449_FinalProject

Jasmin Martinez

2024-05-17

```r
#Load necessary libraries
library(car)
```

```
## Loading required package: carData
```

```r
library(MASS)
library(pROC)
```

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

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

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

```r
# Libraries for visualization
library(ggplot2)
library(pROC)
library(caret)
```

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

State the problem and describe the data set

The data set that I have decided to work with is one that I found on kaggle called Mushroom Data set for Binary Classification where different information on thousands of mushrooms was taken in an effort to determine which factors should be looked at when trying to determine whether the mushroom is edible or poisonous. This data set is cleaned up version of the original Mushroom Dataset for Binary Classification Available at UCI Library. It consists of 8 predictive variable pertaining to the mushrooms, including Cap Diameter, Cap Shape, Gill Attachment, Gill Color, Stem Height, Stem Width, Stem Color and Season. And finally the target class, which contains two values - 0 or 1 - where 0 refers to edible and 1 refers to poisonous.

1. cap-diameter (m): float number in cm
2. cap-shape (n): bell=b, conical=c, convex=x, flat=f, sunken=s, spherical=p, others=o
3. gill-attachment (n): adnate=a, adnexed=x, decurrent=d, free=e, sinuate=s, pores=p, none=f, unknown=?
4. gill-color (n): see cap-color + none=f 5.stem-height (m): float number in cm
5. stem-width (m): float number in mm 7.stem-color (n): see cap-color + none=f 8.season (n): spring=s, summer=u, autumn=a, winter=w

Class Labels edible=e, poisonous=p

```r
mush_data <- read.csv("mushroom_cleaned.csv")
```

Fit a logistic regression model with all predictors

```r
mushroom_full= glm(class ~ cap.diameter+ factor(cap.shape) + factor(gill.attachment) + factor(gill.colo
summary(mushroom_full)
```

```
##
## Call:
## glm(formula = class ~ cap.diameter + factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height + stem.width + factor(stem.color) +
##     season, family = binomial, data = mush_data)
##
## Coefficients: (1 not defined because of singularities)
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -1.551e+01  1.722e+02  -0.090  0.92822
## cap.diameter             -3.989e-04  5.832e-05  -6.840 7.92e-12 ***
## factor(cap.shape)1       -8.310e-01  6.743e-02 -12.324  < 2e-16 ***
## factor(cap.shape)2       -1.183e+00  4.630e-02 -25.562  < 2e-16 ***
## factor(cap.shape)3        9.119e-01  9.879e-02   9.231  < 2e-16 ***
## factor(cap.shape)4       -9.363e-01  6.928e-02 -13.514  < 2e-16 ***
## factor(cap.shape)5       -9.766e-01  5.417e-02 -18.030  < 2e-16 ***
## factor(cap.shape)6       -1.254e+00  4.369e-02 -28.703  < 2e-16 ***
## factor(gill.attachment)1  4.319e-01  3.495e-02  12.359  < 2e-16 ***
## factor(gill.attachment)2 -5.752e-01  4.330e-02 -13.284  < 2e-16 ***
## factor(gill.attachment)3 -6.538e-01  1.065e-01  -6.136 8.44e-10 ***
## factor(gill.attachment)4 -1.900e+00  4.617e-02 -41.156  < 2e-16 ***
## factor(gill.attachment)5  1.987e-01  3.973e-02   5.002 5.67e-07 ***
## factor(gill.attachment)6  4.833e-01  3.484e-02  13.872  < 2e-16 ***
## factor(gill.color)1       1.850e+00  1.191e-01  15.539  < 2e-16 ***
## factor(gill.color)2              NA         NA      NA       NA
## factor(gill.color)3       1.774e-01  9.443e-02   1.879  0.06027 .
## factor(gill.color)4       4.704e-01  1.052e-01   4.472 7.74e-06 ***
## factor(gill.color)5       1.431e+00  9.018e-02  15.866  < 2e-16 ***
## factor(gill.color)6       8.934e-01  9.630e-02   9.277  < 2e-16 ***
## factor(gill.color)7       7.051e-01  9.208e-02   7.657 1.90e-14 ***
## factor(gill.color)8       1.191e+00  1.088e-01  10.945  < 2e-16 ***
## factor(gill.color)9       1.099e+00  1.141e-01   9.628  < 2e-16 ***
## factor(gill.color)10      2.592e-01  8.707e-02   2.977  0.00291 **
## factor(gill.color)11      1.181e+00  9.013e-02  13.102  < 2e-16 ***
## stem.height               1.025e+00  1.927e-02  53.180  < 2e-16 ***
## stem.width               -6.238e-05  2.803e-05  -2.225  0.02605 *
## factor(stem.color)1       1.749e+01  1.722e+02   0.102  0.91910
## factor(stem.color)2       3.029e+01  1.865e+02   0.162  0.87095
## factor(stem.color)3       1.543e+01  1.722e+02   0.090  0.92861
## factor(stem.color)4       3.256e+01  1.962e+02   0.166  0.86820
## factor(stem.color)5       1.599e+01  1.722e+02   0.093  0.92604
## factor(stem.color)6       1.659e+01  1.722e+02   0.096  0.92324
## factor(stem.color)7       1.607e+01  1.722e+02   0.093  0.92565
## factor(stem.color)8       1.790e+01  1.722e+02   0.104  0.91723
## factor(stem.color)9       1.732e+01  1.722e+02   0.101  0.91990
## factor(stem.color)10      1.694e+01  1.722e+02   0.098  0.92166
## factor(stem.color)11      1.565e+01  1.722e+02   0.091  0.92758
## factor(stem.color)12      1.706e+01  1.722e+02   0.099  0.92107
## season                   -6.690e-01  3.685e-02 -18.154  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 74385  on 54034  degrees of freedom
## Residual deviance: 58215  on 53996  degrees of freedom
## AIC: 58293
##
## Number of Fisher Scoring iterations: 15
```

The model summary indicates some coefficients are not defined due to singularities. This often occurs due to high multicollinearity or perfect separation. We need to address these issues and proceed with the other tasks. Given that the Stem color seams to not be a significant variable we will drop it and create a new model that does not contain that variable.

Select the best subset of variables. Perfom a diagnostic on the best model. Perform all possible inferences you can think about.

```r
mushroom_sig= glm(class ~ cap.diameter+ factor(cap.shape) + factor(gill.attachment) + factor(gill.color)
summary(mushroom_sig)
```

```
##
## Call:
## glm(formula = class ~ cap.diameter + factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height + stem.width + season, family = binomial,
##     data = mush_data)
##
## Coefficients: (1 not defined because of singularities)
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               5.886e-01  9.400e-02   6.262 3.80e-10 ***
## cap.diameter             -4.677e-04  5.499e-05  -8.505  < 2e-16 ***
## factor(cap.shape)1       -1.080e+00  6.399e-02 -16.886  < 2e-16 ***
## factor(cap.shape)2       -1.082e+00  4.372e-02 -24.742  < 2e-16 ***
## factor(cap.shape)3        1.099e+00  8.642e-02  12.720  < 2e-16 ***
## factor(cap.shape)4       -9.156e-01  6.639e-02 -13.793  < 2e-16 ***
## factor(cap.shape)5       -7.668e-01  5.116e-02 -14.988  < 2e-16 ***
## factor(cap.shape)6       -1.098e+00  4.121e-02 -26.629  < 2e-16 ***
## factor(gill.attachment)1  4.735e-01  3.340e-02  14.175  < 2e-16 ***
## factor(gill.attachment)2 -6.356e-01  4.140e-02 -15.353  < 2e-16 ***
## factor(gill.attachment)3  5.327e-03  9.659e-02   0.055  0.95601
## factor(gill.attachment)4 -1.368e+00  4.290e-02 -31.894  < 2e-16 ***
## factor(gill.attachment)5  3.968e-01  3.694e-02  10.741  < 2e-16 ***
## factor(gill.attachment)6  4.215e-01  3.285e-02  12.830  < 2e-16 ***
## factor(gill.color)1       2.594e+00  1.107e-01  23.427  < 2e-16 ***
## factor(gill.color)2              NA         NA      NA       NA
## factor(gill.color)3       1.457e-01  8.642e-02   1.686  0.09179 .
## factor(gill.color)4       3.797e-01  9.718e-02   3.907 9.36e-05 ***
## factor(gill.color)5       1.602e+00  8.220e-02  19.494  < 2e-16 ***
## factor(gill.color)6       7.980e-01  8.821e-02   9.046  < 2e-16 ***
## factor(gill.color)7       1.028e+00  8.313e-02  12.360  < 2e-16 ***
## factor(gill.color)8       1.407e+00  9.993e-02  14.084  < 2e-16 ***
## factor(gill.color)9       9.152e-01  1.019e-01   8.977  < 2e-16 ***
## factor(gill.color)10      2.521e-01  7.937e-02   3.177  0.00149 **
## factor(gill.color)11      1.531e+00  8.163e-02  18.759  < 2e-16 ***
## stem.height               9.666e-01  1.822e-02  53.046  < 2e-16 ***
## stem.width               -2.084e-04  2.565e-05  -8.123 4.53e-16 ***
## season                   -5.891e-01  3.471e-02 -16.972  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 74385  on 54034  degrees of freedom
## Residual deviance: 62439  on 54008  degrees of freedom
## AIC: 62493
##
## Number of Fisher Scoring iterations: 4
```

Here we see that although the step color is not significant in the full model, this new model has a higher AIC and therefore is not as good as the full model. However we will perform an Anova analysis on both and compare them.

```
anova(mushroom_full, mushroom_sig, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: class ~ cap.diameter + factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height + stem.width + factor(stem.color) +
##     season
## Model 2: class ~ cap.diameter + factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height + stem.width + season
##   Resid. Df Resid. Dev  Df Deviance  Pr(>Chi)
## 1     53996      58215
## 2     54008      62439 -12  -4224.3 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From the anova test we see that Model 1(mushroom_full), which includes factor(stem.color), fits the data significantly better than Model 2, which excludes this variable. We also see that the large reduction in deviance (-4,224.3) suggests that factor(stem.color) is an important predictor in the logistic regression model for predicting class. Finally the very small p-value (< 2.2e-16) indicates that the improvement in model fit by including factor(stem.color) is not due to random chance and is highly significant. Thus, including the variable factor(stem.color) in the logistic regression model provides a significantly better fit to your data compared to a model without this variable. Therefore, factor(stem.color) is an important predictor for the outcome variable class in your data set.

From this First model we are able to see the variables that have a significant p-value however we will perform a step function in order to determine the best possible combination of variables to get the best model.

```
model_step <- stepAIC(mushroom_full, direction = "both")
```

```
## Start:  AIC=58292.61
## class ~ cap.diameter + factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height + stem.width + factor(stem.color) +
##     season
##
##                           Df Deviance   AIC
## <none>                         58215 58293
## - stem.width               1    58220 58296
## - cap.diameter             1    58261 58337
## - season                   1    58552 58628
## - factor(cap.shape)        6    59754 59820
## - factor(gill.color)      10    60077 60135
## - factor(gill.attachment)  5    61288 61356
## - stem.height              1    61438 61514
## - factor(stem.color)      12    62439 62493
```

```r
summary(model_step)
```

```
##
## Call:
## glm(formula = class ~ cap.diameter + factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height + stem.width + factor(stem.color) +
##     season, family = binomial, data = mush_data)
##
## Coefficients: (1 not defined because of singularities)
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -1.551e+01  1.722e+02  -0.090  0.92822
## cap.diameter             -3.989e-04  5.832e-05  -6.840 7.92e-12 ***
## factor(cap.shape)1       -8.310e-01  6.743e-02 -12.324  < 2e-16 ***
## factor(cap.shape)2       -1.183e+00  4.630e-02 -25.562  < 2e-16 ***
## factor(cap.shape)3        9.119e-01  9.879e-02   9.231  < 2e-16 ***
## factor(cap.shape)4       -9.363e-01  6.928e-02 -13.514  < 2e-16 ***
## factor(cap.shape)5       -9.766e-01  5.417e-02 -18.030  < 2e-16 ***
## factor(cap.shape)6       -1.254e+00  4.369e-02 -28.703  < 2e-16 ***
## factor(gill.attachment)1  4.319e-01  3.495e-02  12.359  < 2e-16 ***
## factor(gill.attachment)2 -5.752e-01  4.330e-02 -13.284  < 2e-16 ***
## factor(gill.attachment)3 -6.538e-01  1.065e-01  -6.136 8.44e-10 ***
## factor(gill.attachment)4 -1.900e+00  4.617e-02 -41.156  < 2e-16 ***
## factor(gill.attachment)5  1.987e-01  3.973e-02   5.002 5.67e-07 ***
## factor(gill.attachment)6  4.833e-01  3.484e-02  13.872  < 2e-16 ***
## factor(gill.color)1       1.850e+00  1.191e-01  15.539  < 2e-16 ***
## factor(gill.color)2              NA         NA      NA       NA
## factor(gill.color)3       1.774e-01  9.443e-02   1.879  0.06027 .
## factor(gill.color)4       4.704e-01  1.052e-01   4.472 7.74e-06 ***
## factor(gill.color)5       1.431e+00  9.018e-02  15.866  < 2e-16 ***
## factor(gill.color)6       8.934e-01  9.630e-02   9.277  < 2e-16 ***
## factor(gill.color)7       7.051e-01  9.208e-02   7.657 1.90e-14 ***
## factor(gill.color)8       1.191e+00  1.088e-01  10.945  < 2e-16 ***
## factor(gill.color)9       1.099e+00  1.141e-01   9.628  < 2e-16 ***
## factor(gill.color)10      2.592e-01  8.707e-02   2.977  0.00291 **
## factor(gill.color)11      1.181e+00  9.013e-02  13.102  < 2e-16 ***
## stem.height               1.025e+00  1.927e-02  53.180  < 2e-16 ***
## stem.width               -6.238e-05  2.803e-05  -2.225  0.02605 *
## factor(stem.color)1       1.749e+01  1.722e+02   0.102  0.91910
## factor(stem.color)2       3.029e+01  1.865e+02   0.162  0.87095
## factor(stem.color)3       1.543e+01  1.722e+02   0.090  0.92861
## factor(stem.color)4       3.256e+01  1.962e+02   0.166  0.86820
## factor(stem.color)5       1.599e+01  1.722e+02   0.093  0.92604
## factor(stem.color)6       1.659e+01  1.722e+02   0.096  0.92324
## factor(stem.color)7       1.607e+01  1.722e+02   0.093  0.92565
## factor(stem.color)8       1.790e+01  1.722e+02   0.104  0.91723
## factor(stem.color)9       1.732e+01  1.722e+02   0.101  0.91990
## factor(stem.color)10      1.694e+01  1.722e+02   0.098  0.92166
## factor(stem.color)11      1.565e+01  1.722e+02   0.091  0.92758
## factor(stem.color)12      1.706e+01  1.722e+02   0.099  0.92107
## season                   -6.690e-01  3.685e-02 -18.154  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 74385  on 54034  degrees of freedom
## Residual deviance: 58215  on 53996  degrees of freedom
## AIC: 58293
##
## Number of Fisher Scoring iterations: 15
```

After performing the step wise function forward and backwards we see that the best model with the lowest AIC is the original one which contains all predictor variables.

We are going to perform one more test which is the drop1 test in order to see if we should in fact keep the full model.

```
drop1(mushroom_full, test="LRT")
```

```
## Single term deletions
##
## Model:
## class ~ cap.diameter + factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height + stem.width + factor(stem.color) +
##     season
##                         Df Deviance   AIC    LRT  Pr(>Chi)
## <none>                       58215 58293
## cap.diameter             1    58261 58337   46.8 7.856e-12 ***
## factor(cap.shape)        6    59754 59820 1539.8 < 2.2e-16 ***
## factor(gill.attachment)  5    61288 61356 3072.9 < 2.2e-16 ***
## factor(gill.color)      10    60077 60135 1862.6 < 2.2e-16 ***
## stem.height              1    61438 61514 3223.4 < 2.2e-16 ***
## stem.width               1    58220 58296    5.0   0.02591 *
## factor(stem.color)      12    62439 62493 4224.3 < 2.2e-16 ***
## season                   1    58552 58628  337.8 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

These results suggest that all the predictors in your model are important, with cap.shape, gill.attachment, gill.color, stem.height, stem.color, and season being especially critical. The predictor stem.width is also important, albeit slightly less so compared to the others. So we again conclude that the model works best when we include all of the variable predictors.

Since we must keep all variables to get the best model I now want to try to have some interaction between some of the predictors in order to see if that has an effect on the model.

```
mushroom_test= glm(class ~ cap.diameter*factor(cap.shape) + factor(gill.attachment) + factor(gill.color
summary(mushroom_test)
```

```
##
## Call:
## glm(formula = class ~ cap.diameter * factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height * stem.width + factor(stem.color) +
##     season, family = binomial, data = mush_data)
##
## Coefficients: (1 not defined because of singularities)
##                                   Estimate Std. Error z value Pr(>|z|)    
## (Intercept)                     -1.571e+01  1.731e+02  -0.091 0.927714    
## cap.diameter                     7.669e-04  2.273e-04   3.374 0.000740 ***
## factor(cap.shape)1              -1.723e+00  1.418e-01 -12.148  < 2e-16 ***
## factor(cap.shape)2              -4.100e-01  8.942e-02  -4.586 4.52e-06 ***
## factor(cap.shape)3               2.886e+00  1.883e-01  15.325  < 2e-16 ***
## factor(cap.shape)4              -4.826e-01  1.281e-01  -3.768 0.000165 ***
## factor(cap.shape)5              -2.239e-01  1.036e-01  -2.160 0.030742 *  
## factor(cap.shape)6              -1.233e+00  8.339e-02 -14.786  < 2e-16 ***
## factor(gill.attachment)1         3.891e-01  3.555e-02  10.945  < 2e-16 ***
## factor(gill.attachment)2        -7.841e-01  4.562e-02 -17.185  < 2e-16 ***
## factor(gill.attachment)3        -1.006e+00  1.089e-01  -9.238  < 2e-16 ***
## factor(gill.attachment)4        -2.059e+00  4.785e-02 -43.024  < 2e-16 ***
## factor(gill.attachment)5         8.904e-02  4.078e-02   2.183 0.029016 *  
## factor(gill.attachment)6         3.538e-01  3.558e-02   9.945  < 2e-16 ***
## factor(gill.color)1              1.745e+00  1.201e-01  14.534  < 2e-16 ***
## factor(gill.color)2                     NA         NA      NA       NA    
## factor(gill.color)3              1.635e-01  9.671e-02   1.690 0.090963 .  
## factor(gill.color)4              4.601e-01  1.068e-01   4.308 1.64e-05 ***
## factor(gill.color)5              1.397e+00  9.168e-02  15.238  < 2e-16 ***
## factor(gill.color)6              8.421e-01  9.817e-02   8.578  < 2e-16 ***
## factor(gill.color)7              5.923e-01  9.398e-02   6.302 2.94e-10 ***
## factor(gill.color)8              9.872e-01  1.111e-01   8.885  < 2e-16 ***
## factor(gill.color)9              1.119e+00  1.163e-01   9.622  < 2e-16 ***
## factor(gill.color)10             2.073e-01  8.852e-02   2.342 0.019182 *  
## factor(gill.color)11             1.111e+00  9.165e-02  12.127  < 2e-16 ***
## stem.height                      1.205e+00  3.624e-02  33.249  < 2e-16 ***
## stem.width                       2.337e-05  3.312e-05   0.706 0.480389    
## factor(stem.color)1              1.735e+01  1.731e+02   0.100 0.920158    
## factor(stem.color)2              2.922e+01  1.872e+02   0.156 0.875949    
## factor(stem.color)3              1.523e+01  1.731e+02   0.088 0.929909    
## factor(stem.color)4              3.261e+01  1.970e+02   0.166 0.868505    
## factor(stem.color)5              1.597e+01  1.731e+02   0.092 0.926522    
## factor(stem.color)6              1.643e+01  1.731e+02   0.095 0.924377    
## factor(stem.color)7              1.590e+01  1.731e+02   0.092 0.926818    
## factor(stem.color)8              1.783e+01  1.731e+02   0.103 0.917995    
## factor(stem.color)9              1.716e+01  1.731e+02   0.099 0.921066    
## factor(stem.color)10             1.673e+01  1.731e+02   0.097 0.923012    
## factor(stem.color)11             1.545e+01  1.731e+02   0.089 0.928897    
## factor(stem.color)12             1.692e+01  1.731e+02   0.098 0.922139    
## season                          -6.619e-01  3.708e-02 -17.849  < 2e-16 ***
## cap.diameter:factor(cap.shape)1  2.652e-03  3.781e-04   7.012 2.34e-12 ***
## cap.diameter:factor(cap.shape)2 -1.792e-03  2.289e-04  -7.827 5.00e-15 ***
## cap.diameter:factor(cap.shape)3 -3.541e-03  3.151e-04 -11.237  < 2e-16 ***
## cap.diameter:factor(cap.shape)4 -1.053e-03  2.547e-04  -4.136 3.54e-05 ***
## cap.diameter:factor(cap.shape)5 -1.723e-03  2.390e-04  -7.210 5.58e-13 ***
## cap.diameter:factor(cap.shape)6 -4.958e-04  2.224e-04  -2.230 0.025768 *  
## stem.height:stem.width          -1.528e-04  2.192e-05  -6.971 3.16e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## (Dispersion parameter for binomial family taken to be 1)
## 
##     Null deviance: 74385  on 54034  degrees of freedom
## Residual deviance: 57518  on 53989  degrees of freedom
## AIC: 57610
##
## Number of Fisher Scoring iterations: 15
```

I first created an interaction between cap.diameter and factor(cap.shape) and I got a better AIC. Then I did the same with stem.height and stem.width and that is where I got the lowest AIC so that is the model I decided to go with.

We will now make a comparison with the null model since our data is un-grouped in order to check the fit of our model H_0: All parameters in model_test not in the null model are zero. H_1: At least one of them is not zero

```r
# Fit the null model
null_model <- glm(class ~ 1, data = mush_data, family = binomial)

# Fit the original model
mushroom_test <- glm(class ~ cap.diameter * factor(cap.shape) +
                     factor(gill.attachment) +
                     factor(gill.color) +
                     stem.height * stem.width +
                     factor(stem.color) +
                     season,
                     data = mush_data, family = binomial)

anova(mushroom_test,null_model,test = "LRT")
```

```
## Analysis of Deviance Table
##
## Model 1: class ~ cap.diameter * factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height * stem.width + factor(stem.color) +
##     season
## Model 2: class ~ 1
##   Resid. Df Resid. Dev  Df Deviance  Pr(>Chi)
## 1     53989      57518
## 2     54034      74385 -45   -16867 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
# Compare AIC values
null_model_aic <- AIC(null_model)
original_model_aic <- AIC(mushroom_test)

# Print AIC values
cat("Null Model AIC:", null_model_aic, "\n")
```

```
## Null Model AIC: 74386.77
```

```r
cat("Original Model AIC:", original_model_aic, "\n")
```

```
## Original Model AIC: 57609.98
```

Since the p-value is much less than the typical significance level (e.g., 0.05 or 0.01), you reject the null hypothesis. This indicates strong evidence that Model 1, which includes the predictor variables, significantly improves the model fit compared to the null model. So our model fit is good. We can also conclude the same when comparing the AIC of the two models, the model we chose has a lower AIC therefore it is better.

Predictions

Probability Predictions: The GLM with a binomial family (logistic regression) will predict the probability that a given mushroom is poisonous (class = 1) or edible (class = 0). The model output will be a probability value between 0 and 1. Class Predictions: Based on the probability predictions, we can classify each mushroom as either poisonous or edible. Typically, a threshold of 0.5 is used, where a probability greater than 0.5 indicates a poisonous mushroom (class = 1), and a probability less than or equal to 0.5 indicates an edible mushroom (class = 0). However, this threshold can be adjusted based on the context and requirements of our problem and this is what we will do in the next step.

But first we will create some new data point in order to make some predictions with our model. The results are as follows:

```r
# New data frame containing new mushroom data
new_mush_data <- data.frame(
  cap.diameter = c(953, 450),
  cap.shape = c(4,2),
  gill.attachment =c(3,1),
  gill.color = c(6, 7),
  stem.height = c(1.5, 3.0),
  stem.width = c(2676, 1747),
  stem.color = c(8,1),
  season = c(0.962, 0.877 )
)

# Predict probabilities
probability_predictions <- predict(mushroom_test, newdata = new_mush_data, type = "response")

# Set a threshold for classification
threshold <- 0.5

# Predict class labels
class_predictions <- ifelse(probability_predictions > threshold, 1, 0)

# Print predictions
print(probability_predictions)
```

```
##         1         2
## 0.8604141 0.9825540
```

```r
print(class_predictions)
```

```
## 1 2
## 1 1
```

The predicted probabilities for the two new mushroom instances are: For the first mushroom: 0.8604141 (approximately 86.04% probability of being poisonous). For the second mushroom: 0.9825540 (approximately 98.26% probability of being poisonous).

Class Predictions: Based on the threshold of 0.5: For the first mushroom: The probability (0.8604141) is greater than 0.5, so it is classified as 1 (poisonous). For the second mushroom: The probability (0.9825540) is greater than 0.5, so it is also classified as 1 (poisonous).

As we can see our model is providing valid predictions indicating that the mushrooms in our new data set are classified as poisonous. The predicted probabilities and class labels are consistent with the behavior of a logistic regression model used for binary classification.

Use different pi__0 as a cut-off point and create a confusion table.

Now we will create a confusion matrix for different π_0 or cut off points:

```r
# Function to create confusion matrix for different thresholds
create_confusion_matrix <- function(actual, predicted_prob, threshold) {
  predicted_class <- ifelse(predicted_prob > threshold, 1, 0)
  return(table(Actual = actual, Predicted = predicted_class))
}

# Different thresholds
thresholds <- seq(0.1, 0.9, by = 0.1)
predicted_probabilities <- predict(model_step, type = "response")

# Create confusion matrices for each threshold
confusion_matrices <- lapply(thresholds, function(thresh) {
  cm <- create_confusion_matrix(mush_data$class, predicted_probabilities, thresh)
  list(threshold = thresh, confusion_matrix = cm)
})

# Print confusion matrices for each threshold
for (cm in confusion_matrices) {
  cat("Threshold:", cm$threshold, "\n")
  print(cm$confusion_matrix)
  cat("\n")
}
```

```
## Threshold: 0.1
##       Predicted
## Actual     0     1
##      0   504 23856
##      1    56 29619
##
## Threshold: 0.2
##       Predicted
## Actual     0     1
##      0  3379 20981
##      1   766 28909
##
## Threshold: 0.3
##       Predicted
## Actual     0     1
##      0  8975 15385
##      1  2268 27407
##
## Threshold: 0.4
##       Predicted
## Actual     0     1
##      0 13905 10455
##      1  4431 25244
##
## Threshold: 0.5
##       Predicted
## Actual     0     1
##      0 17445  6915
##      1  7820 21855
##
## Threshold: 0.6
##       Predicted
## Actual     0     1
##      0 20136  4224
##      1 11266 18409
##
## Threshold: 0.7
##       Predicted
## Actual     0     1
##      0 21685  2675
##      1 14594 15081
##
## Threshold: 0.8
##       Predicted
## Actual     0     1
##      0 22922  1438
##      1 18735 10940
##
## Threshold: 0.9
##       Predicted
## Actual     0     1
##      0 23992   368
##      1 24274  5401
```

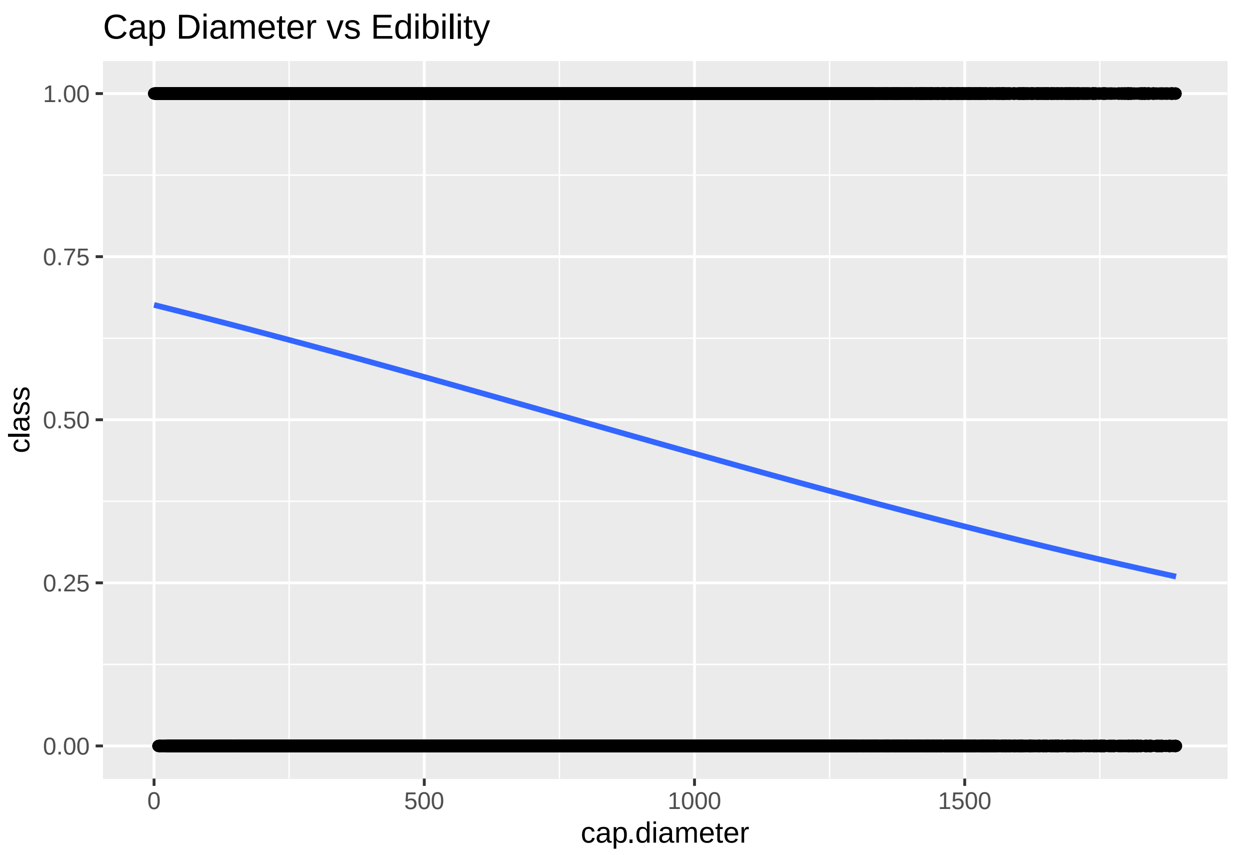
From the confusion matrices, we can observe the trade-off between true positives (correctly identified poisonous mushrooms) and false positives (incorrectly identified edible mushrooms as poisonous) at different thresholds.

Threshold 0.1: High sensitivity (high true positive rate) but very low specificity (high false positive rate). Threshold 0.9: High specificity (high true negative rate) but very low sensitivity (high false negative rate). ### Perform Visualization of Data and Models

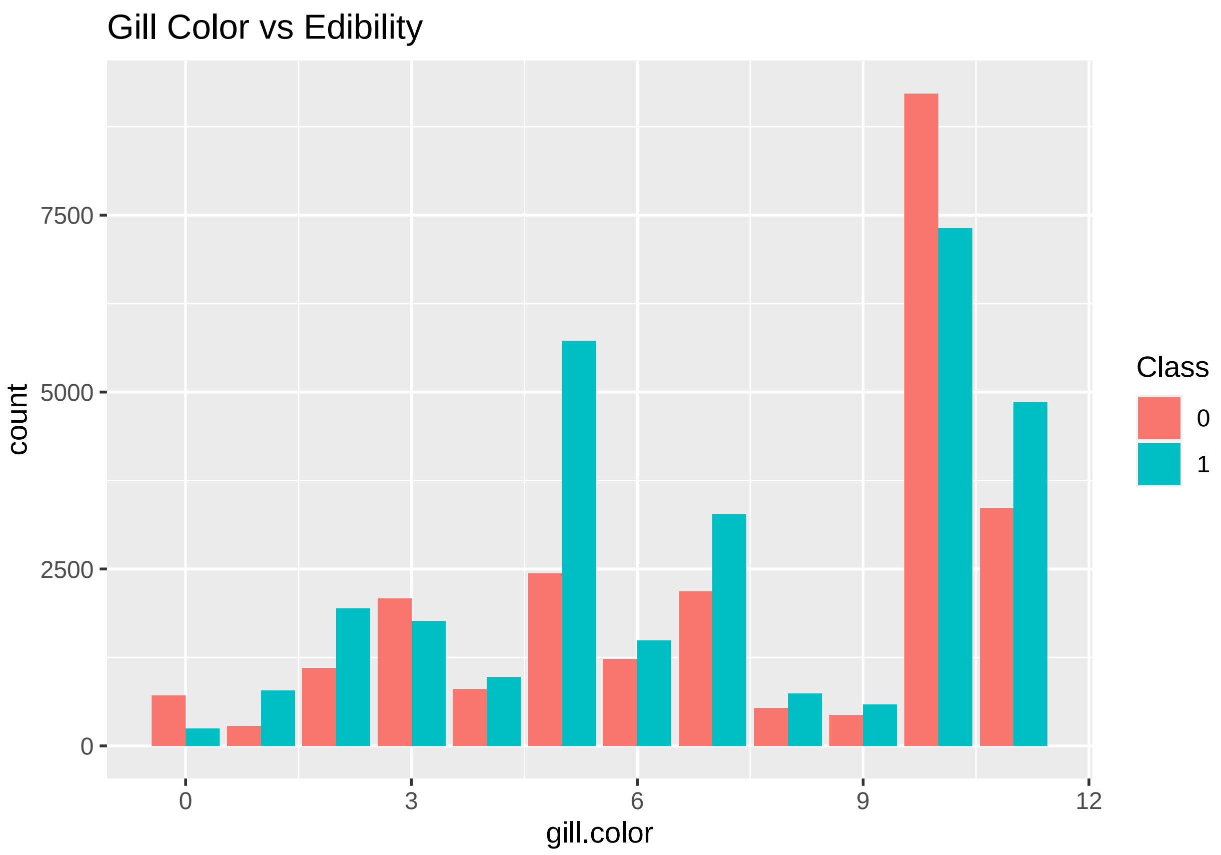
```r
#library(ggplot2)

# Visualize the relationship between some predictors and the class
ggplot(mush_data, aes(x = cap.diameter, y = class)) +
  geom_point() +
  geom_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
  labs(title = "Cap Diameter vs Edibility")
```

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

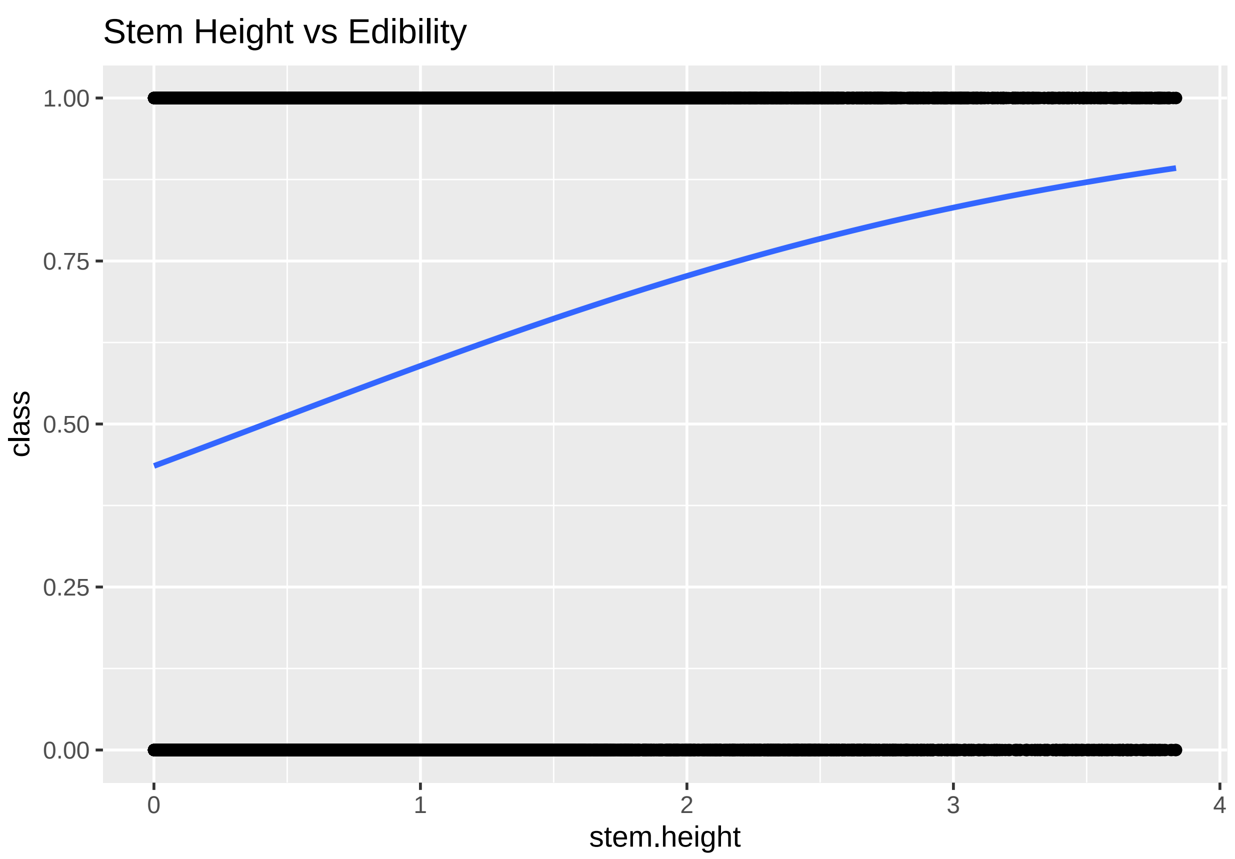


```r
ggplot(mush_data, aes(x = gill.color, fill = as.factor(class))) +
  geom_bar(position = "dodge") +
  labs(title = "Gill Color vs Edibility", fill = "Class")
```



```
ggplot(mush_data, aes(x = stem.height, y = class)) +
  geom_point() +
  geom_smooth(method = "glm", method.args = list(family = "binomial"), se = FALSE) +
  labs(title = "Stem Height vs Edibility")
```

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



These plots allow us to visualize our data a bit more and see how changing some variables changes the edibility of the mushroom.

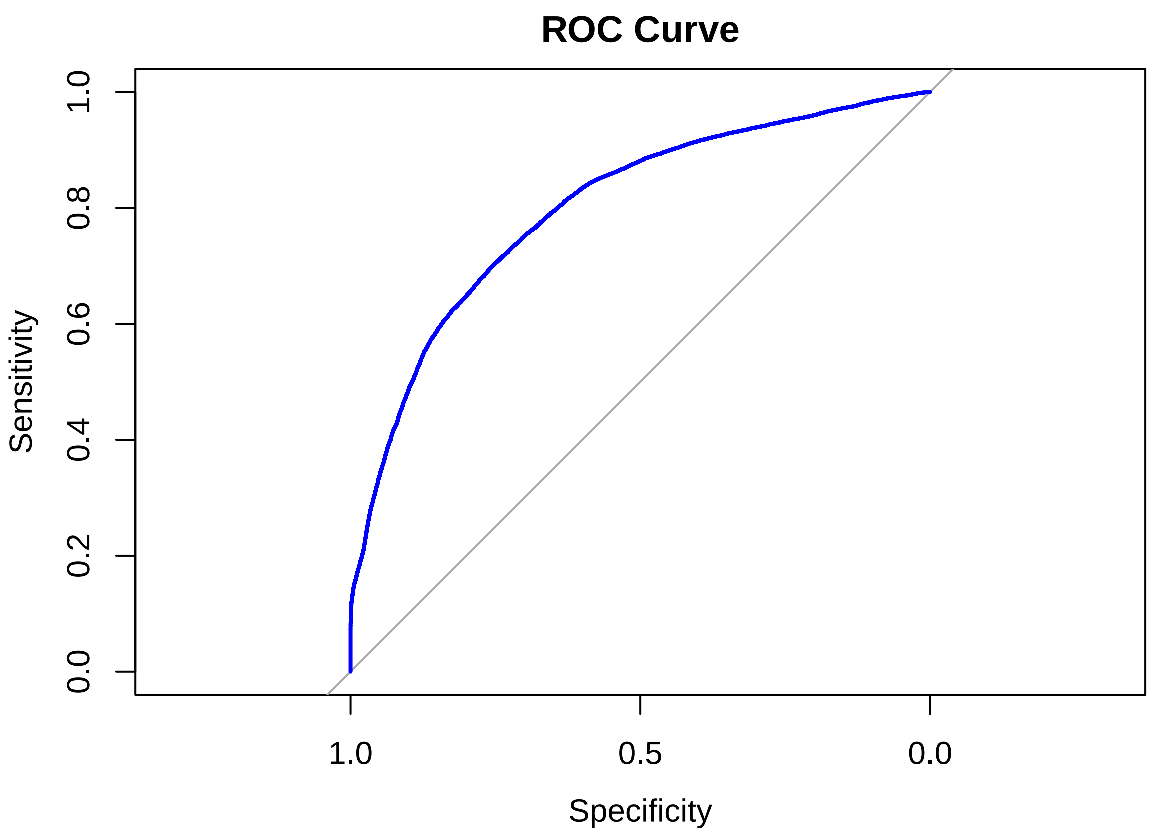
Plot the ROC Curve, Find AUC, and the Best Cutoff Point for Classification

```
# Assuming 'predicted_probabilities' contains the predicted probabilities from the logistic regression
roc_curve <- roc(mush_data$class, predicted_probabilities)
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```
# Plot the ROC curve
plot(roc_curve, main = "ROC Curve", col = "blue")
```



```r
# Calculate AUC
auc_value <- auc(roc_curve)
cat("AUC:", auc_value, "\n")
```

```
## AUC: 0.8000803
```

```r
# Find the best cutoff point
best_cutoff <- coords(roc_curve, "best", ret = "threshold")
cat("Best Cutoff Point:", best_cutoff$threshold, "\n")
```

```
## Best Cutoff Point: 0.5257369
```

```r
# Best cutoff point
best_cutoff <- coords(roc_curve, "best", ret = "threshold")
```

As we see we get an AUC: 0.8000803, indicating a good model performance. and we get the Best Cutoff Point: 0.5257369, suggesting that using this threshold optimizes the balance between sensitivity and specificity.

```r
# LOOCV
#mush_data$class <- as.factor(mush_data$class)
#loocv_control <- trainControl(method = "LOOCV")
#loocv_model <- train(factor(class) ~ cap.diameter*factor(cap.shape) + factor(gill.attachment) + factor
#                     data = mush_data, method = "glm", family = "binomial", trControl = loocv_control)
#print(loocv_model)

# 10-fold cross-validation
kfold_control <- trainControl(method = "cv", number = 10)
kfold_model <- train(factor(class) ~ cap.diameter+factor(cap.shape) + factor(gill.attachment) + factor(
                     data = mush_data, method = "glm", family = "binomial", trControl = kfold_control)
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
print(kfold_model)
```

```
## Generalized Linear Model
##
## 54035 samples
##     8 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 48631, 48632, 48631, 48632, 48632, 48632, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.72664   0.4497846
```

When we tried running the LOOCV model we see that due to the data being so large the function is computationally heavy and therefore we must omit this step. In the case of the k-fold model we see that we get a successful model with pretty good accuracy, however we get the following warning messages. Warning about singularities and warning about non-estimable cases. The first one indicates that one or more predictor variables might be collinear, resulting in a singular fit while the second one suggests that there might be rank-deficient fits, meaning the model couldn't estimate coefficients for some predictor levels. This can happen due to rare or highly imbalanced categories in categorical variables.

So we check for multicolunearity with in the model in order to find highly correlated values.

```r
# Check for multicollinearity
cor_matrix <- cor(mush_data[, -ncol(mush_data)]) # Exclude target variable
high_cor <- findCorrelation(cor_matrix, cutoff = 0.7) # Find highly correlated variables
high_cor_vars <- colnames(mush_data[, -ncol(mush_data)])[high_cor]

# Print highly correlated variables
print(high_cor_vars)
```

```
## [1] "cap.diameter"
```

```r
# Remove highly correlated variables
mush_data_nodiam <- mush_data[, !(colnames(mush_data) %in% high_cor_vars)]

#re-run the kfold model
newkfold_model <- train(factor(class) ~ factor(cap.shape) + factor(gill.attachment) +
                     factor(gill.color) + stem.height + stem.width + factor(stem.color) +
                     season,
                     data = mush_data_nodiam , method = "glm", family = "binomial", trControl = kfold_c
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
# Print the results
print(newkfold_model)
```

```
## Generalized Linear Model
##
## 54035 samples
##     7 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 48631, 48632, 48631, 48632, 48631, 48632, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.7284353  0.4533171
```

```
# Check for highly correlated variables
cor_matrix <- cor(mush_data_nodiam[, -which(names(mush_data_nodiam) == "class")])
highly_correlated <- findCorrelation(cor_matrix, cutoff = 0.75)

# Get the names of highly correlated variables
highly_correlated_vars <- colnames(mush_data_nodiam)[highly_correlated]

# Remove highly correlated variables
mush_data_nocor<- mush_data_nodiam[, !colnames(mush_data_nodiam) %in% highly_correlated_vars]

# Fit the model again
kfold_model_nocor <- train(factor(class) ~ ., data = mush_data_nocor, method = "glm", family = "binomial

# Print the results
print(kfold_model_nocor)
```

```
## Generalized Linear Model
##
## 54035 samples
##     7 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 48631, 48632, 48631, 48632, 48632, 48631, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.6344963  0.2539378
```

After performing that test twice at different cut off points we see that cap.diameter is the variable that is highly correlated. We exclude that for the k-fold model and we end up with an accuracy of 0.6348 which is better than 0.5, meaning out model is better than just guessing.

Since we determined that the variable cap.diameter has high colinearity we will run the original model again with out that variable and see which model is best.

```
mushroom_nocap= glm(class ~ factor(cap.shape) + factor(gill.attachment) + factor(gill.color) + stem.hei
summary(mushroom_nocap)
```

```
##
## Call:
## glm(formula = class ~ factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height * stem.width + factor(stem.color) +
##     season, family = binomial, data = mush_data)
```

```
##
## Coefficients: (1 not defined because of singularities)
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -1.568e+01  1.715e+02  -0.091  0.92714
## factor(cap.shape)1       -8.038e-01  6.734e-02 -11.936  < 2e-16 ***
## factor(cap.shape)2       -1.212e+00  4.625e-02 -26.198  < 2e-16 ***
## factor(cap.shape)3        9.843e-01  9.855e-02   9.987  < 2e-16 ***
## factor(cap.shape)4       -8.981e-01  6.936e-02 -12.948  < 2e-16 ***
## factor(cap.shape)5       -1.029e+00  5.403e-02 -19.049  < 2e-16 ***
## factor(cap.shape)6       -1.269e+00  4.372e-02 -29.016  < 2e-16 ***
## factor(gill.attachment)1  3.901e-01  3.475e-02  11.225  < 2e-16 ***
## factor(gill.attachment)2 -6.819e-01  4.184e-02 -16.299  < 2e-16 ***
## factor(gill.attachment)3 -5.917e-01  1.057e-01  -5.597 2.18e-08 ***
## factor(gill.attachment)4 -1.849e+00  4.632e-02 -39.910  < 2e-16 ***
## factor(gill.attachment)5  1.689e-01  3.985e-02   4.240 2.24e-05 ***
## factor(gill.attachment)6  4.658e-01  3.484e-02  13.371  < 2e-16 ***
## factor(gill.color)1       1.852e+00  1.186e-01  15.619  < 2e-16 ***
## factor(gill.color)2              NA         NA      NA       NA
## factor(gill.color)3       2.132e-01  9.433e-02   2.260  0.02384 *
## factor(gill.color)4       4.920e-01  1.050e-01   4.687 2.78e-06 ***
## factor(gill.color)5       1.428e+00  8.994e-02  15.873  < 2e-16 ***
## factor(gill.color)6       9.180e-01  9.606e-02   9.556  < 2e-16 ***
## factor(gill.color)7       7.477e-01  9.187e-02   8.138 4.01e-16 ***
## factor(gill.color)8       1.199e+00  1.084e-01  11.057  < 2e-16 ***
## factor(gill.color)9       1.158e+00  1.143e-01  10.130  < 2e-16 ***
## factor(gill.color)10      2.823e-01  8.680e-02   3.252  0.00115 **
## factor(gill.color)11      1.155e+00  8.986e-02  12.859  < 2e-16 ***
## stem.height               1.182e+00  3.509e-02  33.703  < 2e-16 ***
## stem.width               -1.198e-04  2.368e-05  -5.061 4.18e-07 ***
## factor(stem.color)1       1.752e+01  1.715e+02   0.102  0.91863
## factor(stem.color)2       2.991e+01  1.858e+02   0.161  0.87210
## factor(stem.color)3       1.544e+01  1.715e+02   0.090  0.92825
## factor(stem.color)4       3.260e+01  1.955e+02   0.167  0.86758
## factor(stem.color)5       1.606e+01  1.715e+02   0.094  0.92537
## factor(stem.color)6       1.661e+01  1.715e+02   0.097  0.92283
## factor(stem.color)7       1.614e+01  1.715e+02   0.094  0.92499
## factor(stem.color)8       1.787e+01  1.715e+02   0.104  0.91700
## factor(stem.color)9       1.734e+01  1.715e+02   0.101  0.91944
## factor(stem.color)10      1.698e+01  1.715e+02   0.099  0.92113
## factor(stem.color)11      1.568e+01  1.715e+02   0.091  0.92716
## factor(stem.color)12      1.711e+01  1.715e+02   0.100  0.92050
## season                   -6.893e-01  3.678e-02 -18.740  < 2e-16 ***
## stem.height:stem.width   -1.183e-04  2.056e-05  -5.754 8.72e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 74385  on 54034  degrees of freedom
## Residual deviance: 58228  on 53996  degrees of freedom
## AIC: 58306
##
## Number of Fisher Scoring iterations: 15
```

Here we see that this model has an AIC of 58337 while the original full model has an AIC of 57610 so while cap.diameter was removed due to collinearity in the k-10 fold step, it seems to contribute to a better model fit according to the AIC.

Trying the Probit Link and the Identity Links to Model Data.

```r
# Probit model
model_probit <- glm(class ~ cap.diameter * factor(cap.shape) + factor(gill.attachment) +
                    factor(gill.color) + stem.height * stem.width + factor(stem.color) +
                    season, family = binomial(link = "probit"), data = mush_data)
summary(model_probit)
```

```
##
## Call:
## glm(formula = class ~ cap.diameter * factor(cap.shape) + factor(gill.attachment) +
##     factor(gill.color) + stem.height * stem.width + factor(stem.color) +
##     season, family = binomial(link = "probit"), data = mush_data)
##
## Coefficients: (1 not defined because of singularities)
##                                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)                     -5.180e+00  4.293e+01  -0.121 0.903964
## cap.diameter                     4.807e-04  1.290e-04   3.727 0.000194 ***
## factor(cap.shape)1              -9.868e-01  8.148e-02 -12.111  < 2e-16 ***
## factor(cap.shape)2              -2.128e-01  5.127e-02  -4.151 3.32e-05 ***
## factor(cap.shape)3               1.658e+00  1.084e-01  15.301  < 2e-16 ***
## factor(cap.shape)4              -2.176e-01  7.524e-02  -2.892 0.003826 **
## factor(cap.shape)5              -1.158e-01  6.010e-02  -1.927 0.053966 .
## factor(cap.shape)6              -7.057e-01  4.755e-02 -14.841  < 2e-16 ***
## factor(gill.attachment)1         2.491e-01  2.124e-02  11.727  < 2e-16 ***
## factor(gill.attachment)2        -4.662e-01  2.695e-02 -17.300  < 2e-16 ***
## factor(gill.attachment)3        -5.329e-01  6.368e-02  -8.368  < 2e-16 ***
## factor(gill.attachment)4        -1.242e+00  2.823e-02 -43.994  < 2e-16 ***
## factor(gill.attachment)5         4.480e-02  2.435e-02   1.840 0.065749 .
## factor(gill.attachment)6         1.843e-01  2.110e-02   8.733  < 2e-16 ***
## factor(gill.color)1              1.059e+00  6.981e-02  15.164  < 2e-16 ***
## factor(gill.color)2                     NA         NA      NA       NA
## factor(gill.color)3              1.322e-01  5.650e-02   2.340 0.019290 *
## factor(gill.color)4              3.221e-01  6.225e-02   5.175 2.28e-07 ***
## factor(gill.color)5              8.766e-01  5.336e-02  16.428  < 2e-16 ***
## factor(gill.color)6              5.379e-01  5.757e-02   9.343  < 2e-16 ***
## factor(gill.color)7              4.052e-01  5.496e-02   7.372 1.67e-13 ***
## factor(gill.color)8              6.445e-01  6.524e-02   9.879  < 2e-16 ***
## factor(gill.color)9              7.233e-01  6.855e-02  10.551  < 2e-16 ***
## factor(gill.color)10             1.533e-01  5.146e-02   2.979 0.002892 **
## factor(gill.color)11             7.281e-01  5.339e-02  13.636  < 2e-16 ***
## stem.height                      7.196e-01  2.117e-02  34.000  < 2e-16 ***
## stem.width                       2.407e-05  1.970e-05   1.222 0.221832
## factor(stem.color)1              6.084e+00  4.293e+01   0.142 0.887309
## factor(stem.color)2              8.946e+00  4.642e+01   0.193 0.847177
## factor(stem.color)3              4.825e+00  4.293e+01   0.112 0.910526
## factor(stem.color)4              1.106e+01  4.872e+01   0.227 0.820474
## factor(stem.color)5              5.270e+00  4.293e+01   0.123 0.902299
## factor(stem.color)6              5.538e+00  4.293e+01   0.129 0.897367
## factor(stem.color)7              5.206e+00  4.293e+01   0.121 0.903491
## factor(stem.color)8              6.310e+00  4.293e+01   0.147 0.883143
## factor(stem.color)9              5.967e+00  4.293e+01   0.139 0.889462
## factor(stem.color)10             5.742e+00  4.293e+01   0.134 0.893604
## factor(stem.color)11             4.955e+00  4.293e+01   0.115 0.908119
## factor(stem.color)12             5.809e+00  4.293e+01   0.135 0.892363
## season                          -3.883e-01  2.195e-02 -17.692  < 2e-16 ***
## cap.diameter:factor(cap.shape)1  1.378e-03  2.167e-04   6.358 2.04e-10 ***
## cap.diameter:factor(cap.shape)2 -1.099e-03  1.299e-04  -8.455  < 2e-16 ***
## cap.diameter:factor(cap.shape)3 -2.142e-03  1.816e-04 -11.792  < 2e-16 ***
## cap.diameter:factor(cap.shape)4 -6.888e-04  1.465e-04  -4.701 2.59e-06 ***
## cap.diameter:factor(cap.shape)5 -1.065e-03  1.362e-04  -7.818 5.36e-15 ***
## cap.diameter:factor(cap.shape)6 -3.354e-04  1.260e-04  -2.663 0.007743 **
## stem.height:stem.width          -9.335e-05  1.306e-05  -7.149 8.74e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 74385  on 54034  degrees of freedom
## Residual deviance: 57582  on 53989  degrees of freedom
## AIC: 57674
##
## Number of Fisher Scoring iterations: 15
```

```
# Identity link model
#model_identity_lm <- lm(class ~ cap.diameter * factor(cap.shape) + factor(gill.attachment) +
                    #  factor(gill.color) + stem.height * stem.width + factor(stem.color) +
                    #  season, data = mush_data, family = binomial(link = "identity"))
#summary(model_identity_lm)

#strt= (coef(model_identity_lm))

#model_identity = glm(class ~ cap.diameter * factor(cap.shape) + factor(gill.attachment) +
      #                factor(gill.color) + stem.height * stem.width + factor(stem.color) +
       #               season, data = mush_data, start = strt, family = binomial(link = "identity"))
```

When trying to run the identity model we need to provide a starting point for the model. However when we try to do this we have some problems due to the fact that there are so many variables so for our case we will stick with the probit model.

```
# AIC values
cat("AIC of Logistic Regression Model:", AIC(mushroom_test), "\n")
```

```
## AIC of Logistic Regression Model: 57609.98
```

```
cat("AIC of Probit Model:", AIC(model_probit), "\n")
```

```
## AIC of Probit Model: 57673.7
```

```
# AUC for probit model
probit_pred <- predict(model_probit, type = "response")
roc_probit <- roc(mush_data$class, probit_pred)
```

```
## Setting levels: control = 0, case = 1
```

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

```r
auc_probit <- auc(roc_probit)
cat("AUC of Probit Model:", auc_probit, "\n")
```

```
## AUC of Probit Model: 0.8049035
```

Preferred Model: Given the lower AIC, the logistic regression model is preferred for this dataset. It provides a better balance of fit and complexity compared to the probit model. Model Performance: Both models demonstrate good classification performance with a high AUC, but the logistic regression model is slightly better in terms of fit. Therefore we can conclude that the logistic regression model if the better model.