,6,7,9,10,12)]
dat<-rename(dat,c("Site.year"="Site","total.yield..t.ha."="Total
Yield",

"relative.yield..total.yield.max.plot.yield.for.trial."=
                    "Relative Yield","Total.N.rate.kg.ha"="Total
N Added",
                   "SPEC_GRAV"="Specific
Gravity","Total.N.uptake.kg.N.ha"=
                    "Total N Uptake"))
y<-dat[,5:8] # response variables
genes<-log(dat[,9:71]) # log transformed gene data

# We now transform the remaining predictors into factors
site<-as.factor(dat$Site)
cul<-as.factor(dat$cultivar)
DAP<-as.factor(dat$DAP.for.leaf.sampling)
TotalN<-as.factor(dat$`Total N Added`)
TotalN<-
revalue(TotalN,c("106"="106/120","120"="106/120","200"="180/200",
"180"="180/200"))
# combine 106 and 120 as well as 180 and 200

x<-data.frame(site,cul,TotalN,DAP) # all categorial predictors in
one data frame

dat<-data.frame(y,x,genes) # the dataset

# only choose data from the first time point
dap1<-dat[(dat$DAP==42|dat$DAP==47|dat$DAP==50|dat$DAP==48),]
~~~

~~~
dap1<-dap1[,-c(1,2,4,8)] # eliminate unwanted response variables
dap1<-dap1[-48,] # eliminating an outlier
dap1.y<-dap1[,1] # choosing Total Yield to be the response
dap1.x<-dap1[,-1]

```


```{r training and testing sets}
dff<-dap1
dff<-order(runif(nrow(dff)))
bins<-rep(1:10,length(dff)/10)
indices<-(split(dff, bins)) # splitting the data into 10 "equal"
parts.

# Set of empty lists
test<-list()
train<-list()
dat.train<-list() # training data for each fold
dat.test<-list() # testing data for each fold

#########
bootstraps=list() # empty list
lass=list()

# A loop to store all training and testing folds into a list.
for (i in 1:10){
  for (j in 1:10){
    test[[j]]<-as.vector(unlist(indices[j]))
    train[[j]]<-as.vector(unlist(indices[-j]))
  }
  dat.train[[i]]<-dap1[train[[i]],] # list with 10 elements
  dat.test[[i]]<-dap1[test[[i]],] # list with 10 elements
}
```


```{r lm full and fac}
# Full model and factors only assessment in each of the ten folds
full.lm.mod<-list()
full.lm.pred<-list()
full.mse<-list()
fac.lm.mod<-list()
fac.lm.pred<-list()
fac.mse<-list()
full.r.sq<-list()
fac.r.sq<-list()

for (i in 1:10){
  full.lm.mod[[i]]<-lm(Total.Yield~.,data = dat.train[[i]])
  full.lm.pred[[i]]<-predict(full.lm.mod[[i]],newdata =
dat.test[[i]])
~~~

```
  full.mse[[i]]<-(mean((full.lm.pred[[i]]-
dat.test[[i]]$Total.Yield)^2))
  full.r.sq[[i]]<-summary(full.lm.mod[[i]])$adj.r.squared

  fac.lm.mod[[i]]<-lm(Total.Yield~site+cul+TotalN, data =
dat.train[[i]])
  fac.lm.pred[[i]]<-predict(fac.lm.mod[[i]],newdata =
dat.test[[i]][,1:4])
  fac.mse[[i]]<-(mean((fac.lm.pred[[i]]-
dat.test[[i]]$Total.Yield)^2))
  fac.r.sq[[i]]<-summary(fac.lm.mod[[i]])$adj.r.squared
}

RSQ.lm.full<-mean(unlist(full.r.sq))
RSQ.lm.fac<-mean(unlist(fac.r.sq))
MSE.lm.full<-mean(unlist(full.mse))
MSE.lm.fac<-mean(unlist(fac.mse))

MSE.lm.full
MSE.lm.fac
RSQ.lm.full
RSQ.lm.fac
```

```{r bootstrapping & coef selection}
# function to resample the data with replacement
resamp<-function(dat){
  dat[sample(dim(dat)[1], dim(dat)[1],replace=TRUE),]
}

##### Bootstrap Iterations
t=100

bootstr<-vector("list",t)
for (i in 1:t){
  bootstr[[i]]<-lapply(dat.train, resamp) # indexed by the number
of bootstrap iterations
}

switched<-function(alist){
  alist<-apply(do.call(rbind, alist), 2, as.list)
  lapply(alist, function(X) X[!sapply(X, is.null)])
}

boot.folds<-switched(bootstr)

lasso.coef<-function(dat1){
  y<-dat1[,1]
  dat.x<-dat1[,2:67]
  dat1<-data.frame(y,dat.x)
  x<-model.matrix(y~.,data=dat1) # creating a design matrix
  lasso.mod<-glmnet(x,y,alpha = 1) # the model (lasso)
  cv.out<-cv.glmnet(x,y,alpha=1,nfolds=10) # 10-fold cv to find
lambda
  bestlam<-cv.out$lambda.min # finding the smallest lambda

lasso.coef=predict(lasso.mod,s=bestlam,newx=x,type='coefficients'
)[1:ncol(dat1),]
  return(names(lasso.coef[lasso.coef!=0]))
}

lassoed <- lapply(boot.folds, function (x) {
  lapply(x, lasso.coef)})

takeitoff<- lapply(lassoed, unlist)
incl<-lapply(takeitoff,count)
```

```{r within fold reduced for lm}

thresh<-0.5 # only pull genes that were selected a certain
proportion of the time.

incl.50<-list()
incl.50names<-list()
names.use<-list()
train.red<-list()
train.y.red<-list()
train.x.red<-list()
test.red<-list()
test.y.red<-list()
test.x.red<-list()
red.lm.mod<-list()
red.lm.pred<-list()
red.mse<-list()
red.r.sq<-list()

 for (i in 1:10){
  incl.50[[i]]<-incl[[i]][(incl[[i]]$freq)>t*thresh,]
  incl.50names[[i]]<-as.vector(incl.50[[i]]$x)
  names.use[[i]]<-names(dap1)[(names(dap1) %in%
incl.50names[[i]])]

  train.red[[i]] <-
data.frame(dat.train[[i]][,1:4],dat.train[[i]][,names.use[[i]]])
  train.y.red[[i]] <-train.red[[i]][,1]
  train.x.red[[i]]<-train.red[[i]][,-1]
  train.red[[i]] <-data.frame(train.y.red[[i]],train.x.red[[i]])

  test.red[[i]]<-
data.frame(dat.test[[i]][,1:4],dat.test[[i]][,names.use[[i]]])
  test.y.red[[i]]<-test.red[[i]][,1]
  test.x.red[[i]]<-test.red[[i]][,-1]
  test.red[[i]]<-data.frame(test.y.red[[i]],test.x.red[[i]])
  red.lm.mod[[i]]<-lm(train.red[[i]]$train.y.red..i.. ~ .,data =
train.red[[i]])
  red.lm.pred[[i]]<-predict(red.lm.mod[[i]],newdata =
test.red[[i]])
  red.mse[[i]]<-(mean((red.lm.pred[[i]]-test.y.red[[i]])^2))

  red.r.sq[[i]]<-summary(red.lm.mod[[i]])$adj.r.squared
}

RSQ.lm.red<-mean(unlist(red.r.sq))
MSE.lm.red<-mean(unlist(red.mse))
MSE.lm.red
RSQ.lm.red
```

```{r full reduced and factors only for rf}
# Set of empty lists
test<-list()
train<-list()
dat.train<-list() # training data for each fold
dat.test<-list() # testing data for each fold

bootstraps=list() # empty list
lass=list()

# A loop to store all training and testing folds into a list.
for (i in 1:10){
  for (j in 1:10){
    test[[j]]<-as.vector(unlist(indices[j]))
    train[[j]]<-as.vector(unlist(indices[-j]))
  }
  dat.train[[i]]<-dap1[train[[i]],] # list with 10 elements
  dat.test[[i]]<-dap1[test[[i]],] # list with 10 elements
}

###

random.potato<-list()
p.random<-list()
rf.mse.full<-list()
imp<-list()
q<-list()
red.train<-list()
red.test<-list()
red.potato<-list()
p.red<-list()
rf.mse.red<-list()
fac.rf.mod<-list()
fac.rf.pred<-list()
fac.rf.mse<-list()
imp.names<-list()
```

```
for(i in 1:10){
  # full set
 random.potato[[i]]<-randomForest(Total.Yield ~ .,
    data =
dat.train[[i]],ntree=1000,mtry=round(ncol(dat.train[[i]])/3),impo
rtance=TRUE)
  p.random[[i]]<-
predict(random.potato[[i]],newdata=dat.test[[i]])
  rf.mse.full[[i]]<-(mean((p.random[[i]]-
dat.test[[i]]$Total.Yield)^2))

  # reduced set
  imp[[i]]<-importance(random.potato[[i]])
  q[[i]]<-(imp[[i]][order(imp[[i]][,1], decreasing =
TRUE),][1:20,]) # top 20 predictors

  imp.names[[i]]<-rownames(q[[i]])

  red.train[[i]]<-
data.frame(dat.train[[i]]$Total.Yield,dat.train[[i]][,(colnames(d
at.train[[i]])=row.names(q[[i]]))])
  red.test[[i]]<-
data.frame(dat.test[[i]]$Total.Yield,dat.test[[i]][,(colnames(dat
.test[[i]])=row.names(q[[i]]))])
  red.potato[[i]]<-
randomForest(dat.train..i...Total.Yield~.,data=red.train[[i]],ntr
ee=1000,mtry=round(ncol(red.train[[i]])/3),importance=TRUE)
  p.red[[i]]<-predict(red.potato[[i]],newdata = red.test[[i]])
  rf.mse.red[[i]]<-(mean((p.red[[i]]-
red.test[[i]]$dat.test..i...Total.Yield)^2))
}

MSE.rf.full<-mean(unlist(rf.mse.full))
MSE.rf.red<-mean(unlist(rf.mse.red))
MSE.rf.full
MSE.rf.red

for (i in 1:10){
  for (j in 1:10){
    test[[j]]<-as.vector(unlist(indices[j]))
    train[[j]]<-as.vector(unlist(indices[-j]))
  }
  dat.train[[i]]<-dap1[train[[i]],] # list with 10 elements
  dat.test[[i]]<-dap1[test[[i]],] # list with 10 elements
}

for (i in 1:10){
  # factors only
  fac.rf.mod[[i]]<-randomForest(Total.Yield~., data =
dat.train[[i]][,1:4],ntree=1000,mtry=1,importance=TRUE)
```

```
  fac.rf.pred[[i]]<-predict(fac.rf.mod[[i]],newdata =
dat.test[[i]][,1:4])
  fac.rf.mse[[i]]<-(mean((fac.rf.pred[[i]]-
dat.test[[i]]$Total.Yield)^2))
}

MSE.rf.fac<-mean(unlist(fac.rf.mse))
MSE.rf.fac

plot(dap1[unlist(test),]$Total.Yield,unlist(p.red)) # reduced
random forest model

```

```{r lasso 10 for lm and rf}
more.than.50<-count(unlist(incl.50names))

more.than.50<-more.than.50[more.than.50$x%in%colnames(genes),]

jj<-more.than.50[more.than.50$freq==10,]
names.use.lasso10<-as.vector(jj$x)

train.lasso10<-list()
train.y.lasso10<-list()
train.x.lasso10<-list()
test.lasso10<-list()
test.y.lasso10<-list()
test.x.lasso10<-list()
lasso10.lm.mod<-list()
lasso10.lm.pred<-list()
lasso10.mse<-list()
lasso10.rf.mod<-list()
p.red<-list()
lasso10.mse.rf<-list()
lasso10.rf.pred<-list()
lasso10.r.sq<-list()

for (i in 1:10){
  train.lasso10[[i]] <-
data.frame(dat.train[[i]][,1:4],dat.train[[i]][,names.use.lasso10
])
  train.y.lasso10[[i]] <-train.lasso10[[i]][,1]
  train.x.lasso10[[i]]<-train.lasso10[[i]][,-1]
  train.lasso10[[i]] <-
data.frame(train.y.lasso10[[i]],train.x.lasso10[[i]])

  test.lasso10[[i]]<-
data.frame(dat.test[[i]][,1:4],dat.test[[i]][,names.use.lasso10])
  test.y.lasso10[[i]]<-test.lasso10[[i]][,1]
  test.x.lasso10[[i]]<-test.lasso10[[i]][,-1]
  test.lasso10[[i]]<-
data.frame(test.y.lasso10[[i]],test.x.lasso10[[i]])
```

```
  lasso10.lm.mod[[i]]<-lm(train.lasso10[[i]]$train.y.lasso10..i..
~ .,data = train.lasso10[[i]])
  lasso10.lm.pred[[i]]<-predict(lasso10.lm.mod[[i]],newdata =
test.lasso10[[i]])
  lasso10.mse[[i]]<-(mean((lasso10.lm.pred[[i]]-
test.y.lasso10[[i]])^2))
  lasso10.r.sq[[i]]<-summary(lasso10.lm.mod[[i]])$adj.r.squared
}

RSQ.lm.lasso10<-mean(unlist(lasso10.r.sq))
MSE.lm.lasso10<-mean(unlist(lasso10.mse))
MSE.lm.lasso10
RSQ.lm.lasso10

for (i in 1:10){
  for (j in 1:10){
    test[[j]]<-as.vector(unlist(indices[j]))
    train[[j]]<-as.vector(unlist(indices[-j]))
  }
  dat.train[[i]]<-dap1[train[[i]],] # list with 10 elements
  dat.test[[i]]<-dap1[test[[i]],] # list with 10 elements
}
###
for (i in 1:10){
  lasso10.rf.mod[[i]]<-
randomForest(train.lasso10[[i]]$train.y.lasso10..i.. ~ .
                     ,data=train.lasso10[[i]],ntree=1000,

mtry=round(length(names.use.lasso10)/3),importance=TRUE)
  lasso10.rf.pred[[i]]<-predict(lasso10.rf.mod[[i]],newdata =
test.lasso10[[i]])
  lasso10.mse.rf[[i]]<-(mean((lasso10.rf.pred[[i]]-
test.y.lasso10[[i]])^2))
}

MSE.rf.lasso10<-mean(unlist(lasso10.mse.rf))
MSE.rf.lasso10
```

```{r rf 10 for lm and rf}
cv.count<-count(unlist(imp.names))
cv.count<-cv.count[cv.count$freq==10,]$x
names.use.rf10<-as.vector(cv.count)

train.rf10<-list()
train.y.rf10<-list()
train.x.rf10<-list()
test.rf10<-list()
test.y.rf10<-list()
test.x.rf10<-list()
rf10.lm.mod<-list()
rf10.lm.pred<-list()
rf10.lm.mse<-list()
rf10.rf.mod<-list()
p.red<-list()
rf10.mse.rf<-list()
rf10.rf.pred<-list()
rf10.r.sq<-list()

for (i in 1:10){
  train.rf10[[i]] <-
data.frame(dat.train[[i]][,1],dat.train[[i]][,names.use.rf10])
  train.y.rf10[[i]] <-train.rf10[[i]][,1]
  train.x.rf10[[i]]<-train.rf10[[i]][,-1]
  train.rf10[[i]] <-
data.frame(train.y.rf10[[i]],train.x.rf10[[i]])

  test.rf10[[i]]<-
data.frame(dat.test[[i]][,1],dat.test[[i]][,names.use.rf10])
  test.y.rf10[[i]]<-test.rf10[[i]][,1]
  test.x.rf10[[i]]<-test.rf10[[i]][,-1]
  test.rf10[[i]]<-data.frame(test.y.rf10[[i]],test.x.rf10[[i]])

  rf10.lm.mod[[i]]<-lm(train.rf10[[i]]$train.y.rf10..i.. ~ .,data
= train.rf10[[i]])
  rf10.lm.pred[[i]]<-predict(rf10.lm.mod[[i]],newdata =
test.rf10[[i]])
  rf10.lm.mse[[i]]<-(mean((rf10.lm.pred[[i]]-
test.y.rf10[[i]])^2))
  rf10.r.sq[[i]]<-summary(rf10.lm.mod[[i]])$adj.r.squared
}

RSQ.lm.rf10<-mean(unlist(rf10.r.sq))
MSE.lm.rf10<-mean(unlist(rf10.lm.mse))
MSE.lm.rf10
RSQ.lm.rf10

for (i in 1:10){
  for (j in 1:10){
    test[[j]]<-as.vector(unlist(indices[j]))
    train[[j]]<-as.vector(unlist(indices[-j]))
  }
  dat.train[[i]]<-dap1[train[[i]],] # list with 10 elements
  dat.test[[i]]<-dap1[test[[i]],] # list with 10 elements
}
###
for (i in 1:10){
  rf10.rf.mod[[i]]<-
randomForest(train.rf10[[i]]$train.y.rf10..i.. ~ .
                     ,data=train.rf10[[i]],ntree=1000,
mtry=round(length(names.use.rf10)/3),importance=TRUE)
  rf10.rf.pred[[i]]<-predict(rf10.rf.mod[[i]],newdata =
test.rf10[[i]])
  rf10.mse.rf[[i]]<-(mean((rf10.rf.pred[[i]]-
test.y.rf10[[i]])^2))
}

MSE.rf.rf10<-mean(unlist(rf10.mse.rf))
MSE.rf.rf10
```

```{r}
final.results<-
matrix(c(MSE.lm.full,MSE.lm.red,MSE.lm.fac,MSE.lm.lasso10,MSE.lm.
rf10,MSE.rf.full,MSE.rf.red,MSE.rf.fac,MSE.rf.lasso10,MSE.rf.rf10
),ncol = 5,nrow = 2,byrow = TRUE)
rownames(final.results)<-c("Linear Model","Random Forest")
colnames(final.results)<-c("full","reduced","factors","lasso
10","rf 10")
final.results

adj.r.squared<-
matrix(c(RSQ.lm.full,RSQ.lm.red,RSQ.lm.fac,RSQ.lm.lasso10,RSQ.lm.
rf10),ncol = 5,nrow = 1,byrow = TRUE)
colnames(adj.r.squared)<-c("full","reduced","factors","lasso
10","rf 10")
adj.r.squared

plot(dap1[unlist(test),]$Total.Yield,unlist(full.lm.pred),
     xlab = "Actual",ylab="Predicted",main="Total Yield Full
Model",xlim = c(-4,60),ylim = c(-4,60)) # full linear model
abline(lm(unlist(full.lm.pred)~dap1[unlist(test),]$Total.Yield),c
ol="red")
legend("bottomright", legend=c(round(RSQ.lm.full,2)))

plot(unlist(test.y.lasso10),unlist(lasso10.lm.pred),
     xlab = "Actual",ylab="Predicted",main="Total Yield Reduced
Model",xlim = c(-4,60),ylim = c(-4,60)) # lasso 10 linear
abline(lm(unlist(lasso10.lm.pred)~unlist(test.y.lasso10)),col="re
d")
legend("bottomright", legend=c(round(RSQ.lm.lasso10,2)))

# Genes that were used for the 10 times reduction
names.use.lasso10 # genes that were used more that 50% of the
time for all 10 folds
names.use.rf10 # genes that were the top 20 most important
predictors for all 10 folds
```