Assignment 5

Nishanth Gandhidoss

11/20/2017

Question 1

Lets first load the data in here and check whether I have the data. I am just printing the first few elements so that it wont create cluttering in my submission.

```
## [1] Mild   None   None   Severe Mild   Severe
## Levels: Mild None Severe
```

```
##    Z1  Z2  Z3 Z4 Z5
## 1 4.0 4.0 4.0  6  4
## 2 4.0 4.5 4.0  6  4
## 3 4.0 4.0 4.0  6  4
## 4 4.9 4.0 4.5  4  4
## 5 4.0 4.0 4.0  6  4
## 6 4.0 4.0 4.0  6  4
```

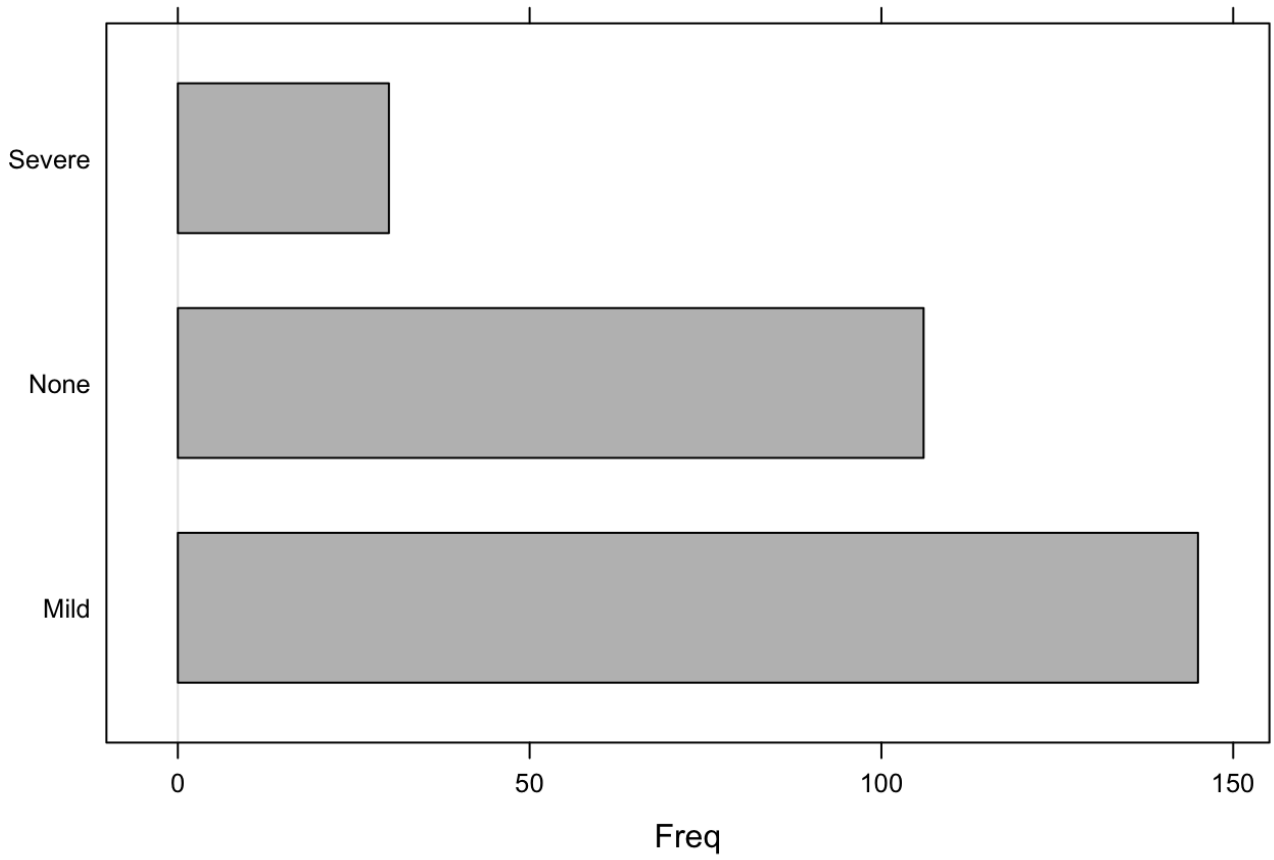
```
##   X1     X2 X3 X4 X5
## 1  0  2.497  0  0 18
## 2  0  0.153  0  0 15
## 3  0 -2.046  0  0 12
## 4  0  3.657  0  0 25
## 5  0  2.332  0  0 28
## 6  0  2.056  0  0 24
```

Thus we see above that we have the data loaded properly.

Section (a)

Lets have look at the distributions of injury which is our responding variable.

```
## injury
##   Mild   None Severe
##    145    106     30
```



From the above table and barchart, we can see that the frequency of Mild is higher than both severe and None. Thus there is no equal distribution of the classes here. I am going to check near zero variance and remove those predictors which have near zero variance.

```
## [1] "Biological : Reducing the 82 zero variance columns from 184 predcitors (fra
ction =  0.445652)"
```

```
## [1] "Chemical : Reducing the 58 zero variance columns from 192 predcitors (fract
ion =  0.302083)"
```

```
## [1] "Combined : Reducing the 140 zero variance columns from 376 predcitors (frac
tion =  0.372340)"
```

Thus we have the predictors with only non zero variance.

I am going to split the data into train and test set for all three set of predictors here which is biological only, chemical only, biological and chemical. Since we have imbalance in the responding variable, I am going to use stratified sampling to split the dataset using createDataPartition.

```
## [1] "All three set of predictors and responding variables train and test split i
s completed using stratified sampling"
```

Thus given the imbalance in the dataset, we have split the train and test set using strartified sampling.

Section (b)

The reponding variable here is inujury having three classes “Severe”, “Mild”, “None”. My aim is that I wanted

to create a model that best fits the data so that we can predict all the classes equally. For that purpose, I am here concern about the accuracy of the model. I am going to accuracy for my model prediction. And also to mention that the question in scetion C is asking about how much information is given by the best model on the hepatic data. It is also one more reason I choose to use accuracy.

Thus the classifcation statistics that I am going to use here is Accuracy.

Section (c)

We have split the data into test and train set already and also removed the near zero variance predictors. I will be using those to create the models for biological and chemical predictors seperately.

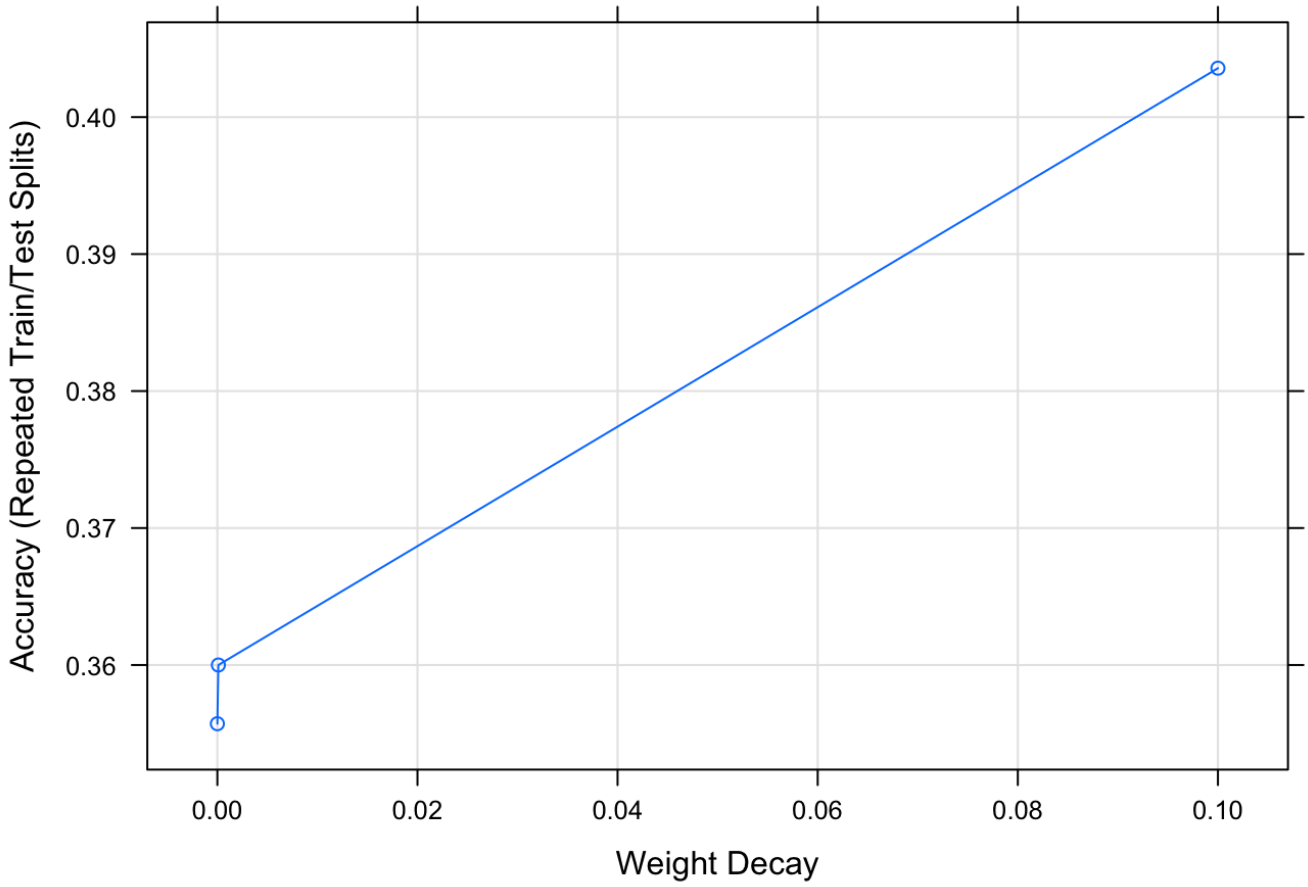
Biological Predictors

First lets see about biological predictors effects on hepatic toxicity.

Logistic Regression

For the logistic regression, I have removed the highly correlated predcitor out of the predictors set with cut off threshold of 0.9. Lets view the results of this now.

```
## Penalized Multinomial Regression
##
## 225 samples
##  95 predictor
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (95), scaled (95)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   decay  Accuracy   Kappa
##   0e+00  0.3557143  -0.016078882
##   1e-04  0.3600000  -0.018147409
##   1e-01  0.4035714  -0.000937217
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was decay = 0.1.
```




```
## [1] "Test set Accuracy:  0.4643"
```

Thus the model has accuracy of 0.4643 on the test set with decay parameter = 0.1

Linear Discriminant Analysis

Similar to logistic regression, I have removed the highly correlated values out of the predictors and then passed on to the model. Since there are no tuning paramter for this model, I am not showing any plots. Lets see the results

```
## Linear Discriminant Analysis
##
## 225 samples
##  95 predictor
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (95), scaled (95)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.3857143  -0.007088105
```

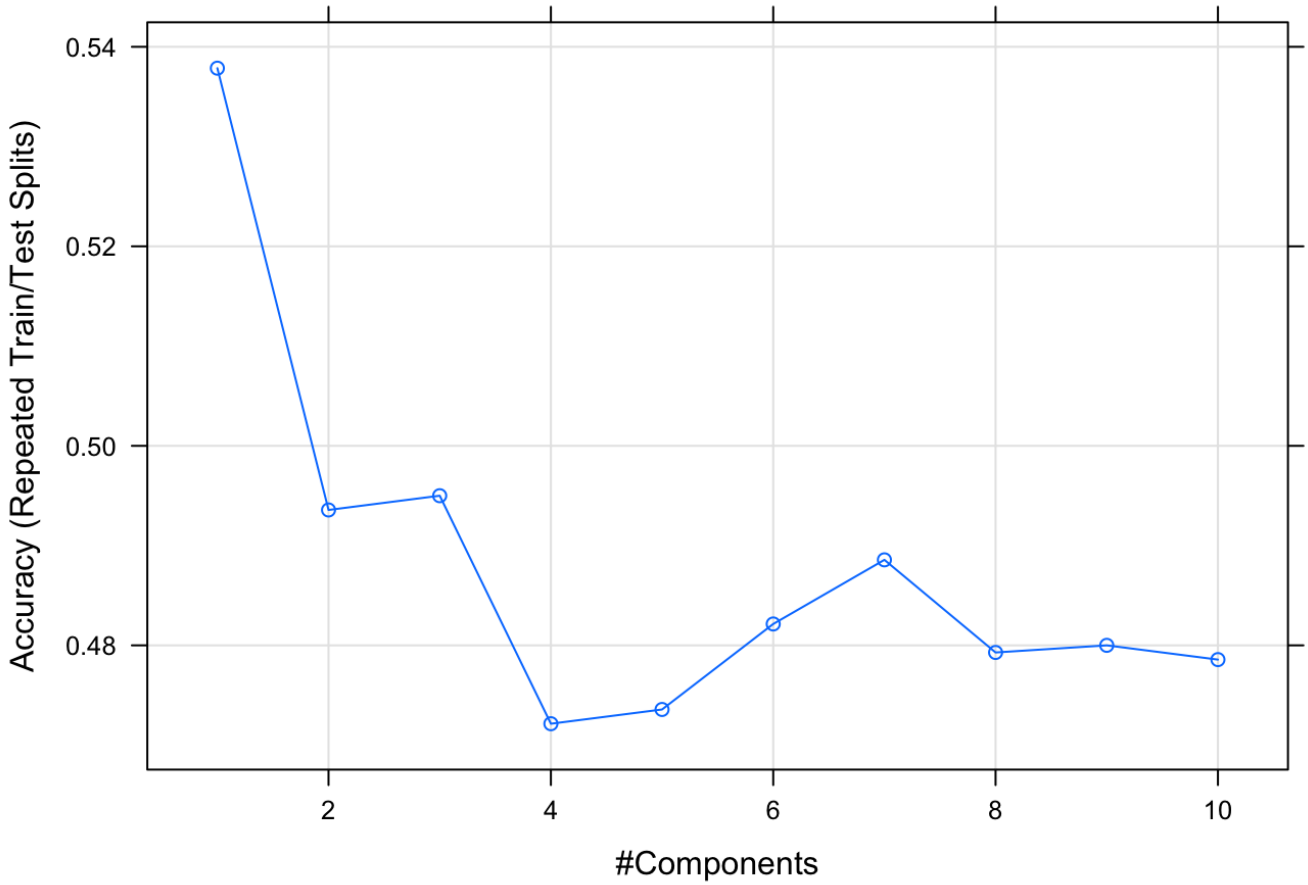
```
## [1] "Test set Accuracy:  0.5"
```

Thus the model has accuracy of 0.5 on the test set.

Partial Least Squares Discriminant Analysis

Let see what is the results for partial least squares discriminant model

```
## Partial Least Squares
##
## 225 samples
## 102 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (102), scaled (102)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   ncomp  Accuracy   Kappa
##    1     0.5378571  0.088735073
##    2     0.4935714  0.029512837
##    3     0.4950000  0.042314809
##    4     0.4721429  0.009987867
##    5     0.4735714  0.015108113
##    6     0.4821429  0.036087483
##    7     0.4885714  0.049967748
##    8     0.4792857  0.037603937
##    9     0.4800000  0.045399743
##   10     0.4785714  0.043833580
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was ncomp = 1.
```




```
## [1] "Test set Accuracy:  0.4643"
```

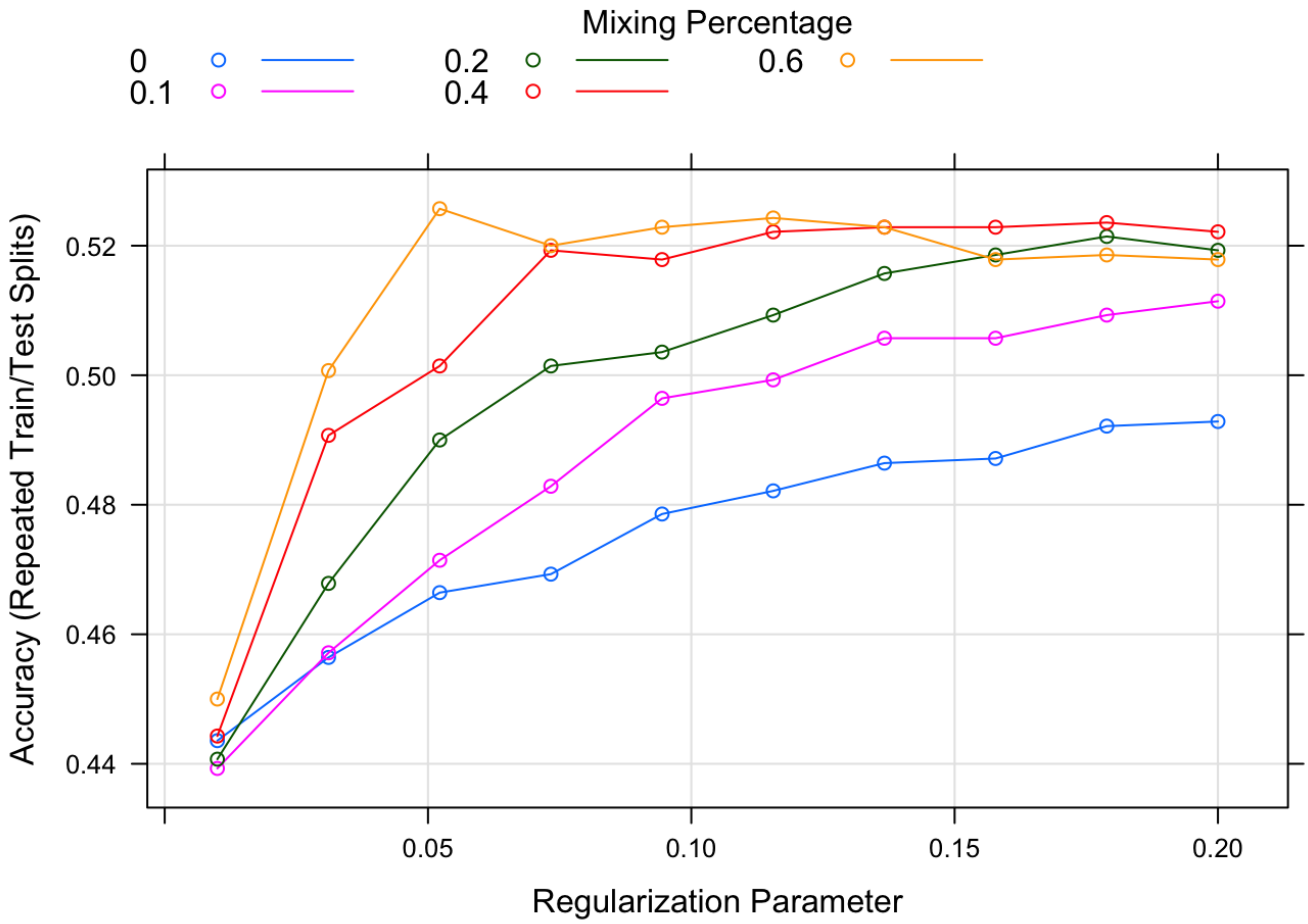
Thus the model has accuracy of 0.4643 on the test set with one component.

Penalized Methods

Let see whst is the result for penalized model

```
## glmnet
##
## 225 samples
## 102 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (102), scaled (102)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda      Accuracy   Kappa
##   0.0    0.01000000  0.4435714  0.029486476
##   0.0    0.03111111  0.4564286  0.029086777
##   0.0    0.05222222  0.4664286  0.026634400
##   0.0    0.07333333  0.4692857  0.024459869
##   0.0    0.09444444  0.4785714  0.032555223
##   0.0    0.11555556  0.4821429  0.033341463
##   0.0    0.13666667  0.4864286  0.036837006
##   0.0    0.15777778  0.4871429  0.033881851
##   0.0    0.17888889  0.4921429  0.039624562
```

```
##   0.0    0.17000000  0.4921429  0.059024502
##   0.0    0.20000000  0.4928571  0.039094109
##   0.1    0.01000000  0.4392857  0.018075956
##   0.1    0.03111111  0.4571429  0.016538325
##   0.1    0.05222222  0.4714286  0.021047328
##   0.1    0.07333333  0.4828571  0.027438616
##   0.1    0.09444444  0.4964286  0.044000467
##   0.1    0.11555556  0.4992857  0.044083784
##   0.1    0.13666667  0.5057143  0.051013605
##   0.1    0.15777778  0.5057143  0.044826967
##   0.1    0.17888889  0.5092857  0.045286935
##   0.1    0.20000000  0.5114286  0.045433977
##   0.2    0.01000000  0.4407143  0.017372585
##   0.2    0.03111111  0.4678571  0.023667602
##   0.2    0.05222222  0.4900000  0.039507069
##   0.2    0.07333333  0.5014286  0.049724835
##   0.2    0.09444444  0.5035714  0.043132186
##   0.2    0.11555556  0.5092857  0.046006484
##   0.2    0.13666667  0.5157143  0.052717229
##   0.2    0.15777778  0.5185714  0.052834811
##   0.2    0.17888889  0.5214286  0.052798402
##   0.2    0.20000000  0.5192857  0.045512269
##   0.4    0.01000000  0.4442857  0.016084557
##   0.4    0.03111111  0.4907143  0.041532852
##   0.4    0.05222222  0.5014286  0.044492331
##   0.4    0.07333333  0.5192857  0.064369560
##   0.4    0.09444444  0.5178571  0.050008852
##   0.4    0.11555556  0.5221429  0.050606372
##   0.4    0.13666667  0.5228571  0.043780908
##   0.4    0.15777778  0.5228571  0.034380911
##   0.4    0.17888889  0.5235714  0.029144801
##   0.4    0.20000000  0.5221429  0.017394123
##   0.6    0.01000000  0.4500000  0.015883423
##   0.6    0.03111111  0.5007143  0.049247615
##   0.6    0.05222222  0.5257143  0.079492245
##   0.6    0.07333333  0.5200000  0.050822379
##   0.6    0.09444444  0.5228571  0.044974504
##   0.6    0.11555556  0.5242857  0.035539643
##   0.6    0.13666667  0.5228571  0.019929279
##   0.6    0.15777778  0.5178571  0.002813405
##   0.6    0.17888889  0.5185714  0.001684211
##   0.6    0.20000000  0.5178571  0.000000000
##
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were alpha = 0.6 and lambda
##  = 0.05222222.
```



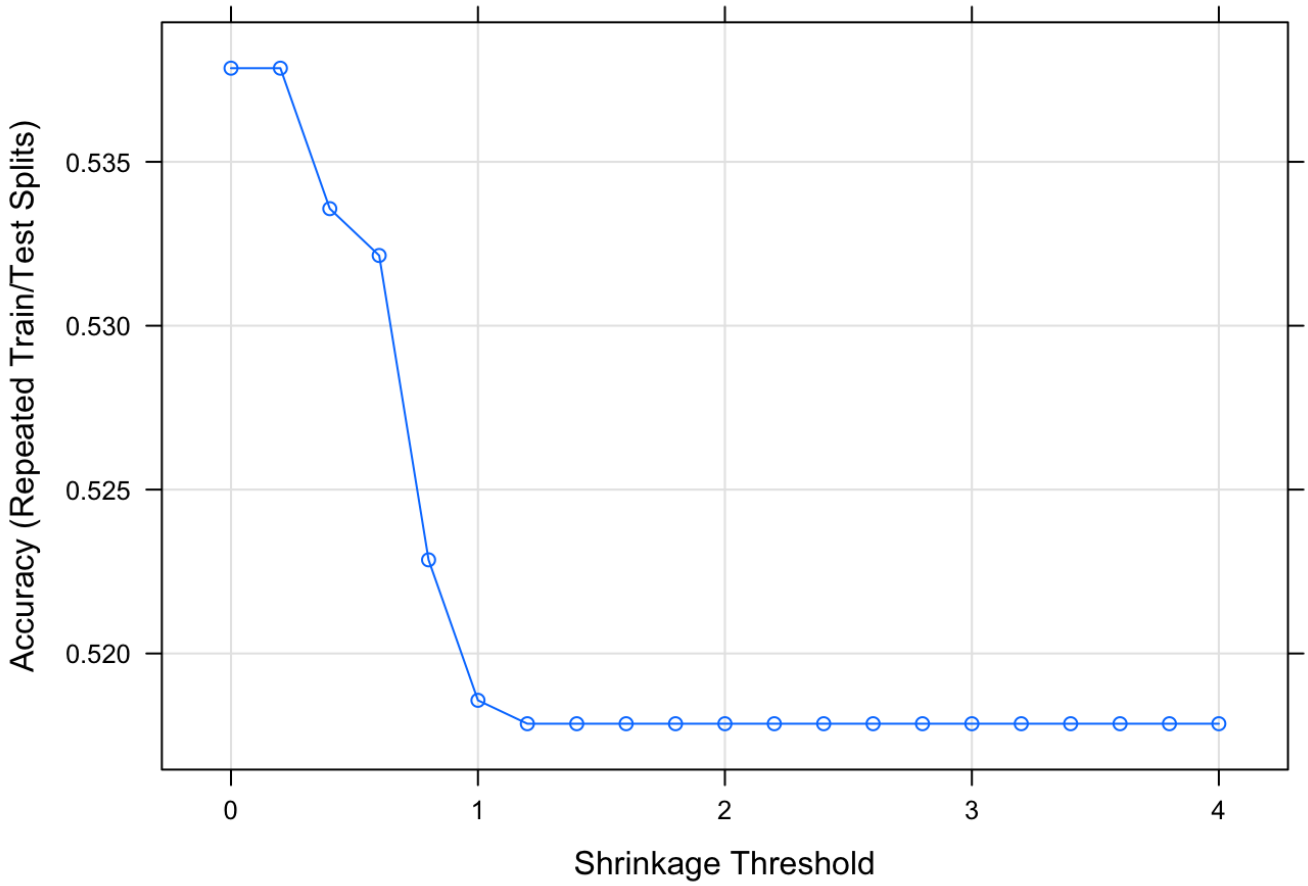

```
## [1] "Test set Accuracy:  0.5"
```

Thus the model has accuracy of 0.5 on the test set with alpha = 0.6 and lambda = 0.05222222.

Nearest shrunken Centroids

And finally, lets see the results of Nearest shrunken Centroids for the biological predictors

```
## Nearest Shrunken Centroids
##
## 225 samples
## 102 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (102), scaled (102)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   threshold  Accuracy   Kappa
##   0.0        0.5378571  0.104485291
##   0.2        0.5378571  0.099908189
##   0.4        0.5335714  0.084803018
##   0.6        0.5321429  0.072818004
##   0.8        0.5228571  0.034766318
##   1.0        0.5185714  0.007949487
##   1.2        0.5178571  0.000000000
##   1.4        0.5178571  0.000000000
##   1.6        0.5178571  0.000000000
##   1.8        0.5178571  0.000000000
##   2.0        0.5178571  0.000000000
##   2.2        0.5178571  0.000000000
##   2.4        0.5178571  0.000000000
##   2.6        0.5178571  0.000000000
##   2.8        0.5178571  0.000000000
##   3.0        0.5178571  0.000000000
##   3.2        0.5178571  0.000000000
##   3.4        0.5178571  0.000000000
##   3.6        0.5178571  0.000000000
##   3.8        0.5178571  0.000000000
##   4.0        0.5178571  0.000000000
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was threshold = 0.
```




```
## [1] "Test set Accuracy:  0.4464"
```

Thus the model has accuracy of 0.4464 on the test set with threshold = 0.

Below table shows the compilation of the information regarding the above models all together.

Model	Parameter	Training Accuracy	Testing Accuracy
Logistic Regression	decay = 0.1	0.4035714	0.4643
Linear Discriminant Analysis	None	0.3857	0.5
Partial Least Squares Discriminant Analysis	ncomp = 1	0.5379	0.4643
Penalized Methods	alpha = 0.6 & lambda = 0.05222222	0.5257143	0.5
Nearest shrunken Centroids	threshold = 0	0.5379	0.4464

From the above table we can see that Linear Discriminant Analysis(LDA) and Penalized Methods has better predictive ability with test set accuracy of 0.5 for the biological predictors.

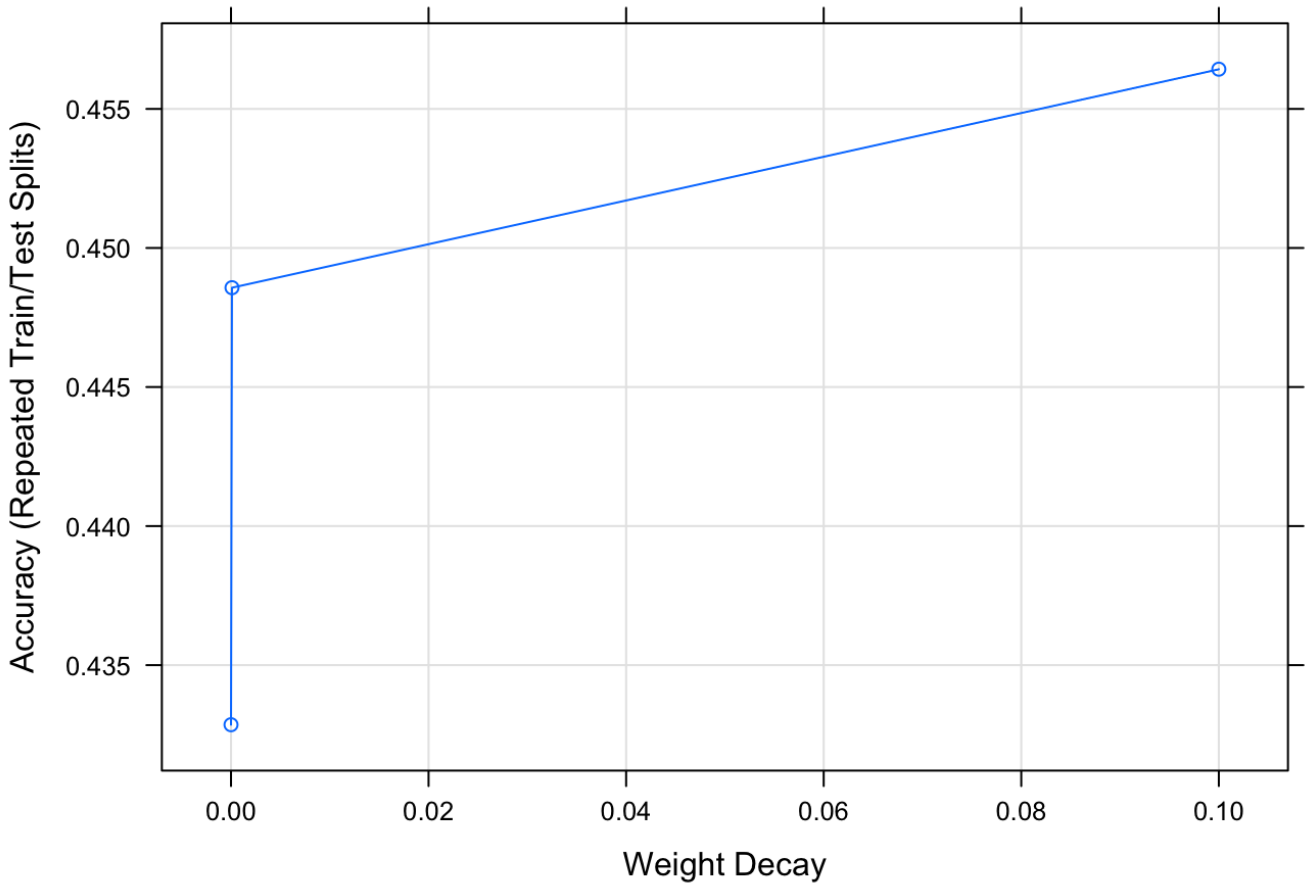
Chemical Predcitor

Now lets see about chemical predictors effects on hepatic toxicity.

Logistic Regression

For the logistic regression, I have removed the highly correlated predcitor out of the predictors set with cut off threshold of 0.9. Lets view the results of this now.

```
## Penalized Multinomial Regression
##
## 225 samples
## 108 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (108), scaled (108)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   decay  Accuracy   Kappa
##   0e+00  0.4328571  0.07112906
##   1e-04  0.4485714  0.07644722
##   1e-01  0.4564286  0.07527396
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was decay = 0.1.
```




```
## [1] "Test set Accuracy:  0.5714"
```

Thus the model has accuracy of 0.5714 on the test set with decay 0.1.

Linear Discriminant Analysis

Similar to logistic regression, I have removed the highly correlated values out of the predictors and then passed on to the model. Since there are no tuning paramter for this model, I am not showing any plots. Lets see the results

```
## Linear Discriminant Analysis
##
## 225 samples
## 108 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (108), scaled (108)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.4564286  0.09741566
```

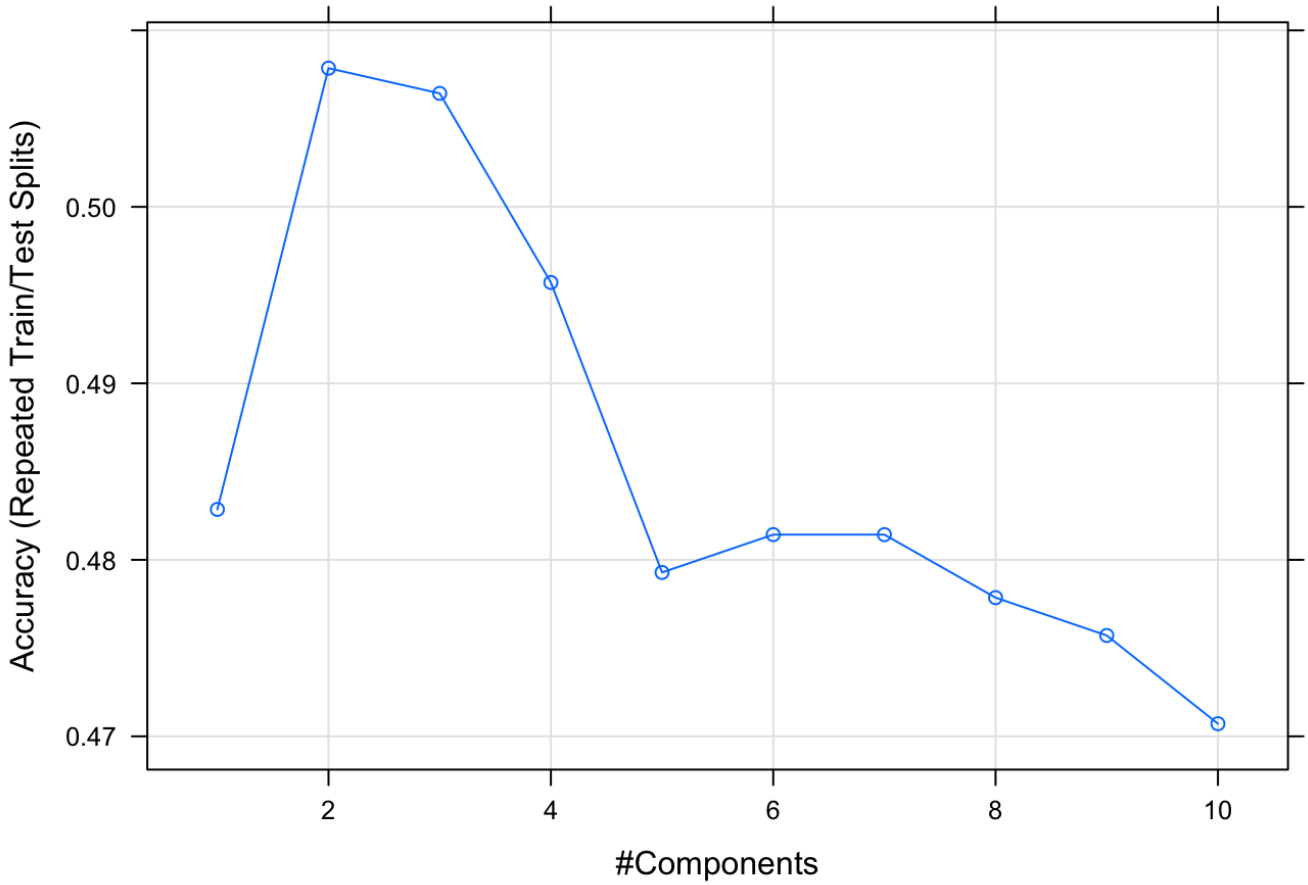
```
## [1] "Test set Accuracy:  0.6607"
```

Thus the model has accuracy of 0.6607 on the test set.

Partial Least Squares Discriminant Analysis

Let see what is the results for partial least squares discriminant model

```
## Partial Least Squares
##
## 225 samples
## 134 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (134), scaled (134)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   ncomp  Accuracy   Kappa
##    1     0.4828571  0.001087308
##    2     0.5078571  0.069807254
##    3     0.5064286  0.075710790
##    4     0.4957143  0.062499467
##    5     0.4792857  0.041294725
##    6     0.4814286  0.049245278
##    7     0.4814286  0.054136522
##    8     0.4778571  0.052656825
##    9     0.4757143  0.051882364
##   10     0.4707143  0.048352711
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was ncomp = 2.
```




```
## [1] "Test set Accuracy:  0.5536"
```

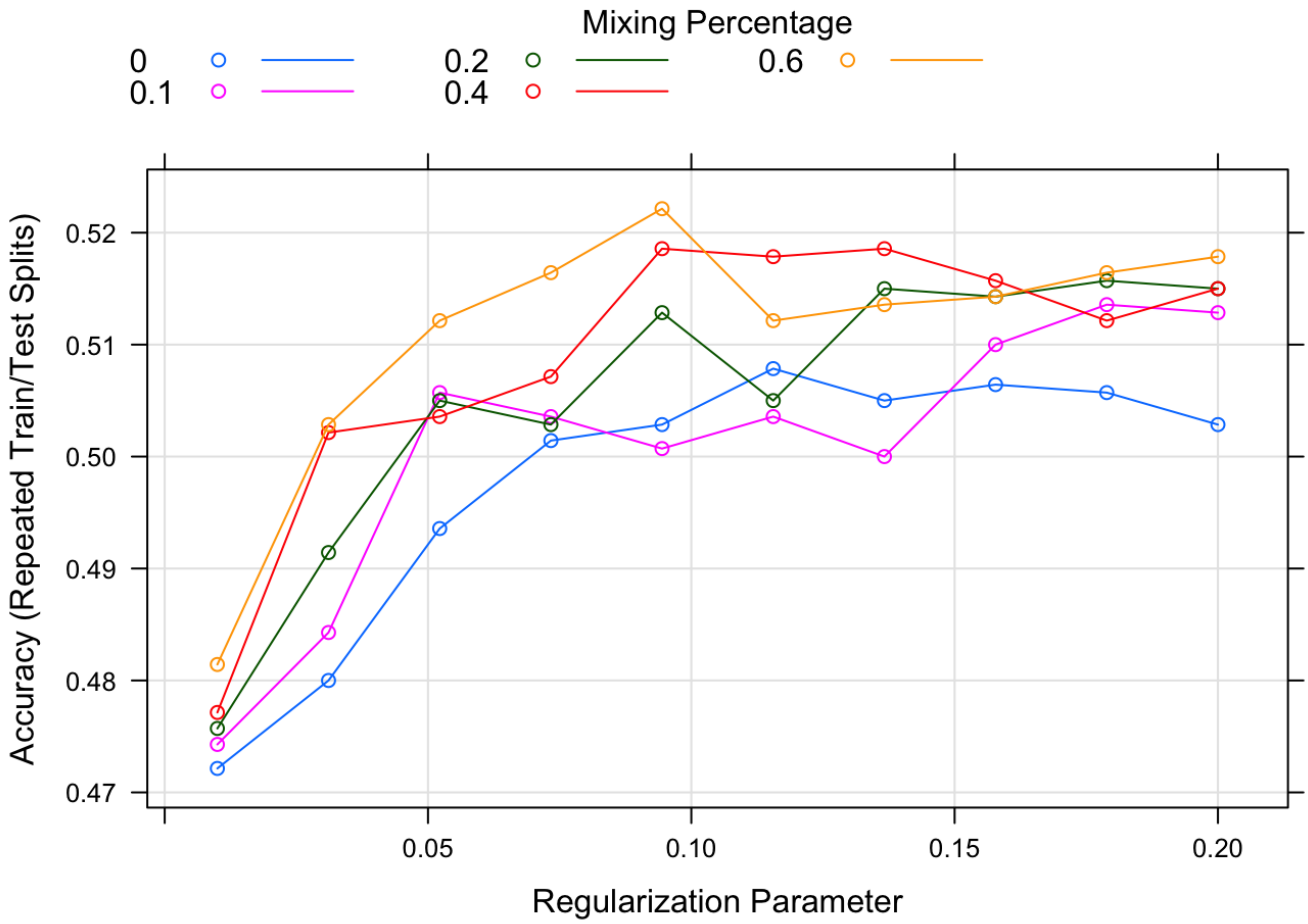
Thus the model has accuracy of 0.5536 on the test set with two components.

Penalized Methods

Let see whst is the result for penalized model

```
## glmnet
##
## 225 samples
## 134 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (134), scaled (134)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda      Accuracy   Kappa
##   0.0    0.01000000  0.4721429   0.0808032889
##   0.0    0.03111111  0.4800000   0.0795303004
##   0.0    0.05222222  0.4935714   0.0907825118
##   0.0    0.07333333  0.5014286   0.0985207917
##   0.0    0.09444444  0.5028571   0.0955469251
##   0.0    0.11555556  0.5078571   0.0971603174
##   0.0    0.13666667  0.5050000   0.0903764774
##   0.0    0.15777778  0.5064286   0.0900602865
##   0.0    0.17888889  0.5057143   0.0864654396
```

```
##   0.0    0.20000000  0.5028571   0.0784133946
##   0.1    0.01000000  0.4742857   0.0852601678
##   0.1    0.03111111  0.4842857   0.0758405007
##   0.1    0.05222222  0.5057143   0.1003822586
##   0.1    0.07333333  0.5035714   0.0835278685
##   0.1    0.09444444  0.5007143   0.0707766405
##   0.1    0.11555556  0.5035714   0.0696321343
##   0.1    0.13666667  0.5000000   0.0570279375
##   0.1    0.15777778  0.5100000   0.0677204415
##   0.1    0.17888889  0.5135714   0.0717612264
##   0.1    0.20000000  0.5128571   0.0648264125
##   0.2    0.01000000  0.4757143   0.0839267652
##   0.2    0.03111111  0.4914286   0.0820853743
##   0.2    0.05222222  0.5050000   0.0856835224
##   0.2    0.07333333  0.5028571   0.0682334228
##   0.2    0.09444444  0.5128571   0.0769800564
##   0.2    0.11555556  0.5050000   0.0529279251
##   0.2    0.13666667  0.5150000   0.0617801964
##   0.2    0.15777778  0.5142857   0.0521493104
##   0.2    0.17888889  0.5157143   0.0467041975
##   0.2    0.20000000  0.5150000   0.0378869460
##   0.4    0.01000000  0.4771429   0.0806163964
##   0.4    0.03111111  0.5021429   0.0853846448
##   0.4    0.05222222  0.5035714   0.0640413617
##   0.4    0.07333333  0.5071429   0.0546891124
##   0.4    0.09444444  0.5185714   0.0576525166
##   0.4    0.11555556  0.5178571   0.0407369193
##   0.4    0.13666667  0.5185714   0.0256465639
##   0.4    0.15777778  0.5157143   0.0097480304
##   0.4    0.17888889  0.5121429  -0.0039809468
##   0.4    0.20000000  0.5150000  -0.0026676377
##   0.6    0.01000000  0.4814286   0.0856957109
##   0.6    0.03111111  0.5028571   0.0730058692
##   0.6    0.05222222  0.5121429   0.0658931063
##   0.6    0.07333333  0.5164286   0.0489712067
##   0.6    0.09444444  0.5221429   0.0351582599
##   0.6    0.11555556  0.5121429  -0.0006284534
##   0.6    0.13666667  0.5135714  -0.0055233023
##   0.6    0.15777778  0.5142857  -0.0060496319
##   0.6    0.17888889  0.5164286  -0.0025130890
##   0.6    0.20000000  0.5178571   0.0000000000
##
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were alpha = 0.6 and lambda
##  = 0.09444444.
```



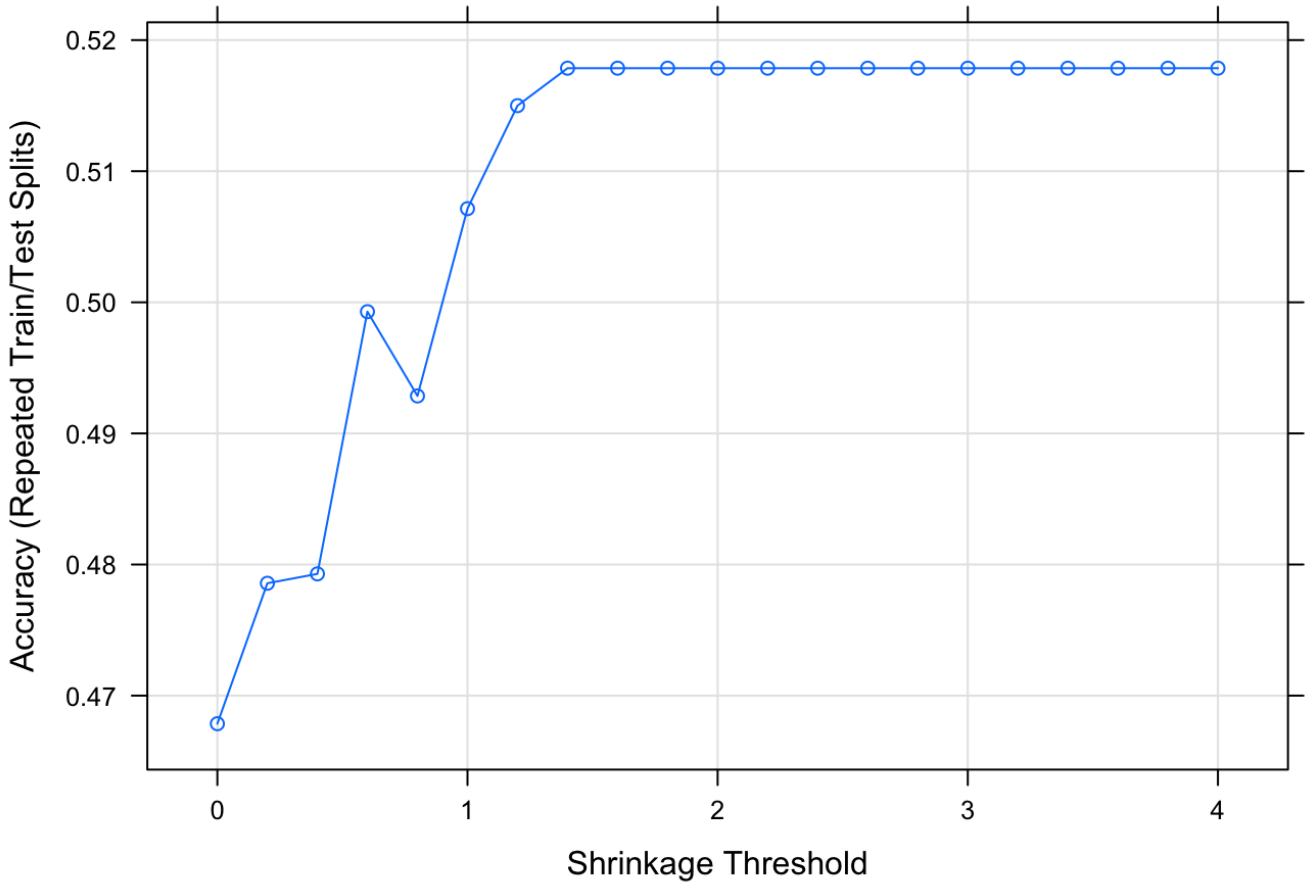

```
## [1] "Test set Accuracy:  0.5536"
```

Thus the model has accuracy of 0.5536 on the test set with alpha = 0.6 and lambda = 0.09444444.

Nearest shrunken Centroids

And finally, lets see the results of Nearest shrunken Centroids for the biological predictors

```
## Nearest Shrunken Centroids
##
## 225 samples
## 134 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (134), scaled (134)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   threshold  Accuracy   Kappa
##   0.0        0.4678571   0.055830247
##   0.2        0.4785714   0.060309066
##   0.4        0.4792857   0.028584018
##   0.6        0.4992857   0.031711456
##   0.8        0.4928571  -0.015832863
##   1.0        0.5071429  -0.009923811
##   1.2        0.5150000  -0.003629763
##   1.4        0.5178571   0.000000000
##   1.6        0.5178571   0.000000000
##   1.8        0.5178571   0.000000000
##   2.0        0.5178571   0.000000000
##   2.2        0.5178571   0.000000000
##   2.4        0.5178571   0.000000000
##   2.6        0.5178571   0.000000000
##   2.8        0.5178571   0.000000000
##   3.0        0.5178571   0.000000000
##   3.2        0.5178571   0.000000000
##   3.4        0.5178571   0.000000000
##   3.6        0.5178571   0.000000000
##   3.8        0.5178571   0.000000000
##   4.0        0.5178571   0.000000000
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was threshold = 1.4.
```




```
## [1] "Test set Accuracy:  0.5179"
```

Thus the model has accuracy of 0.5179 on the test set with threshold = 1.4.

Below table shows the compilation of the information regarding the above models all together.

Model	Parameter	Training Accuracy	Testing Accuracy
Logistic Regression	decay = 0.1	0.4564	0.5714
Linear Discriminant Analysis	None	0.4564	0.6607
Partial Least Squares Discriminant Analysis	ncomp = 2	0.5079	0.5536
Penalized Methods	alpha = 0.6 & lambda = 0.09444444	0.5221	0.5536
Nearest shrunken Centroids	threshold = 1.4	0.5179	0.5179

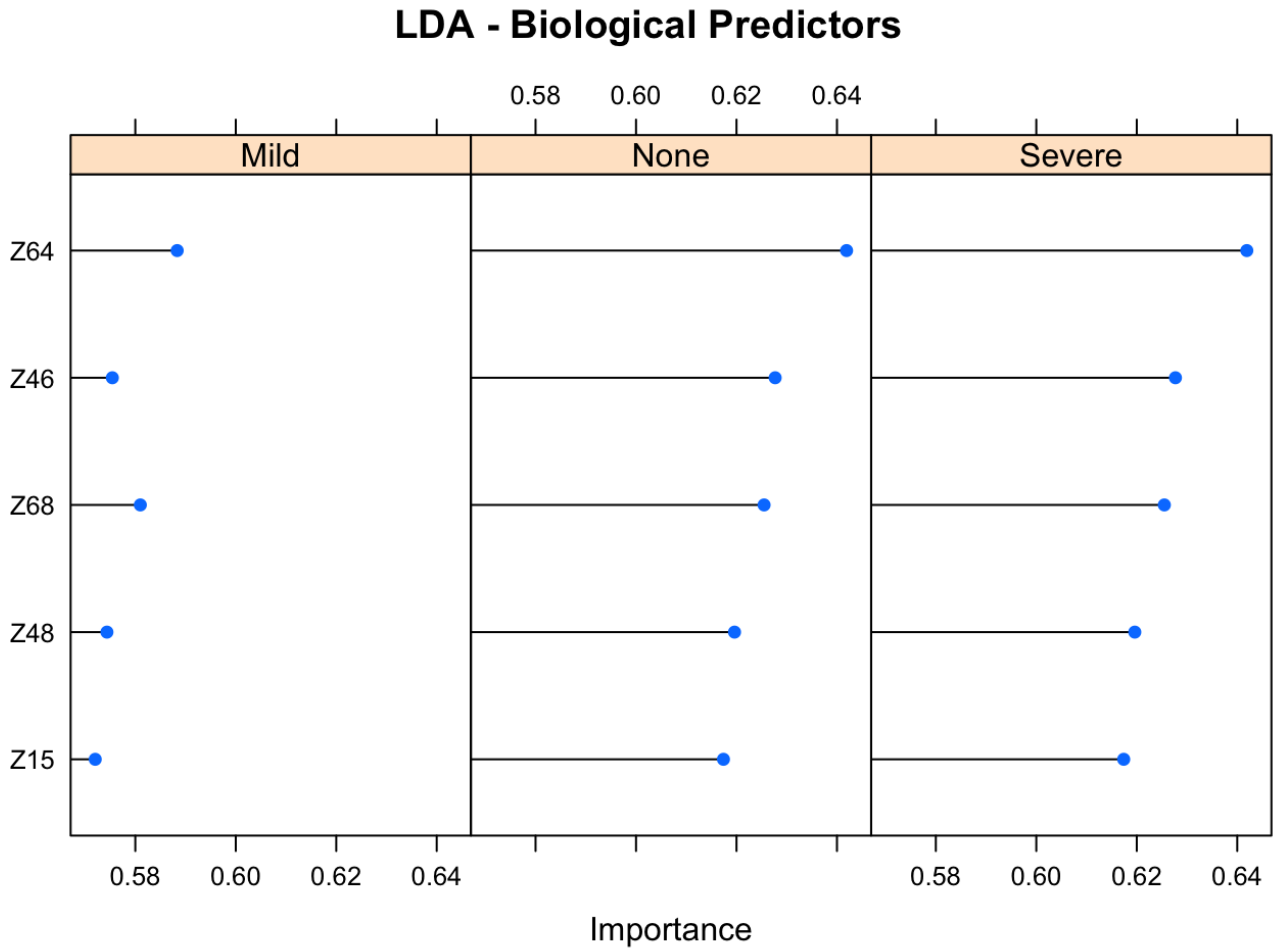
From the above table we can see that Linear Discriminant Analysis(LDA) model has better predictive ability with test set accuracy of 0.6607 for the chemical predictors.

Thus on comparing the chemical and biological predictors results and its model accuracies, it looks like the Linear Discriminant Analysis(LDA) model using the Chemical predictors contains the most information about hepatic toxicity.

Section (d)

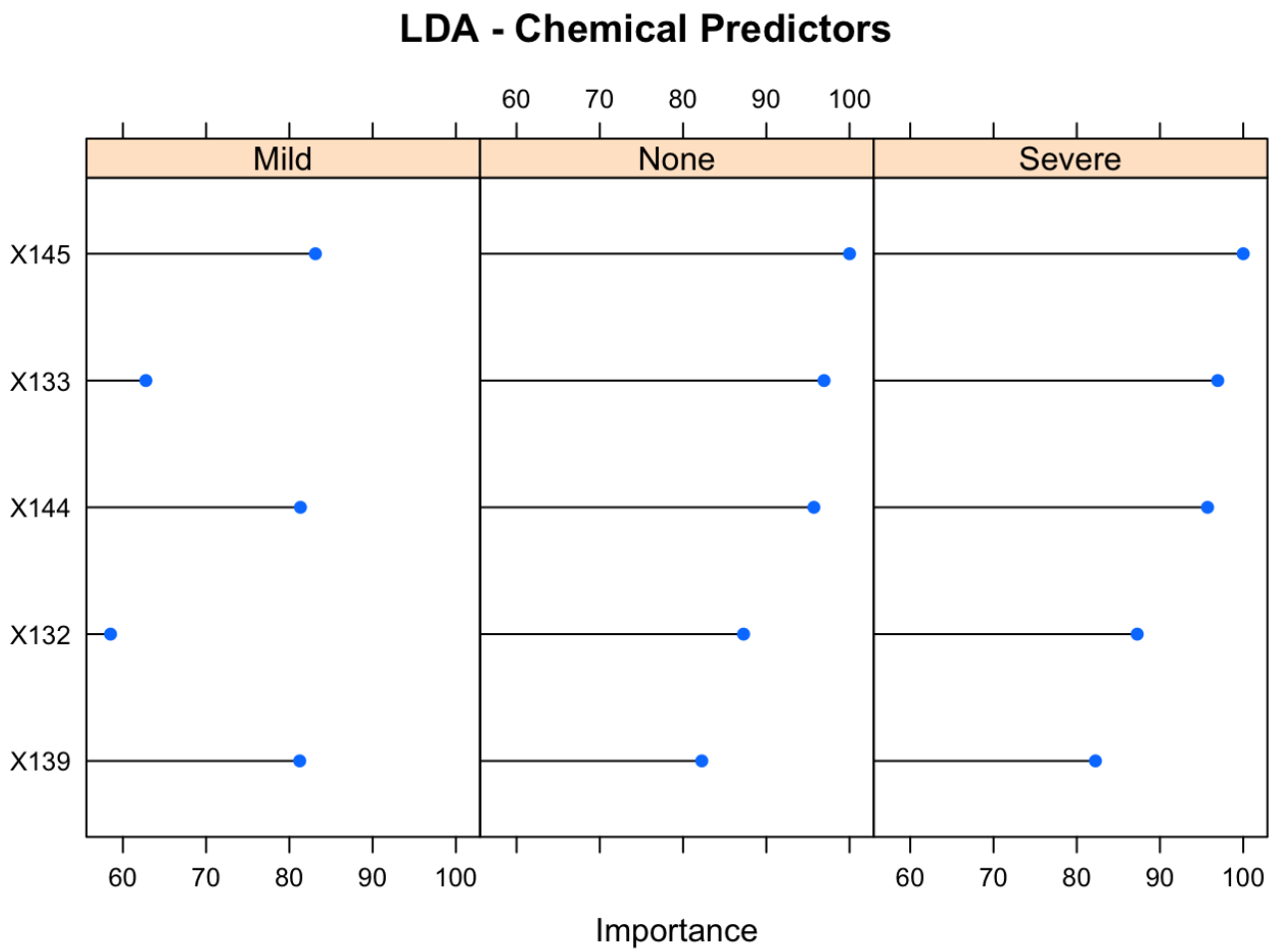
Now let see the optimal models for both the biological and chemical predictors, the top five important predictors. The best model for both biological and chemical predictor is Linear Discriminant Analysis.

```
##           Mild      None    Severe
## Z64  0.5883367 0.6419118 0.6419118
## Z100 0.6115302 0.6137255 0.6137255
## Z68  0.5809986 0.6254902 0.6254902
## Z46  0.5754310 0.6276961 0.6276961
## Z48  0.5743534 0.6196078 0.6196078
```



From the above table, we can see the top 5 important predictors for the biological set of predictors.

```
##          Mild      None    Severe
## X145 83.12645 100.00000 100.00000
## X144 81.33207  95.71459  95.71459
## X133 62.76019  96.93899  96.93899
## X139 81.24235  82.24615  82.24615
## X132 58.51805  87.26620  87.26620
```



From the above table, we can see the top 5 important predictors for the chemical set of predictors.

Section (e)

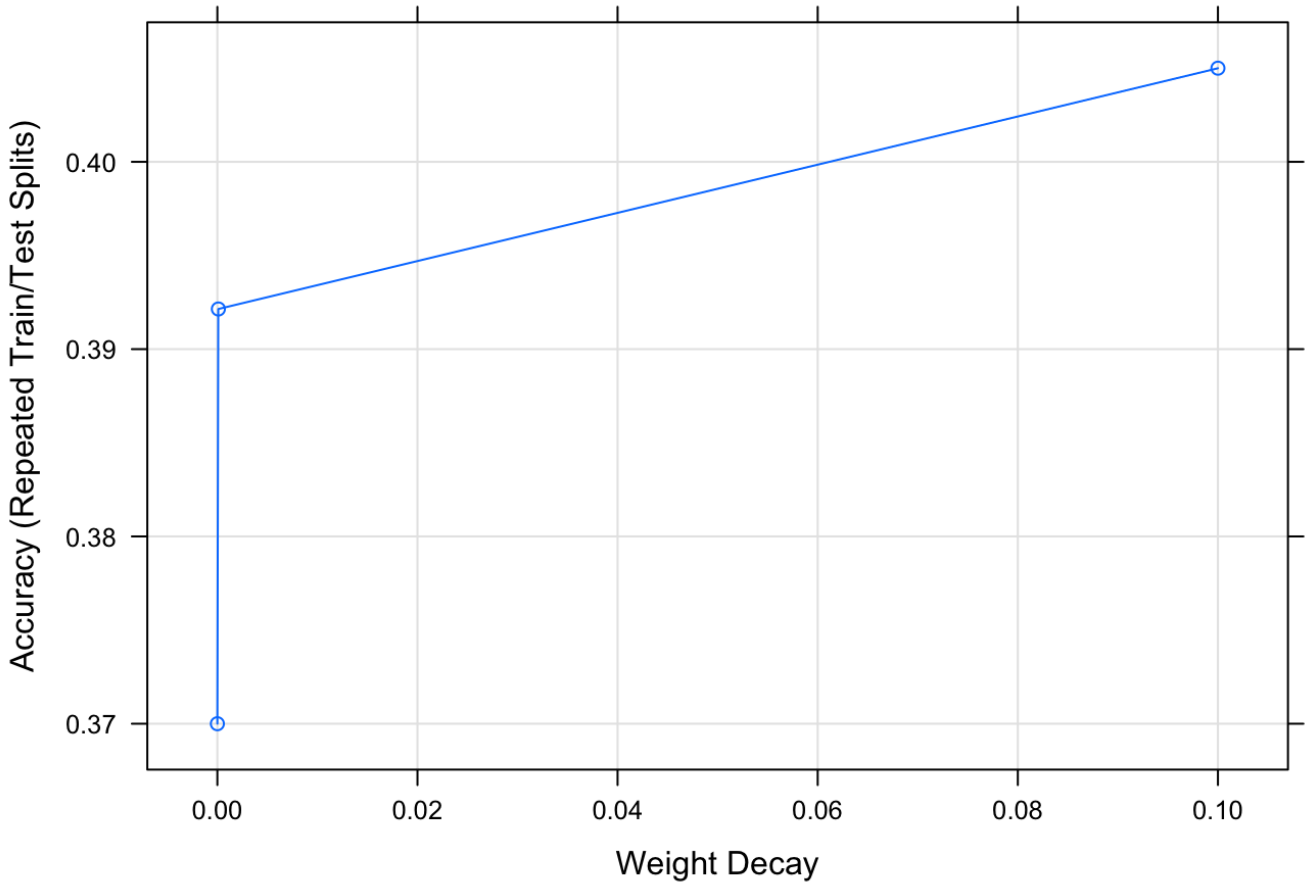
Combined Predictors

I have combined both chemical and biological predictor into one. Let try to predict the hepatic toxcity with this combined predictors

Logistic Regression

For the logistic regression, I have removed the highly correlated predcitor out of the predictors set with cut off threshold of 0.9. Lets view the results of this now.

```
## Penalized Multinomial Regression
##
## 225 samples
## 203 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (203), scaled (203)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   decay  Accuracy   Kappa
##   0e+00  0.3700000  -0.012565330
##   1e-04  0.3921429  -0.008620933
##   1e-01  0.4050000  -0.007890071
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was decay = 0.1.
```




```
## [1] "Test set Accuracy:  0.5893"
```

Thus the model has accuracy of 0.5893 on the test set with decay = 0.1.

Linear Discriminant Analysis

Similar to logistic regression, I have removed the highly correlated values out of the predictors and then passed on to the model. Since there are no tuning paramter for this model, I am not showing any plots. Lets

see the results

```
## Linear Discriminant Analysis
##
## 225 samples
## 203 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (203), scaled (203)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.3514286  -0.01342954
```

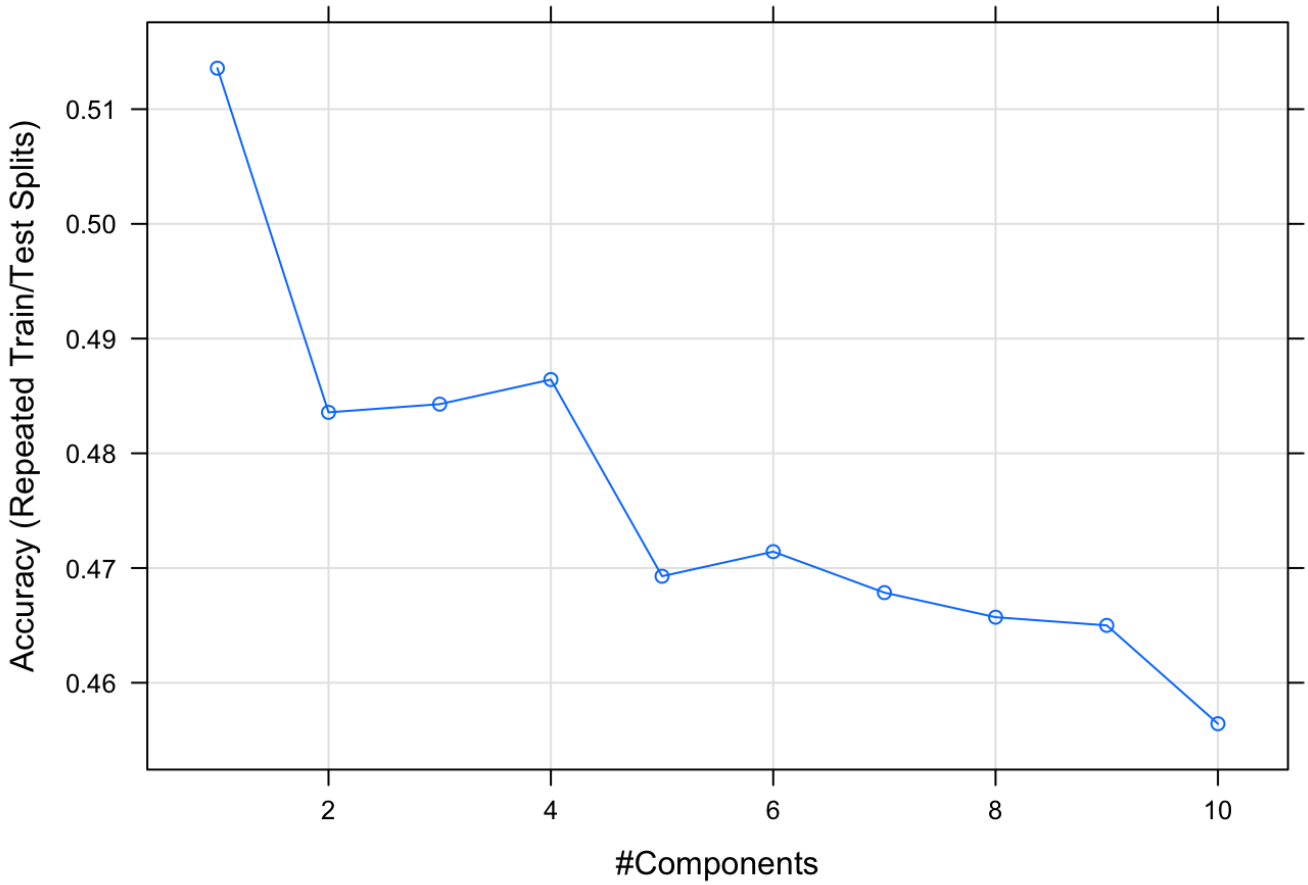
```
## [1] "Test set Accuracy:  0.3929"
```

Thus the model has accuracy of 0.3929 on the test set.

Partial Least Squares Discriminant Analysis

Let see what is the results for partial least squares discriminant model

```
## Partial Least Squares
##
## 225 samples
## 236 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (236), scaled (236)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   ncomp  Accuracy   Kappa
##    1     0.5135714  0.06089109
##    2     0.4835714  0.02235697
##    3     0.4842857  0.03513239
##    4     0.4864286  0.04109171
##    5     0.4692857  0.01671872
##    6     0.4714286  0.03130051
##    7     0.4678571  0.03143297
##    8     0.4657143  0.03026566
##    9     0.4650000  0.03165091
##   10     0.4564286  0.01841661
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was ncomp = 1.
```




```
## [1] "Test set Accuracy:  0.4643"
```

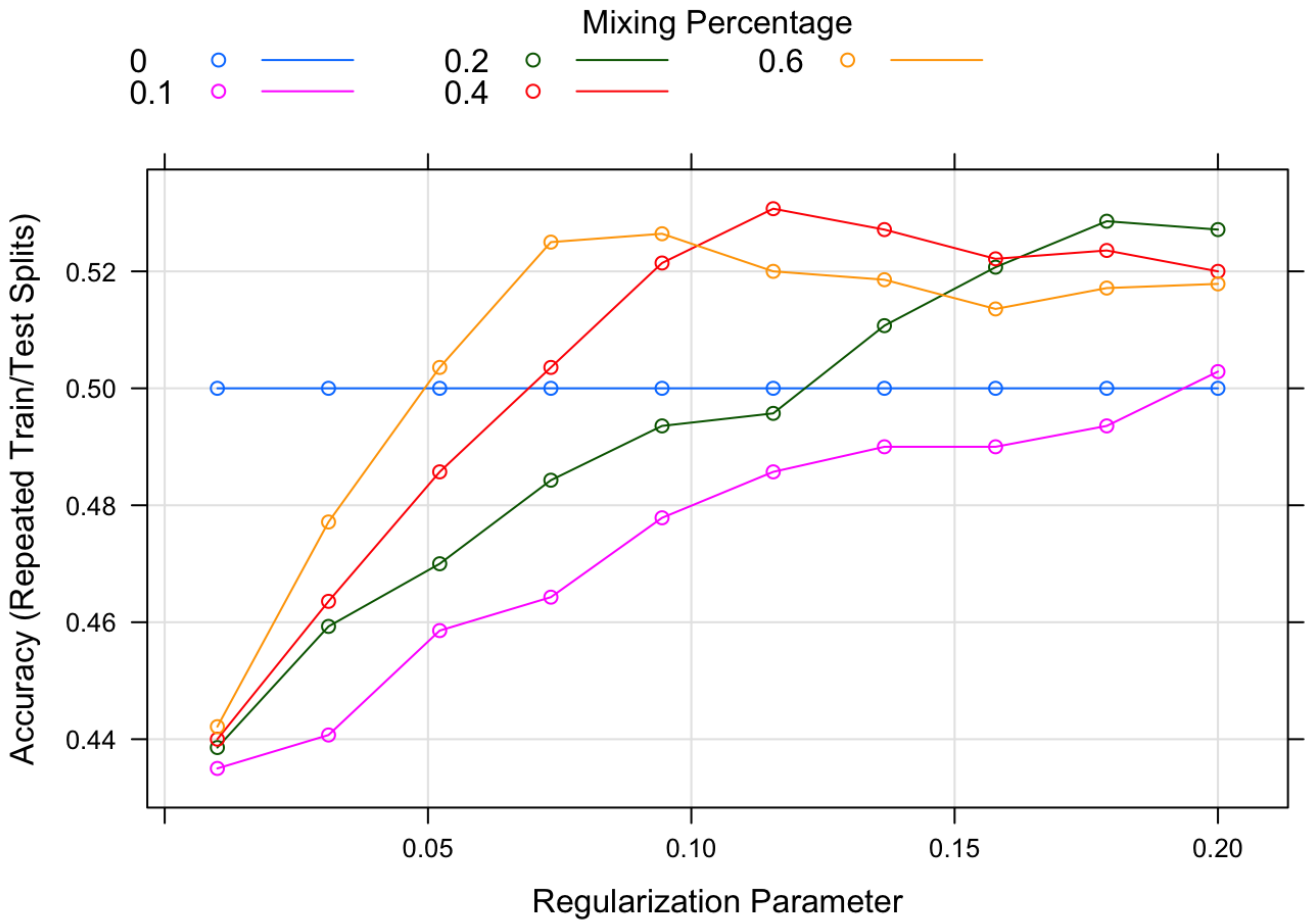
Thus the model has accuracy of 0.4643 on the test set with ncomp = 1.

Penalized Methods

Let see whst is the result for penalized model

```
## glmnet
##
## 225 samples
## 236 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (236), scaled (236)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda      Accuracy   Kappa
##   0.0    0.01000000  0.5000000   0.0426512103
##   0.0    0.03111111  0.5000000   0.0426512103
##   0.0    0.05222222  0.5000000   0.0426512103
##   0.0    0.07333333  0.5000000   0.0426512103
##   0.0    0.09444444  0.5000000   0.0426512103
##   0.0    0.11555556  0.5000000   0.0426512103
##   0.0    0.13666667  0.5000000   0.0426512103
##   0.0    0.15777778  0.5000000   0.0426512103
##   0.0    0.17888889  0.5000000   0.0426512103
```

```
##   0.0    0.17888889  0.5000000   0.0426512103
##   0.0    0.20000000  0.5000000   0.0426512103
##   0.1    0.01000000  0.4350000   0.0039588902
##   0.1    0.03111111  0.4407143   0.0022780604
##   0.1    0.05222222  0.4585714   0.0223548246
##   0.1    0.07333333  0.4642857   0.0259622785
##   0.1    0.09444444  0.4778571   0.0400009478
##   0.1    0.11555556  0.4857143   0.0472975830
##   0.1    0.13666667  0.4900000   0.0485777250
##   0.1    0.15777778  0.4900000   0.0420169159
##   0.1    0.17888889  0.4935714   0.0417440461
##   0.1    0.20000000  0.5028571   0.0558828243
##   0.2    0.01000000  0.4385714   0.0086507359
##   0.2    0.03111111  0.4592857   0.0287362793
##   0.2    0.05222222  0.4700000   0.0347704686
##   0.2    0.07333333  0.4842857   0.0470047269
##   0.2    0.09444444  0.4935714   0.0537130423
##   0.2    0.11555556  0.4957143   0.0470304686
##   0.2    0.13666667  0.5107143   0.0643312778
##   0.2    0.15777778  0.5207143   0.0746612547
##   0.2    0.17888889  0.5285714   0.0850413220
##   0.2    0.20000000  0.5271429   0.0742306790
##   0.4    0.01000000  0.4400000   0.0130243162
##   0.4    0.03111111  0.4635714   0.0276307712
##   0.4    0.05222222  0.4857143   0.0425065381
##   0.4    0.07333333  0.5035714   0.0588468146
##   0.4    0.09444444  0.5214286   0.0780304788
##   0.4    0.11555556  0.5307143   0.0788302891
##   0.4    0.13666667  0.5271429   0.0600928599
##   0.4    0.15777778  0.5221429   0.0405221197
##   0.4    0.17888889  0.5235714   0.0342157530
##   0.4    0.20000000  0.5200000   0.0183594478
##   0.6    0.01000000  0.4421429   0.0121912593
##   0.6    0.03111111  0.4771429   0.0396760116
##   0.6    0.05222222  0.5035714   0.0604033238
##   0.6    0.07333333  0.5250000   0.0790243168
##   0.6    0.09444444  0.5264286   0.0620090630
##   0.6    0.11555556  0.5200000   0.0328365635
##   0.6    0.13666667  0.5185714   0.0159880354
##   0.6    0.15777778  0.5135714  -0.0040397651
##   0.6    0.17888889  0.5171429  -0.0008288785
##   0.6    0.20000000  0.5178571   0.0000000000
##
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were alpha = 0.4 and lambda
##  = 0.1155556.
```



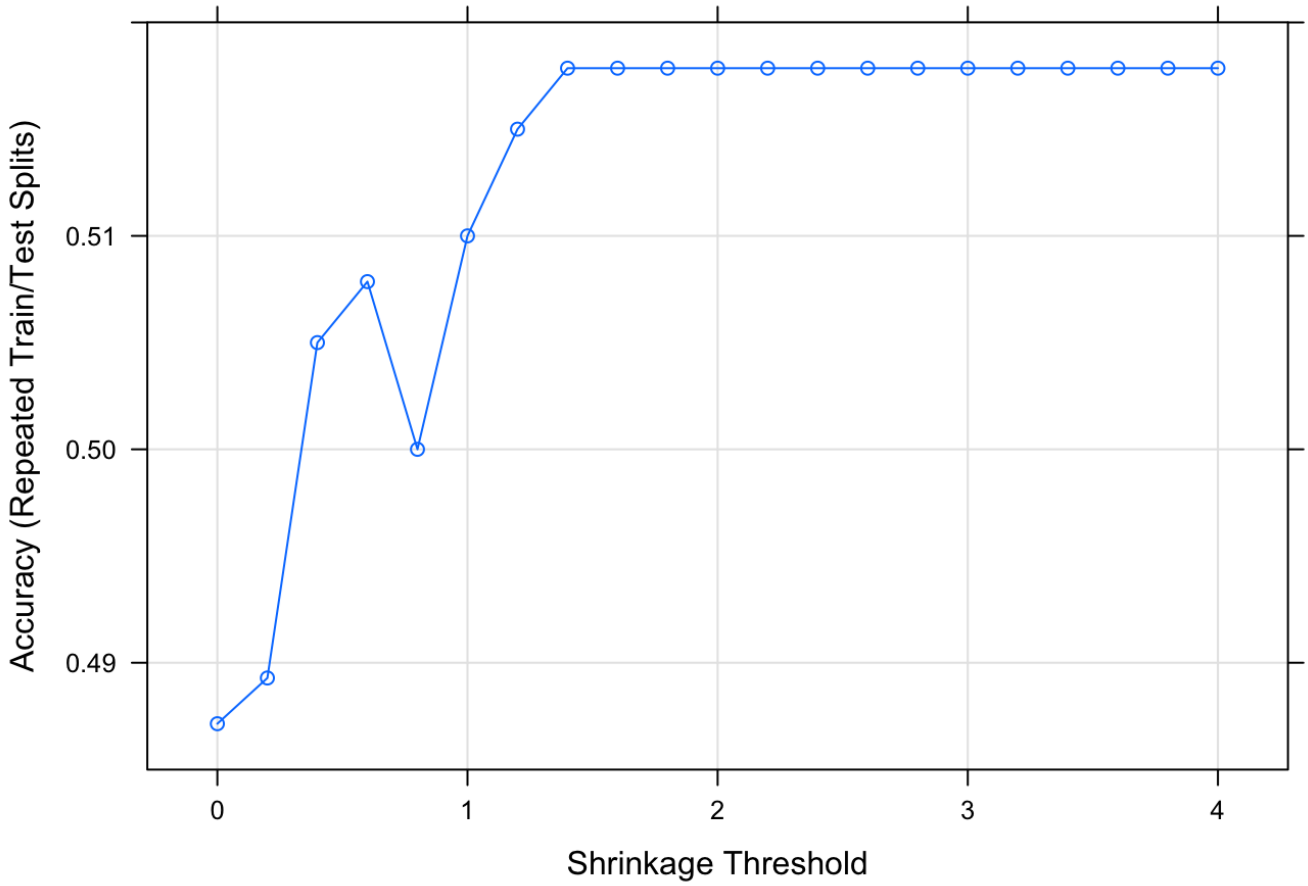

```
## [1] "Test set Accuracy:  0.5"
```

Thus the model has accuracy of 0.5 on the test set with alpha = 0.4 and lambda = 0.1155556.

Nearest shrunken Centroids

And finally, lets see the results of Nearest shrunken Centroids for the biological predictors

```
## Nearest Shrunken Centroids
##
## 225 samples
## 236 predictors
##   3 classes: 'Mild', 'None', 'Severe'
##
## Pre-processing: centered (236), scaled (236)
## Resampling: Repeated Train/Test Splits Estimated (25 reps, 75%)
## Summary of sample sizes: 169, 169, 169, 169, 169, 169, ...
## Resampling results across tuning parameters:
##
##   threshold  Accuracy   Kappa
##   0.0        0.4871429   0.104032980
##   0.2        0.4892857   0.103118264
##   0.4        0.5050000   0.107864435
##   0.6        0.5078571   0.083138981
##   0.8        0.5000000   0.020402765
##   1.0        0.5100000   0.003359881
##   1.2        0.5150000  -0.003629763
##   1.4        0.5178571   0.000000000
##   1.6        0.5178571   0.000000000
##   1.8        0.5178571   0.000000000
##   2.0        0.5178571   0.000000000
##   2.2        0.5178571   0.000000000
##   2.4        0.5178571   0.000000000
##   2.6        0.5178571   0.000000000
##   2.8        0.5178571   0.000000000
##   3.0        0.5178571   0.000000000
##   3.2        0.5178571   0.000000000
##   3.4        0.5178571   0.000000000
##   3.6        0.5178571   0.000000000
##   3.8        0.5178571   0.000000000
##   4.0        0.5178571   0.000000000
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was threshold = 1.4.
```




```
## [1] "Test set Accuracy:  0.5179"
```

Thus the model has accuracy of 0.5179 on the test set with threshold = 1.4.

Below table shows the compilation of the information regarding the above models all together.

Model	Parameter	Training Accuracy	Testing Accuracy
Logistic Regression	decay = 0.1	0.4050	0.5893
Linear Discriminant Analysis	None	0.3514	0.3929
Partial Least Squares Discriminant Analysis	ncomp = 1	0.5136	0.4643
Penalized Methods	alpha = 0.4 & lambda = 0.1155556	0.5307	0.5
Nearest shrunken Centroids	threshold = 1.4	0.5179	0.5179

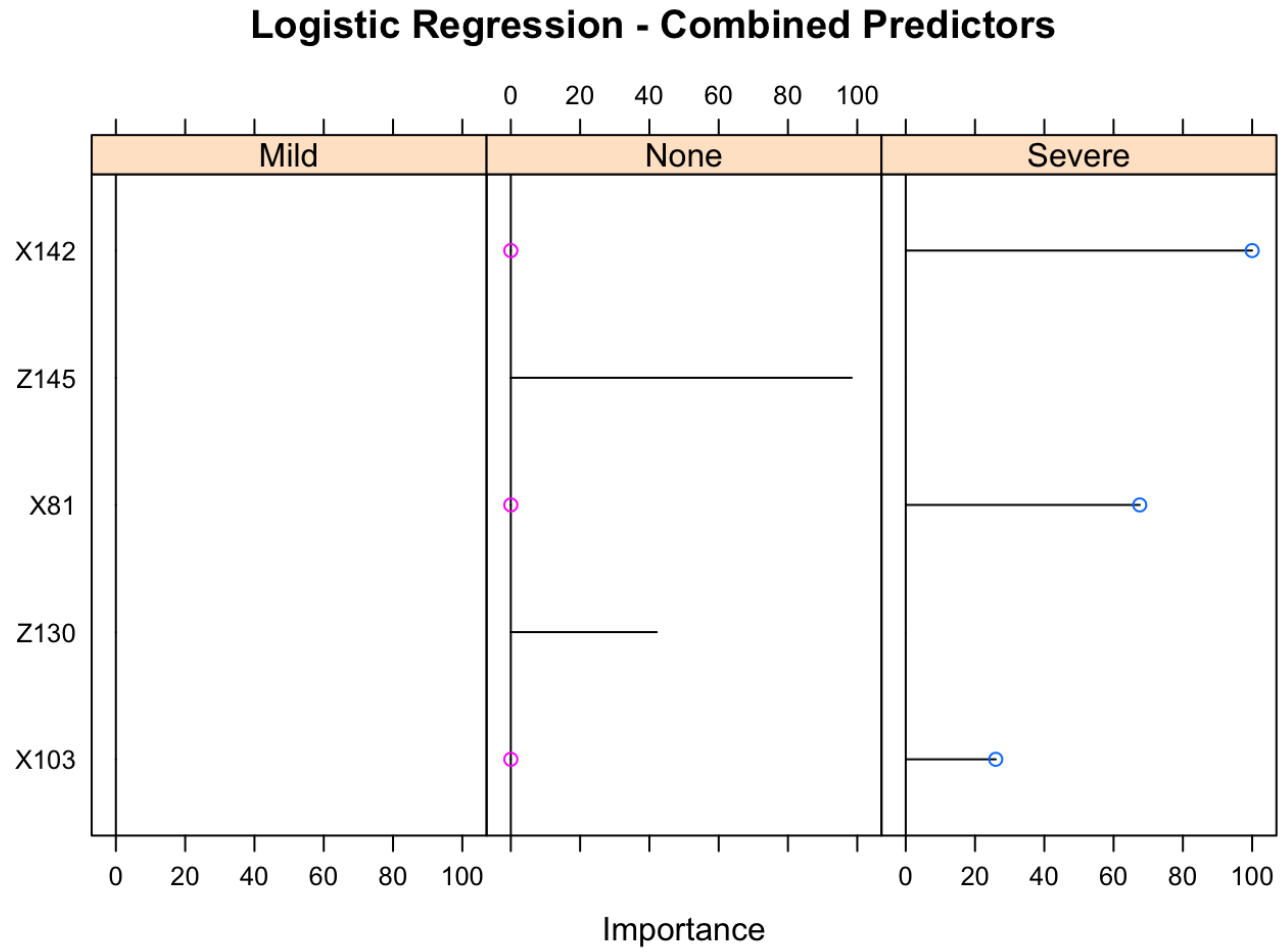
From the above table we can see that Logistic Regression model has better predictive ability with test set accuracy of 0.5893 for the combined predictors of both chemical and biological predictors.

But however it does not perform better than other model from above predictor set especially Linear Discriminant Analysis from the biological predictor which has the accuracy of 0.6607.

Important Predictors

Lets see the top 5 important predictors of the combined predcitors for the best model in this set that is logistic regression.

```
##       Mild     None    Severe
## X142     0  0.00000 100.00000
## Z145     0 98.38722   0.00000
## X81      0  0.00000  67.54500
## Z130     0 42.12206   0.00000
## X103     0  0.00000  25.96863
```



Above we have the important predictors for the combined predictors model.

On camparing with the previous models from the chemical only and biological only predictors, the models on the combined predictors performs bit less. This might be due to the biological predcitors addition because we saw before that the biological predictors models doesn't have huge accuracies values for hepatic toxcity.

Section (e)

I won't be recommending any of this model to be used in real time because the highest accuracy achived from developing all this models is 0.6607 which is not huge number that could be recommended to be used with real world applications. This being determining the hepatic toxicity that goes into the sector of health care, I don't think that 0.6607 accuracy is enough for recommending using to predict compounds.

Question 2

Lets first load the data in here and check whether I have the data porperly. I am just printing the first few elements so that it wont create cluttering in my submission.

```
##   Palmitic Stearic Oleic Linoleic Linolenic Eicosanoic Eicosenoic
## 1      9.7     5.2  31.0     52.7       0.4        0.4        0.1
## 2     11.1     5.0  32.9     49.8       0.3        0.4        0.1
## 3     11.5     5.2  35.0     47.2       0.2        0.4        0.1
## 4     10.0     4.8  30.4     53.5       0.3        0.4        0.1
## 5     12.2     5.0  31.1     50.5       0.3        0.4        0.1
## 6      9.8     4.2  43.0     39.2       2.4        0.4        0.5
```

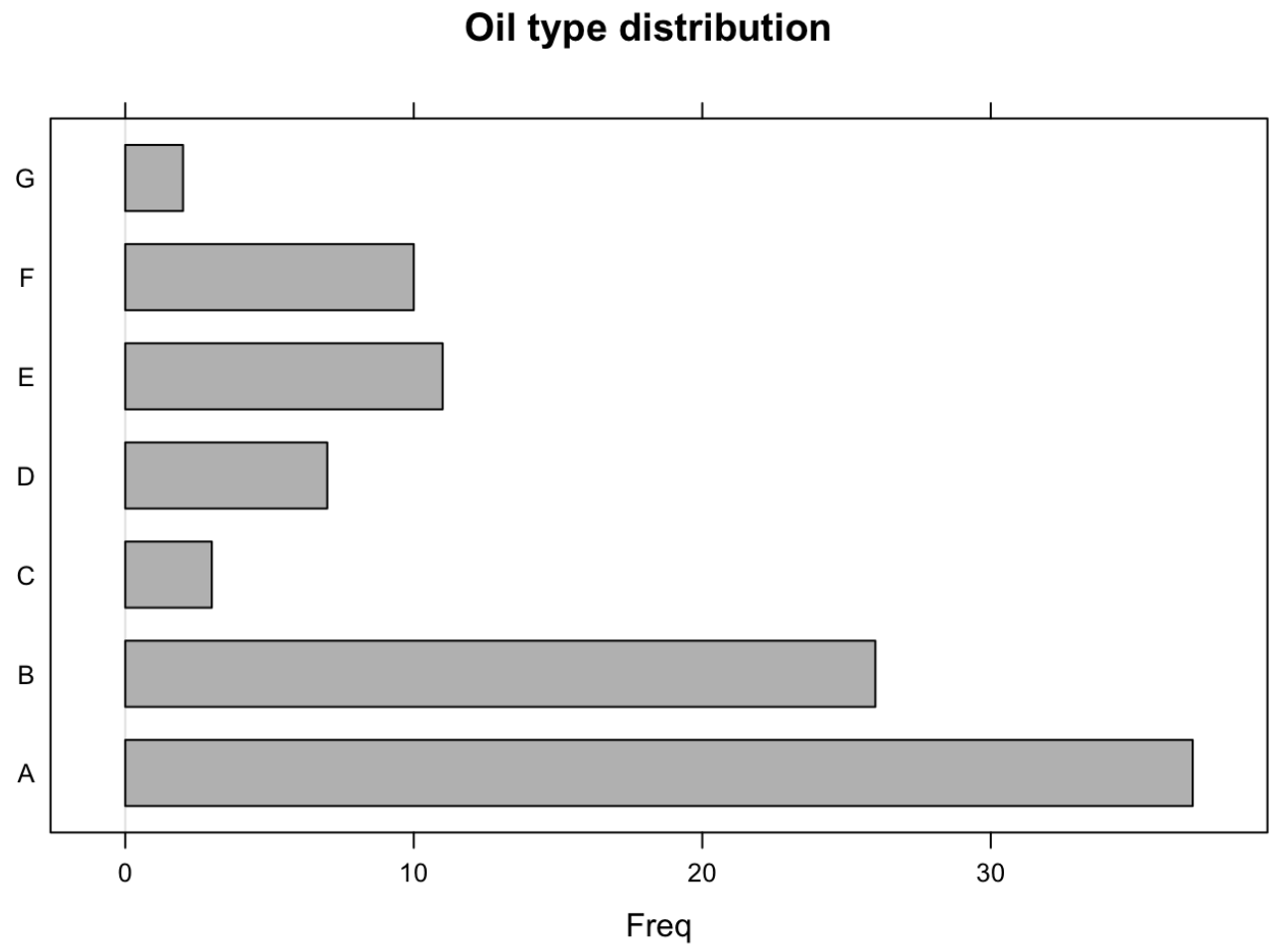
```
## [1] A A A A A A
## Levels: A B C D E F G
```

Thus we see above that we have the data loaded properly.

Section (a)

Lets have look at the distributions of oil type which is our responding variable

```
## oilType
##  A  B  C  D  E  F  G
## 37 26  3  7 11 10  2
```



From the above table and barchart, we can see that the frequency of our responding variable is not distributed mutually. Thus there is no equal distribution of the classes here. I am going to check for near zero variance and remove those predictors which have near zero variance.

```
## [1] "Biological : Reducing the 0 zero variance columns from 7 predcitors (fracti
on =  0.000000)"
```

Thus we can see that there is no near zero variance predictors and we will be using all the predictors for buliding the models. Since we have very less number of samples I am not going to split the data into train and test.

Section (b)

Since we are targeting about how accurate our model is and also how well it is identifying all the classes in the responding variable, It will be more appropriate to use accuracy as the classification statistics for this exercise.

Section (c)

Lets bulid the models, make predcition and calculate the accuracy to decide which model is having the best and worst performance etc.

Logistic Regression

For the logistic regression, I have removed the highly correlated predcitor out of the predictors set with cut off threshold of 0.9. Lets view the results of this now.

```
## Penalized Multinomial Regression
##
## 96 samples
##  6 predictor
##  7 classes: 'A', 'B', 'C', 'D', 'E', 'F', 'G'
##
## Pre-processing: centered (6), scaled (6)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 96, 96, 96, 96, 96, 96, ...
## Resampling results across tuning parameters:
##
##   decay  Accuracy   Kappa
##   0e+00  0.9435810  0.9235240
##   1e-04  0.9551933  0.9383216
##   1e-01  0.9425688  0.9219081
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was decay = 1e-04.
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  A  B  C  D  E  F  G
##          A 37  0  0  0  0  0  0
##          B  0 26  0  0  0  0  0
##          C  0  0  3  0  0  0  0
##          D  0  0  0  7  0  0  0
##          E  0  0  0  0 11  0  0
##          F  0  0  0  0  0 10  0
##          G  0  0  0  0  0  0  2
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9623, 1)
##     No Information Rate : 0.3854
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E Class: F
## Sensitivity            1.0000   1.0000  1.00000  1.00000   1.0000   1.0000
## Specificity            1.0000   1.0000  1.00000  1.00000   1.0000   1.0000
## Pos Pred Value         1.0000   1.0000  1.00000  1.00000   1.0000   1.0000
## Neg Pred Value         1.0000   1.0000  1.00000  1.00000   1.0000   1.0000
## Prevalence             0.3854   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Rate         0.3854   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Prevalence   0.3854   0.2708  0.03125  0.07292   0.1146   0.1042
## Balanced Accuracy      1.0000   1.0000  1.00000  1.00000   1.0000   1.0000
##                      Class: G
## Sensitivity           1.00000
## Specificity           1.00000
## Pos Pred Value        1.00000
## Neg Pred Value        1.00000
## Prevalence            0.02083
## Detection Rate        0.02083
## Detection Prevalence  0.02083
## Balanced Accuracy     1.00000
```

```
## [1] "Accuracy on the whole training set:  1"
```

Thus the model has a accuracy of 1 on the whole training set with decay = 1e-04.

Linear Discriminant Analysis

Similar to logistic regression, I have removed the highly correlated values out of the predictors and then passed on to the model. Since there are no tuning paramter for this model, I am not showing any plots. Lets see the results

```
## Linear Discriminant Analysis
##
## 96 samples
##  6 predictor
##  7 classes: 'A', 'B', 'C', 'D', 'E', 'F', 'G'
##
## Pre-processing: centered (6), scaled (6)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 96, 96, 96, 96, 96, 96, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9558989  0.9397047
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  A  B  C  D  E  F  G
##          A 34  0  0  0  0  0  0
##          B  2 26  0  0  0  0  0
##          C  0  0  3  0  0  0  0
##          D  0  0  0  7  0  0  0
##          E  1  0  0  0 11  0  0
##          F  0  0  0  0  0 10  0
##          G  0  0  0  0  0  0  2
##
## Overall Statistics
##
##                Accuracy : 0.9688
##                  95% CI : (0.9114, 0.9935)
##     No Information Rate : 0.3854
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9585
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E Class: F
## Sensitivity            0.9189   1.0000  1.00000  1.00000   1.0000   1.0000
## Specificity            1.0000   0.9714  1.00000  1.00000   0.9882   1.0000
## Pos Pred Value         1.0000   0.9286  1.00000  1.00000   0.9167   1.0000
## Neg Pred Value         0.9516   1.0000  1.00000  1.00000   1.0000   1.0000
## Prevalence             0.3854   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Rate         0.3542   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Prevalence   0.3542   0.2917  0.03125  0.07292   0.1250   0.1042
## Balanced Accuracy      0.9595   0.9857  1.00000  1.00000   0.9941   1.0000
##                      Class: G
## Sensitivity           1.00000
## Specificity           1.00000
## Pos Pred Value        1.00000
## Neg Pred Value        1.00000
## Prevalence            0.02083
## Detection Rate        0.02083
## Detection Prevalence  0.02083
## Balanced Accuracy     1.00000
```

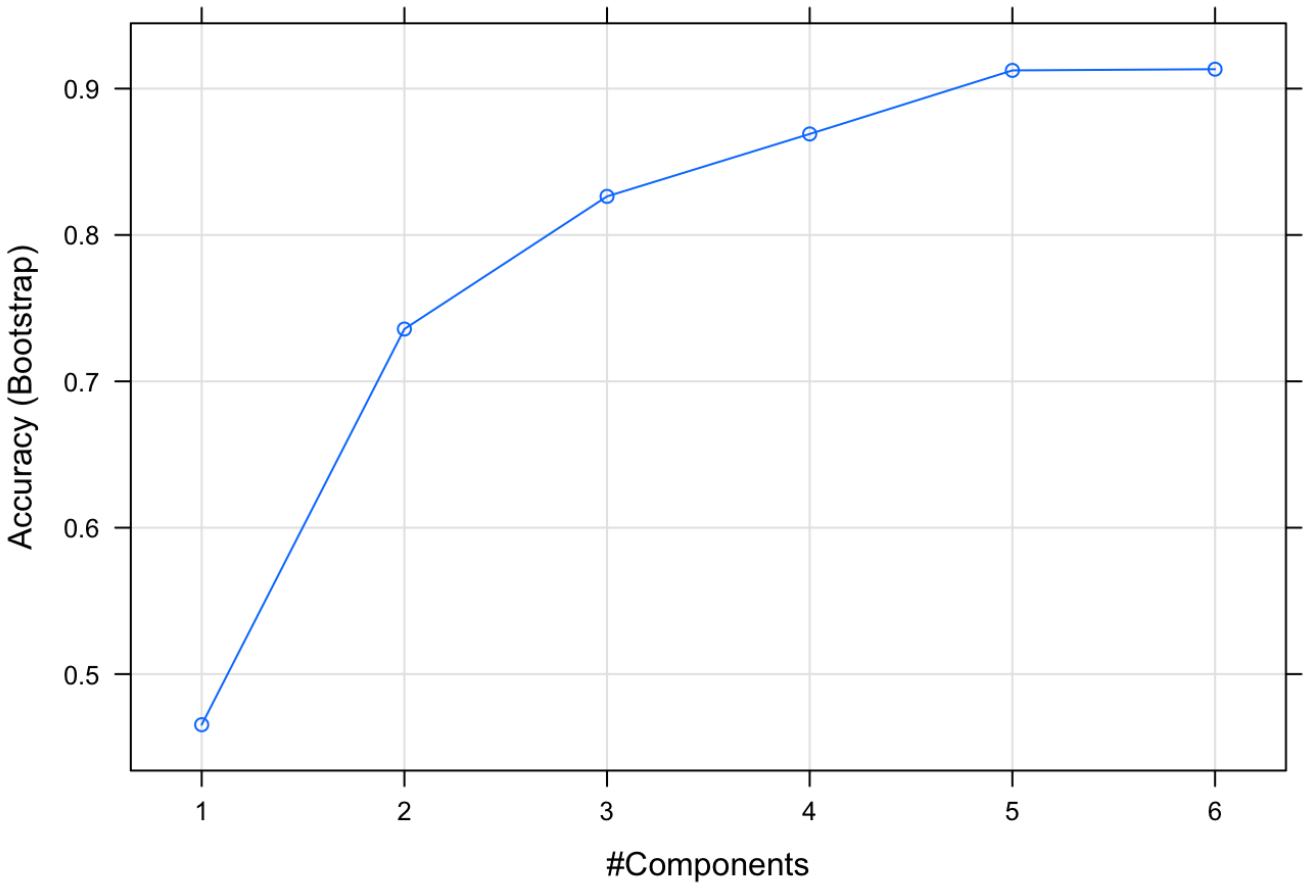
```
## [1] "Accuracy on the whole training set:  0.9688"
```

Thus the model has a accuracy of 0.9688 on the whole training set.

Partial Least Squares Discriminant Analysis

Let see what is the results for partial least squares discriminant model

```
## Partial Least Squares
##
## 96 samples
##  7 predictor
##  7 classes: 'A', 'B', 'C', 'D', 'E', 'F', 'G'
##
## Pre-processing: centered (7), scaled (7)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 96, 96, 96, 96, 96, 96, ...
## Resampling results across tuning parameters:
##
##   ncomp  Accuracy   Kappa
##   1      0.4654105  0.2274574
##   2      0.7357312  0.6142652
##   3      0.8263763  0.7551163
##   4      0.8690209  0.8169309
##   5      0.9124506  0.8794215
##   6      0.9132750  0.8807045
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was ncomp = 6.
```



```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  A  B  C  D  E  F  G
##          A 35  0  0  0  0  0  0
##          B  2 26  0  0  0  0  1
##          C  0  0  3  0  0  0  0
##          D  0  0  0  7  0  0  0
##          E  0  0  0  0 11  0  1
##          F  0  0  0  0  0 10  0
##          G  0  0  0  0  0  0  0
##
## Overall Statistics
##
##                Accuracy : 0.9583
##                  95% CI : (0.8967, 0.9885)
##     No Information Rate : 0.3854
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9442
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E Class: F
## Sensitivity            0.9459   1.0000  1.00000  1.00000   1.0000   1.0000
## Specificity            1.0000   0.9571  1.00000  1.00000   0.9882   1.0000
## Pos Pred Value         1.0000   0.8966  1.00000  1.00000   0.9167   1.0000
## Neg Pred Value         0.9672   1.0000  1.00000  1.00000   1.0000   1.0000
## Prevalence             0.3854   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Rate         0.3646   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Prevalence   0.3646   0.3021  0.03125  0.07292   0.1250   0.1042
## Balanced Accuracy      0.9730   0.9786  1.00000  1.00000   0.9941   1.0000
##                      Class: G
## Sensitivity           0.00000
## Specificity           1.00000
## Pos Pred Value            NaN
## Neg Pred Value        0.97917
## Prevalence            0.02083
## Detection Rate        0.00000
## Detection Prevalence  0.00000
## Balanced Accuracy     0.50000
```

```
## [1] "Accuracy on the whole training set:  0.9583"
```

Thus the model has a accuracy of 0.9583 on the whole training set with six components.

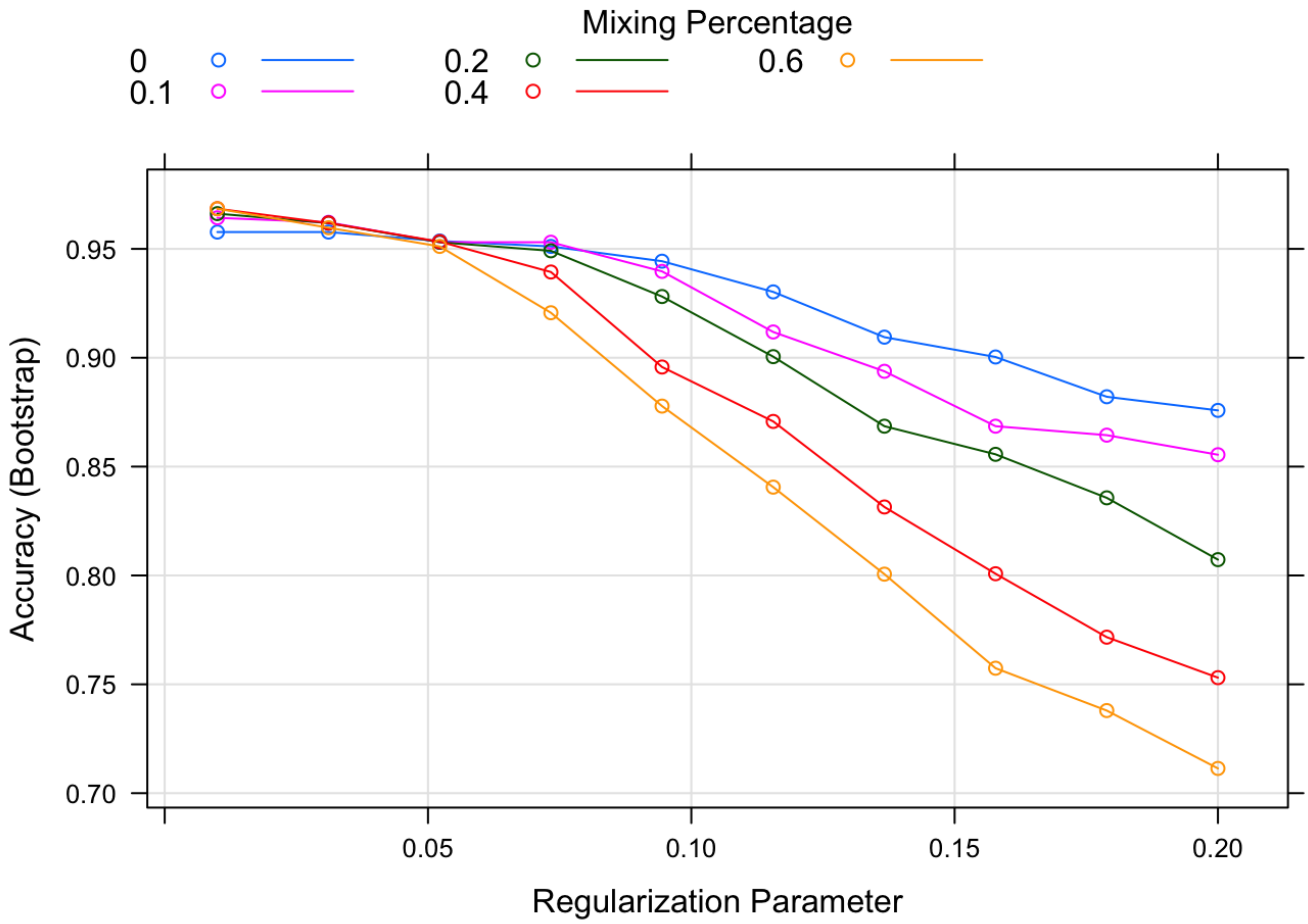
Penalized Methods

Let see whst is the result for penalized model

```
## glmnet
##
## 96 samples
##  7 predictor
```

```
##  7 classes: 'A', 'B', 'C', 'D', 'E', 'F', 'G'
##
## Pre-processing: centered (7), scaled (7)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 96, 96, 96, 96, 96, 96, ...
## Resampling results across tuning parameters:
##
##   alpha  lambda      Accuracy   Kappa
##   0.0    0.01000000  0.9577325  0.9419257
##   0.0    0.03111111  0.9577325  0.9419257
##   0.0    0.05222222  0.9536117  0.9364238
##   0.0    0.07333333  0.9511303  0.9331706
##   0.0    0.09444444  0.9443441  0.9234297
##   0.0    0.11555556  0.9302419  0.9044037
##   0.0    0.13666667  0.9094547  0.8750760
##   0.0    0.15777778  0.9003414  0.8614004
##   0.0    0.17888889  0.8820799  0.8348280
##   0.0    0.20000000  0.8758154  0.8255880
##   0.1    0.01000000  0.9642718  0.9508526
##   0.1    0.03111111  0.9621928  0.9479806
##   0.1    0.05222222  0.9530533  0.9357606
##   0.1    0.07333333  0.9530533  0.9357606
##   0.1    0.09444444  0.9396649  0.9169376
##   0.1    0.11555556  0.9118348  0.8784229
##   0.1    0.13666667  0.8937749  0.8529728
##   0.1    0.15777778  0.8685619  0.8151925
##   0.1    0.17888889  0.8644410  0.8091165
##   0.1    0.20000000  0.8554520  0.7955853
##   0.2    0.01000000  0.9661949  0.9534167
##   0.2    0.03111111  0.9618534  0.9473662
##   0.2    0.05222222  0.9530533  0.9357606
##   0.2    0.07333333  0.9491060  0.9303311
##   0.2    0.09444444  0.9281087  0.9016057
##   0.2    0.11555556  0.9004976  0.8627174
##   0.2    0.13666667  0.8685619  0.8151925
##   0.2    0.15777778  0.8556394  0.7960870
##   0.2    0.17888889  0.8356538  0.7660707
##   0.2    0.20000000  0.8072694  0.7256667
##   0.4    0.01000000  0.9684573  0.9564757
##   0.4    0.03111111  0.9618534  0.9473662
##   0.4    0.05222222  0.9533281  0.9360470
##   0.4    0.07333333  0.9393813  0.9168904
##   0.4    0.09444444  0.8957357  0.8556526
##   0.4    0.11555556  0.8707510  0.8185971
##   0.4    0.13666667  0.8314560  0.7630986
##   0.4    0.15777778  0.8007911  0.7134266
##   0.4    0.17888889  0.7716565  0.6684281
##   0.4    0.20000000  0.7530649  0.6383755
##   0.6    0.01000000  0.9684573  0.9564757
##   0.6    0.03111111  0.9597114  0.9446785
##   0.6    0.05222222  0.9511303  0.9331798
##   0.6    0.07333333  0.9206645  0.8916223
##   0.6    0.09444444  0.8777870  0.8306255
##   0.6    0.11555556  0.8405679  0.7760320
##   0.6    0.13666667  0.8006002  0.7130987
##   0.6    0.15777778  0.7573994  0.6460251
##   0.6    0.17888889  0.7379493  0.6132371
```

```
##   0.6    0.20000000  0.7113694  0.5693025
##
## Accuracy was used to select the optimal model using  the largest value.
## The final values used for the model were alpha = 0.4 and lambda = 0.01.
```



```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  A  B  C  D  E  F  G
##          A 36  0  0  0  0  0  0
##          B  1 26  0  0  0  0  0
##          C  0  0  3  0  0  0  0
##          D  0  0  0  7  0  0  0
##          E  0  0  0  0 11  0  0
##          F  0  0  0  0  0 10  0
##          G  0  0  0  0  0  0  2
##
## Overall Statistics
##
##                Accuracy : 0.9896
##                  95% CI : (0.9433, 0.9997)
##     No Information Rate : 0.3854
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9861
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E Class: F
## Sensitivity            0.9730   1.0000  1.00000  1.00000   1.0000   1.0000
## Specificity            1.0000   0.9857  1.00000  1.00000   1.0000   1.0000
## Pos Pred Value         1.0000   0.9630  1.00000  1.00000   1.0000   1.0000
## Neg Pred Value         0.9833   1.0000  1.00000  1.00000   1.0000   1.0000
## Prevalence             0.3854   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Rate         0.3750   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Prevalence   0.3750   0.2812  0.03125  0.07292   0.1146   0.1042
## Balanced Accuracy      0.9865   0.9929  1.00000  1.00000   1.0000   1.0000
##                      Class: G
## Sensitivity           1.00000
## Specificity           1.00000
## Pos Pred Value        1.00000
## Neg Pred Value        1.00000
## Prevalence            0.02083
## Detection Rate        0.02083
## Detection Prevalence  0.02083
## Balanced Accuracy     1.00000
```

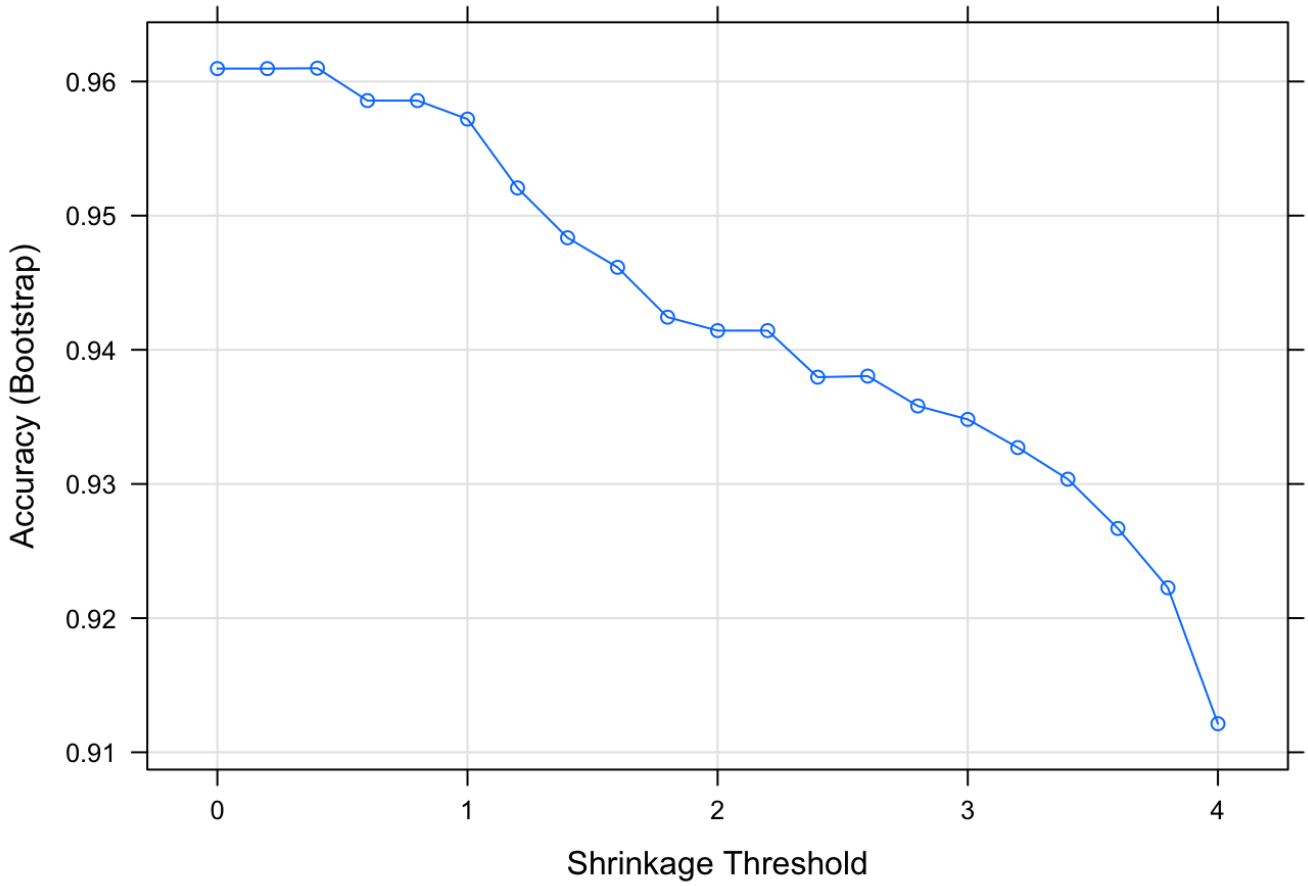
```
## [1] "Accuracy on the whole training set:  0.9896"
```

Thus the model has a accuracy of 0.9896 on the whole training set with alpha = 0.4 and lambda = 0.01.

Nearest shrunken Centroids

And finally, lets see the results of Nearest shrunken Centroids for the biological predictors

```
## Nearest Shrunken Centroids
##
## 96 samples
##  7 predictor
##  7 classes: 'A', 'B', 'C', 'D', 'E', 'F', 'G'
##
## Pre-processing: centered (7), scaled (7)
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 96, 96, 96, 96, 96, 96, ...
## Resampling results across tuning parameters:
##
##   threshold  Accuracy   Kappa
##   0.0        0.9609612  0.9467681
##   0.2        0.9609612  0.9467681
##   0.4        0.9609913  0.9467295
##   0.6        0.9585769  0.9434192
##   0.8        0.9585769  0.9434192
##   1.0        0.9571975  0.9415578
##   1.2        0.9520675  0.9345438
##   1.4        0.9483507  0.9294215
##   1.6        0.9461486  0.9264873
##   1.8        0.9424318  0.9215103
##   2.0        0.9414318  0.9201769
##   2.2        0.9414318  0.9201769
##   2.4        0.9379622  0.9156424
##   2.6        0.9380432  0.9157109
##   2.8        0.9358141  0.9127104
##   3.0        0.9348141  0.9113547
##   3.2        0.9327122  0.9085780
##   3.4        0.9303572  0.9055031
##   3.6        0.9266834  0.9005310
##   3.8        0.9222637  0.8946302
##   4.0        0.9121297  0.8813096
##
## Accuracy was used to select the optimal model using  the largest value.
## The final value used for the model was threshold = 0.4.
```


Accuracy (Bootstrap)
0.96
0.95
0.94
0.93
0.92
0.91
0
1
2
3
4
Shrinkage Threshold

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  A  B  C  D  E  F  G
##          A 35  0  0  0  0  0  0
##          B  2 26  0  0  0  0  0
##          C  0  0  3  0  0  0  0
##          D  0  0  0  7  0  0  0
##          E  0  0  0  0 11  0  0
##          F  0  0  0  0  0 10  0
##          G  0  0  0  0  0  0  2
##
## Overall Statistics
##
##                Accuracy : 0.9792
##                  95% CI : (0.9268, 0.9975)
##     No Information Rate : 0.3854
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9722
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: A Class: B Class: C Class: D Class: E Class: F
## Sensitivity            0.9459   1.0000  1.00000  1.00000   1.0000   1.0000
## Specificity            1.0000   0.9714  1.00000  1.00000   1.0000   1.0000
## Pos Pred Value         1.0000   0.9286  1.00000  1.00000   1.0000   1.0000
## Neg Pred Value         0.9672   1.0000  1.00000  1.00000   1.0000   1.0000
## Prevalence             0.3854   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Rate         0.3646   0.2708  0.03125  0.07292   0.1146   0.1042
## Detection Prevalence   0.3646   0.2917  0.03125  0.07292   0.1146   0.1042
## Balanced Accuracy      0.9730   0.9857  1.00000  1.00000   1.0000   1.0000
##                      Class: G
## Sensitivity           1.00000
## Specificity           1.00000
## Pos Pred Value        1.00000
## Neg Pred Value        1.00000
## Prevalence            0.02083
## Detection Rate        0.02083
## Detection Prevalence  0.02083
## Balanced Accuracy     1.00000
```

```
## [1] "Accuracy on the whole training set:  0.9792"
```

Thus the model has a accuracy of 0.9792 on the whole training set with threshold = 0.4.

Below table shows the compilation of the information regarding the above models all together.

Model	Parameter	Best Resampling Accuracy	Accuracy on whole data
Logistic Regression	decay = 1e-04	0.9426	1
Linear Discriminant Analysis	None	0.9559	0.9688

Partial Least Squares Discriminant Analysis	ncomp = 6	0.9133	0.9583
Model	**Parameter**	**Best Resampling Accuracy**	**Accuracy on whole data**
Penalized Methods	alpha = 0.4 & lambda = 0.01	0.9685	0.9896
Nearest shrunken Centroids	threshold = 0.4	0.961	0.9792

From the above table we can see that Logistic Regression model has better predictive ability on the whole dataset with the value of 1 as accuracy. Let see the confusion matrix of the logistic regression model to do further analysis.

```
##           Reference
## Prediction  A  B  C  D  E  F  G
##          A 37  0  0  0  0  0  0
##          B  0 26  0  0  0  0  0
##          C  0  0  3  0  0  0  0
##          D  0  0  0  7  0  0  0
##          E  0  0  0  0 11  0  0
##          F  0  0  0  0  0 10  0
##          G  0  0  0  0  0  0  2
```

We can see the logistic regression predicts all the oil type accurately. Thus all the classes can be considered to be most predicting classes in this case. Since there is no missclassification in the confusion matrix, this model does have any least predicting classes.

*** End of Solution ***

Appendix - Coding

installing the packages

installNewPackage <- function(packageName) {

```
    if(packageName  %in% rownames(installed.packages()) == FALSE)

    {

            install.packages(packageName, repos = "http://cran.us.r-project.org", d
ependencies=TRUE)

    }
```

}

installNewPackage(“caret”)

installNewPackage(“AppliedPredictiveModeling”)

installNewPackage(“pROC”)

installNewPackage(“Metrics”)

installNewPackage(“ModelMetrics”)

library(caret)

library(AppliedPredictiveModeling)

library(pROC)

library(Metrics)

library(ModelMetrics)

Question 1

data(hepatic)

head(injury)

print(head(bio[, 1:5]))

print(head(chem[, 1:5]))

Section (a)

table(injury)

barchart(injury, main = “Injury distribution”, col = “gray”)

Biological

nzv <- nearZeroVar(bio)

bio_nzv <- bio[, -nzv]

print(sprintf(“Biological : Reducing the %d zero variance columns from %d predcitors (fraction =%10.6f)”, length(nzv), dim(bio)[2], length(nzv)/dim(bio)[2]))

Chemical

nzv <- nearZeroVar(chem)

chem_nzv <- chem[, -nzv]

print(sprintf(“Chemical : Reducing the %d zero variance columns from %d predcitors (fraction =%10.6f)”, length(nzv), dim(chem)[2], length(nzv)/dim(chem)[2]))

Combined

bio_chem <- cbind(bio, chem)

nzv <- nearZeroVar(bio_chem)

bio_chem_nzv <- bio_chem[, -nzv]

print(sprintf(“Combined : Reducing the %d zero variance columns from %d predcitors (fraction =%10.6f)”,

```
          length(nzv), dim(bio_chem)[2], length(nzv)/dim(bio_chem)[2]))
```

Set the seed

set.seed(1)

Split train and test set for all three set of predictors

cv_index <- createDataPartition(injury, p = 0.8, list = FALSE)

bio_train <- bio_nzv[cv_index,]

bio_test <- bio_nzv[-cv_index,]

chem_train <- chem_nzv[cv_index,]

chem_test <- chem_nzv[-cv_index,]

bio_chem_train <- bio_chem_nzv[cv_index,]

bio_chem_test <- bio_chem_nzv[-cv_index,]

injury_train <- injury[cv_index]

injury_test <- injury[-cv_index]

print("All three set of predictors and responding variables train and test split is completed using stratified sampling")

Custom analysis function

analysis <- function(classifier, X, y) {

```
\# Compute the prediction probablities for each classes

probs <- predict(classifier, X, type = "prob")

\# Make the predcit by taking max probabiltiy class

pred <- as.factor(colnames(probs)[apply(probs, 1, which.max)])

\# Get the ROC and AUC

acc <- accuracy(actual = as.numeric(y), predicted = as.numeric(pred))

return (list(classifier = classifier, acc = acc))
```

}

Custom function to build all the models

build_models <- function(X_train, y_train, X_test, y_test, seed_value=1) {

```
\# Calling MUlticlass summary for train control with Class prob set to TRUE

ctrl <- trainControl(method = "LGOCV")

\# Removing the highly correlated predictors out for

\# logistic and LDA models

too_high <- findCorrelation(cor(X_train), 0.9)

X_train_cor <- as.data.frame(X_train[, -too_high])

X_test_cor <- as.data.frame(X_test[, -too_high])

\# Logistic Regression Model:

set.seed(seed_value)

glm.classifier <- train(X_train_cor, y_train, method = "multinom", preProcess = c("
center", "scale"), metric = "Accuracy", trControl = ctrl)
```

```
glm <- analysis(glm.classifier, X_test_cor, y_test)

\# Linear Discriminant Analysis:

set.seed(seed_value)

lda.classifier <- train(X_train_cor, y_train, method = "lda", preProc = c("center",
"scale"), metric = "Accuracy", trControl = ctrl)

lda <- analysis(lda.classifier, X_test_cor, y_test)

\# Partial Least Squares Discriminant Analysis

set.seed(seed_value)

plsda.classifier <- train(X_train, y_train, method = "pls", tuneGrid = expand.grid(
.ncomp=1:10),

                          preProc = c("center","scale"), metric = "Accuracy", trCon
trol = ctrl)

plsda <- analysis(plsda.classifier, X_test, y_test)

\# Penalized Methods

glmnGrid <- expand.grid(.alpha = c(0, .1, .2, .4, .6), .lambda = seq(.01, .2, lengt
h = 10))

set.seed(seed_value)

glmnet.classifier <- train(X_train, y_train, method = "glmnet", tuneGrid = glmnGrid
, preProc = c("center","scale"),

                           metric = "Accuracy", trControl = ctrl)

glmnet <- analysis(glmnet.classifier, X_test, y_test)

\# Nearest shrunken Centroids:

nscGrid <- data.frame(.threshold = seq(0,4, by=0.2))

set.seed(seed_value)

nsc.classifier <- train(X_train, y_train, method = "pam", tuneGrid = nscGrid, prePr
oc = c("center","scale"), metric = "Accuracy",

                        trControl = ctrl)

nsc <- analysis(nsc.classifier, X_test, y_test)

result <- list(glm = glm, lda = lda, plsda = plsda, glmnet = glmnet, nsc = nsc )

return (result)
```

}

Check the file exists and load to variables # else bulid and store the model

if(file.exists(“bio_results.rds”)) {

```
bio_results <- readRDS("bio_results.rds")
```

} else {

```
bio_results <- build_models(bio_train, injury_train, bio_test, injury_test)

saveRDS(bio_results, "bio_results.rds")
```

}

Logistic Regression

bio_resultsglmclassifier

plot(bio_resultsglmclassifier)

print(paste(“Test set Accuracy:”, round(bio_resultsglmacc, 4)))

Linear Discriminant Analysis

bio_resultsldaclassifier

print(paste(“Test set Accuracy:”, round(bio_resultsldaacc, 4)))

Partial Least Squares Discriminant Analysis

bio_results$plsda$classifier

plot(bio_results$plsda$classifier)

print(paste(“Test set Accuracy:”, round(bio_results$plsda$acc, 4)))

Penalized Methods

bio_results$glmnet$classifier

plot(bio_results$glmnet$classifier)

print(paste(“Test set Accuracy:”, round(bio_results$glmnet$acc, 4)))

Nearest shrunken Centroids

bio_resultsnscclassifier

plot(bio_resultsnscclassifier)

print(paste(“Test set Accuracy:”, round(bio_resultsnscacc, 4)))

Chemical Predcitor

Check the file exists and load to variables # else bulid and store the model

if(file.exists(“chem_results.rds”)) {

```
chem_results <- readRDS("chem_results.rds")
```

} else {

```
chem_results <- build_models(chem_train, injury_train, chem_test, injury_test)

saveRDS(chem_results, "chem_results.rds")
```

}

Logistic Regression

chem_resultsglmclassifier

plot(chem_resultsglmclassifier)

print(paste("Test set Accuracy:", round(chem_resultsglmacc, 4)))

Linear Discriminant Analysis

chem_resultsldaclassifier

print(paste("Test set Accuracy:", round(chem_resultsldaacc, 4)))

Partial Least Squares Discriminant Analysis

chem_results$plsda$classifier

plot(chem_results$plsda$classifier)

print(paste("Test set Accuracy:", round(chem_results$plsda$acc, 4)))

Penalized Methods

chem_results$glmnet$classifier

plot(chem_results$glmnet$classifier)

print(paste("Test set Accuracy:", round(chem_results$glmnet$acc, 4)))

Nearest shrunken Centroids

chem_resultsnscclassifier

plot(chem_resultsnscclassifier)

print(paste("Test set Accuracy:", round(chem_resultsnscacc, 4)))

Section (d)

Biological Predictors

bio_imp <- varImp(bio_resultsldaclassifier, scale = FALSE, top = 5)

plot(bio_imp, top = 5, main = "LDA - Biological Predictors")

Chemical Predictors

chem_imp <- varImp(chem_resultsldaclassifier)

plot(chem_imp, top = 5, main = “PLSDA - Chemical Predictors”)

Section (e)

Combined Predictors

Check the file exists and load to variables

else bulid and store the model

if(file.exists(“bio_chem_results.rds”)) {

```
bio_chem_results <- readRDS("bio_chem_results.rds")
```

} else {

```
bio_chem_results <- build_models(bio_chem_train, injury_train, bio_chem_test, injur
y_test)

saveRDS(bio_chem_results, "bio_chem_results.rds")
```

}

Logistic Regression

bio_chem_resultsglmclassifier

plot(bio_chem_resultsglmclassifier)

print(paste(“Test set Accuracy:”, round(bio_chem_resultsglmacc, 4)))

Linear Discriminant Analysis

bio_chem_resultsldaclassifier

print(paste(“Test set Accuracy:”, round(bio_chem_resultsldaacc, 4)))

Partial Least Squares Discriminant Analysis

bio_chem_results$plsda$classifier

plot(bio_chem_results$plsda$classifier)

print(paste(“Test set Accuracy:”, round(bio_chem_results$plsda$acc, 4)))

Penalized Methods

bio_chem_results$glmnet$classifier

plot(bio_chem_results$glmnet$classifier)

print(paste(“Test set Accuracy:”, round(bio_chem_results$glmnet$acc, 4)))

Nearest shrunken Centroids

bio_chem_resultsnscclassifier

plot(bio_chem_resultsnscclassifier)

print(paste("Test set Accuracy:", round(bio_chem_resultsnscacc, 4)))

Important Predictors

bio_chem_imp <- varImp(bio_chem_resultsnscclassifier)

plot(bio_chem_imp, top = 5, main = "Logistic Regression - Combined Predictors")

Question 2

data(oil)

head(fattyAcids)

head(oilType)

Section (a)

table(oilType)

barchart(oilType, main = "Oil type distribution", col = "gray")

nzv <- nearZeroVar(fattyAcids)

print(sprintf("Biological : Reducing the %d zero variance columns from %d predcitors (fraction =%10.6f)", length(nzv), dim(fattyAcids)[2],

```
          length(nzv)/dim(fattyAcids)[2]))
```

Custom accuracy analysis function

analysis_acc <- function(classifier, X, y) {

```
\# Make the predictions

pred <- predict(classifier, X)

\# Compute the confusion matrix

cm <- caret::confusionMatrix(data = pred, reference = y)

acc <- cm\$overall[1]

return (list(classifier = classifier, confusionMatrix = cm, accuracy = acc))
```

}

Custome models build model with accuracy

build_linear_models <- function(X, y, seed_value = 1){

```
\# Removing the highly correlated predictors out for

\# logistic and LDA models
```

```
too_high <- findCorrelation(cor(X), 0.9)

X_cor <- as.data.frame(X[, -too_high])

\# Logistic Regression Model:

set.seed(seed_value)

glm.classifier <- train(X_cor, y, method = "multinom", preProcess = c("center", "sc
ale"))

glm <- analysis_acc(glm.classifier, X, y)

\# Linear Discriminant Analysis:

set.seed(seed_value)

lda.classifier <- train(X_cor, y, method = "lda", preProc = c("center", "scale"))

lda <- analysis_acc(lda.classifier, X, y)

\# Partial Least Squares Discriminant Analysis

set.seed(seed_value)

plsda.classifier <- train(X, y, method = "pls", tuneGrid = expand.grid(.ncomp=1:6),
preProc = c("center","scale"))

plsda <- analysis_acc(plsda.classifier, X, y)

\# Penalized Methods:

glmnGrid <- expand.grid(.alpha = c(0, .1, .2, .4, .6), .lambda = seq(.01, .2, lengt
h = 10))

set.seed(seed_value)

glmnet.classifier <- train(X, y, method = "glmnet", tuneGrid = glmnGrid, preProc =
c("center", "scale"))

glmnet <- analysis_acc(glmnet.classifier, X, y)

\# Nearest Shrunken Centroids:

nscGrid <- data.frame(.threshold = seq(0,4, by=0.2))

set.seed(seed_value)

nsc.classifier <- train(X, y, method = "pam", tuneGrid = nscGrid, preProc = c("cent
er", "scale"))

nsc <- analysis_acc(nsc.classifier, X, y)

return (list(glm = glm, plsda = plsda, lda=lda, glmnet=glmnet, nsc=nsc))
```

}

Check the file exists and load to variables

else bulid and store the model

if(file.exists(“fatty_results.rds”)) {

```
fatty_results <- readRDS("fatty_results.rds")
```

} else {

```
fatty_results <- build_linear_models(fattyAcids, oilType)

saveRDS(fatty_results, "fatty_results.rds")
```

}

Logistic Regression

fatty_resultsglmclassifier

fatty_resultsglmconfusionMatrix

print(paste(“Accuracy on the whole training set:”, round(fatty_resultsglmaccuracy, 4)))

Linear Discriminant Analysis

fatty_resultsldaclassifier

fatty_resultsldaconfusionMatrix

print(paste(“Accuracy on the whole training set:”, round(fatty_resultsldaaccuracy, 4)))

Partial Least Squares Discriminant Analysis

fatty_results$plsda$classifier

plot(fatty_results$plsda$classifier)

fatty_results$plsda$confusionMatrix

print(paste(“Accuracy on the whole training set:”, round(fatty_results$plsda$accuracy, 4)))

Penalized Methods

fatty_results$glmnet$classifier

plot(fatty_results$glmnet$classifier)

fatty_results$glmnet$confusionMatrix

print(paste(“Accuracy on the whole training set:”, round(fatty_results$glmnet$accuracy, 4)))

Nearest shrunken Centroids

fatty_resultsnscclassifier

plot(fatty_resultsnscclassifier)

fatty_resultsnscconfusionMatrix

```
print(paste("Accuracy on the whole training set:", round(fatty_results$nsc$accuracy, 4)))

fatty_results$glm$confusionMatrix$table
```

End of the Assignemnt