dpahw2.R

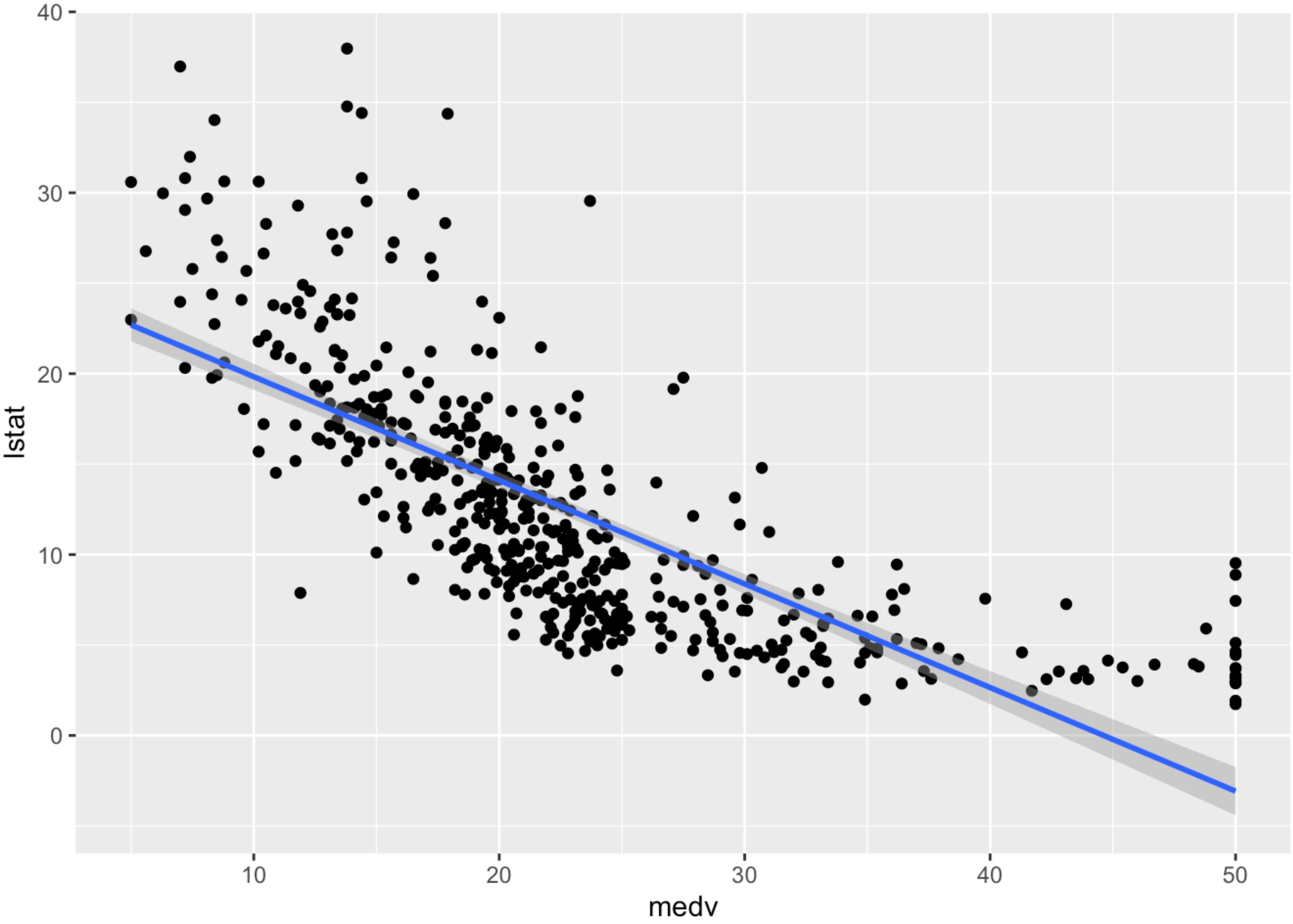
renahaswah

2019-10-24

```
library(MASS)
library(ggplot2)
library(boot)

#Load the Boston sample dataset into R using a dataframe
data(Boston)
#Use lm to fit a regression between medv and lstat
model<-lm(medv~lstat, data=Boston)
summary(model)
```

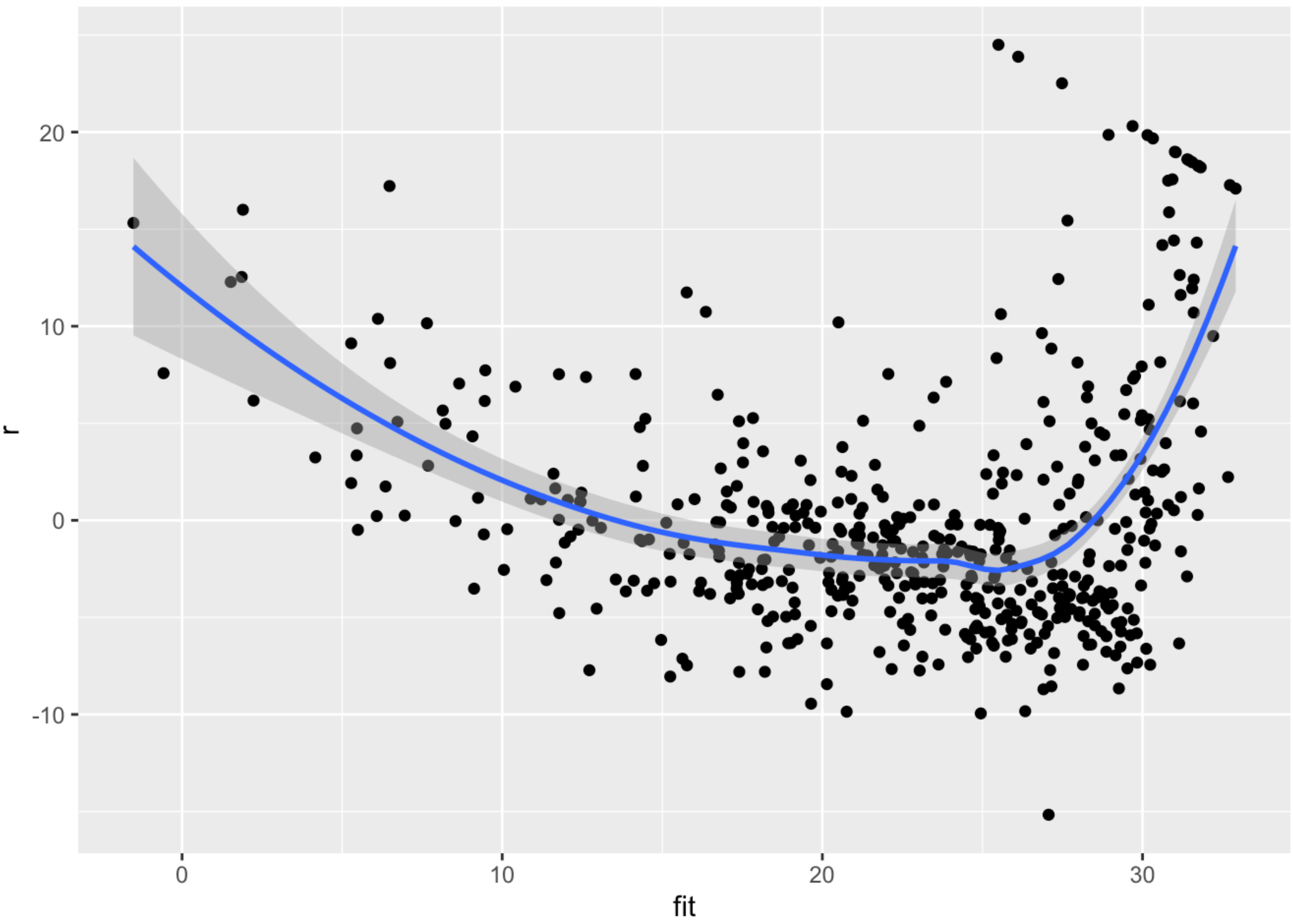
```
## 
## Call:
## lm(formula = medv ~ lstat, data = Boston)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -15.168  -3.990  -1.318   2.034  24.500 
## 
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 34.55384    0.56263   61.41   <2e-16 ***
## lstat       -0.95005    0.03873  -24.53   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 6.216 on 504 degrees of freedom
## Multiple R-squared:  0.5441, Adjusted R-squared:  0.5432 
## F-statistic: 601.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

```
# plot the resulting fit
ggplot(Boston,aes(x=medv,y=lstat)) +
  geom_point() + geom_smooth(method=lm)
```



```
#and show a plot of fitted values vs. residuals.
metadata<-data.frame("fit"=fitted(model),
                     "r"=resid(model))
ggplot(data = metadata)+
  geom_point(mapping = aes(x = fit,y=r))+
  geom_smooth(mapping = aes(x = fit,y=r))
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



```
#Is there a possible non-linear relationship between
#the predictor and response?
#Yeah. That looks very non-linear.


#Use the predict function to calculate values response
#values for lstat of 5, 10, and 15
#obtain confidence intervals as well as prediction intervals
#for the results
predict(model,data.frame(lstat=c(5,10,15)), interval = 'confidence',level=.95)
```

```
##        fit      lwr      upr
## 1 29.80359 29.00741 30.59978
## 2 25.05335 24.47413 25.63256
## 3 20.30310 19.73159 20.87461
```

```
predict(model,data.frame(lstat=c(5,10,15)), interval = 'prediction',level=.95)
```

```
##        fit       lwr      upr
## 1 29.80359 17.565675 42.04151
## 2 25.05335 12.827626 37.27907
## 3 20.30310  8.077742 32.52846
```

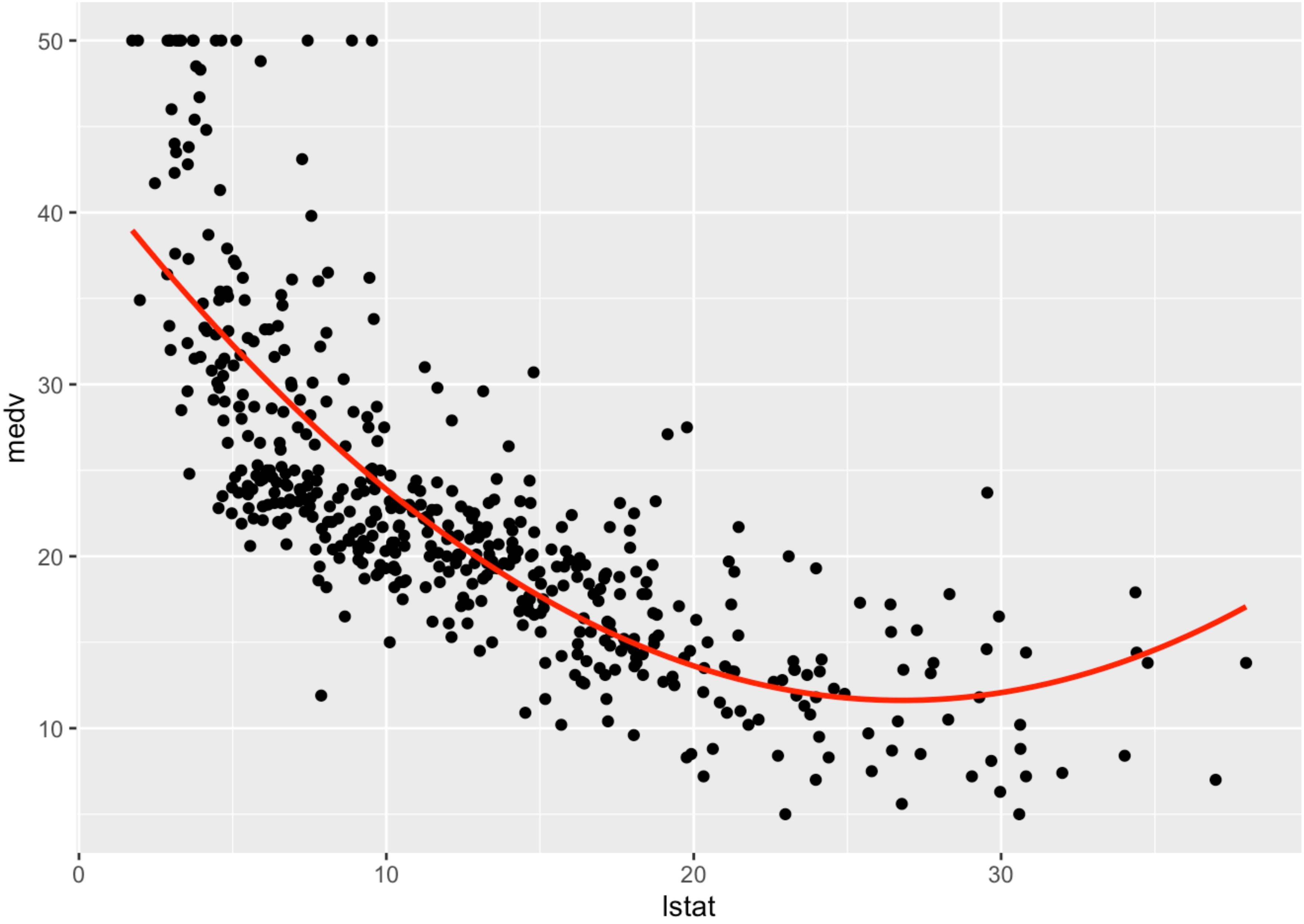
```
#are they the same? Why or why not?
#No. The prediction interval is actually much wider. This is because it is
#taking into account the variance of the error term for new response values.



#Modify the regression to include lstat2
mod2<-lm(medv~lstat+I(lstat^2), data=Boston)
#compare the R2 between the linear and non-linear fit
summary(mod2)$r.squared
```

```
## [1] 0.6407169
```

```
summary(model)$r.squared
```

```
## [1] 0.5441463
```

```
#use ggplot2 and stat smooth to plot the relationship.
ggplot(Boston, aes(x = lstat, y = medv)) +
  geom_point() + stat_smooth(formula = y~x+I(x^2),
                             method = "lm", se= FALSE, color = "red")
```



```
#PRACTICUM PROBLEM #2
#Load the abalone sample dataset from the UCI Machine Learning
#Repository (abalone.data) into R using a dataframe.
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:boot':
##
##     melanoma
```

```
col_names=c("Sex", "Length", "Diameter", "Height", "Whole_Weight","Shucked_Weight", "
Viscera_Weight", "Shell_Weight", "Rings")
df = read.csv('https://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abal
one.data', header=FALSE, sep=',', col.names = col_names)
head(df)
```

```
##   Sex Length Diameter Height Whole_Weight Shucked_Weight Viscera_Weight
## 1   M  0.455    0.365  0.095       0.5140         0.2245         0.1010
## 2   M  0.350    0.265  0.090       0.2255         0.0995         0.0485
## 3   F  0.530    0.420  0.135       0.6770         0.2565         0.1415
## 4   M  0.440    0.365  0.125       0.5160         0.2155         0.1140
## 5   I  0.330    0.255  0.080       0.2050         0.0895         0.0395
## 6   I  0.425    0.300  0.095       0.3515         0.1410         0.0775
##   Shell_Weight Rings
## 1        0.150    15
## 2        0.070     7
## 3        0.210     9
## 4        0.155    10
## 5        0.055     7
## 6        0.120     8
```

```
#Remove all observations in the Infant category, keeping the Male/Female classes.
d= df[df$Sex!='I',]
d$Sex <- factor(d$Sex, labels = c("Male","Female"))

#Using the caret package, use createDataPartition to perform an 80/20 test-train spli
t
splitdata <- createDataPartition(d$Sex, p=0.8, list=FALSE, times=1)
train <- d[ splitdata,]
test  <- d[-splitdata,]

#Fit a logistic regression using all feature variables
logfit<-glm(Sex~.,data=train,family=binomial)
summary(logfit)
```

```
##
## Call:
## glm(formula = Sex ~ ., family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6784  -1.2087   0.8999   1.1133   1.4649
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)      2.58745    0.50858   5.088 3.63e-07 ***
## Length          -2.30243    2.27247  -1.013  0.31097
## Diameter        -3.06324    2.71584  -1.128  0.25936
## Height          -3.78253    2.25576  -1.677  0.09358 .
## Whole_Weight     0.20787    0.81378   0.255  0.79838
## Shucked_Weight   2.76007    0.99543   2.773  0.00556 **
## Viscera_Weight -2.83654    1.42996  -1.984  0.04729 *
## Shell_Weight     0.53653    1.24470   0.431  0.66643
## Rings           -0.01016    0.01787  -0.569  0.56948
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3131.7  on 2268  degrees of freedom
## Residual deviance: 3079.4  on 2260  degrees of freedom
## AIC: 3097.4
##
## Number of Fisher Scoring iterations: 4
```

```
#Length, Diameter, and Height are most significant.
exp(coef(logfit))
```

```
##    (Intercept)         Length       Diameter         Height   Whole_Weight
##    13.29589004     0.10001535     0.04673622     0.02276491     1.23105649
## Shucked_Weight Viscera_Weight   Shell_Weight          Rings
##    15.80089280     0.05862795     1.71007074     0.98988771
```

```
#Do the confidence intervals for the predictors
#contain 0 within the range?
#Diameter, Height, Shucked_Weight do not contain 0.
confint(logfit)
```

```
## Waiting for profiling to be done...
##                      2.5 %      97.5 %
## (Intercept)     1.60446189  3.59937635
## Length         -6.76216311  2.15226841
## Diameter       -8.39733600  2.25758465
## Height         -8.60997544 -0.07682158
## Whole_Weight   -1.39075477  1.81157579
## Shucked_Weight  0.81262605  4.72314172
## Viscera_Weight -5.65314194 -0.04043521
## Shell_Weight   -1.90808795  2.98240293
## Rings          -0.04522802  0.02486421
```

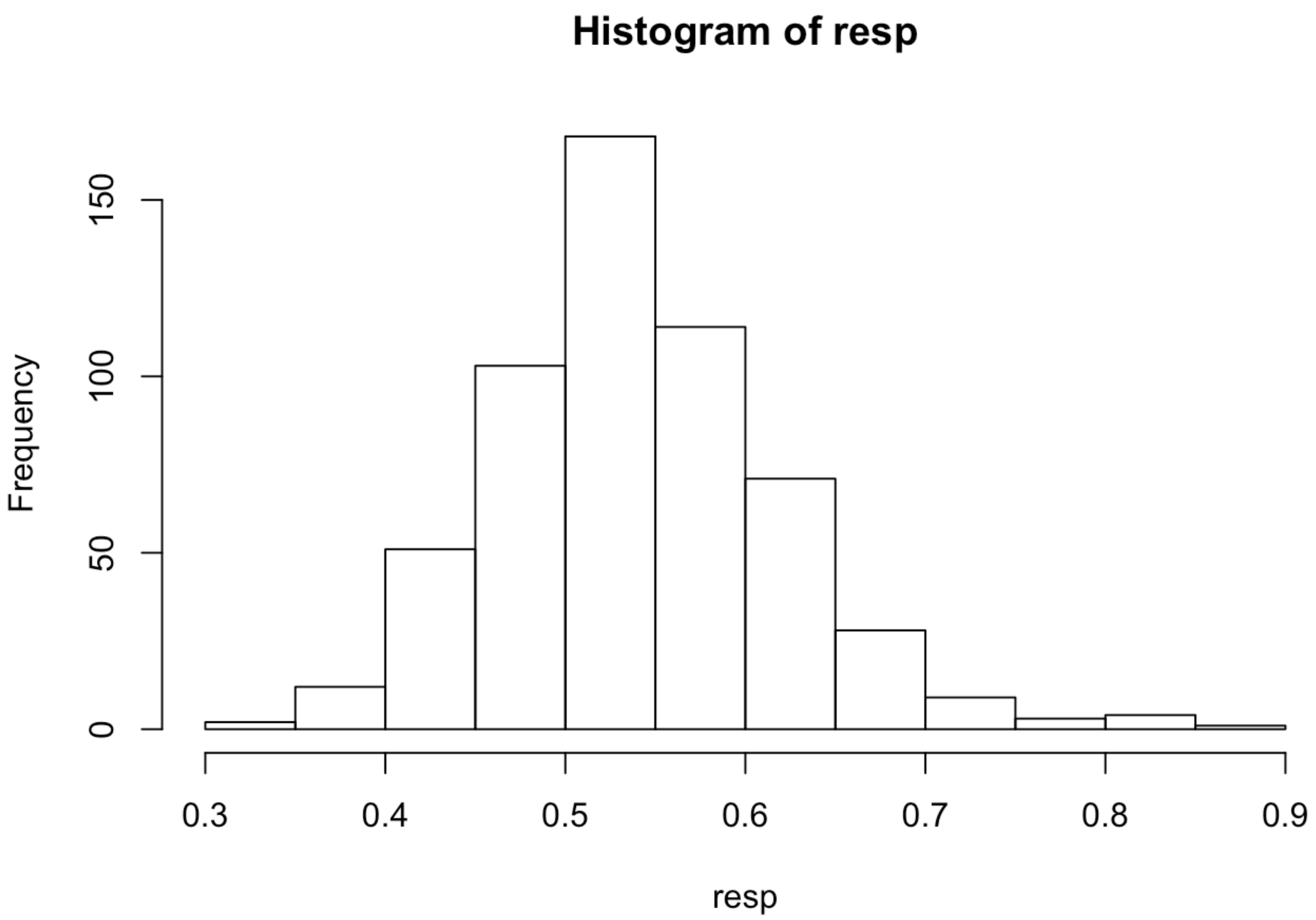
```
#How does this relate to the null hypothesis?
#Since zero is in the interval, the null CANNOT be rejected for this confidence level
!
#The ones without zero in the interval are significant for the regression,
# and thus are good to reject null, though.

#Use the confusionMatrix function in caret to observe testing results
#tofix
pred<-predict(logfit,newdata = test)
pred.dt<-ifelse(pred>0.50, "M","F")
Pred <- as.factor(pred.dt)
Predicted <- ordered(Pred, levels = c("M", "F"))
Actual <- ordered(test$Sex,levels = c("M", "F"))
install.packages('e1071', dependencies=TRUE, repos = "http://cran.us.r-project.org")
```

```
##
## The downloaded binary packages are in
##  /var/folders/20/cfr03_6909g0l0pj45zvcc440000gn/T//Rtmp1oO804/downloaded_packages
```

```
confusionMatrix(table(Predicted,Actual))
```

```
## Confusion Matrix and Statistics
##
##          Actual
## Predicted M F
##         M 0 0
##         F 0 0
##
##                Accuracy : NaN
##                  95% CI : (NA, NA)
##     No Information Rate : NA
##     P-Value [Acc > NIR] : NA
##
##                   Kappa : NaN
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity :  NA
##             Specificity :  NA
##          Pos Pred Value :  NA
##          Neg Pred Value :  NA
##              Prevalence : NaN
##          Detection Rate : NaN
##    Detection Prevalence : NaN
##       Balanced Accuracy :  NA
##
##        'Positive' Class : M
##
```

```
#how does the accuracy compare to a random classifier ROC curve?
resp <- predict(logfit, test, type = "response")
test$resp=resp
hist(resp)
```



```
head(test)
```

```
##       Sex Length Diameter Height Whole_Weight Shucked_Weight
## 9  Female  0.475    0.370  0.125       0.5095         0.2165
## 14   Male  0.535    0.405  0.145       0.6845         0.2725
## 19 Female  0.365    0.295  0.080       0.2555         0.0970
## 23   Male  0.565    0.440  0.155       0.9395         0.4275
## 24   Male  0.550    0.415  0.135       0.7635         0.3180
## 35   Male  0.705    0.550  0.200       1.7095         0.6330
##    Viscera_Weight Shell_Weight Rings      resp
## 9          0.1125        0.165     9 0.5668461
## 14         0.1710        0.205    10 0.4960716
## 19         0.0430        0.100     7 0.6730877
## 23         0.2140        0.270    12 0.5358305
## 24         0.2100        0.200     9 0.4989508
## 35         0.4115        0.490    13 0.3986651
```

```
pred <- ifelse(resp > 0.5, "Male", "Female")

#ROC:
test$Sex
```

```
##   [1] Female Male   Female Male   Male   Male   Male   Female Male   Female
##  [11] Female Male   Female Male   Male   Male   Male   Male   Female Female
##  [21] Female Female Female Male   Male   Female Male   Female Male   Male
##  [31] Female Male   Male   Male   Male   Female Male   Male   Male   Female
##  [41] Female Male   Female Female Male   Female Female Female Male   Female
##  [51] Male   Female Male   Male   Male   Female Female Male   Male   Female
##  [61] Female Female Female Female Female Male   Male   Female Female Female
##  [71] Male   Male   Male   Female Female Male   Male   Male   Male   Female
##  [81] Female Female Male   Male   Male   Male   Male   Male   Male   Female
##  [91] Male   Female Female Male   Male   Female Female Male   Female Female
## [101] Male   Female Female Female Female Female Female Female Female Female
## [111] Female Female Female Female Female Male   Female Male   Female Male
## [121] Female Male   Female Male   Female Female Female Male   Female Female
## [131] Female Male   Male   Female Male   Male   Female Male   Male   Female
## [141] Male   Male   Male   Female Female Female Female Male   Female Female
## [151] Female Female Female Male   Female Female Male   Male   Male   Male
## [161] Female Male   Female Male   Male   Female Male   Female Male   Male
## [171] Male   Female Female Female Male   Female Female Female Female Male
## [181] Male   Female Female Female Female Female Male   Male   Male   Male
## [191] Male   Female Male   Female Female Female Female Male   Female Male
## [201] Female Female Female Female Female Female Female Male   Male   Female
## [211] Female Female Female Female Male   Male   Male   Male   Female Female
## [221] Male   Female Female Male   Male   Female Female Male   Male   Male
## [231] Female Female Female Female Male   Female Male   Male   Female Male
## [241] Female Female Male   Male   Male   Female Female Female Male   Female
## [251] Female Male   Female Male   Female Female Female Female Female Female
## [261] Female Female Male   Female Female Male   Male   Male   Female Male
## [271] Male   Female Male   Female Male   Female Female Female Male   Female
## [281] Female Female Female Female Male   Female Male   Female Male   Male
## [291] Female Female Female Male   Male   Male   Female Male   Female Female
## [301] Female Male   Male   Female Male   Male   Female Male   Male   Female
## [311] Male   Female Male   Male   Female Female Male   Male   Male   Male
## [321] Male   Male   Male   Female Male   Female Female Female Male   Male
## [331] Female Female Female Male   Female Male   Male   Male   Male   Female
## [341] Female Male   Female Female Female Male   Male   Female Male   Male
## [351] Female Female Female Male   Female Male   Male   Male   Male   Female
## [361] Female Female Male   Male   Male   Female Male   Male   Female Male
## [371] Male   Female Male   Male   Male   Female Female Male   Male   Male
## [381] Male   Female Female Male   Male   Female Male   Female Female Male
## [391] Male   Male   Male   Male   Female Female Female Male   Female Male
## [401] Female Female Female Female Male   Male   Female Female Female Male
## [411] Female Female Female Female Female Male   Female Male   Female Male
## [421] Female Male   Female Female Male   Female Female Male   Male   Female
## [431] Male   Female Female Male   Female Female Female Male   Male   Male
## [441] Female Male   Male   Female Male   Male   Female Female Male   Female
## [451] Female Male   Male   Female Male   Male   Female Female Female Female
## [461] Male   Female Female Female Female Female Female Female Female Female
## [471] Male   Male   Female Female Male   Male   Male   Female Male   Male
## [481] Female Female Male   Female Male   Female Female Female Male   Male
## [491] Male   Male   Female Female Female Female Male   Male   Male   Female
## [501] Male   Male   Male   Male   Female Female Female Male   Female Male  
## [511] Female Female Female Male   Female Male   Female Male   Male   Female
## [521] Male   Male   Female Male   Female Female Male   Female Female Female
## [531] Female Male   Female Male   Male   Female Male   Female Female Male  
## [541] Female Female Male   Female Male   Male   Male   Female Male   Female
## [551] Female Male   Female Male   Male   Female Male   Male   Female Female
## [561] Female Female Female Female Female Male  
## Levels: Male Female
```

```
test$resp
```

```
##   [1] 0.5668461 0.4960716 0.6730877 0.5358305 0.4989508 0.3986651 0.5327921
##   [8] 0.5465939 0.5519790 0.6027607 0.5915892 0.5450719 0.5989986 0.4773770
##  [15] 0.5515009 0.6045070 0.4935867 0.4644712 0.4959118 0.4570417 0.4904618
##  [22] 0.5118166 0.5357412 0.5388965 0.5975725 0.6966376 0.5743316 0.5995505
##  [29] 0.4468983 0.4629013 0.4693113 0.4790409 0.3723357 0.4677308 0.4985956
##  [36] 0.5057422 0.5926422 0.5785229 0.3954664 0.3997277 0.4364846 0.4962850
##  [43] 0.5277116 0.4892106 0.4484078 0.5130537 0.4851080 0.5108863 0.4778066
##  [50] 0.5227131 0.3609073 0.6319151 0.4521440 0.5599465 0.4439347 0.4460299
##  [57] 0.5516953 0.4838954 0.4873282 0.5022715 0.5230501 0.5392199 0.5079915
##  [64] 0.4762562 0.4686946 0.4962906 0.6246858 0.5144318 0.4913747 0.3777182
##  [71] 0.4578004 0.4630344 0.5349405 0.4704122 0.3756351 0.4638392 0.5199512
##  [78] 0.3954648 0.3138060 0.4227928 0.8110550 0.8356953 0.4474765 0.6020609
##  [85] 0.5675315 0.4902499 0.6034984 0.5153192 0.6565797 0.4826696 0.4069514
##  [92] 0.6665347 0.6950090 0.5907337 0.5166201 0.7080127 0.5689831 0.4528486
##  [99] 0.4373119 0.5522286 0.5958949 0.6239086 0.5756415 0.6126107 0.6881769
## [106] 0.6277426 0.5486538 0.5039892 0.5694987 0.5454492 0.5140351 0.5192635
## [113] 0.5183255 0.5381994 0.4189532 0.4085597 0.3895280 0.4138439 0.4818681
## [120] 0.4266829 0.5069127 0.4696461 0.5176903 0.5009904 0.5131377 0.4650143
## [127] 0.4886918 0.5691620 0.5285100 0.5113298 0.6597389 0.6128928 0.6012789
## [134] 0.6377931 0.5215542 0.5825566 0.4888447 0.5609445 0.6181928 0.6062767
## [141] 0.4273131 0.5118755 0.5224525 0.5811636 0.6384335 0.5812892 0.6003559
## [148] 0.5881560 0.5600896 0.6368609 0.5863594 0.5865697 0.6015941 0.5245141
## [155] 0.5279483 0.5232452 0.5641314 0.4604865 0.5342330 0.6268894 0.5820080
## [162] 0.5365822 0.5458880 0.4912851 0.4649940 0.6195446 0.5381344 0.5383016
## [169] 0.6701721 0.5219838 0.5514067 0.6378951 0.5978535 0.4994032 0.5465142
## [176] 0.5832939 0.6163933 0.5519511 0.7411869 0.5018496 0.4824240 0.5194899
## [183] 0.4309046 0.5264860 0.4691957 0.6957803 0.6348307 0.6814613 0.8170313
## [190] 0.5654373 0.5752739 0.5337844 0.4973612 0.6663700 0.5385175 0.6050935
## [197] 0.5975231 0.5071481 0.4907683 0.6425435 0.4316076 0.4275418 0.4817984
## [204] 0.5388186 0.5558027 0.7046328 0.4997711 0.3964985 0.7141036 0.6502045
## [211] 0.6533877 0.5650119 0.6548199 0.5283922 0.5495859 0.5420135 0.4446882
## [218] 0.5692062 0.5680549 0.5791304 0.6083295 0.5165882 0.4941555 0.4945150
## [225] 0.5695682 0.5366016 0.5106388 0.4770799 0.5114976 0.4705981 0.5117680
## [232] 0.4828353 0.6224516 0.5468350 0.5148939 0.6000628 0.5360544 0.4186976
## [239] 0.4104806 0.4656092 0.5918900 0.5674636 0.6052896 0.4500909 0.4212853
## [246] 0.5645647 0.6101206 0.6801329 0.5603880 0.6580269 0.5531182 0.5361599
## [253] 0.5259771 0.5223608 0.6027272 0.5456447 0.5472643 0.5730244 0.4419292
## [260] 0.4742960 0.5428039 0.4634055 0.4291198 0.4958026 0.6022004 0.4637316
## [267] 0.5321754 0.4858280 0.6013107 0.4495430 0.5681463 0.5017213 0.4368982
## [274] 0.6004428 0.5488779 0.5563226 0.5401149 0.5220219 0.4770501 0.5390987
## [281] 0.5273185 0.6102815 0.5519725 0.6176311 0.5617166 0.5515622 0.5450798
## [288] 0.6003451 0.5507025 0.5124573 0.5261486 0.6494026 0.5146149 0.4227128
## [295] 0.6782276 0.5397209 0.7171645 0.4974716 0.5656917 0.6025836 0.6773673
## [302] 0.4455426 0.5269940 0.5623361 0.5489416 0.4890293 0.4980994 0.5170351
## [309] 0.4788278 0.4483091 0.6086038 0.6671528 0.5065670 0.5227582 0.6584395
## [316] 0.5998179 0.5336215 0.4348662 0.4531279 0.5184189 0.6666672 0.5541386
## [323] 0.5029361 0.6239893 0.5505178 0.5058692 0.5212779 0.4754955 0.4452410
## [330] 0.5153242 0.5431390 0.4621017 0.5274619 0.5176004 0.8522798 0.5959155
## [337] 0.4811937 0.4027476 0.5632189 0.6830280 0.7595541 0.5543974 0.4800078
## [344] 0.4983571 0.5625611 0.4775785 0.4287312 0.5841513 0.6785123 0.5943173
## [351] 0.5846544 0.5679659 0.5284499 0.5958469 0.5281384 0.5196447 0.5590798
## [358] 0.5057508 0.5892816 0.5676690 0.6049538 0.5289583 0.5475118 0.5604929
## [365] 0.6097502 0.5780149 0.6124521 0.5104815 0.6267954 0.4919442 0.5319482
## [372] 0.4328362 0.4794286 0.5097583 0.5415964 0.5000037 0.6083891 0.4517206
## [379] 0.5456059 0.5032181 0.5312279 0.5600430 0.5487348 0.5288404 0.5183110
## [386] 0.6796343 0.6079723 0.5324881 0.5752316 0.4720937 0.6745949 0.5667463
## [393] 0.5270127 0.5897118 0.6090649 0.5718066 0.5596999 0.5574734 0.5367171
## [400] 0.5608882 0.4567221 0.4951750 0.4587398 0.4683309 0.5239084 0.5375612
## [407] 0.4681507 0.6428294 0.6801376 0.5867196 0.6147538 0.5481027 0.7523405
## [414] 0.5212894 0.5170109 0.4790053 0.4729754 0.5260836 0.3349775 0.5950359
## [421] 0.5377399 0.5145007 0.4821556 0.4339931 0.6301857 0.5729816 0.5547086
## [428] 0.4792009 0.5650211 0.6209984 0.7280724 0.5076152 0.4558907 0.4192054
## [435] 0.5686477 0.5859468 0.4436649 0.5347547 0.5178395 0.3735690 0.3925919
## [442] 0.5866155 0.4654788 0.5156148 0.5509512 0.4148879 0.4093090 0.7129822
## [449] 0.4822100 0.5260716 0.5951227 0.4313875 0.4286531 0.7424876 0.5022237
## [456] 0.4656201 0.6111545 0.6142241 0.6253090 0.7460378 0.5224069 0.5683970
## [463] 0.5328523 0.5263307 0.4394232 0.4783275 0.5324153 0.6249662 0.5459555
## [470] 0.6496697 0.5954142 0.5346528 0.5753489 0.5850074 0.5487011 0.5741550
## [477] 0.5131028 0.6252055 0.5266987 0.5828964 0.6183121 0.5903238 0.5071813
## [484] 0.8093021 0.6066854 0.6304964 0.5407485 0.5431264 0.4109017 0.5206235
## [491] 0.5258712 0.5116619 0.5213651 0.5345704 0.5891908 0.5110020 0.5127329
## [498] 0.4788563 0.4806856 0.6335516 0.4326144 0.7678839 0.5662434 0.4790134
## [505] 0.5188705 0.4655628 0.6090363 0.5506087 0.4542586 0.5092487 0.6356343
## [512] 0.5651199 0.5261478 0.4316245 0.5491233 0.6534972 0.4446646 0.4921123
## [519] 0.4875861 0.5661690 0.5679725 0.4930596 0.4360353 0.4864510 0.5647570
## [526] 0.4256548 0.5323221 0.5066074 0.4416362 0.5411465 0.5139911 0.4326765
## [533] 0.5043469 0.5267957 0.5657197 0.4893932 0.5221741 0.6052269 0.6170162
## [540] 0.5292689 0.5303404 0.4742491 0.5343209 0.5549216 0.6011834 0.6325069
## [547] 0.5585572 0.4627075 0.4812736 0.5424293 0.6308661 0.5337569 0.5165221
## [554] 0.6049066 0.5506557 0.4604029 0.4483433 0.5670880 0.6041710 0.4156524
## [561] 0.6815567 0.5729395 0.5674229 0.5075898 0.5578981 0.4584650
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

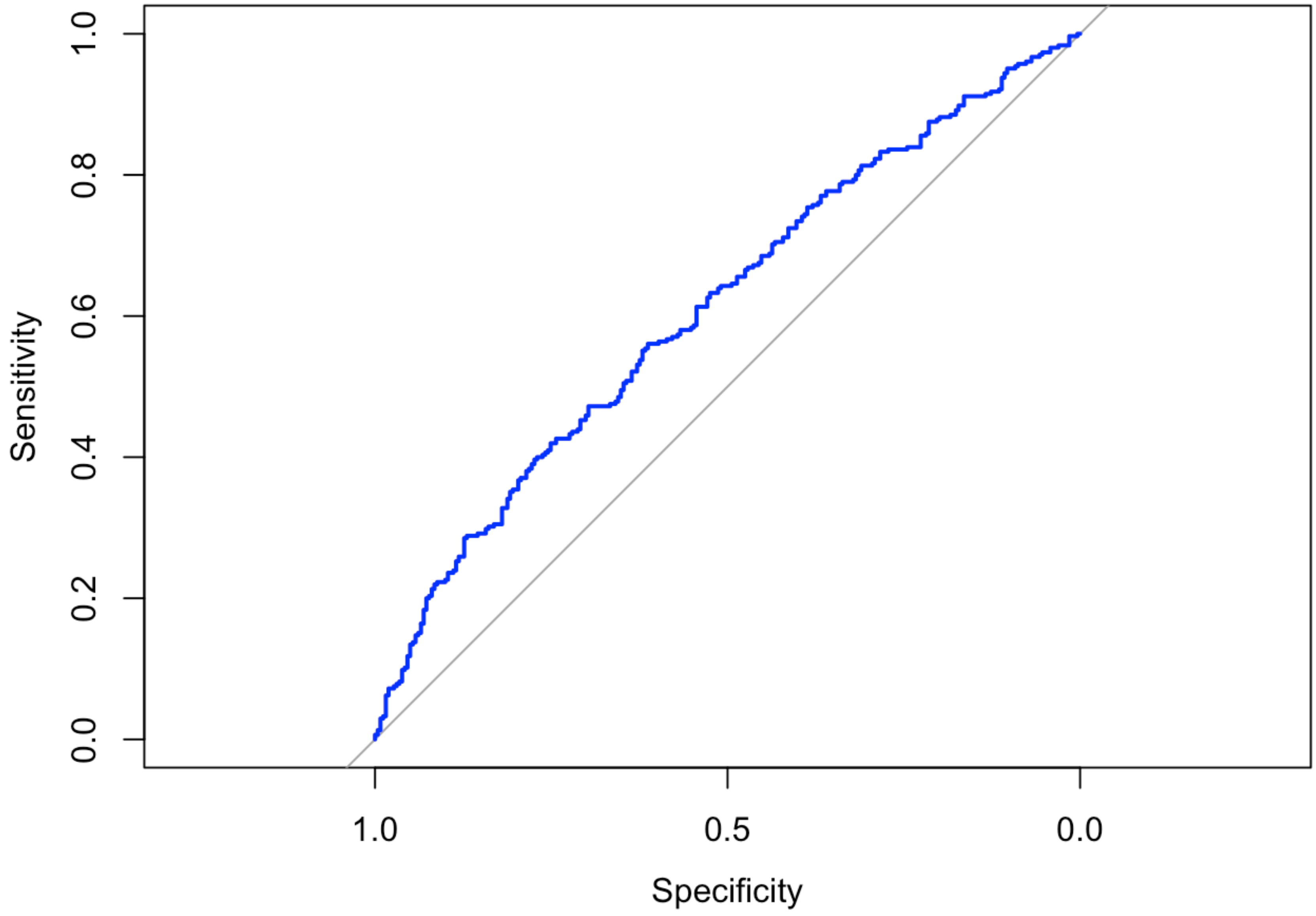
```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
ROC1 <- roc(as.numeric(test$Sex), test$resp)
```

```
## Setting levels: control = 1, case = 2
```

```
## Setting direction: controls < cases
```

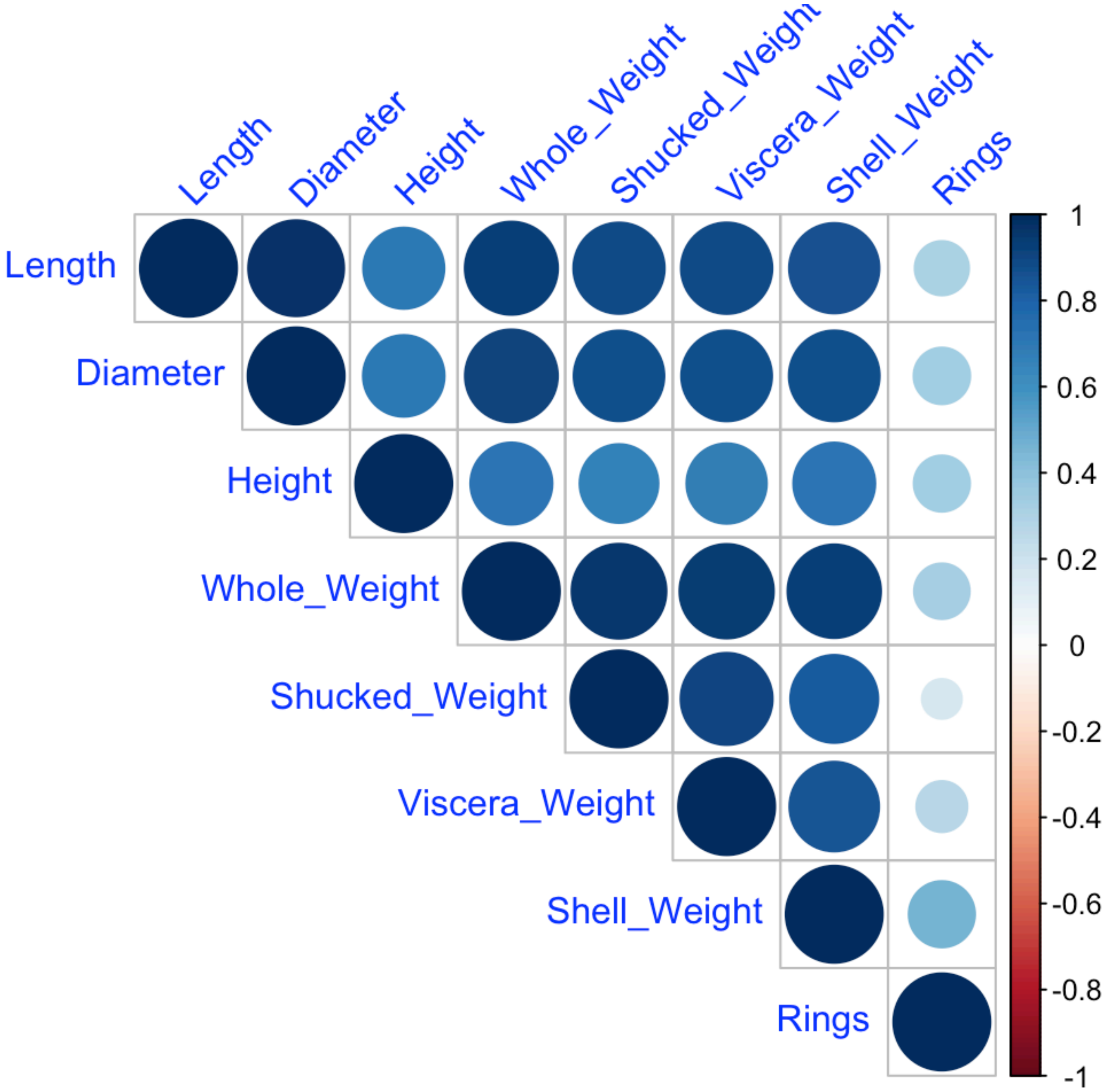
```
plot(ROC1, col = "blue")
```



```
#We calculate the corr matrix
predictors<-d[c(-1)]
corMatrix <- cor(predictors)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
corrplot(corMatrix,type = "upper", method = "circle",diag = TRUE, tl.col = "blue",tl.
srt = 45,)
```



```
#Load the mushroom sample dataset from the UCI Machine Learning Repository
names <- c("edibility","cap-shape","cap-surface","cap-color","bruises","odor","gill-attachment","gill-spacing","gill-size",
           "gill- color","stalk-shape","stalk-root","stalk-surface-above-ring","stalk-surface-below-ring","stalk-color-above-ring",
           "stalk-color- below-ring", "veil-type","veil-color","ring-number","ring-type","spore-print-color","population","habitat")
mush<-read.csv("/Users/renahaswah/Desktop/Data Prep/HW/HW2/agaricus-lepiota.data",header=FALSE,col.names = names)
#Some values in stalk.root have "?" values.
mush[mush == '?'] <- NA
num<-is.na(mush$stalk.root)
sum(num)
```

```
## [1] 2480
```

```
# Mean, Median, Mode Imputation is acceptable, but  KNN is
#recently taught in this course, so let's do that.
library(VIM)
```

```
## Loading required package: colorspace
```

```
##
## Attaching package: 'colorspace'
```

```
## The following object is masked from 'package:pROC':
##
##     coords
```

```
## Loading required package: grid
```

```
## Loading required package: data.table
```

```
## VIM is ready to use.
##  Since version 4.0.0 the GUI is in its own package VIMGUI.
##
##           Please use the package to use the new (and old) GUI.
```

```
## Suggestions and bug-reports can be submitted at: https://github.com/alexkowa/VIM/issues
```

```
##
## Attaching package: 'VIM'
```

```
## The following object is masked from 'package:datasets':
##
##     sleep
```

```
y<-kNN(mush,variable = colnames(mush[12]), k=5)
y[y=='?']<-NA
anyNA(y)
```

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

```
summary(mush)
```

```
##  edibility cap.shape cap.surface   cap.color    bruises       odor     
##  e:4208    b: 452    f:2320      n      :2284   f:4748   n      :3528  
##  p:3916    c:   4    g:   4      g      :1840   t:3376   f      :2160  
##            f:3152    s:2556      e      :1500            s      : 576  
##            k: 828    y:3244      y      :1072            y      : 576  
##            s:  32                w      :1040            a      : 400  
##            x:3656                b      : 168            l      : 400  
##                                  (Other): 220            (Other): 484  
##  gill.attachment gill.spacing gill.size  gill..color   stalk.shape
##  a: 210          c:6812       b:5612    b      :1728   e:3516     
##  f:7914          w:1312       n:2512    p      :1492   t:4608     
##                                         w      :1202              
##                                         n      :1048              
##                                         g      : 752              
##                                         h      : 732              
##                                         (Other):1170              
##  stalk.root  stalk.surface.above.ring stalk.surface.below.ring
##  ?   :   0   f: 552                   f: 600                  
##  b   :3776   k:2372                   k:2304                  
##  c   : 556   s:5176                   s:4936                  
##  e   :1120   y:  24                   y: 284                  
##  r   : 192                                                    
##  NA's:2480                                                    
##                                                               
##  stalk.color.above.ring stalk.color..below.ring veil.type veil.color
##  w      :4464           w      :4384            p:8124    n:  96    
##  p      :1872           p      :1872                      o:  96    
##  g      : 576           g      : 576                      w:7924    
##  n      : 448           n      : 512                      y:   8    
##  b      : 432           b      : 432                                
##  o      : 192           o      : 192                                
##  (Other): 140           (Other): 156                                
##  ring.number ring.type spore.print.color population habitat
##  n:  36      e:2776    w      :2388      a: 384     d:3148 
##  o:7488      f:  48    n      :1968      c: 340     g:2148 
##  t: 600      l:1296    k      :1872      n: 400     l: 832 
##              n:  36    h      :1632      s:1248     m: 292 
##              p:3968    r      :  72      v:4040     p:1144 
##                        b      :  48      y:1712     u: 368 
##                        (Other): 144                 w: 192 
```

```
library(caret)
library(e1071)
library(caTools)
#Create a Naive Bayes classifier using the e1071 package, using
#the sample func- tion to split the data between 80% for training and 20% for testing
.
train_sample <- sample(8124, floor(.8*8124))
train <- mush[train_sample, ]
test  <- mush[-train_sample, ]

library(naivebayes)
```

```
## naivebayes 0.9.6 loaded
```

```
##
## Attaching package: 'naivebayes'
```

```
## The following object is masked from 'package:data.table':
##
##     tables
```

```
nbmodl<-naiveBayes(train$edibility~., train)

#With the target class of interest being edible mushrooms, calculate the accuracy of
the
#classifier both in-training and in-test.
#Accuracy is the percentage of values the model predicted correclty.
#In training
pt<-predict(nbmodl,train,type="class")
cmmush<-table(pt, train$edibility,dnn=c("Prediction","Actual"))
n<-sum(cmmush)
dig<-diag(cmmush)
acc<-sum(dig)/n
acc
```

```
## [1] 0.9422988
```

```
#In test
p<-predict(nbmodl, test, type = "class")
cmmush<-table(p, test$edibility,dnn=c("Prediction","Actual"))
n<-sum(cmmush)
dig<-diag(cmmush)
acc<-sum(dig)/n
acc
```

```
## [1] 0.9507692
```

```
#Use the table function to create a con- fusion matrix of predicted vs. actual classe
s -
table(p, test$edibility,dnn=c("Prediction","Actual"))
```

```
##           Actual
## Prediction   e   p
##          e 846  72
##          p   8 699
```

```
#how many false positives did the model produce?
#Let's say edible is true.
#There are 89 values that were falsely identified as true edible.
```