Testing ChemModLab Code

Jeremy Ash

October 5, 2016

In this documnet I am trying to identify why there are differences between Random Forest and SVM predictions found by the ChemModLab code I run on my machines (Linux and Windows OS) and the output I was originally provided from a run in 2009. I was never able to completely reproduce the output I was provided, even when I run the original code without any modifications an my machines. However, I was able to generate output whose differences in prediction and predicted probabilities are nearly identical in distribution to the differences I see between the original output and runs on my machines. I was able to reproduce these similar differences by simply changing the sequence of random numbers being used by Random Forest and SVM. I conclude that the differences in prediction and predicted probabilties have been produced in a difference in random number generation, and are not major cause for concern. I also demonstrate that the random number seeds are being set properly in my current code and that there are no differences in prediction and predicted probabilities between my current code and the original code I was provided.

Data: Burden Number Descriptors and Active/Inactive Response

```
##        CID Outcome WBN_GC_L_0.25 WBN_GC_H_0.25 WBN_GC_L_0.50 WBN_GC_H_0.50
## 1  5388992       1      -2.40010       1.98339      -2.52864       2.50835
## 2  5388983       1      -2.40010       1.98240      -2.52868       2.50398
## 3   663143       1      -2.41650       1.32890      -2.53910       2.05778
## 4    10607       1      -2.38337       2.17677      -2.52643       2.33232
## 5  5388972       1      -2.29039       1.97468      -2.41743       2.46177
## 6 11970251       1      -2.29039       2.22488      -2.41748       2.56161
```

Sevaral different runs of ChemModLab were compared:

- The original output I was provided
- The original code with the 2008 Random Forest and SVM packages
  - Attempting to recreate the state of the packages when the original output was generated
- The original code with original packages on my linux machine
  - When I was unable to recreate the output with the original packages, I checked to see if the reason was that I was running the code on a windows machine
- The current code with the current packages
- The current code with the current packages with variable importance turned on
  - To demonstrate that a change in the sequence of the random numbers recreates the problem
- A second iteration of the current code with the current packages
  - When it became clear that the problem was being cause by random number generation, I made sure the results are the same when the seed is the same

Old Code with 2008 packages

Installing the most current packages at the time the original ChemModLab output was generated.

```
#install 2008 versions of sum and random Forest
install.packages("C:/Users/Vestige/Downloads/randomForest_4.5-28.tar.gz", repos = NULL, type="source")
install.packages("C:/Users/Vestige/Downloads/e1071_1.5-18.tar.gz", repos = NULL, type="source")

source("background_test.R")

bb<-back.bench(yfilein="AID_364.csv",
               xfilein="BurdenNumbers.csv",
               filepred="Split1/BurdenNumbers/pred_old_oldp.csv",
               fileimpdesc="Split1/BurdenNumbers/varimp_old_oldp.csv",
               fileprob="Split1/BurdenNumbers/prob_old_oldp.csv",
               filesummary="Split1/BurdenNumbers/summary_old_oldp.txt",
               nfolds=10,idcol=1,infofile="info.txt",
               logfile="log_old_oldp.txt",seed.in=11111)
```

Printing the version of the loaded packages to show that the 2008 packages were loaded.

```
library(e1071)
packageVersion("e1071")
```

```
## [1] '1.6.7'
```

```
library(randomForest)
```

```
## randomForest 4.6-12
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
packageVersion("randomForest")
```

```
## [1] '4.6.12'
```

New Code with New Packages

I have added the "rpart"" package to the current code. This packages was made by the authors of mvpart. They have moved the rpart function to this package. As you will see, I have implemented the model in the exact same way it was implemented in the original code, thought the syntax is different. The prediction and predicted probabilties are unchanged.

```
  #-----Recursive partitioning using "rpart" with splitting criterion "information" and
  # minbucket=5, minimum leaf size
  # minsplit=10, minimum parent size
  # maxcompete=0, don't get information on competitive splits
  # maxsurrogate=0, don't get information on surrogate splits
  # Possible modifications that have NOT been pursued here:
  # many ...

# new syntax
```

```
rpart(as.factor(y)~.,data=work.data,subset=(fold.id!=id), method="class",
      parms=list(split="information"), control =
        rpart.control(minsplit=10, minbucket=5, maxcompete=0, maxsurrogate=0))

# old syntax

rpart( as.factor(y)~.,data=work.data,subset=(fold.id!=id), method="class",
       parms=list(split="information"), minsplit=10, minbucket=5, maxcompete=0, maxsurrogate=0 )
```

```
# install.packages("randomForest")
# install.packages("e1017")

source("C:/Users/Vestige/Dropbox/ChemModLab/background.R")

bb <- ModelTrain(data, idcol=1,
                 models = c("NNet","PCR","ENet","PLS","Ridge",
                            "LARs","PLSLDA","RPart","Tree","SVM","KNN","Forest"),
                 nfolds=10, seed.in=c(11111))


write.csv(bb$all.preds[[1]][[1]],
          "Split1/BurdenNumbers/pred_new_newp.csv")
write.csv(bb$all.probs[[1]][[1]],
          "Split1/BurdenNumbers/prob_new_newp.csv")
```

Runing 2nd iteration to check set.seed

```
source("C:/Users/Vestige/Dropbox/ChemModLab/background.R")

bb <- ModelTrain(data, idcol=1,
                 models = c("NNet","PCR","ENet","PLS","Ridge",
                            "LARs","PLSLDA","RPart","Tree","SVM","KNN","Forest"),
                 nfolds=10, seed.in=c(11111))

write.csv(bb$all.preds[[1]][[1]],
          "Split1/BurdenNumbers/pred_new_newp_rep.csv")
write.csv(bb$all.probs[[1]][[1]],
          "Split1/BurdenNumbers/prob_new_newp_rep.csv")
```

New Code, New Packages with variable importance turned on for RF and SVM

When variable importance measure is turned on, random numbers are generated and used for the permutation of the predicted probabilties. The prediction accuracy for the permuted data is used as a baseline to which the prediction of accuracy of the model is compared. For random forests, this results in a different sequence of random numbers used for sampling variables at each split. I still need to look into how this affects SVM.

```
source("C:/Users/Vestige/Dropbox/ChemModLab/background_varimp.R")

bb <- ModelTrain(data, idcol=1,
                 models = c("NNet","PCR","ENet","PLS","Ridge","LARs",
```

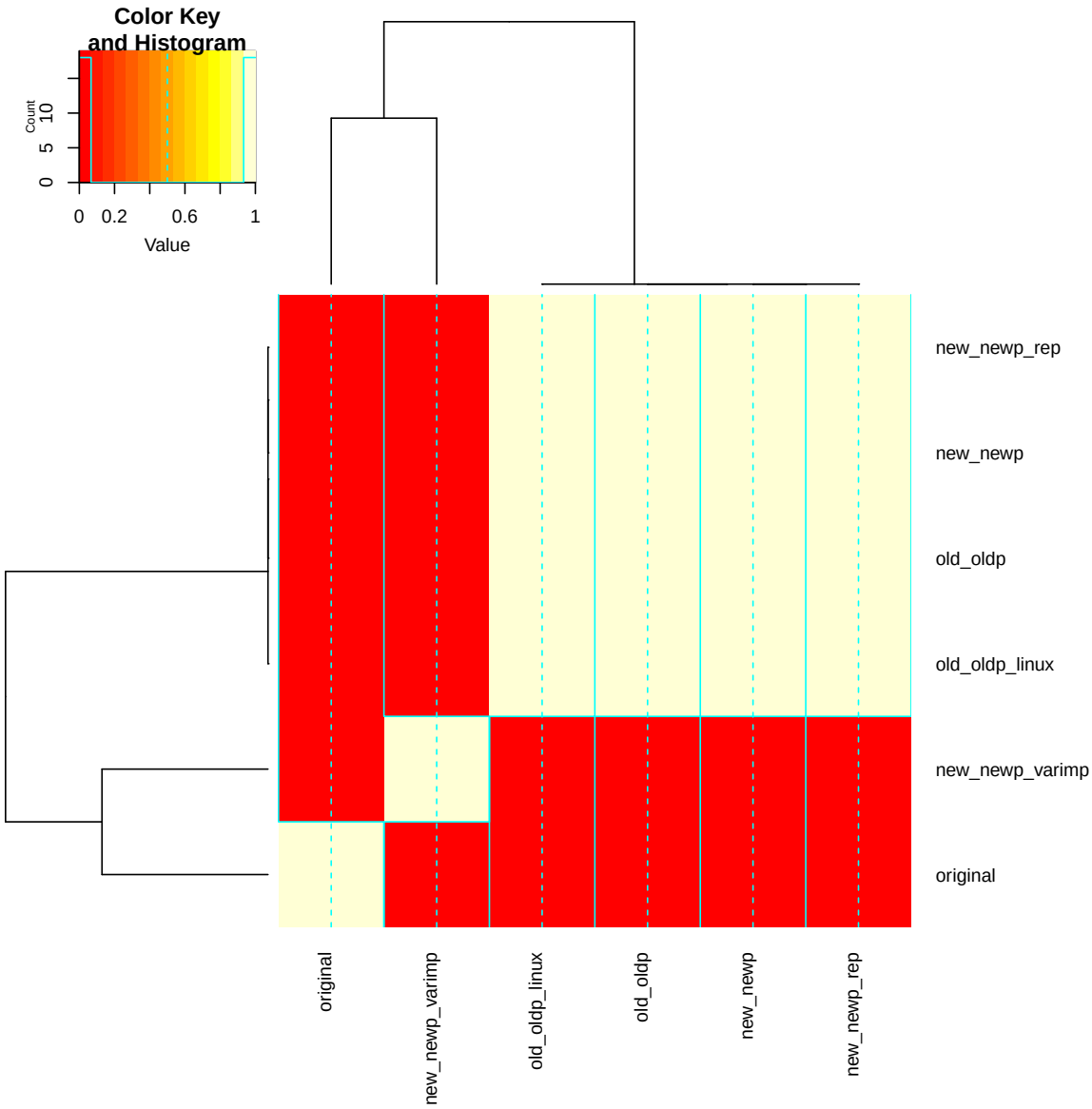
```
                            "PLSLDA","RPart","Tree","SVM","KNN","Forest"),
                nfolds=10, seed.in=c(11111))

write.csv(bb$all.preds[[1]][[1]],
          "Split1/BurdenNumbers/pred_new_newp_varimp.csv")
write.csv(bb$all.probs[[1]][[1]],
          "Split1/BurdenNumbers/prob_new_newp_varimp.csv")
```

Comparing new predictions to original output

```
pred_new_newp <- read.csv("Split1/BurdenNumbers/pred_new_newp.csv", row.names = 1)
pred_old_oldp <- read.csv("Split1/BurdenNumbers/pred_old_oldp.csv", row.names = 1, skip = 1)
pred_old_oldp_linux <- read.csv("Split1/BurdenNumbers/pred_old_oldp_linux.csv", row.names = 1, skip = 1
pred_new_newp_varimp <- read.csv("Split1/BurdenNumbers/pred_new_newp_varimp.csv", row.names = 1)
pred_new_newp_rep <- read.csv("Split1/BurdenNumbers/pred_new_newp_rep.csv", row.names = 1)
pred_orig <- read.csv("Split1/BurdenNumbers/pred_orig.csv", row.names = 1)
```

Heatmap showing the comparison of the predictions for each run. Red means there is a difference between runs, white means no difference. (all.equal used for comparison of each column of each matrix)

There are no differences in predictions when the new code is run with the most recent packages and the old code is run with the 2008 packages.



The number of different predictions between original and new code is close to the number of different predictions between new code and new code when only the variable importance is turned on.

```
which(pred_new_newp_varimp$RF != pred_orig$RF)
```

```
## [1] 18 20 26 28
```

```
which(pred_new_newp$RF != pred_new_newp_varimp$RF)
```

```
## [1]  9 20 26
```

```r
which(pred_new_newp$RF != pred_orig$RF)
```

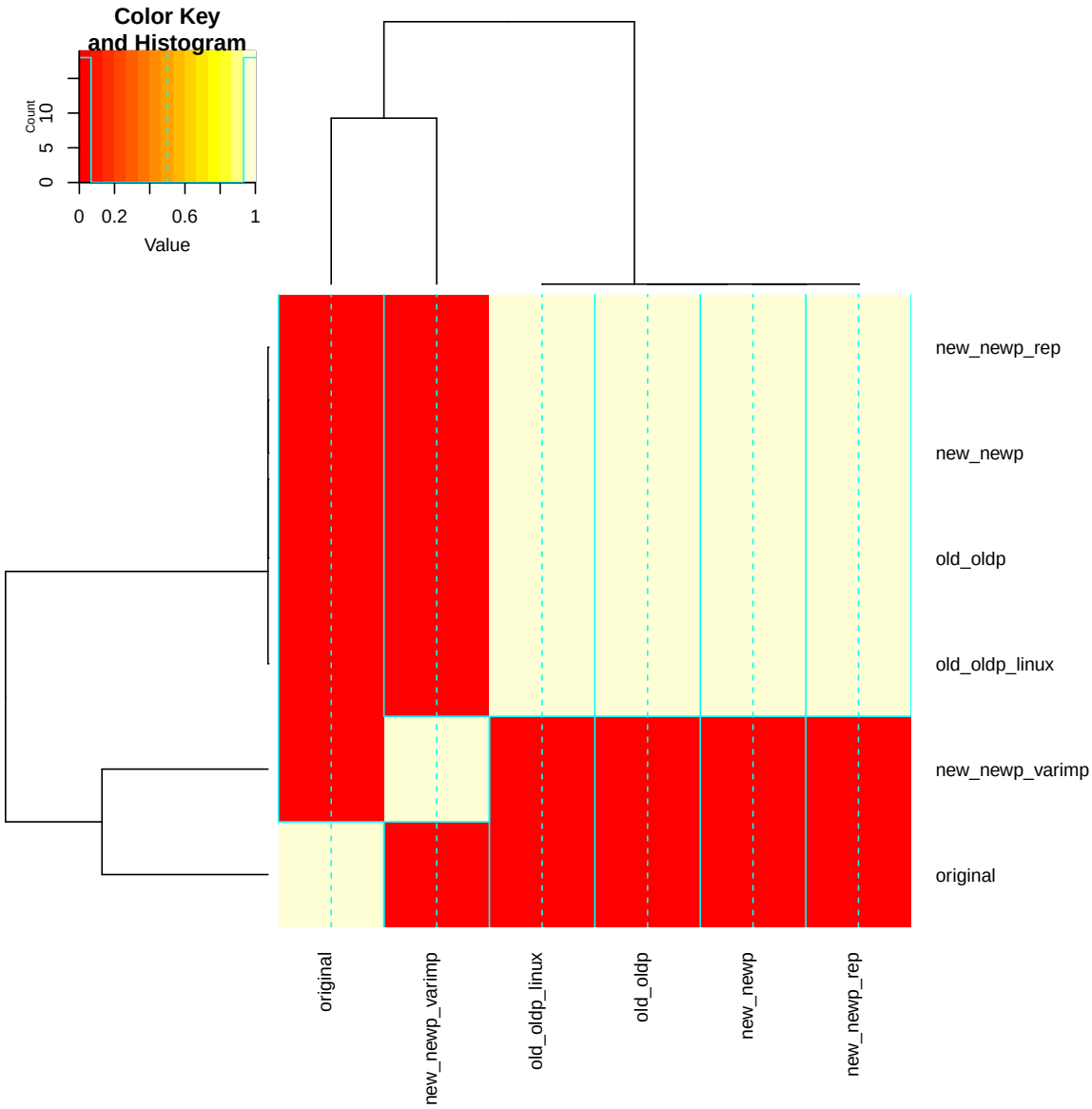
```
## [1]  9 18 28
```

Comparing new predicted probabilities to original output

```r
prob_new_newp <- read.csv("Split1/BurdenNumbers/prob_new_newp.csv", row.names = 1)
prob_old_oldp <- read.csv("Split1/BurdenNumbers/prob_old_oldp.csv", row.names = 1, skip = 1)
prob_old_oldp_linux <- read.csv("Split1/BurdenNumbers/prob_old_oldp_linux.csv", row.names = 1, skip = 1)
prob_new_newp_varimp <- read.csv("Split1/BurdenNumbers/prob_new_newp_varimp.csv", row.names = 1)
prob_new_newp_rep <- read.csv("Split1/BurdenNumbers/prob_new_newp_rep.csv", row.names = 1)


prob_orig <- read.csv("Split1/BurdenNumbers/prob_orig.csv", row.names = 1)


# there are some miniscule differences in predicted probabilities
# for NNet when I use my linux machine

all.equal(prob_old_oldp,prob_old_oldp_linux, scale = 1)
```

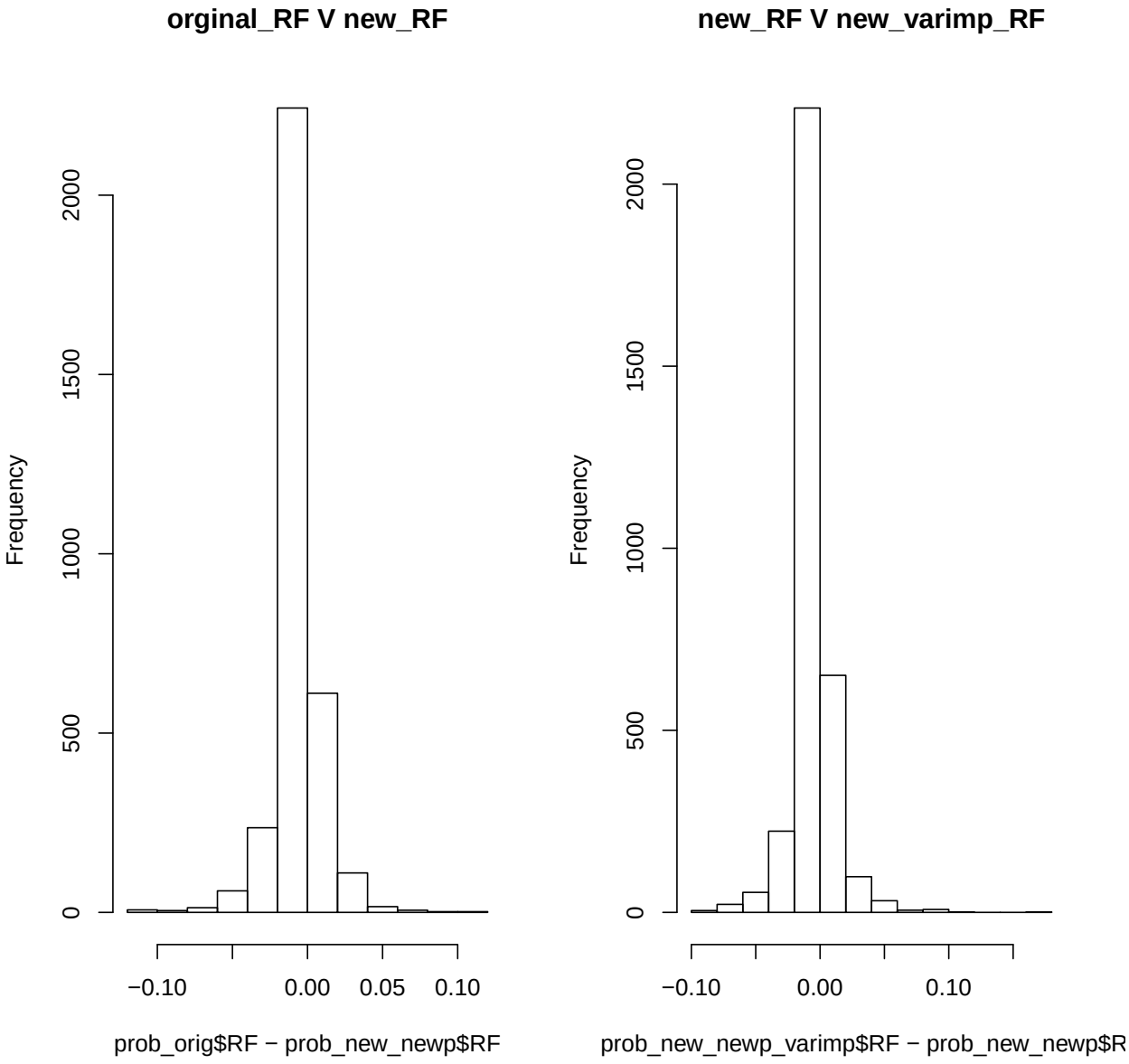
```
## [1] "Component \"NNet\": Mean absolute difference: 1.471322e-06"
```

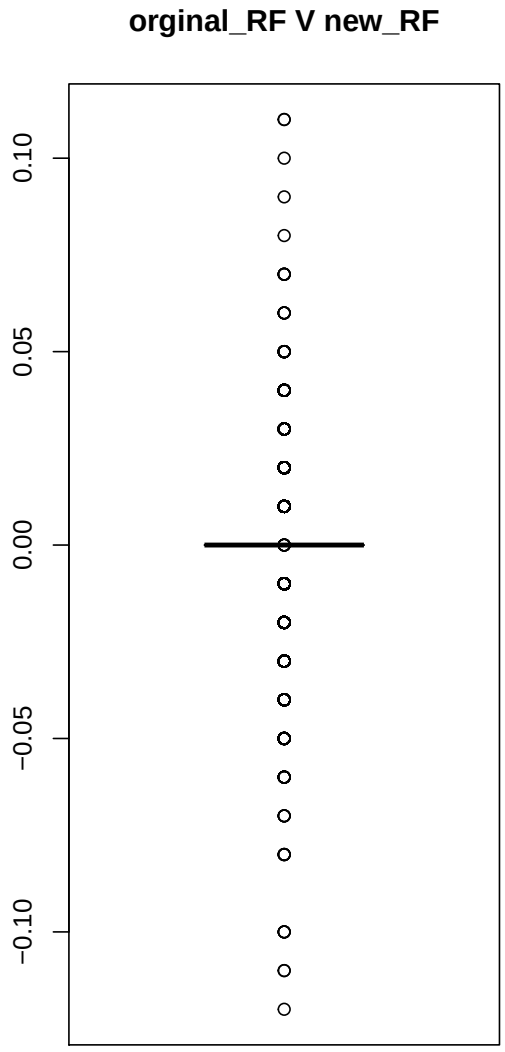
Heatmap showing the comparison of all the predicted probabilties for each run. Red means there is a difference between runs, white means no difference. Miniscule differences in predicted probabilities for NNet were disregarded.

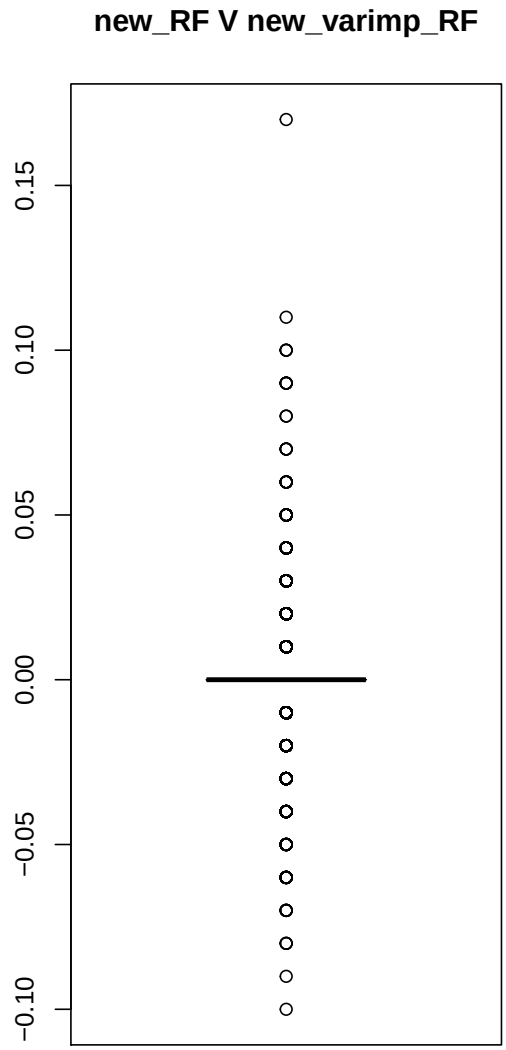


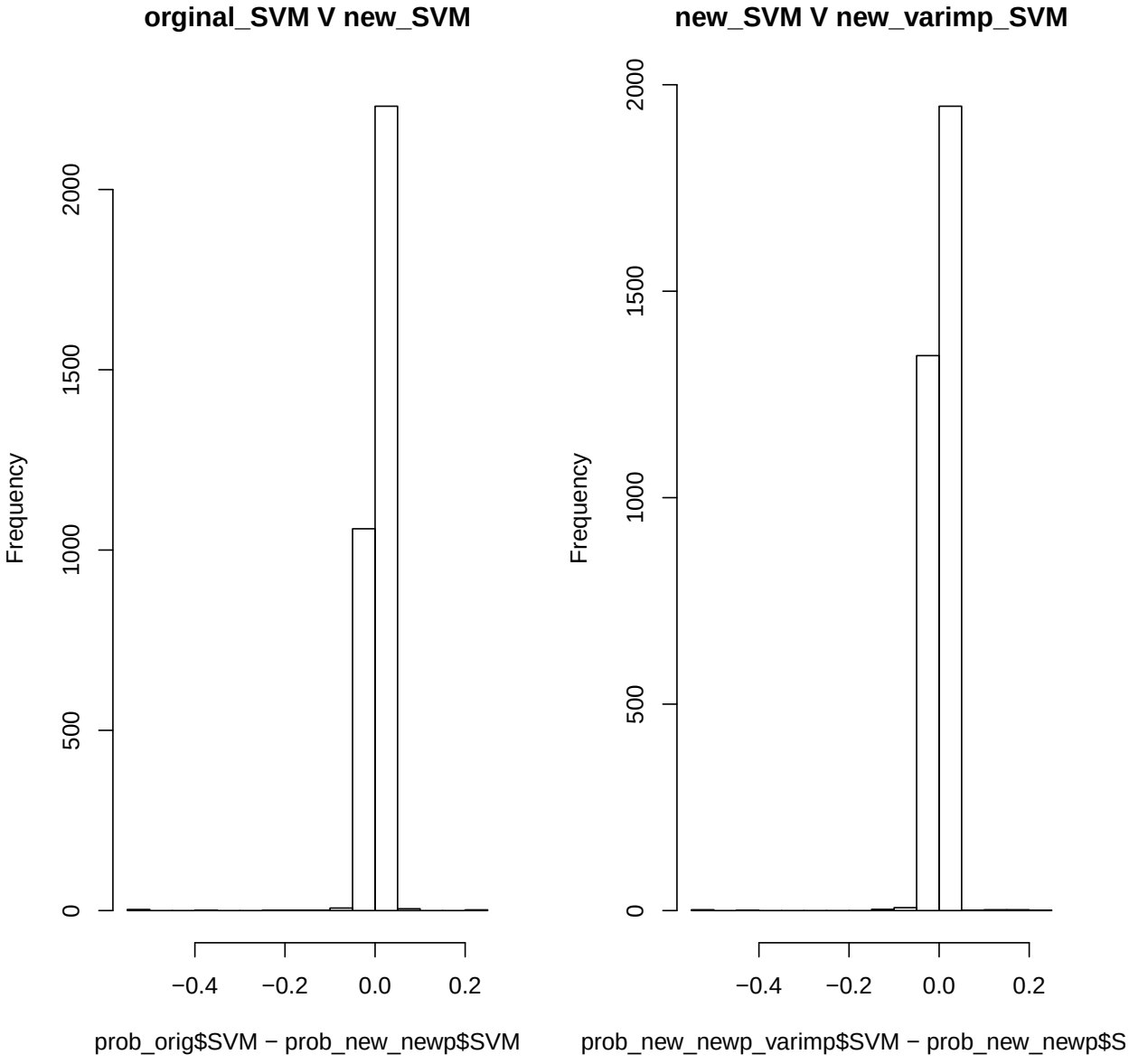
There are no differences in predicted probabilities when the new code is run with the most recent packages and the old code is run with the most recent packages in 2009. This along with no difference in prediction suggests that there have not been any changes to the defaults in the packages that were originally used in ChemModLab in 2009.

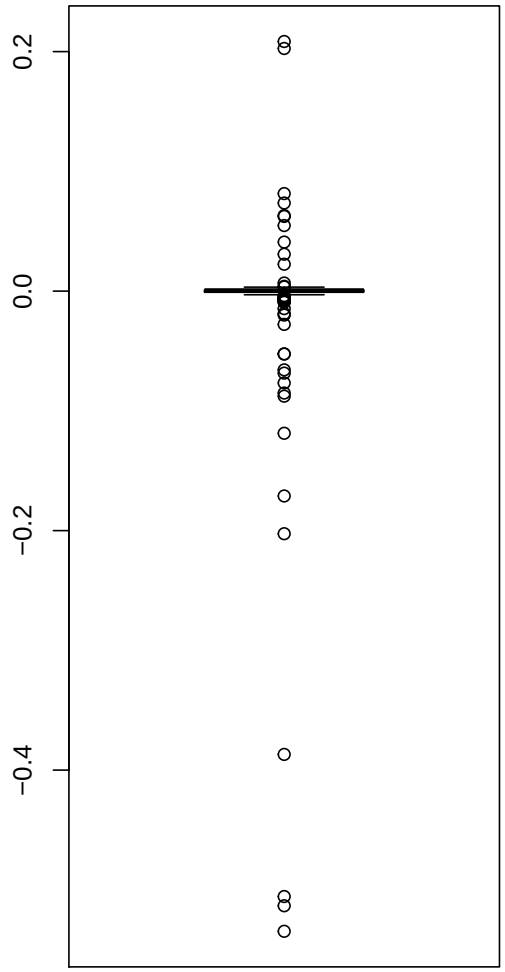
The distributions of the differences of the predicted probabilities between the original and new code resemble differences between new code and new code when only the variable importance measure is turned on.

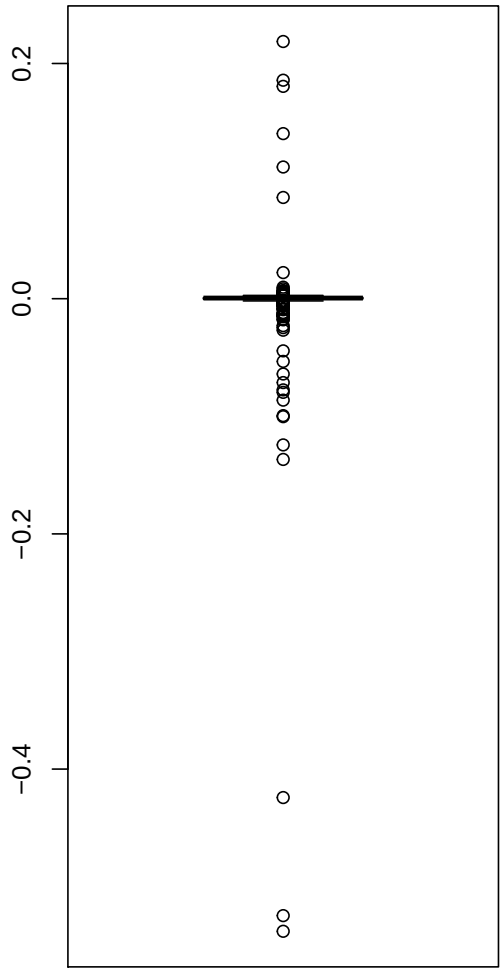

orginal_RF V new_RF
Frequency
0
500
1000
1500
2000
−0.10
0.00
0.05
0.10
prob_orig$RF − prob_new_newp$RF
new_RF V new_varimp_RF
Frequency
0
500
1000
1500
2000
−0.10
0.00
0.10
prob_new_newp_varimp$RF − prob_new_newp$R


orginal_RF V new_RF
0.10
0.05
0.00
−0.05
−0.10


new_RF V new_varimp_RF
0.15
0.10
0.05
0.00
−0.05
−0.10


orginal_SVM V new_SVM
Frequency
0
500
1000
1500
2000
−0.4
−0.2
0.0
0.2
prob_orig$SVM − prob_new_newp$SVM
new_SVM V new_varimp_SVM
Frequency
0
500
1000
1500
2000
−0.4
−0.2
0.0
0.2
prob_new_newp_varimp$SVM − prob_new_newp$S


orginal_SVM V new_SVM
0.2
0.0
−0.2
−0.4


new_SVM V new_varimp_SVM
0.2
0.0
−0.2
−0.4