Final Project Part 1

Andrew Marshall

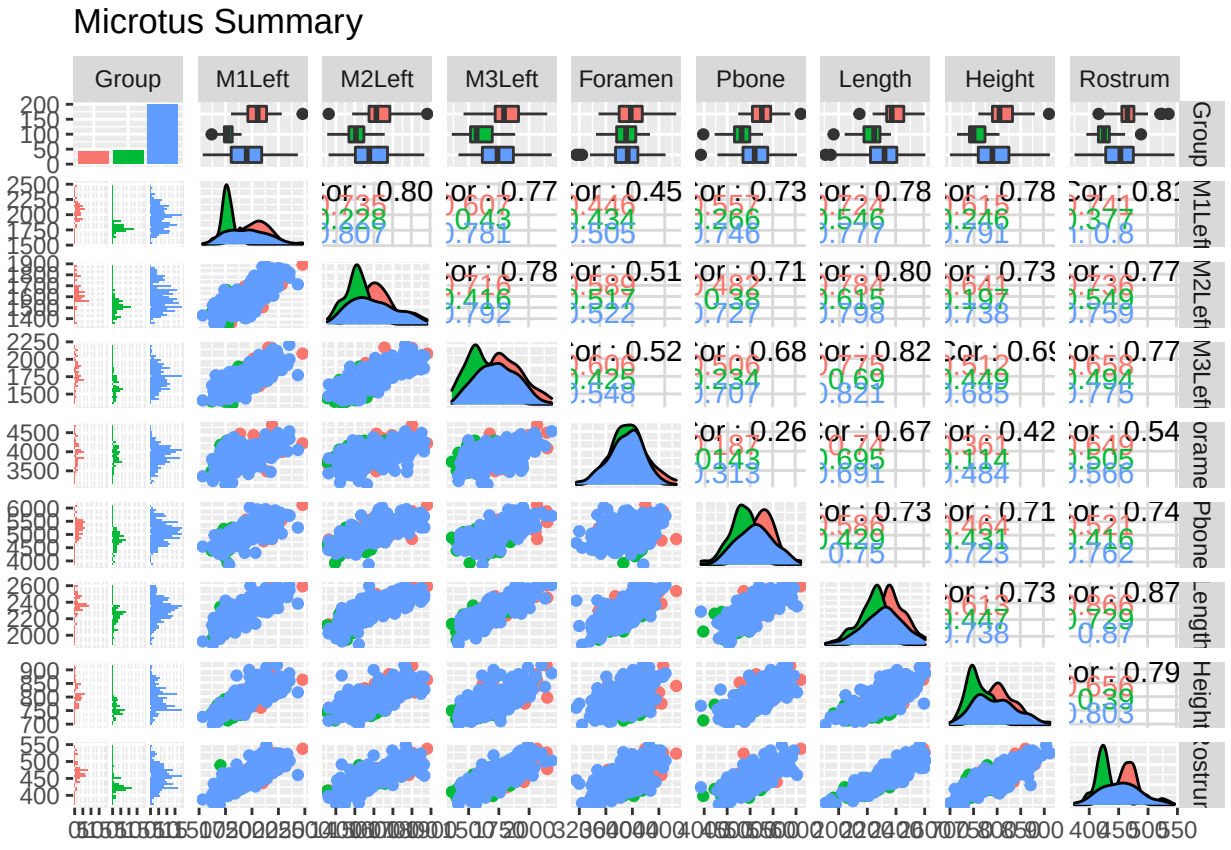
12/15/2019

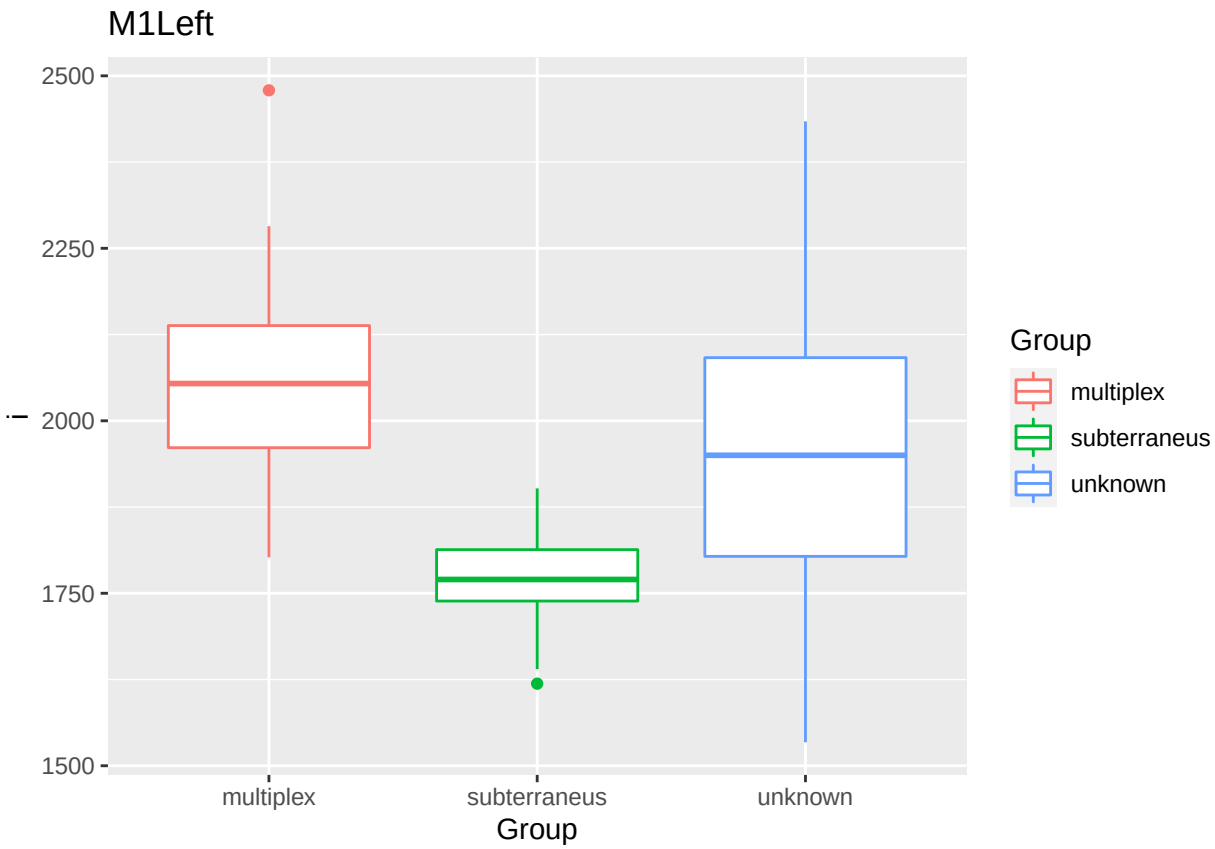
```
data("microtus",package = "Flury")
```

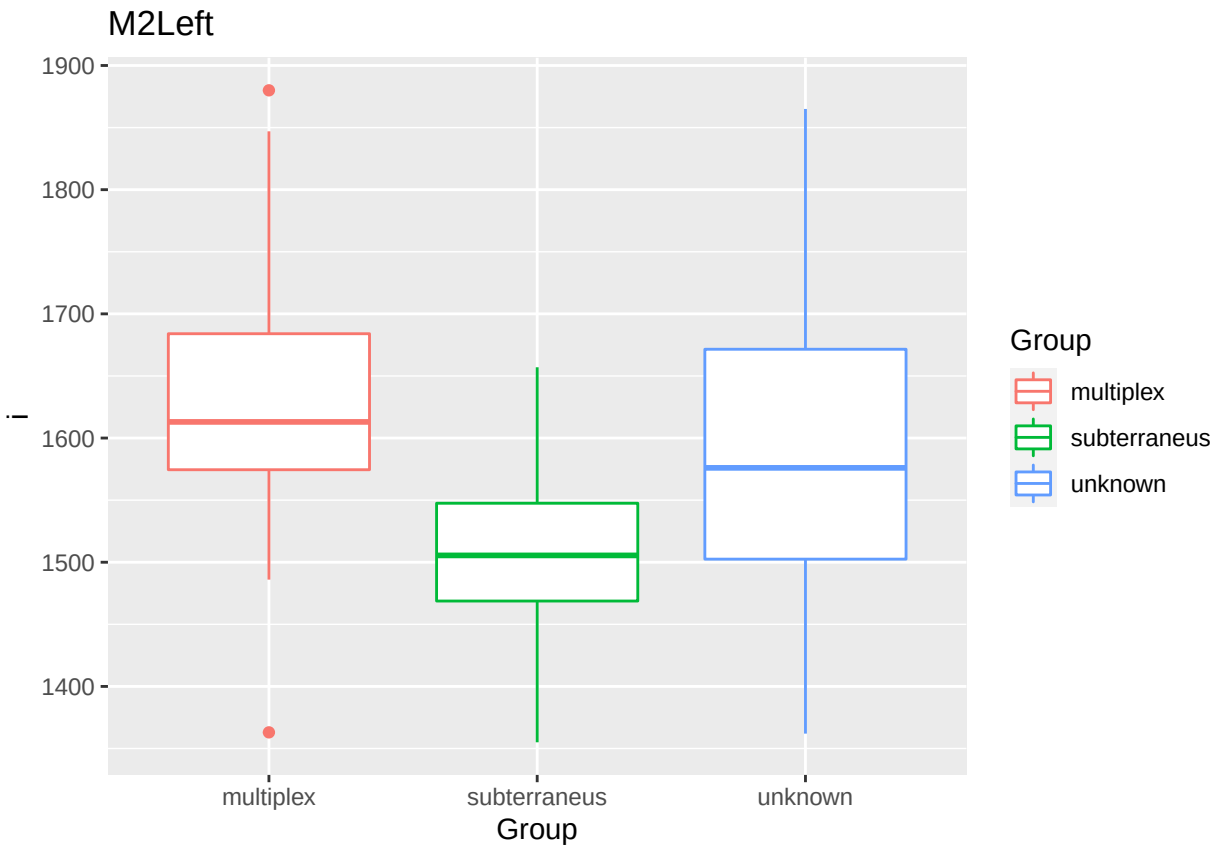
#Exploratory Analysis #Univaritate

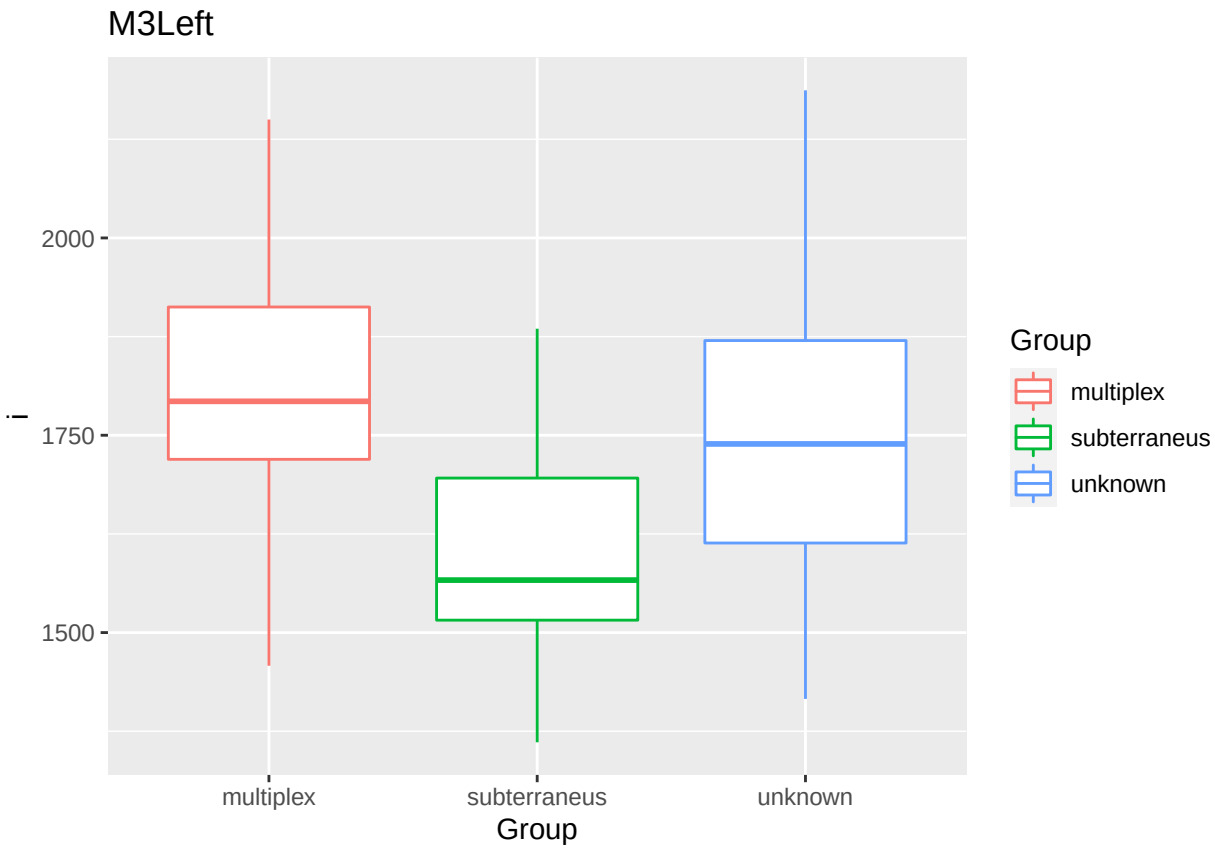
```
##           Group         M1Left          M2Left          M3Left        Foramen
##  multiplex   : 43   Min.   :1534   Min.   :1355   Min.   :1361   Min.   :3155
##  subterraneus: 46   1st Qu.:1783   1st Qu.:1503   1st Qu.:1595   1st Qu.:3751
##  unknown     :199   Median :1923   Median :1570   Median :1724   Median :3932
##                     Mean   :1935   Mean   :1589   Mean   :1727   Mean   :3913
##                     3rd Qu.:2074   3rd Qu.:1660   3rd Qu.:1856   3rd Qu.:4080
##                     Max.   :2479   Max.   :1880   Max.   :2187   Max.   :4662
##      Pbone          Length         Height         Rostrum
##  Min.   :3928   Min.   :1908   Min.   :700.0   Min.   :375.0
##  1st Qu.:4815   1st Qu.:2227   1st Qu.:759.2   1st Qu.:425.0
##  Median :5079   Median :2312   Median :789.0   Median :450.0
##  Mean   :5082   Mean   :2309   Mean   :790.8   Mean   :451.2
##  3rd Qu.:5328   3rd Qu.:2388   3rd Qu.:817.8   3rd Qu.:475.0
##  Max.   :6104   Max.   :2605   Max.   :912.0   Max.   :545.0
```

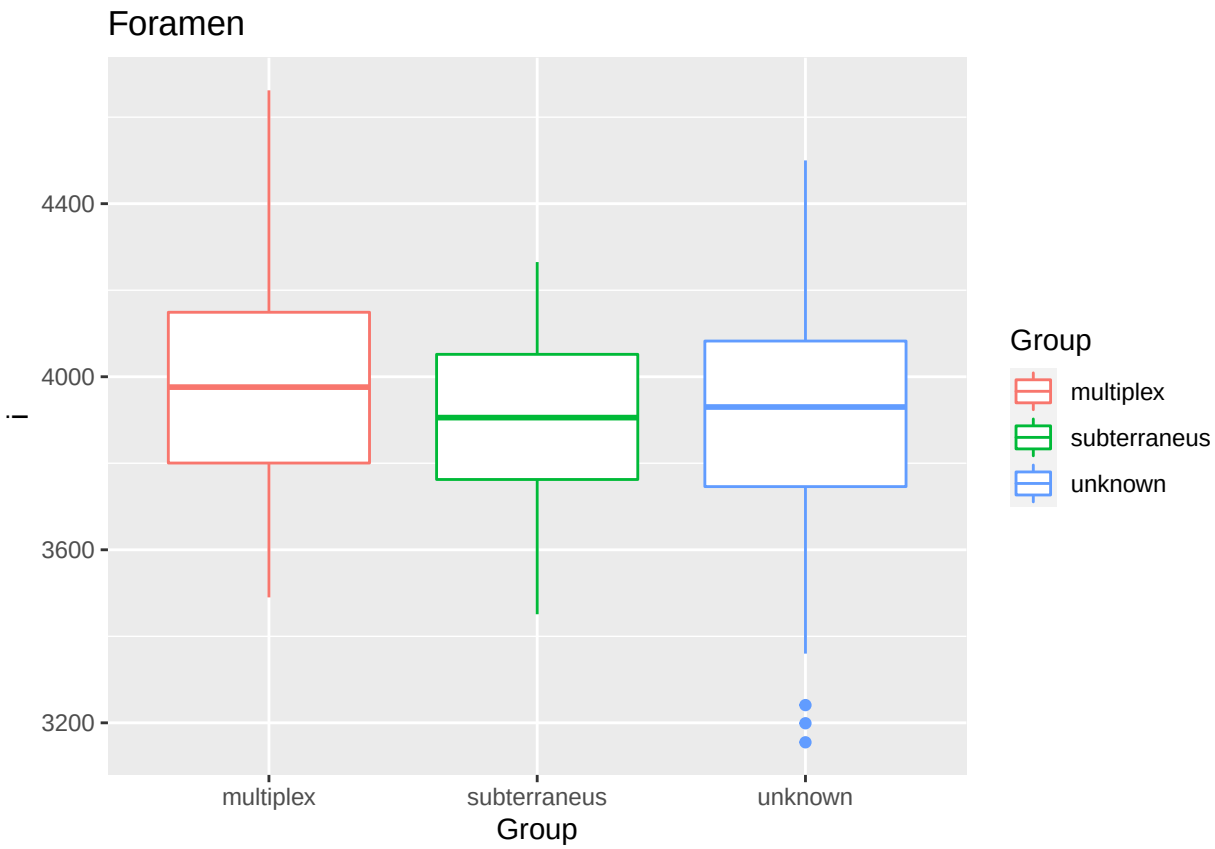
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

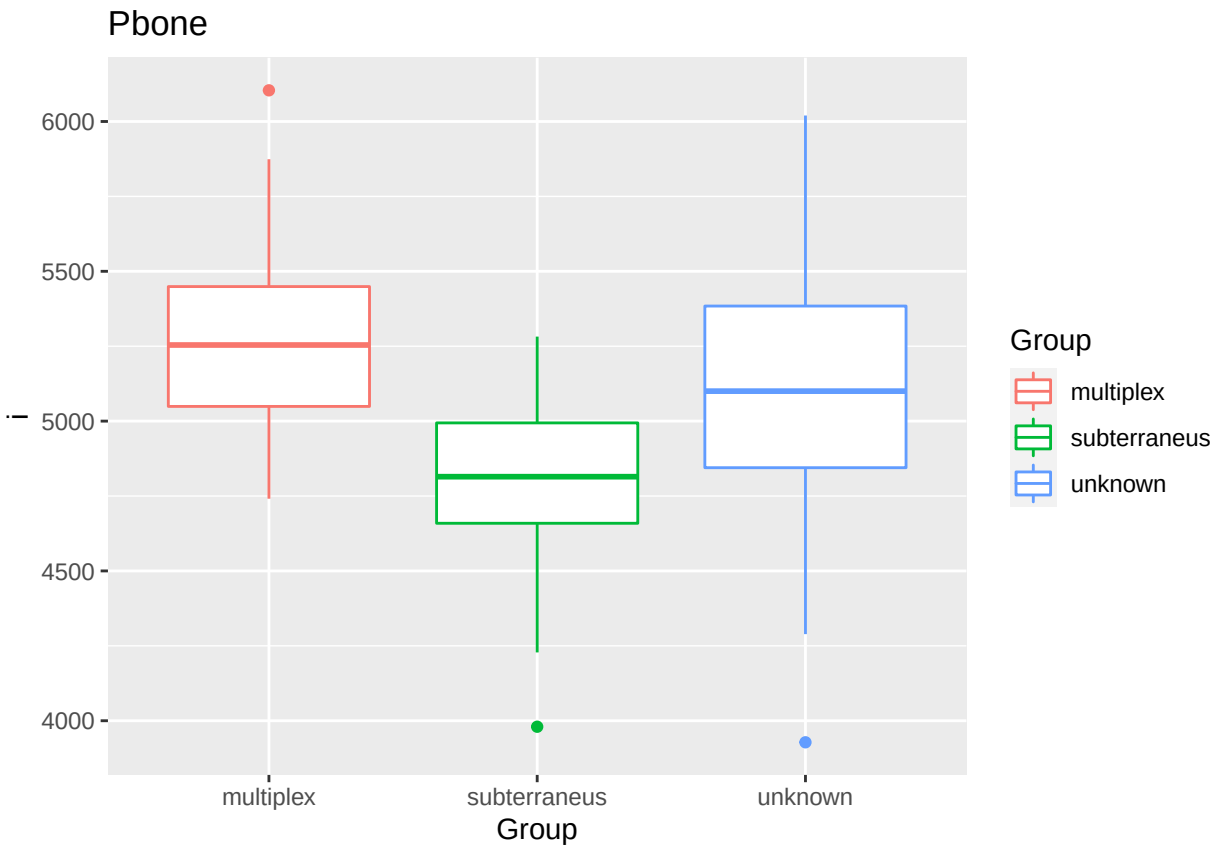

Microtus Summary
Group
M1Left
M2Left
M3Left
Foramen
Pbone
Length
Height
Rostrum

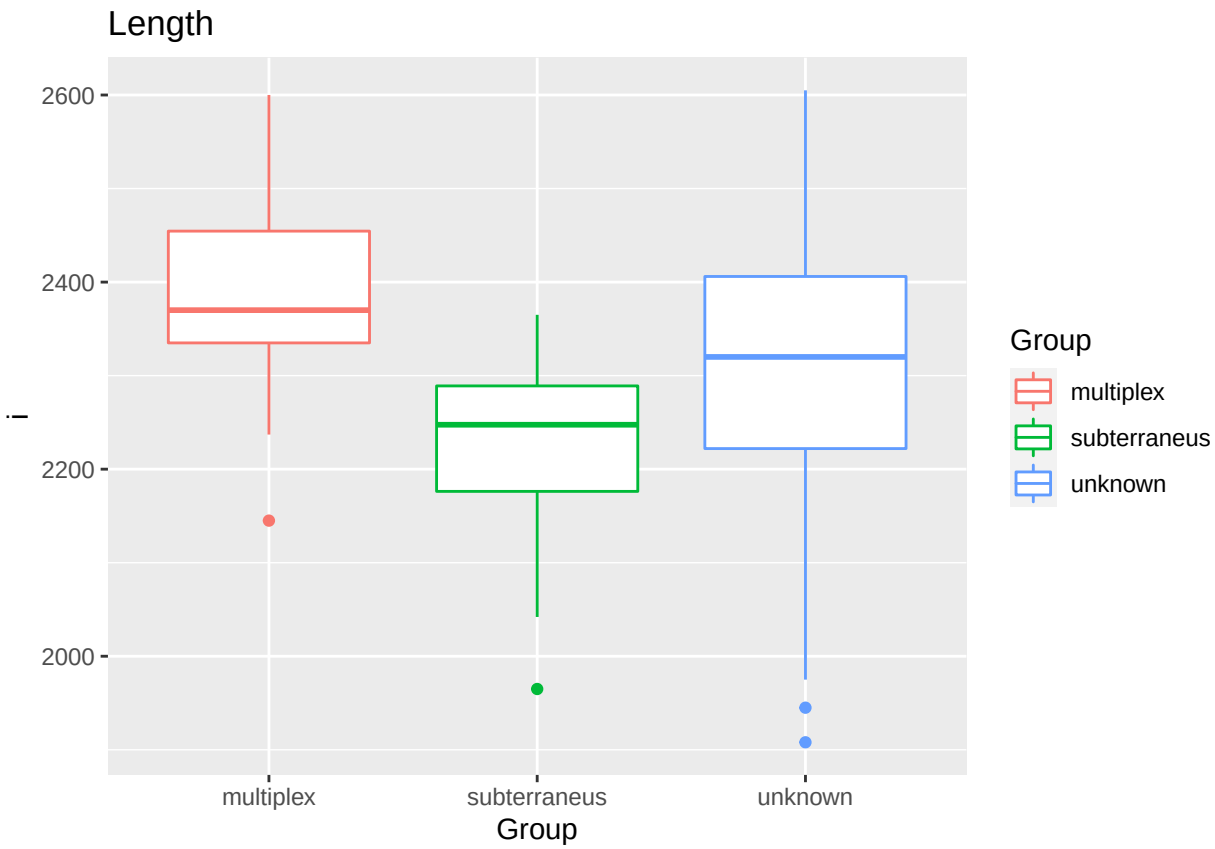

M1Left
2500
2250
2000
1750
1500
multiplex
subterraneus
unknown
Group
Group
multiplex
subterraneus
unknown

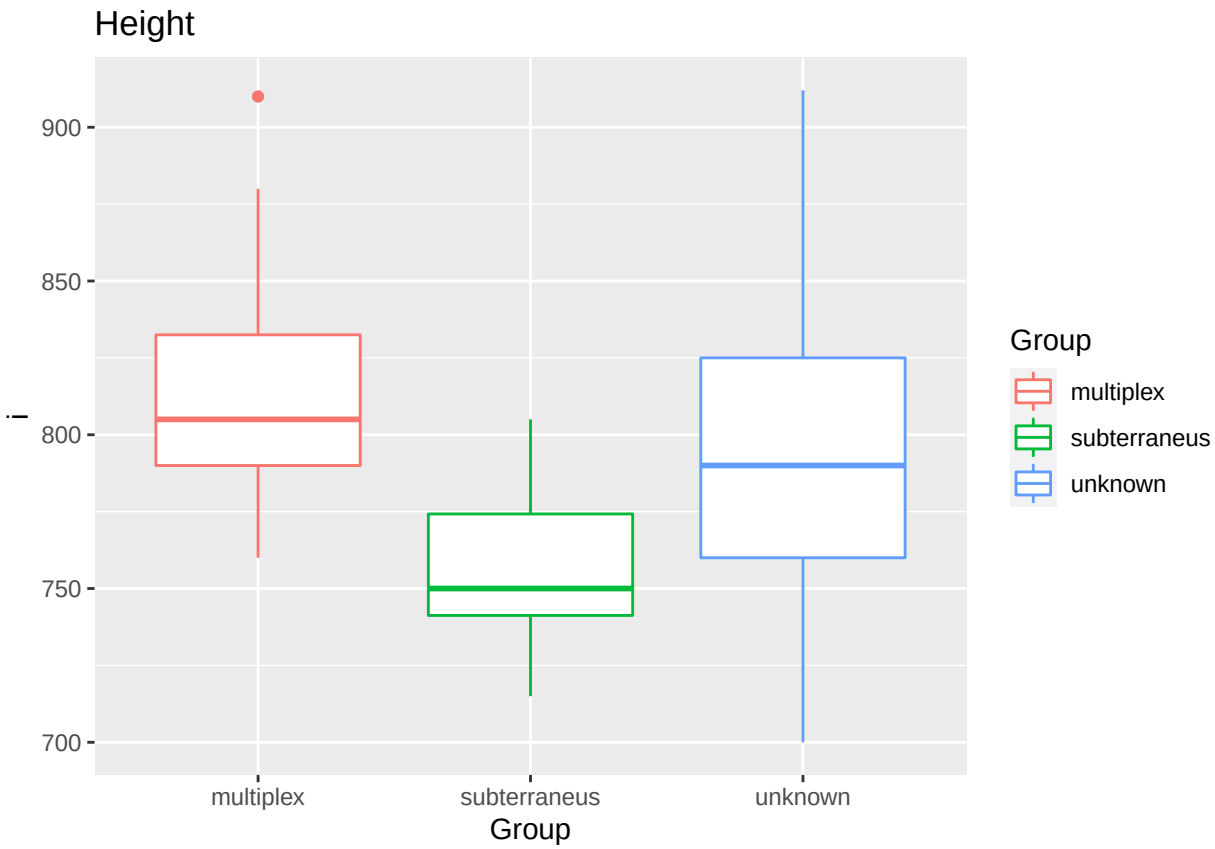

M2Left
1900
1800
1700
1600
1500
1400
i_
multiplex
subterraneus
unknown
Group
Group
multiplex
subterraneus
unknown

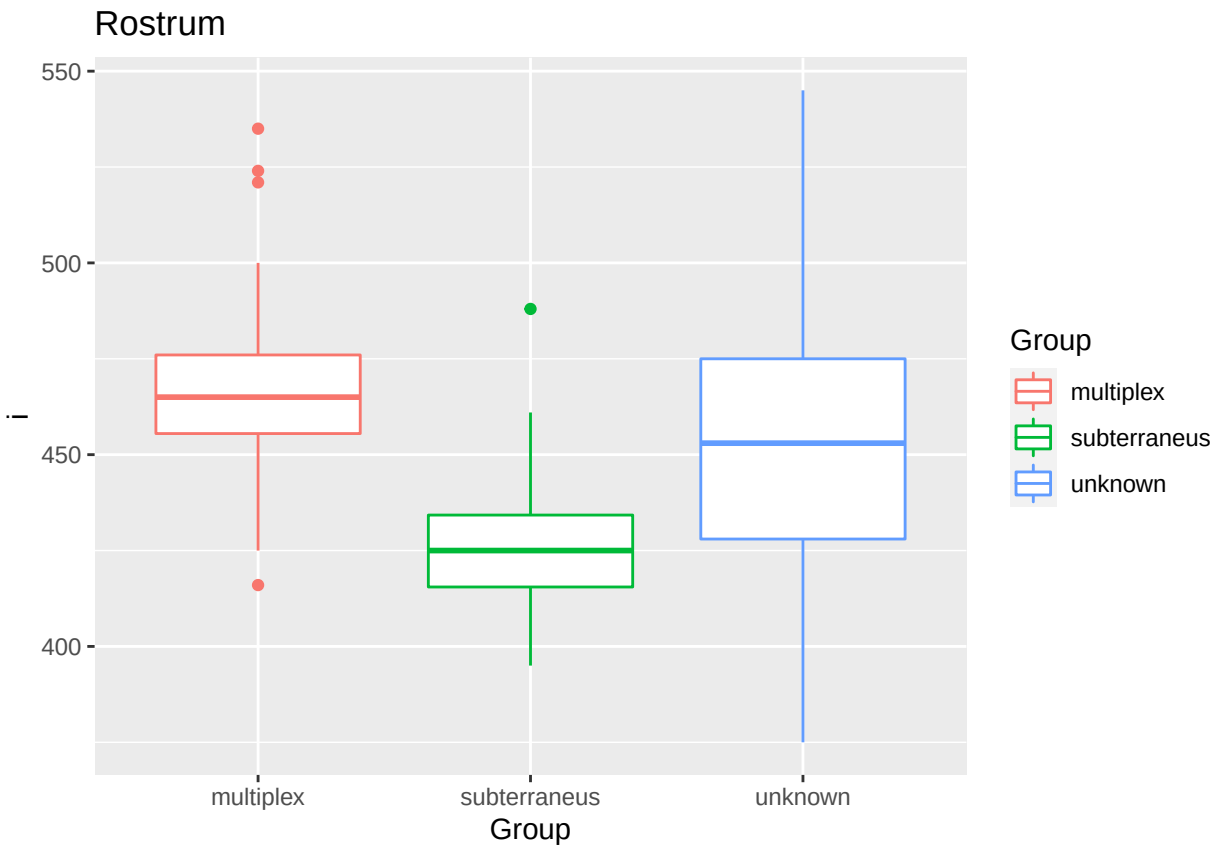

M3Left
2000
1750
1500
multiplex
subterraneus
unknown
Group
Group
multiplex
subterraneus
unknown

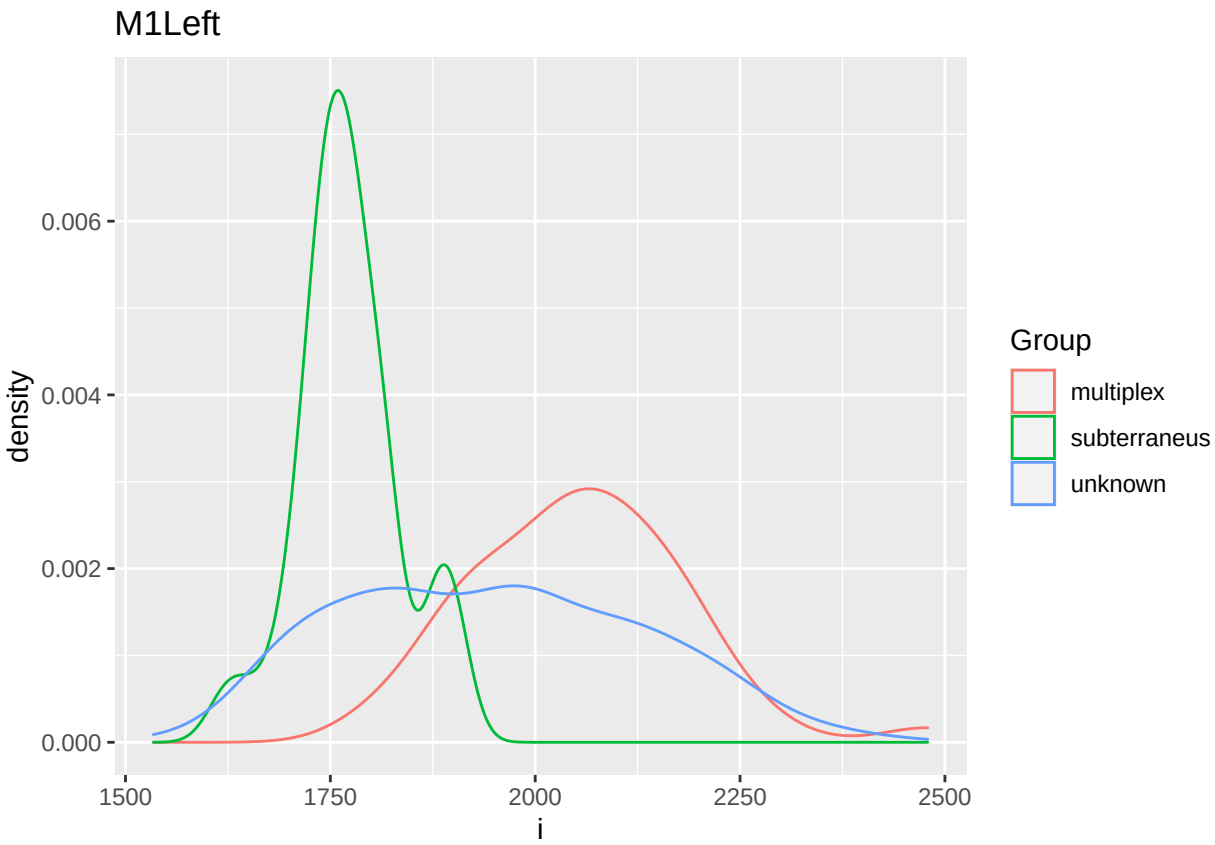

Foramen
4400
4000
3600
3200
multiplex
subterraneus
unknown
Group
Group
multiplex
subterraneus
unknown

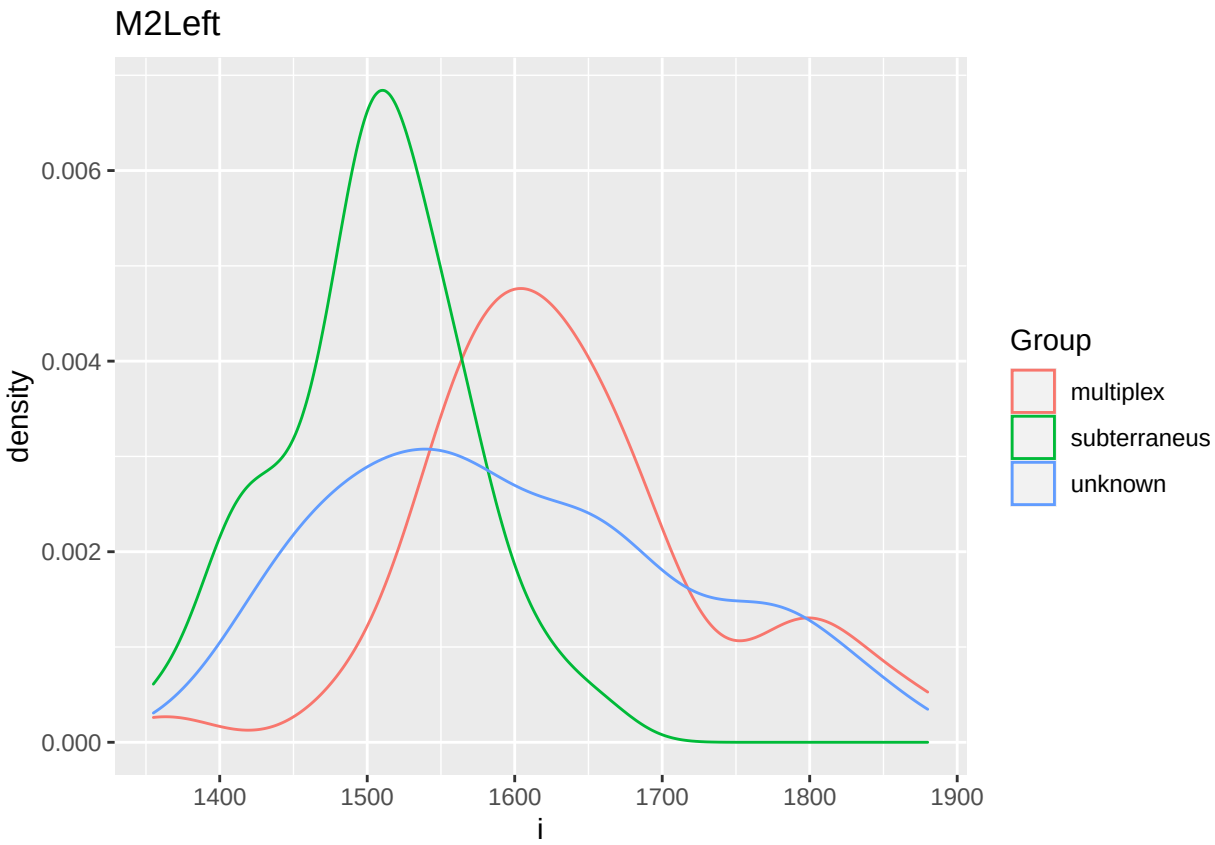

Pbone
6000
5500
5000
4500
4000
multiplex
subterraneus
unknown
Group
Group
multiplex
subterraneus
unknown

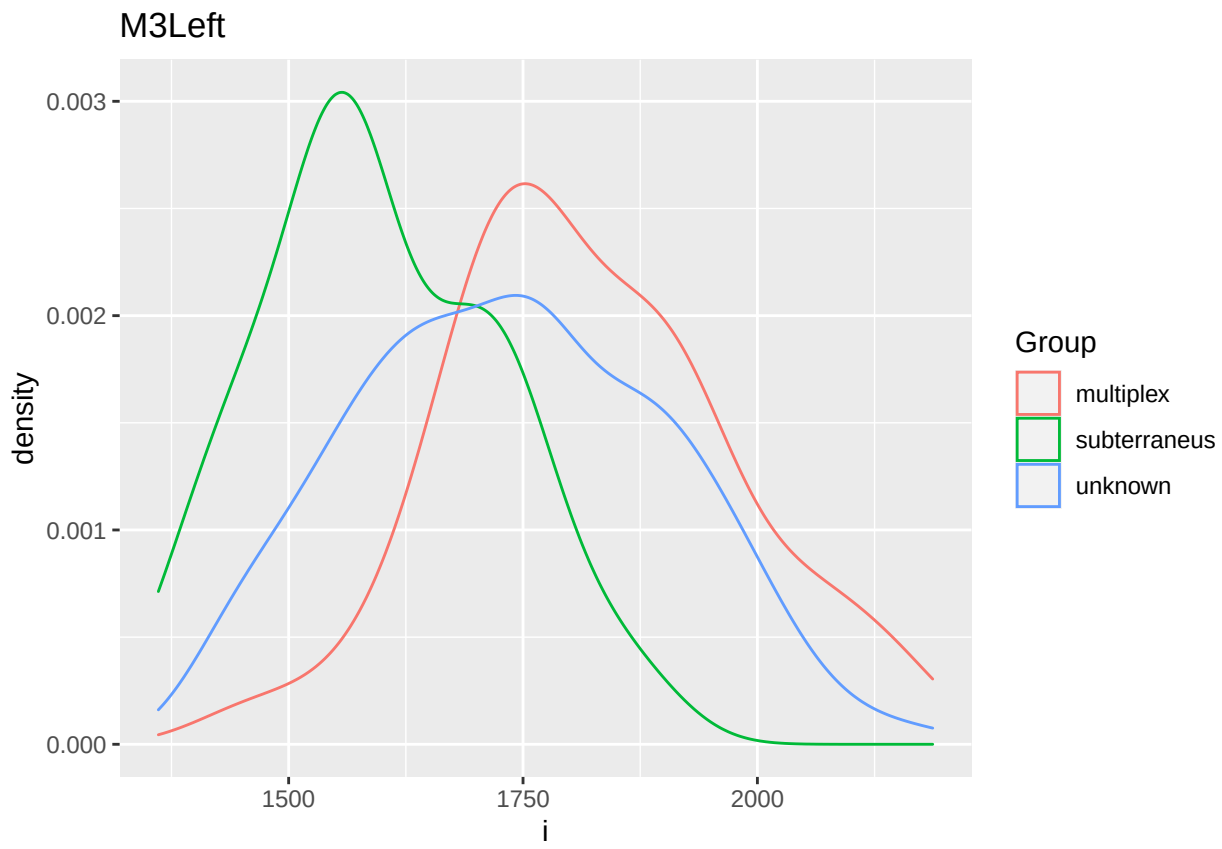

Length
2600
2400
2200
2000
multiplex
subterraneus
unknown
Group
Group
multiplex
subterraneus
unknown

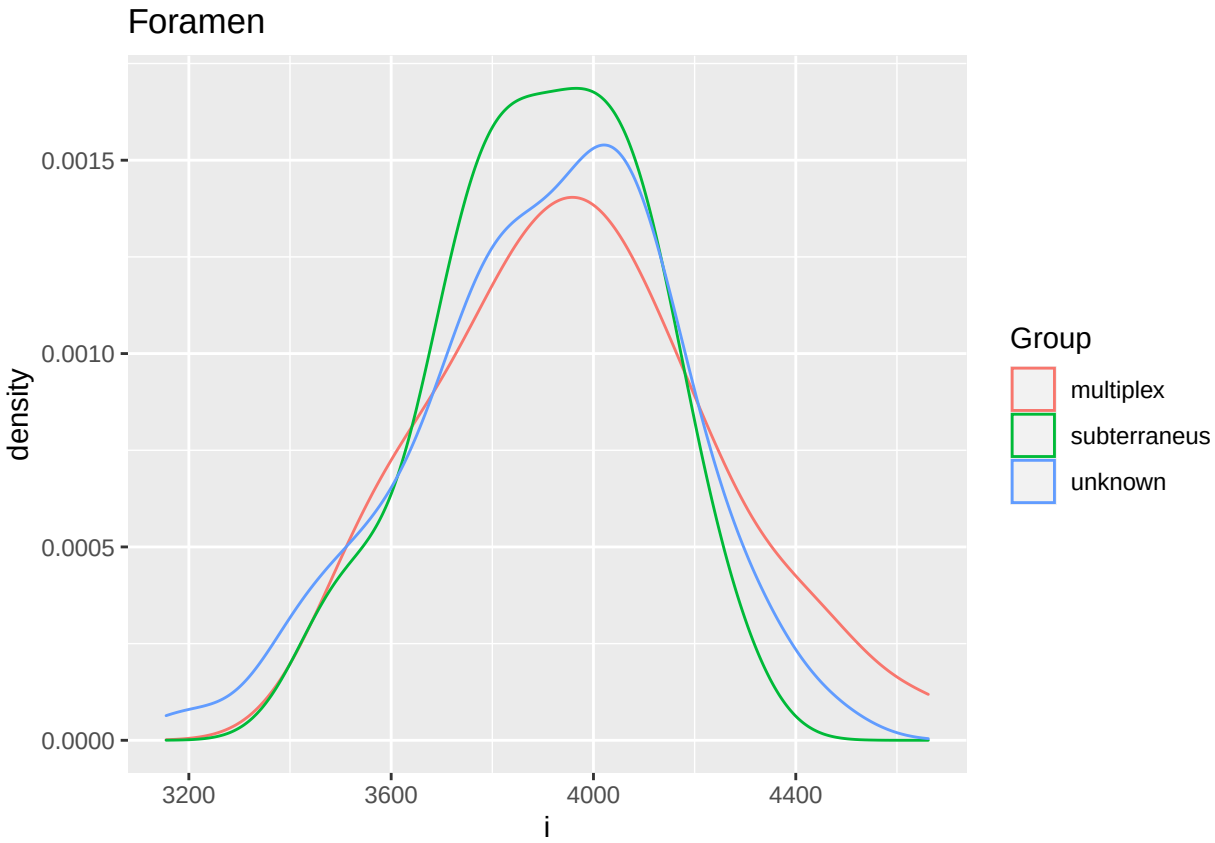

Height
900
850
800
750
700
multiplex
subterraneus
unknown
Group
Group
multiplex
subterraneus
unknown

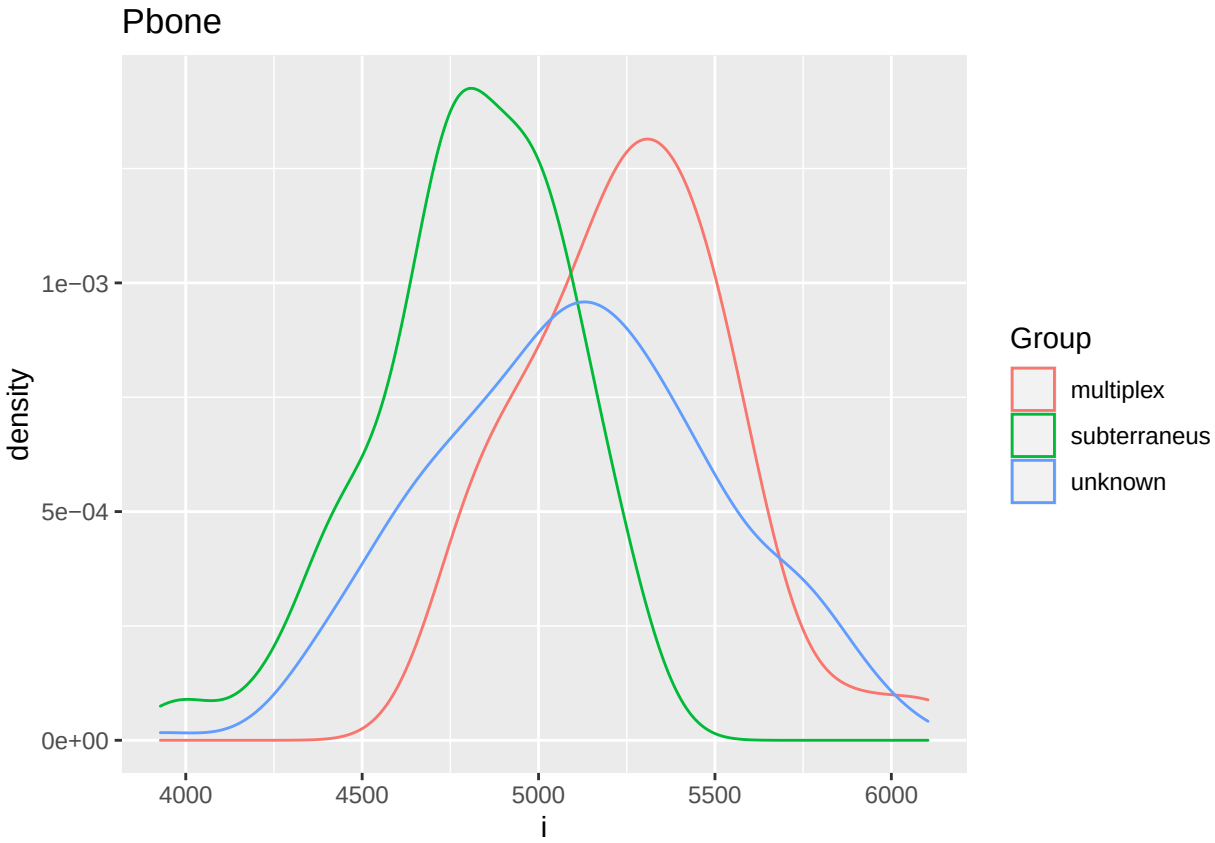

Rostrum
550
500
450
400
multiplex
subterraneus
unknown
Group
Group
multiplex
subterraneus
unknown

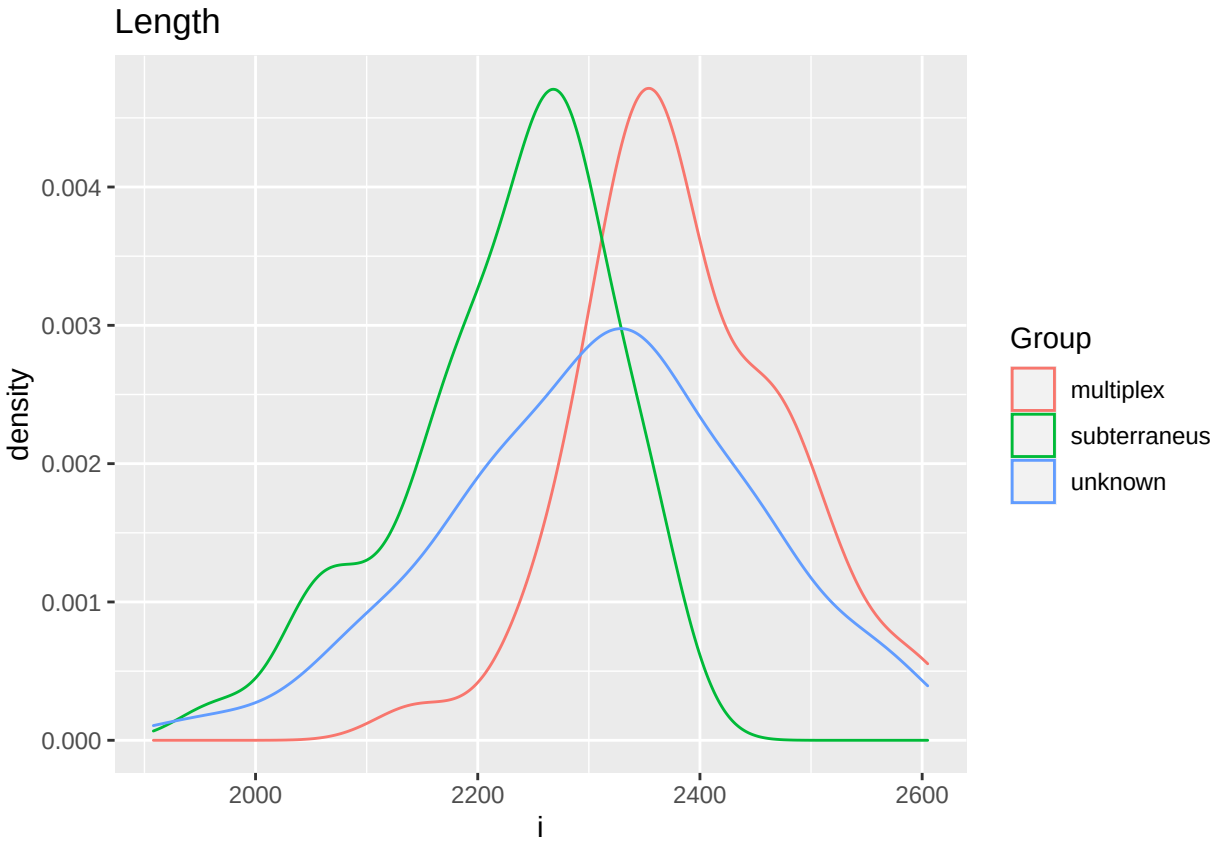

M1Left
density
0.006
0.004
0.002
0.000
1500
1750
2000
2250
2500
i
Group
multiplex
subterraneus
unknown

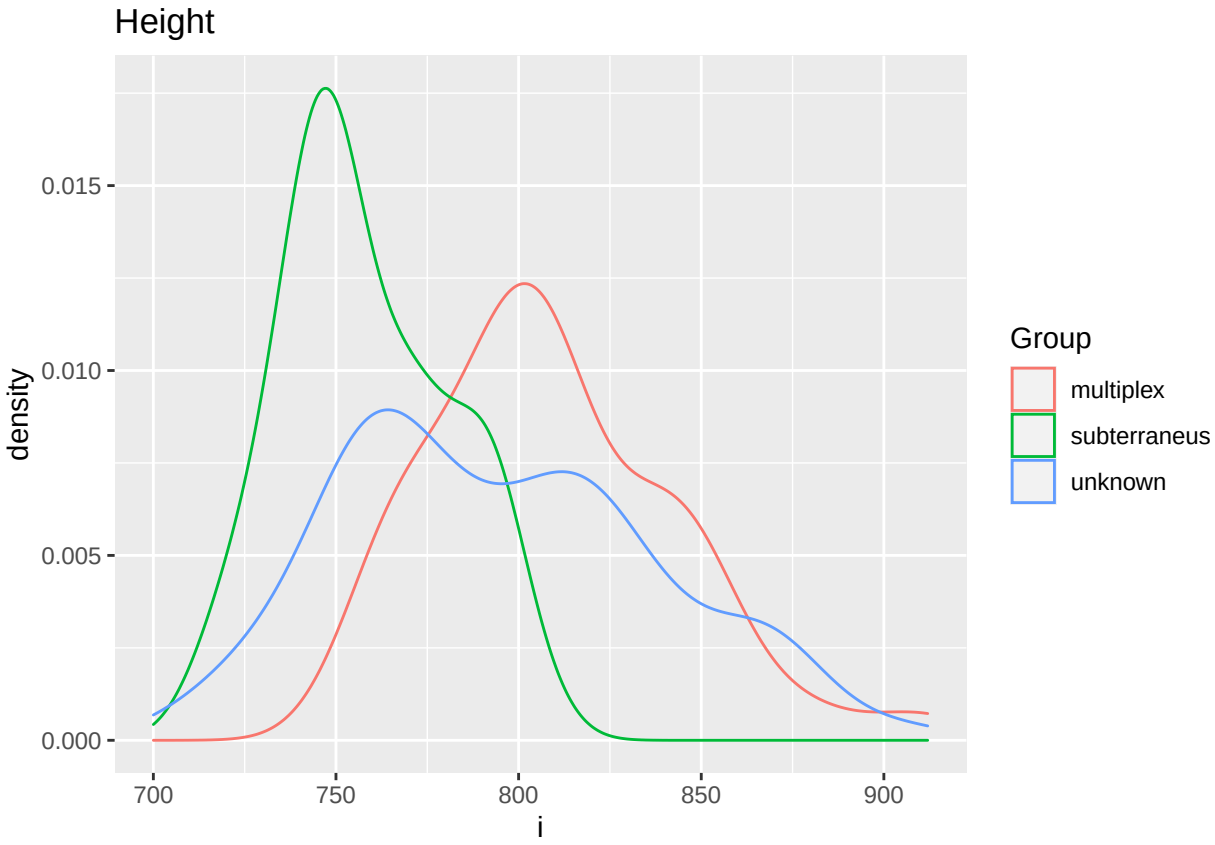

M2Left
density
0.006
0.004
0.002
0.000
1400
1500
1600
1700
1800
1900
i
Group
multiplex
subterraneus
unknown

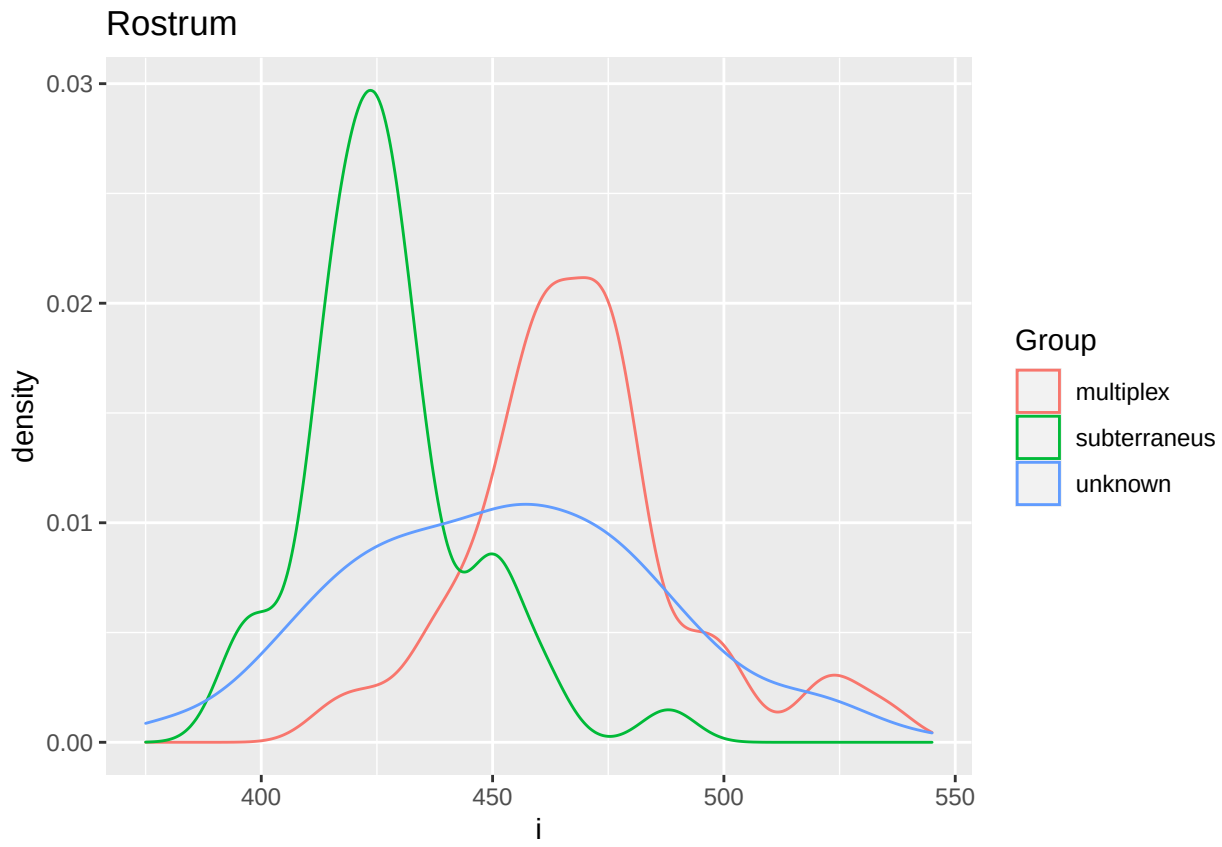

M3Left
0.003
0.002
0.001
0.000
density
1500
1750
2000
i
Group
multiplex
subterraneus
unknown


Foramen
density
0.0015
0.0010
0.0005
0.0000
3200
3600
4000
4400
i
Group
multiplex
subterraneus
unknown


Pbone
density
1e−03
5e−04
0e+00
4000
4500
5000
5500
6000
i
Group
multiplex
subterraneus
unknown


Length
density
0.004
0.003
0.002
0.001
0.000
2000
2200
2400
2600
i
Group
multiplex
subterraneus
unknown


Height
density
0.015
0.010
0.005
0.000
700
750
800
850
900
i
Group
multiplex
subterraneus
unknown



#correlation analysis #variables selection

#build base models #logistic regression

```
##           Group        M3Left         Foramen          Pbone          Height
##  multiplex   :43   Min.   :1361   Min.   :3451   Min.   :3980   Min.   :715.0
##  subterraneus:46   1st Qu.:1561   1st Qu.:3764   1st Qu.:4773   1st Qu.:750.0
##                    Median :1712   Median :3941   Median :5004   Median :776.0
##                    Mean   :1705   Mean   :3932   Mean   :5025   Mean   :782.9
##                    3rd Qu.:1815   3rd Qu.:4078   3rd Qu.:5254   3rd Qu.:805.0
##                    Max.   :2150   Max.   :4662   Max.   :6104   Max.   :910.0
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.14945  -0.43614   0.08254   0.46772   1.82569
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) 48.097613  18.405891   2.613  0.00897 **
## M3Left      -0.003767   0.003981  -0.946  0.34395
## Foramen      0.001583   0.002219   0.713  0.47558
## Pbone       -0.004549   0.002050  -2.219  0.02647 *
```

```
## Height      -0.031933   0.018843  -1.695  0.09013 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 74.786  on 53  degrees of freedom
## Residual deviance: 37.200  on 49  degrees of freedom
## AIC: 47.2
##
## Number of Fisher Scoring iterations: 6

##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.333333e-01   6.657497e-01   7.070588e-01   9.208456e-01   5.185185e-01
## AccuracyPValue  McnemarPValue
##   1.356562e-06   1.000000e+00
```

#Manual Feature Elimination

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6886  -0.6734   0.2551   0.5739   1.8052
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 44.41944   12.10017   3.671 0.000242 ***
## Height      -0.05693    0.01556  -3.660 0.000252 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 74.786  on 53  degrees of freedom
## Residual deviance: 47.757  on 52  degrees of freedom
## AIC: 51.757
##
## Number of Fisher Scoring iterations: 5
```

#Built in Feature selection with glmnet

```
## 5 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept)  2.362528858
## M3Left       .
## Foramen      .
## Pbone       -0.000219100
## Height      -0.001514223
```

#GLM Summary

```
#binary lables to pass to ROCR for ROC curve
microtus_Train <- microtus_Train %>% mutate(
  group_flag = if_else(Group == "multiplex", 1, 0)
)

#create predictions with probabilities
glm_1_pred = predict(glm_1, type = "prob")
glm_2_pred = predict(glm_2, type = "prob")
glm_3_pred = predict(glm_3, type = "prob")

#create prediction objects with multiplex column (note need to select just one column)
pred_1 <- prediction(glm_1_pred$multiplex, as.numeric(microtus_Train$group_flag))
pred_2 <- prediction(glm_2_pred$multiplex, as.numeric(microtus_Train$group_flag))
pred_3 <- prediction(glm_3_pred$multiplex, as.numeric(microtus_Train$group_flag))

#?performance
roc.perf_1 <- performance(pred_1, measure = "tpr", x.measure = "fpr")
roc.perf_1_AUC <- performance(pred_1, measure = "auc")
glm_1_pred_AUC<- roc.perf_1_AUC@y.values

roc.perf_2 <- performance(pred_2, measure = "tpr", x.measure = "fpr")
roc.perf_2_AUC <- performance(pred_2, measure = "auc")
glm_2_pred_AUC<- roc.perf_2_AUC@y.values

roc.perf_3 <- performance(pred_3, measure = "tpr", x.measure = "fpr")
roc.perf_3_AUC <- performance(pred_3, measure = "auc")
glm_3_pred_AUC<- roc.perf_3_AUC@y.values


#calc AUC
"glm_1_pred_AUC"
```

```
## [1] "glm_1_pred_AUC"
```

```
glm_1_pred_AUC
```

```
## [[1]]
## [1] 0.9258242
```

```
"glm_2_pred_AUC"
```

```
## [1] "glm_2_pred_AUC"
```

```
glm_2_pred_AUC
```

```
## [[1]]
## [1] 0.8736264
```

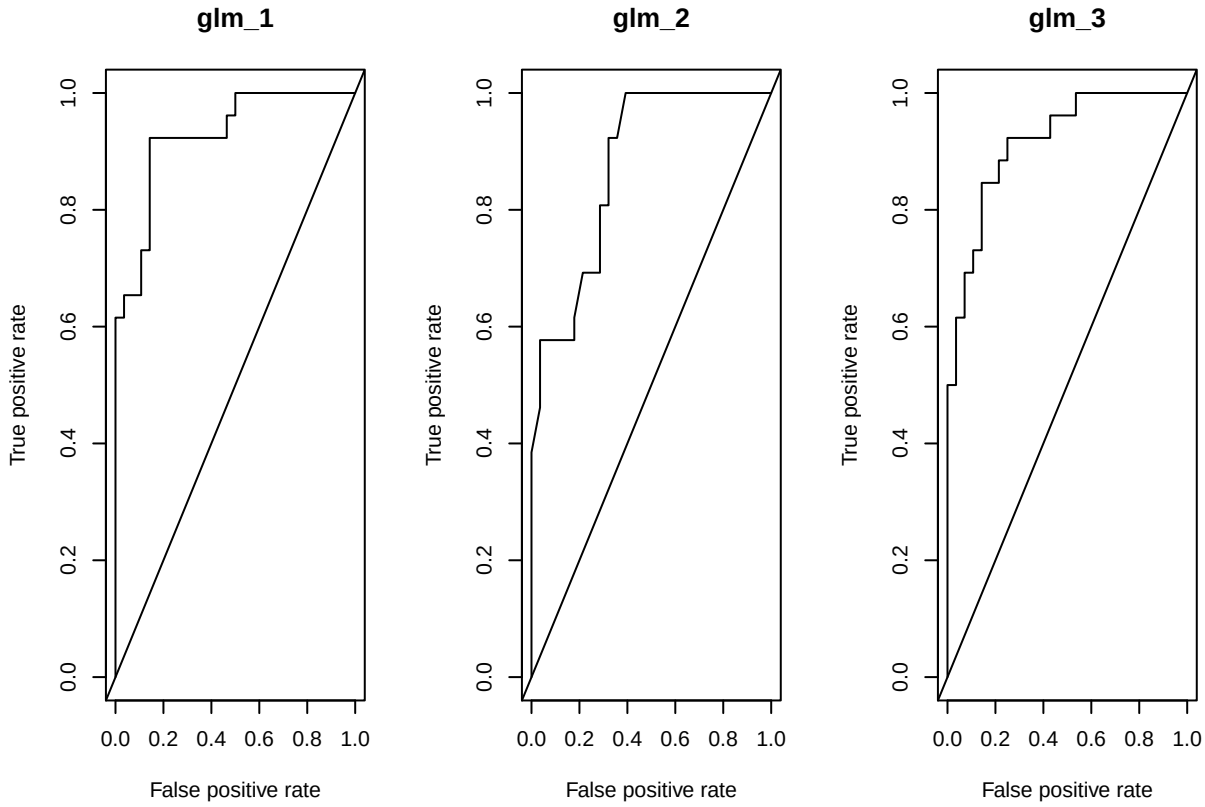
```
"glm_3_pred_AUC"
```

```
## [1] "glm_3_pred_AUC"
```

```
glm_3_pred_AUC
```

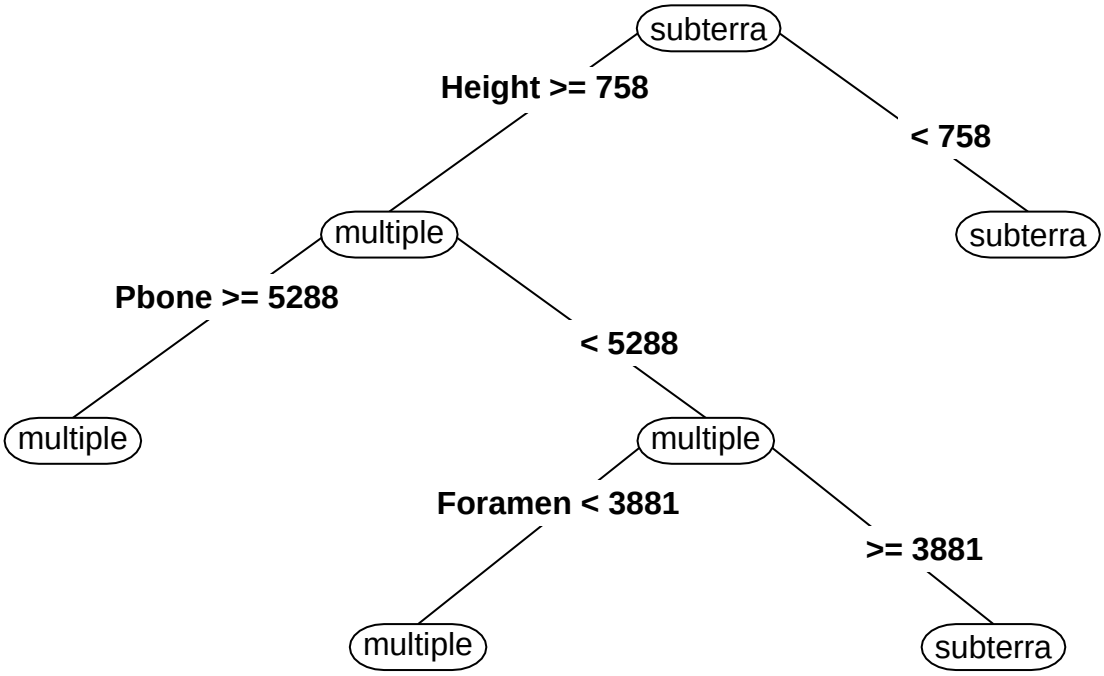
```
## [[1]]
## [1] 0.9148352
```

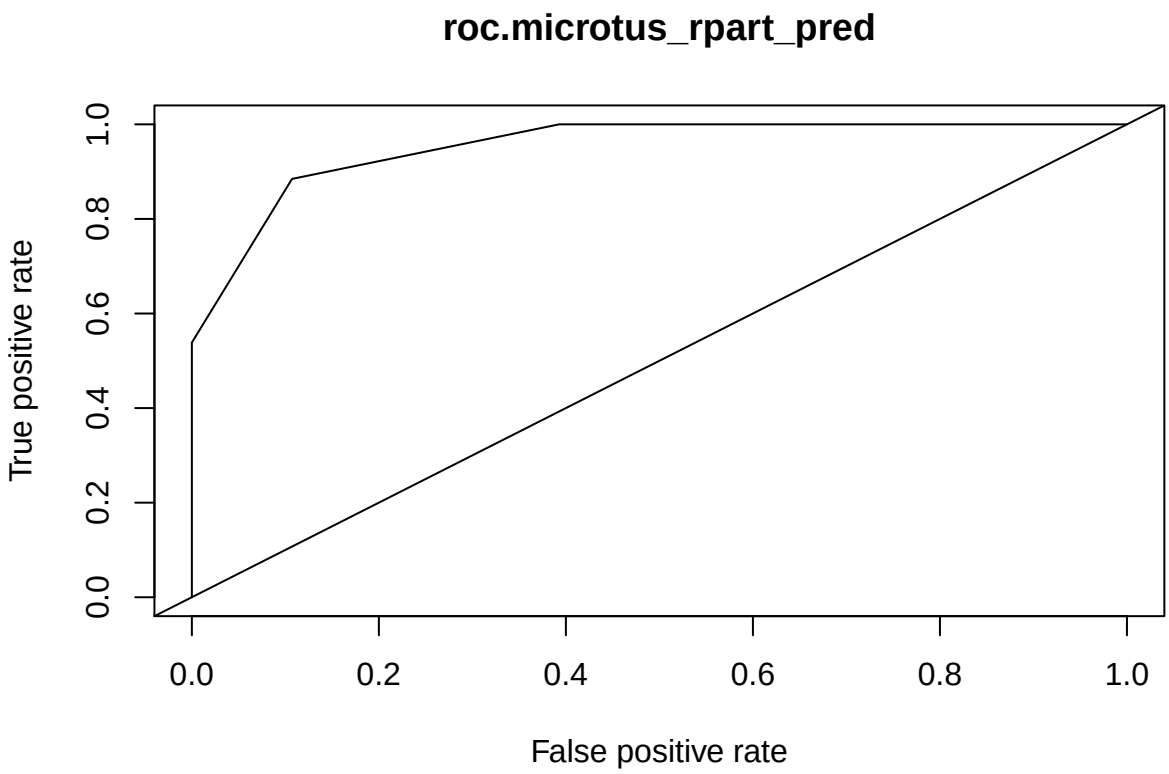
```
#plot data
par(mfrow=c(1,3))
plot(roc.perf_1, main = "glm_1")
abline(0,1)
plot(roc.perf_2, main = "glm_2")
abline(0,1)
plot(roc.perf_3, main = "glm_3")
abline(0,1)
```



#Tree Based Methods (recursive partitioning)

```
## Warning in train.default(x, y, weights = w, ...): The metric "Accuracy" was not
## in the result set. ROC will be used instead.
```


subterra
Height >= 758
< 758
multiple
subterra
Pbone >= 5288
< 5288
multiple
multiple
Foramen < 3881
>= 3881
multiple
subterra

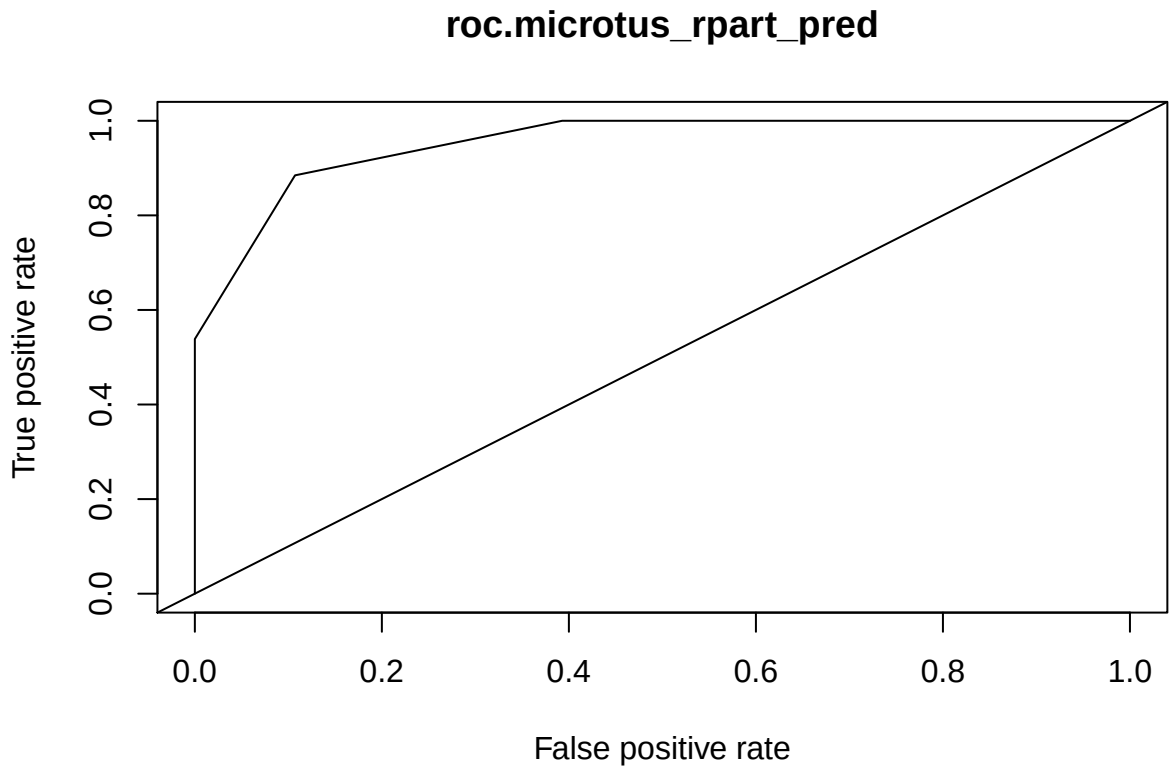



```
## [[1]]
## [1] 0.9526099
```

```
tree_ctrl <- trainControl(method = "cv", number = 10,
                          returnResamp = "all",
                          classProbs = TRUE,
                          summaryFunction = twoClassSummary,
                          seeds = seed)
#treebag method used
microtus_tree_bag <- train(Group ~ .,
                           data = microtus_Train_no_flag,
                           method = "treebag",
                           trControl = tree_ctrl,
                           metric = "ROC",
                           nbagg = 10)

#microtus_tree_bag
#summary(microtus_tree_bag)

microtus_tree_bag_pred <- predict(microtus_tree_bag, newdata = microtus_Train_no_flag)
microtus_tree_bag_pred_cf <- confusionMatrix(microtus_tree_bag_pred, microtus_Train_no_flag$Group)
```

```
#Calculate Area Under the Curve for model

microtus_tree_bag_pred <- predict(microtus_tree_bag, newdata = microtus_Train_no_flag)
```

```
#confusionMatrix(microtus_tree_bag_pred, microtus_Train_no_flag$Group)

#create predictions with probabilities
microtus_tree_bag_pred = predict(microtus_tree_bag, type = "prob")

#create prediction objects with multiplex column (note need to select just one column)
pred_tree_bag <- prediction(microtus_tree_bag_pred$multiplex, as.numeric(microtus_Train$group_flag))

roc.microtus_tree_bag_pred <- performance(pred_rpart, measure = "tpr", x.measure = "fpr")

#plot data
#par(mfrow=c(1,3))
plot(roc.microtus_tree_bag_pred, main = "roc.microtus_rpart_pred")
abline(0,1)
```




```
roc.microtus_tree_bag_pred_AUC <- performance(pred_tree_bag, measure = "auc")
microtus_tree_bag_pred_AUC<- roc.microtus_tree_bag_pred_AUC@y.values
microtus_tree_bag_pred_AUC
```

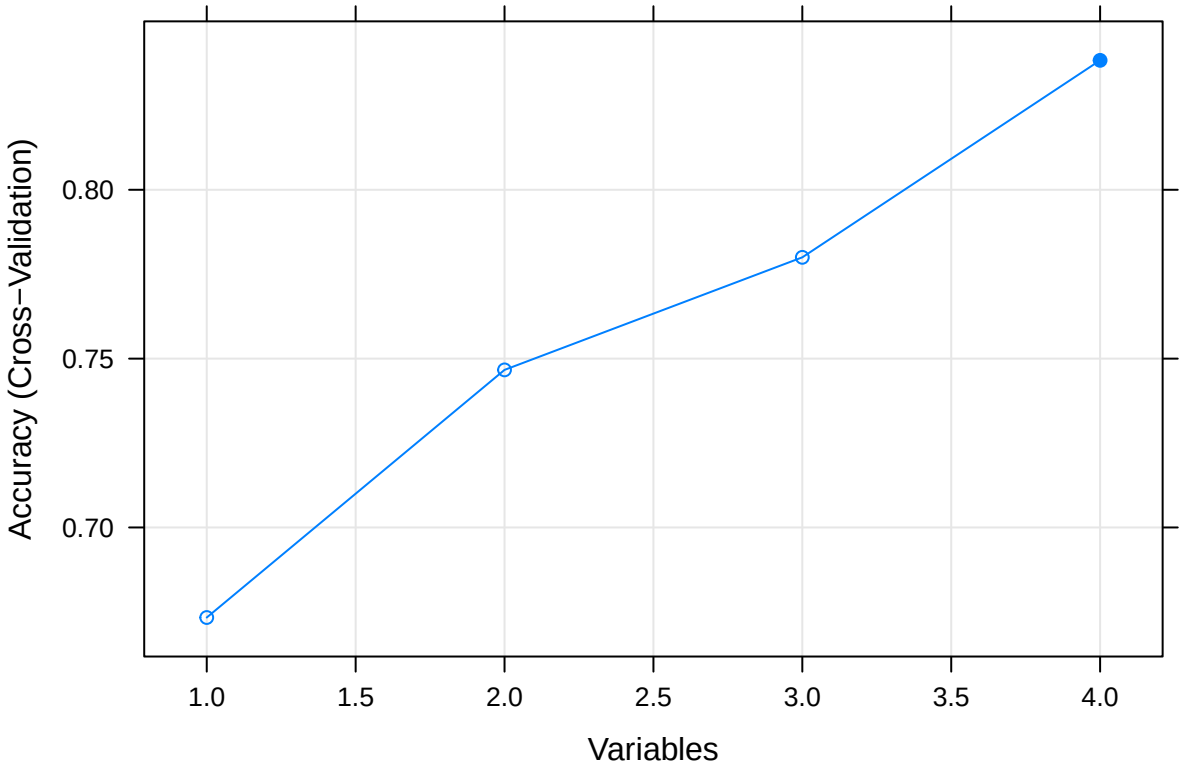
```
## [[1]]
## [1] 1
```

#recursive feature elimination wrapper method to fit random forest

```r
# define the control using a recursive feature elimination (backwards) selection function

# define the control using a random forest selection function
rfe_controller <- rfeControl(functions=rfFuncs, method="cv", number=10)
# run the RFE algorithm
x=microtus_Train_no_flag[,2:5]
y=microtus_Train_no_flag[,1]
rfe_results <- rfe(x=x, y, sizes=c(1:4), rfeControl=rfe_controller)
# summarize the results
print(rfe_results)
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          1   0.6733 0.3212     0.1647  0.3405
##          2   0.7467 0.4879     0.2103  0.4195
##          3   0.7800 0.5545     0.1926  0.3849
##          4   0.8383 0.6712     0.1866  0.3748        *
##
## The top 4 variables (out of 4):
##    Pbone, Height, M3Left, Foramen
```

```r
# list the chosen features
#predictors(rfe_results)
# plot the results
plot(rfe_results, type=c("g", "o"))
```




```
rfe_results
```

```
##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold)
##
## Resampling performance over subset size:
##
##  Variables Accuracy  Kappa AccuracySD KappaSD Selected
##          1   0.6733 0.3212     0.1647  0.3405
##          2   0.7467 0.4879     0.2103  0.4195
##          3   0.7800 0.5545     0.1926  0.3849
##          4   0.8383 0.6712     0.1866  0.3748        *
##
## The top 4 variables (out of 4):
##    Pbone, Height, M3Left, Foramen
```

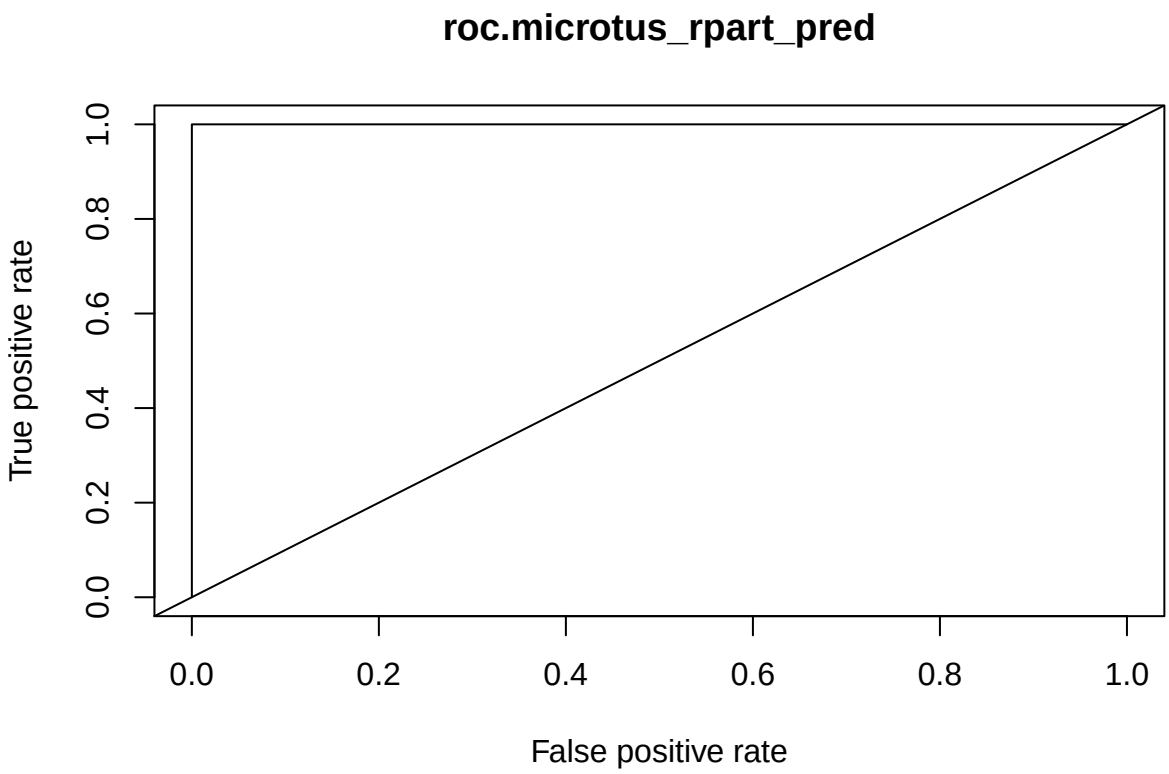
```
microtus_tree_ranFor_rfe <- rfe_results$fit
microtus_tree_ranFor_rfe
```

```
##
## Call:
##  randomForest(x = x, y = y, importance = TRUE)
##                Type of random forest: classification
```

```
##                       Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 16.67%
## Confusion matrix:
##              multiplex subterraneus class.error
## multiplex           21            5   0.1923077
## subterraneus         4           24   0.1428571
```

```
#microtus_tree_ranForRFE_pred
```

```
#Calculate Area Under the Curve for model

microtus_tree_ranFor_pred <- predict(microtus_tree_ranFor, newdata = microtus_Train_no_flag)

#confusionMatrix(microtus_tree_ranFor_pred, microtus_Train_no_flag$Group)
#create predictions with probabilities
microtus_tree_ranFor_pred = predict(microtus_tree_ranFor, type = "prob")

#create prediction objects with multiplex column (note need to select just one column)
pred_tree_ranFor <- prediction(microtus_tree_ranFor_pred$multiplex, as.numeric(microtus_Train$group_flag

roc.microtus_tree_ranFor_pred <- performance(pred_tree_ranFor, measure = "tpr", x.measure = "fpr")

#plot data
#par(mfrow=c(1,3))
plot(roc.microtus_tree_ranFor_pred, main = "roc.microtus_rpart_pred")
abline(0,1)
```




```
roc.microtus_tree_ranFor_pred_AUC <- performance(pred_tree_ranFor, measure = "auc")
microtus_tree_ranFor_pred_AUC<- roc.microtus_tree_ranFor_pred_AUC@y.values
microtus_tree_ranFor_pred_AUC
```

```
## [[1]]
## [1] 1
```

```
#Classification Metrics
"glm_1_pred_cf"
```

```
## [1] "glm_1_pred_cf"
```

```
glm_1_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.333333e-01   6.657497e-01   7.070588e-01   9.208456e-01   5.185185e-01
## AccuracyPValue  McnemarPValue
##   1.356562e-06   1.000000e+00
```

```
"glm_2_pred_cf"
```

```
## [1] "glm_2_pred_cf"
```

```
glm_2_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##    0.722222222    0.442916094    0.583564988    0.835445294    0.518518519
## AccuracyPValue  McnemarPValue
##    0.001827128    1.000000000
```

```
"glm_3_pred_cf"
```

```
## [1] "glm_3_pred_cf"
```

```
glm_3_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   0.7592592593   0.5090909091   0.6235947542   0.8651365787   0.5185185185
## AccuracyPValue  McnemarPValue
##   0.0002456857   0.0008740872
```

```
"microtus_rpart_pred_cf"
```

```
## [1] "microtus_rpart_pred_cf"
```

```
microtus_rpart_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.888889e-01   7.774725e-01   7.736868e-01   9.581162e-01   5.185185e-01
## AccuracyPValue  McnemarPValue
##   7.515084e-09   1.000000e+00
```

```
"microtus_tree_bag_pred_cf"
```

```
## [1] "microtus_tree_bag_pred_cf"
```

```
microtus_tree_bag_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   1.000000e+00   1.000000e+00   9.339685e-01   1.000000e+00   5.185185e-01
## AccuracyPValue  McnemarPValue
##   3.956131e-16            NaN
```

```
"microtus_tree_ranFor_pred_cf"
```

```
## [1] "microtus_tree_ranFor_pred_cf"
```

```
microtus_tree_ranFor_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   1.000000e+00   1.000000e+00   9.339685e-01   1.000000e+00   5.185185e-01
## AccuracyPValue  McnemarPValue
##   3.956131e-16            NaN
```

fit best training models to test datasets

```
#glmnet
glm_3_test_pred <- predict(glm_3, newdata = microtus_Test, type = "raw")
glm_3_test_pred_cf <- confusionMatrix(data = glm_3_test_pred, reference = microtus_Test$Group)

glm_3_test_pred <- predict(glm_3, newdata = microtus_Test, type = "prob")

#create prediction objects with multiplex column (note need to select just one column)
pred_glm_3_test <- prediction(glm_3_test_pred$multiplex, as.numeric(microtus_Test$group_flag))

glm_3_test_pred_AUC <- performance(pred_glm_3_test, measure = "auc")
glm_3_test_pred_AUC@y.values
```

```
## [[1]]
## [1] 0.9444444
```

```
#bagged Tree
microtus_tree_bag_test_pred <- predict(microtus_tree_bag, newdata = microtus_Test)
microtus_tree_bag_test_pred_cf <- confusionMatrix(microtus_tree_bag_test_pred, microtus_Test$Group)

microtus_tree_bag_test_pred <- predict(microtus_tree_bag, newdata = microtus_Test, type = "prob")

#create prediction objects with multiplex column (note need to select just one column)
pred_tree_bag_test <- prediction(microtus_tree_bag_test_pred$multiplex, as.numeric(microtus_Test$group_

tree_bag_test_AUC <- performance(pred_glm_3_test, measure = "auc")
tree_bag_test_AUC@y.values
```

```
## [[1]]
## [1] 0.9444444
```

```
#Random Forest
microtus_tree_ranFor_test_pred <- predict(microtus_tree_ranFor, newdata = microtus_Test)
microtus_tree_ranFor_test_pred_cf <- confusionMatrix(microtus_tree_ranFor_test_pred, microtus_Test$Grou

# microtus_tree_ranFor_test_pred <- predict(microtus_tree_ranFor, newdata = microtus_Test, type = "prob
# #create prediction objects with multiplex column (note need to select just one column)
# pred_tree_ranFor_test <- prediction(microtus_tree_ranFor_test_pred$multiplex, as.numeric(microtus_Tes
#
# microtus_tree_ranFor_test_pred_cf <- confusionMatrix(pred_tree_ranFor_test, microtus_Test$Group)
```

```
glm_3_test_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##    0.742857143    0.477611940    0.567441150    0.875106028    0.514285714
## AccuracyPValue  McnemarPValue
##    0.004919482    0.007660761
```

```
microtus_tree_bag_test_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##    0.771428571    0.542483660    0.598636744    0.895789568    0.514285714
## AccuracyPValue  McnemarPValue
##    0.001626461    1.000000000
```

```
microtus_tree_ranFor_test_pred_cf$overall
```

```
##       Accuracy          Kappa  AccuracyLower  AccuracyUpper   AccuracyNull
##   8.571429e-01   7.135843e-01   6.974286e-01   9.519392e-01   5.142857e-01
## AccuracyPValue  McnemarPValue
##   2.275361e-05   1.000000e+00
```

##use GLMNET to predict all classes based on Kappa Score

```
# microtus <- microtus %>% mutate(
#   final_pred_flag = if_else(microtus$Group == "unknown",1,0))

microtus <- microtus %>% mutate(
  final_pred = if_else(microtus$Group == "unknown",
                       predict(glm_3, newdata = microtus),
                       microtus$Group))
summary(microtus)
```

```
##           Group         M1Left          M2Left          M3Left        Foramen
##  multiplex   : 43   Min.   :1534   Min.   :1355   Min.   :1361   Min.   :3155
##  subterraneus: 46   1st Qu.:1783   1st Qu.:1503   1st Qu.:1595   1st Qu.:3751
##  unknown     :199   Median :1923   Median :1570   Median :1724   Median :3932
##                     Mean   :1935   Mean   :1589   Mean   :1727   Mean   :3913
##                     3rd Qu.:2074   3rd Qu.:1660   3rd Qu.:1856   3rd Qu.:4080
##                     Max.   :2479   Max.   :1880   Max.   :2187   Max.   :4662
##      Pbone          Length         Height         Rostrum
##  Min.   :3928   Min.   :1908   Min.   :700.0   Min.   :375.0
##  1st Qu.:4815   1st Qu.:2227   1st Qu.:759.2   1st Qu.:425.0
##  Median :5079   Median :2312   Median :789.0   Median :450.0
##  Mean   :5082   Mean   :2309   Mean   :790.8   Mean   :451.2
##  3rd Qu.:5328   3rd Qu.:2388   3rd Qu.:817.8   3rd Qu.:475.0
##  Max.   :6104   Max.   :2605   Max.   :912.0   Max.   :545.0
##        final_pred
##  multiplex   :124
##  subterraneus:164
##
##
##
##
```