Classification

Code

Dataset: Secondary Mushroom Dataset (https://archive.ics.uci.edu/ml/datasets/Secondary+Mushroom+Dataset)

This dataset contains points simulated for a binary classification problem between edible and poisonous mushrooms.

Libraries

Hide

```
library(Metrics)
library(ggplot2)
```

```
Need help? Try Stackoverflow: https://stackoverflow.com/tags/ggplot2
```

Hide

```
library(ROCR)
library(caret)
```

```
Loading required package: lattice
Registered S3 method overwritten by 'data.table':
  method           from
  print.data.table

Attaching package: ‘caret’

The following objects are masked from ‘package:Metrics’:

    precision, recall
```

Hide

```
library(e1071)
```

Hide

```
df <- data.frame(read.csv("secondary_data.csv", sep=";"))
head(df)
```

cla… <chr>	cap.diameter <dbl>	cap.shape <chr>	cap.surface <chr>	cap.color <chr>	does.bruise.or.bleed <chr>	gill.attachmen <chr>
1 p	15.26	x	g	o	f	e
2 p	16.60	x	g	o	f	e
3 p	14.07	x	g	o	f	e

cla… <chr>	cap.diameter <dbl>	cap.shape <chr>	cap.surface <chr>	cap.color <chr>	does.bruise.or.bleed <chr>	gill.attachmen <chr>
4 p	14.17	f	h	e	f	e
5 p	14.64	x	h	o	f	e
6 p	15.34	x	g	o	f	e

6 rows | 1-8 of 21 columns

Hide

```
print(paste("# of rows =",nrow(df)))
```

```
[1] "# of rows = 61069"
```

Hide

```
summary(df)
```

```
    class            cap.diameter     cap.shape         cap.surface         cap.color         do
es.bruise.or.bleed gill.attachment
 Length:61069       Min.   : 0.380   Length:61069       Length:61069       Length:61069       Le
ngth:61069         Length:61069
 Class :character   1st Qu.: 3.480   Class :character   Class :character   Class :character   Cl
ass :character     Class :character
 Mode  :character   Median : 5.860   Mode  :character   Mode  :character   Mode  :character   Mo
de  :character     Mode  :character
                    Mean   : 6.734
                    3rd Qu.: 8.540
                    Max.   :62.340
 gill.spacing        gill.color         stem.height       stem.width      stem.root         ste
m.surface        stem.color
 Length:61069       Length:61069       Min.   : 0.000   Min.   :  0.00   Length:61069       Leng
th:61069       Length:61069
 Class :character   Class :character   1st Qu.: 4.640   1st Qu.:  5.21   Class :character   Clas
s :character   Class :character
 Mode  :character   Mode  :character   Median : 5.950   Median : 10.19   Mode  :character   Mode
:character   Mode  :character
                                      Mean   : 6.582   Mean   : 12.15
                                      3rd Qu.: 7.740   3rd Qu.: 16.57
                                      Max.   :33.920   Max.   :103.91
  veil.type          veil.color          has.ring          ring.type         spore.print.color
habitat             season
 Length:61069       Length:61069       Length:61069       Length:61069       Length:61069
Length:61069       Length:61069
 Class :character   Class :character   Class :character   Class :character   Class :character
Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character
Mode  :character   Mode  :character
```

There are many categorical variables in this dataset. In order to prepare these for exploratory data analysis and later machine learning, these need to be converted into numerical representation.

Hide

```
# converts character to numeric encoding
convert.to.numeric <- function(vector) {
  vector <- as.numeric(as.factor(vector))
}
```

Hide

```
# get all categorical columns
cols = names(subset(df, select= -c(cap.diameter, stem.height, stem.width)))

# convert each to numeric
for(col in cols) {
  df[,col] <- convert.to.numeric(df[,col])-1
}

# view
head(df)
```

	cla…	cap.diameter	cap.shape	cap.surface	cap.color	does.bruise.or.bleed	gill.attach
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
1	1	15.26	6	3	6	0	
2	1	16.60	6	3	6	0	
3	1	14.07	6	3	6	0	
4	1	14.17	2	4	1	0	
5	1	14.64	6	4	6	0	
6	1	15.34	6	3	6	0	

6 rows | 1-8 of 21 columns

Hide

```
summary(df)
```

```
     class         cap.diameter      cap.shape      cap.surface       cap.color      does.bruis
e.or.bleed gill.attachment  gill.spacing
 Min.   :0.0000   Min.   : 0.380   Min.   :0.000   Min.   : 0.000   Min.   : 0.000   Min.   :0.0
000       Min.   :0.000   Min.   :0.0000
 1st Qu.:0.0000   1st Qu.: 3.480   1st Qu.:2.000   1st Qu.: 1.000   1st Qu.: 5.000   1st Qu.:0.0
000       1st Qu.:1.000   1st Qu.:0.0000
 Median :1.0000   Median : 5.860   Median :5.000   Median : 4.000   Median : 5.000   Median :0.0
000       Median :2.000   Median :1.0000
 Mean   :0.5549   Mean   : 6.734   Mean   :4.042   Mean   : 4.984   Mean   : 6.145   Mean   :0.1
734       Mean   :2.948   Mean   :0.8324
 3rd Qu.:1.0000   3rd Qu.: 8.540   3rd Qu.:6.000   3rd Qu.: 9.000   3rd Qu.:10.000   3rd Qu.:0.0
000       3rd Qu.:5.000   3rd Qu.:1.0000
 Max.   :1.0000   Max.   :62.340   Max.   :6.000   Max.   :11.000   Max.   :11.000   Max.   :1.0
000       Max.   :7.000   Max.   :3.0000
   gill.color      stem.height       stem.width       stem.root       stem.surface     stem.colo
r       veil.type         veil.color
 Min.   : 0.000   Min.   : 0.000   Min.   :  0.00   Min.   :0.0000   Min.   :0.000   Min.   : 0.
000   Min.   :0.00000   Min.   :0.0000
 1st Qu.: 5.000   1st Qu.: 4.640   1st Qu.:  5.21   1st Qu.:0.0000   1st Qu.:0.000   1st Qu.: 6.
000   1st Qu.:0.00000   1st Qu.:0.0000
 Median : 7.000   Median : 5.950   Median : 10.19   Median :0.0000   Median :0.000   Median :11.
000   Median :0.00000   Median :0.0000
 Mean   : 7.339   Mean   : 6.582   Mean   : 12.15   Mean   :0.4798   Mean   :2.061   Mean   : 8.
429   Mean   :0.05202   Mean   :0.5634
 3rd Qu.:10.000   3rd Qu.: 7.740   3rd Qu.: 16.57   3rd Qu.:0.0000   3rd Qu.:4.000   3rd Qu.:11.
000   3rd Qu.:0.00000   3rd Qu.:0.0000
 Max.   :11.000   Max.   :33.920   Max.   :103.91   Max.   :5.0000   Max.   :8.000   Max.   :12.
000   Max.   :1.00000   Max.   :6.0000
    has.ring        ring.type     spore.print.color    habitat           season
 Min.   :0.0000   Min.   :0.000   Min.   :0.0000    Min.   :0.0000   Min.   :0.000
 1st Qu.:0.0000   1st Qu.:2.000   1st Qu.:0.0000    1st Qu.:0.0000   1st Qu.:0.000
 Median :0.0000   Median :2.000   Median :0.0000    Median :0.0000   Median :1.000
 Mean   :0.2486   Mean   :2.369   Mean   :0.3804    Mean   :0.6237   Mean   :1.053
 3rd Qu.:0.0000   3rd Qu.:2.000   3rd Qu.:0.0000    3rd Qu.:1.0000   3rd Qu.:2.000
 Max.   :1.0000   Max.   :8.000   Max.   :7.0000    Max.   :7.0000   Max.   :3.000
```

The target in this dataset is the ‘class’ column which depicts if the mushroom is edible “0” or poisonous “1”

Hide

```
classes <- unique(df[,'class'])
print(paste("Edible",classes[2]))
```

```
[1] "Edible 0"
```

Hide

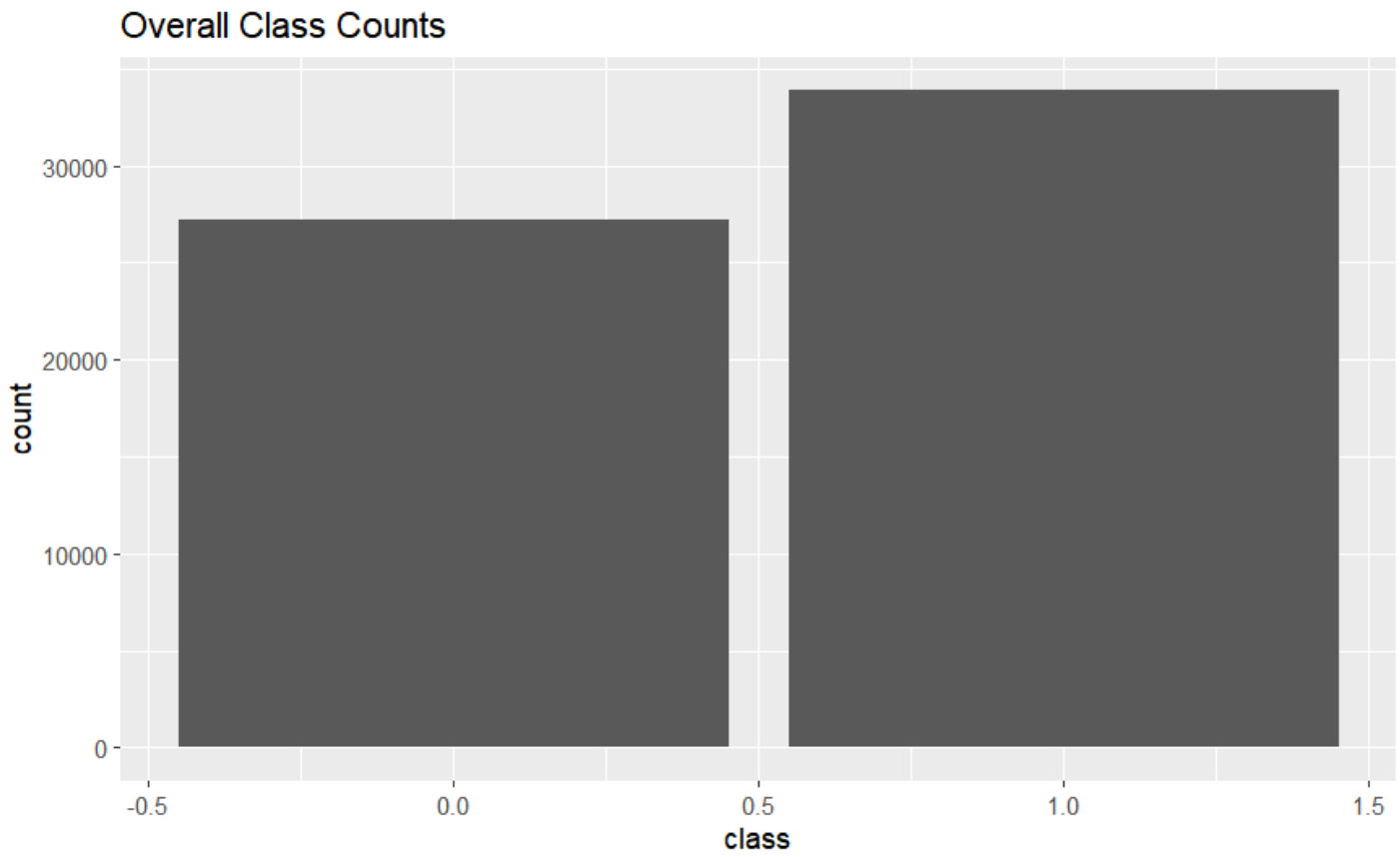
```
print(paste("Poisonous",classes[1]))
```

```
[1] "Poisonous 1"
```

Before moving forward, let's see explore the balance between these two classes

Hide

```
ggplot(df, aes(x = class)) +
  geom_bar() +
  ggtitle("Overall Class Counts")
```



It appears there are more poisonous mushrooms in the a dataset than there are edible mushrooms. An adjustment to this could be made by generating synthetic examples of the edible mushrooms through oversampling. Before doing so however, let's consider the distribution of each of them and what percentage of the data set belongs to poisonous versus edible mushrooms.

Hide

```
total.rows <- nrow(df)
edible.rows <- sum(df[,'class']==0)
poisonous.rows <- total.rows - edible.rows

print(paste("Edible % =", round((edible.rows/total.rows)*100, 2)))
```

```
[1] "Edible % = 44.51"
```

Hide

```
print(paste("Poisonous % =", round((poisonous.rows/total.rows)*100, 2)))
```

```
[1] "Poisonous % = 55.49"
```

It appears there is a slight imbalance to the data. While ideally, we'd like a 50/50 split, the proportion of edible vs poisonous mushrooms isn't that far off. I will leave this for now and continue doing exploratory data analysis and data preparation before the machine learning.

Hide

```
str(df)
```

```
'data.frame':   61069 obs. of  21 variables:
 $ class               : num  1 1 1 1 1 1 1 1 1 1 ...
 $ cap.diameter        : num  15.3 16.6 14.1 14.2 14.6 ...
 $ cap.shape           : num  6 6 6 2 6 6 2 6 2 2 ...
 $ cap.surface         : num  3 3 3 4 4 3 4 4 3 3 ...
 $ cap.color           : num  6 6 6 1 6 6 6 1 6 1 ...
 $ does.bruise.or.bleed: num  0 0 0 0 0 0 0 0 0 0 ...
 $ gill.attachment     : num  3 3 3 3 3 3 3 3 3 3 ...
 $ gill.spacing        : num  0 0 0 0 0 0 0 0 0 0 ...
 $ gill.color          : num  10 10 10 10 10 10 10 10 10 10 ...
 $ stem.height         : num  16.9 18 17.8 15.8 16.5 ...
 $ stem.width          : num  17.1 18.2 17.7 16 17.2 ...
 $ stem.root           : num  5 5 5 5 5 5 5 5 5 5 ...
 $ stem.surface        : num  8 8 8 8 8 8 8 8 8 8 ...
 $ stem.color          : num  11 11 11 11 11 11 11 11 11 11 ...
 $ veil.type           : num  1 1 1 1 1 1 1 1 1 1 ...
 $ veil.color          : num  5 5 5 5 5 5 5 5 5 5 ...
 $ has.ring            : num  1 1 1 1 1 1 1 1 1 1 ...
 $ ring.type           : num  3 3 3 6 6 6 3 6 6 6 ...
 $ spore.print.color   : num  0 0 0 0 0 0 0 0 0 0 ...
 $ habitat             : num  0 0 0 0 0 0 0 0 0 0 ...
 $ season              : num  3 2 3 3 3 2 3 2 0 3 ...
```

All variables have been transformed into encoded values where the categoricals hold low n-ary discrete values and the continuous variables are left untouch represent those values in the real number spectrum.

Train/Test Split

Hide

```
set.seed(123)

# 80/20 split
i <- sample(1:nrow(df), nrow(df)*0.8, replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Hide

```
print(paste("# rows (train) =", nrow(train)))
```

```
[1] "# rows (train) = 48855"
```

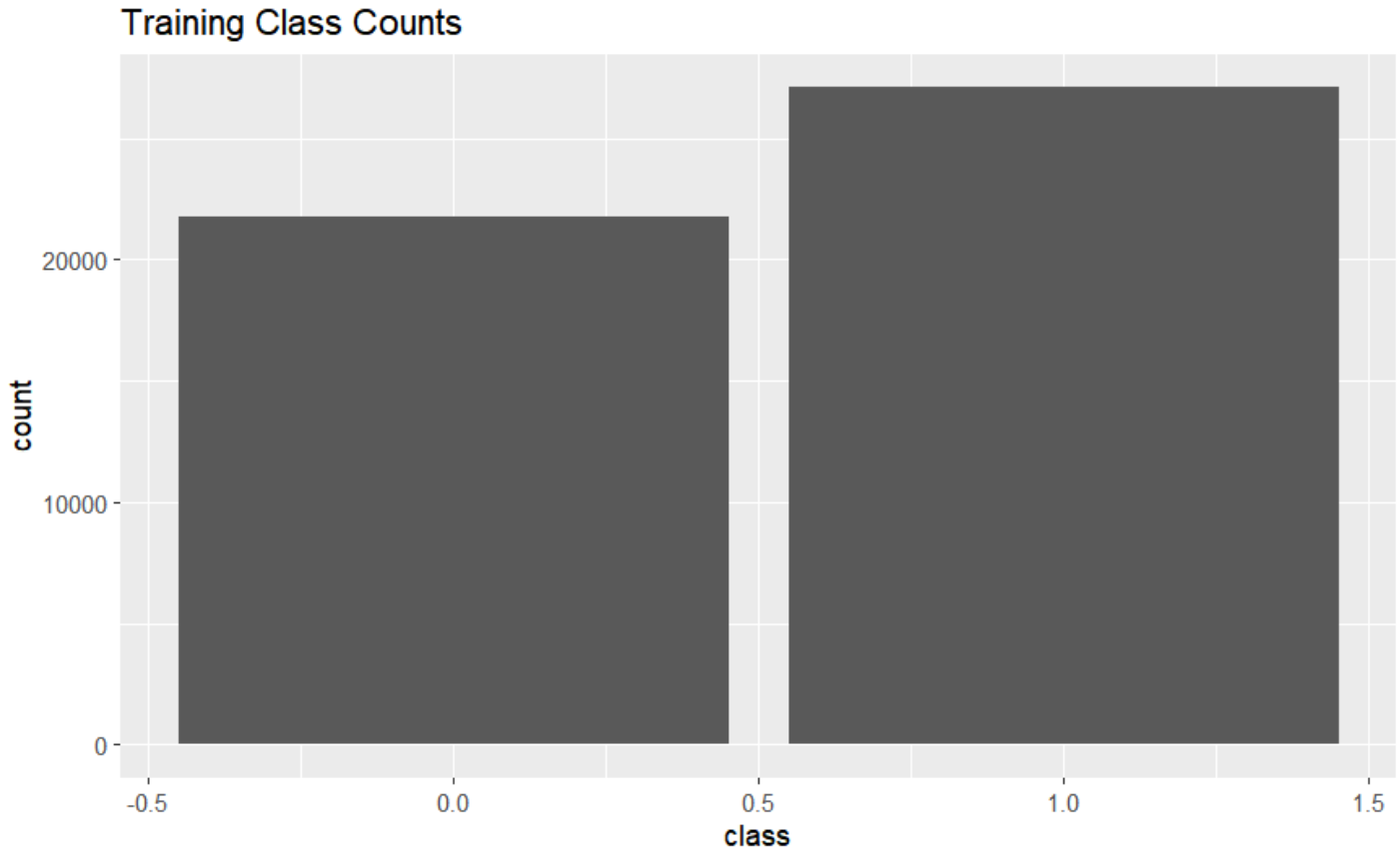
Hide

```
print(paste("# rows (test) =", nrow(test)))
```

```
[1] "# rows (test) = 12214"
```

Hide

```
ggplot(train, aes(x = class)) +
  geom_bar() +
  ggtitle("Training Class Counts")
```



Hide

```
NA
NA
```

Logistic Regression

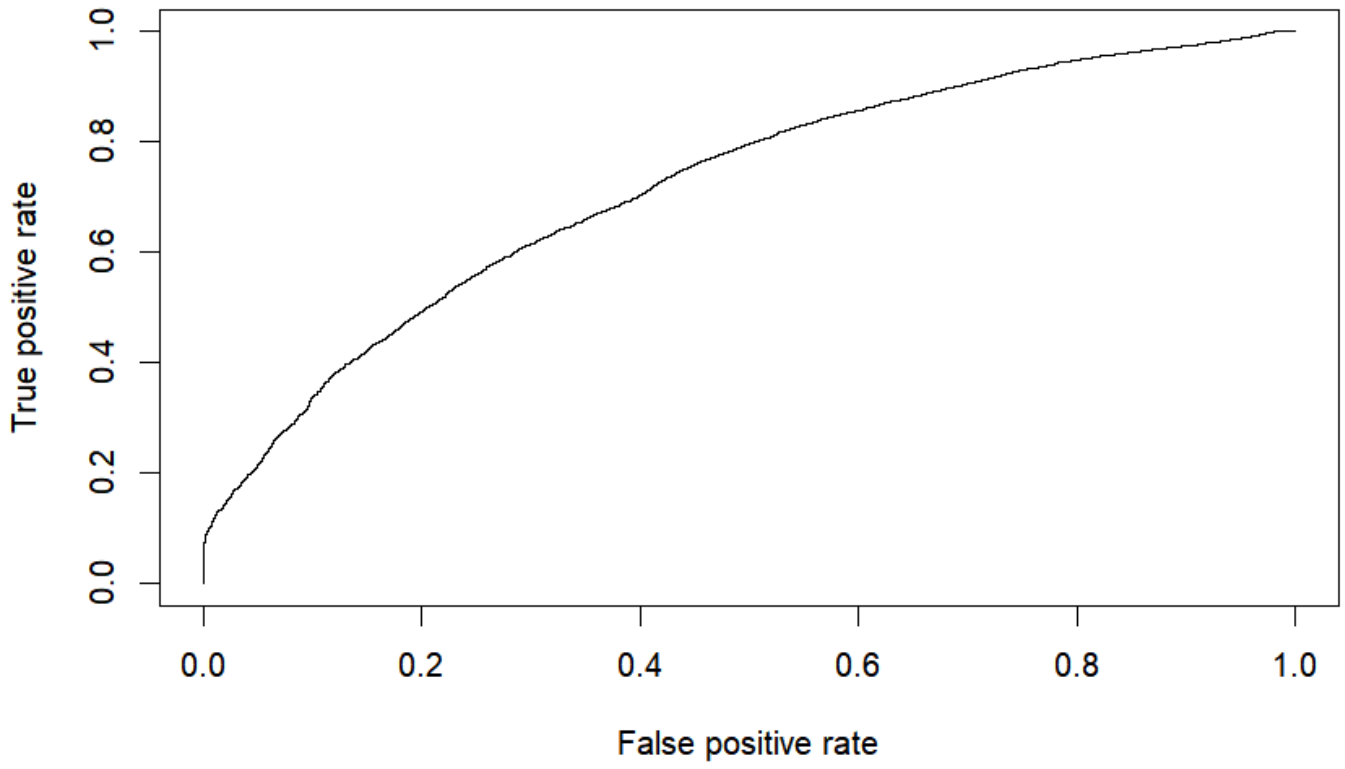
Hide

```
mod <- glm(class ~ ., data = train, family = "binomial")
summary(mod)
```

```
Call:
glm(formula = class ~ ., family = "binomial", data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2661  -1.0945   0.5857   1.0122   2.2871

Coefficients:
                     Estimate Std. Error z value Pr(>|z|)
(Intercept)          2.346287   0.055956  41.931  < 2e-16 ***
cap.diameter        -0.078093   0.003662 -21.324  < 2e-16 ***
cap.shape           -0.119752   0.004936 -24.261  < 2e-16 ***
cap.surface          0.010048   0.002639   3.807 0.000140 ***
cap.color            0.058614   0.003313  17.692  < 2e-16 ***
does.bruise.or.bleed 0.115944   0.028155   4.118 3.82e-05 ***
gill.attachment     -0.023271   0.004397  -5.293 1.20e-07 ***
gill.spacing        -0.223256   0.013044 -17.116  < 2e-16 ***
gill.color          -0.009151   0.003400  -2.691 0.007122 **
stem.height         -0.121176   0.004292 -28.231  < 2e-16 ***
stem.width          -0.003502   0.001659  -2.111 0.034769 *
stem.root            0.368328   0.009696  37.986  < 2e-16 ***
stem.surface         0.013943   0.003703   3.765 0.000166 ***
stem.color          -0.081646   0.003374 -24.195  < 2e-16 ***
veil.type            1.644869   0.065722  25.028  < 2e-16 ***
veil.color          -0.242807   0.009537 -25.460  < 2e-16 ***
has.ring             0.509632   0.030829  16.531  < 2e-16 ***
ring.type            0.109494   0.007546  14.509  < 2e-16 ***
spore.print.color    0.140634   0.008483  16.578  < 2e-16 ***
habitat             -0.165278   0.008131 -20.327  < 2e-16 ***
season              -0.137476   0.009003 -15.270  < 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: 67138  on 48854  degrees of freedom
Residual deviance: 59776  on 48834  degrees of freedom
AIC: 59818

Number of Fisher Scoring iterations: 4
```

Hide

```
pred <- predict(mod, newdata= test, select= -c(class), type="response")
pred <- prediction(pred, test$class)
perf <- performance(pred, measure="tpr", x.measure="fpr")
plot(perf)
```



Hide

```
print(paste("AUC =", performance(pred,measure="auc")@y.values[[1]]))
```

```
[1] "AUC = 0.720367426954618"
```

It appears the model is distinguishing between the classes. The ROC is slightly curved in a log-linear fashion. The area under the curve is in the lower 70% range. Let's see what the accuracy is of our model when evaluated.

Hide

```
probas <- predict(mod, newdata= test, select= -c(class), type="response")
pred <- ifelse(probas>0.5,1,0)
confusionMatrix(as.factor(pred), reference=as.factor(test$class))
```

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2985 1633
         1 2448 5148

               Accuracy : 0.6659
                 95% CI : (0.6574, 0.6742)
    No Information Rate : 0.5552
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.3133

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.5494
            Specificity : 0.7592
         Pos Pred Value : 0.6464
         Neg Pred Value : 0.6777
             Prevalence : 0.4448
         Detection Rate : 0.2444
   Detection Prevalence : 0.3781
      Balanced Accuracy : 0.6543

       'Positive' Class : 0
```

Using the logistic regression model to predict the class of the mushroom, it appears the model was able to distingush between the two classes at a roughly 70% given the area under the curve. The overall accuracy of the model, how many true positive and true negative predictions over the total number of predictions made, was edging closer to 66.6%. The model appears to be worse at predicting edible mushrooms (sensitivity) than it is at predicting poisonous mushrooms (specificity). This makes sense given the training data had more poisonous examples than edible examples in it. Had the classes been balanced, it would be interesting to see if these numbers would change at all.

In some sense this is good. Suppose this model was deployed and being used in an application identifying edible and poisonous mushrooms. Wouldn't you rather have the model predict that something was poisonous when it was actually edible, than the other way around? The cost of misclassification in the latter case could be the difference between life or death. On the contrary, using starvation as an argument, classifying something as poisonous when it was actually edible, could be a deal breaker as well, as it could be the difference between having a meal or not.

Nonetheless, let's explore naive bayes to see if these numbers improve.

Naive Bayes

Hide

```
# create model over sample train sample as logistic regression
nb_mod <- naiveBayes(class ~., train)
nb_mod
```

```
Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
       0        1
0.445154 0.554846

Conditional probabilities:
   cap.diameter
Y       [,1]     [,2]
  0 7.790217 6.348063
  1 5.888382 3.968284

   cap.shape
Y       [,1]     [,2]
  0 4.293222 1.982114
  1 3.837201 2.237376

   cap.surface
Y       [,1]     [,2]
  0 4.942569 4.065254
  1 5.020474 3.876181

   cap.color
Y       [,1]     [,2]
  0 6.052832 3.141446
  1 6.213008 3.188547

   does.bruise.or.bleed
Y        [,1]      [,2]
  0 0.1829134 0.3866044
  1 0.1682960 0.3741358

   gill.attachment
Y       [,1]     [,2]
  0 3.209812 2.307246
  1 2.730476 2.418805

   gill.spacing
Y        [,1]     [,2]
  0 0.8722641 0.874661
  1 0.8040727 0.848661

   gill.color
Y       [,1]     [,2]
  0 7.536693 3.239052
  1 7.147194 3.140848
```

```
   stem.height
Y       [,1]     [,2]
  0 7.025352 3.570694
  1 6.215035 3.140294

   stem.width
Y       [,1]      [,2]
  0 14.36844 11.046095
  1 10.39143  8.761006

   stem.root
Y        [,1]     [,2]
  0 0.3355711 1.118553
  1 0.5913602 1.415576

   stem.surface
Y       [,1]     [,2]
  0 1.896588 2.909800
  1 2.175047 2.935196

   stem.color
Y       [,1]     [,2]
  0 8.775841 3.064790
  1 8.135242 3.306867

   veil.type
Y         [,1]      [,2]
  0 0.03977377 0.1954318
  1 0.06186594 0.2409163

   veil.color
Y        [,1]     [,2]
  0 0.6028141 1.663750
  1 0.5292729 1.464773

   has.ring
Y        [,1]      [,2]
  0 0.2203421 0.4144869
  1 0.2691925 0.4435483

   ring.type
Y       [,1]     [,2]
  0 2.197949 1.202319
  1 2.497731 1.750680

   spore.print.color
Y        [,1]     [,2]
  0 0.2712893 1.206034
  1 0.4702844 1.324260

   habitat
Y        [,1]     [,2]
```

```
  0 0.6882012 1.418640
  1 0.5764563 1.131739

   season
Y        [,1]     [,2]
  0 1.1145852 1.132636
  1 0.9951673 1.070908
```

Hide

```
# predict on test for classes
p1 <- predict(nb_mod, newdata= test, select= -c(class), type="class")

# view confusion matrix
table(p1, test$class)
```

```
p1     0    1
  0 2806 2005
  1 2627 4776
```

Hide

```
# calculate accuracy
acc <- mean(p1==test$class)
print(paste("accuracy % =", acc))
```

```
[1] "accuracy % = 0.620763058785001"
```

Hide

```
head(test)
```

	cla…	cap.diameter	cap.shape	cap.surface	cap.color	does.bruise.or.bleed	gill.attac
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
1	1	15.26	6	3	6	0	
4	1	14.17	2	4	1	0	
9	1	12.85	2	3	6	0	
11	1	14.17	2	4	1	0	
13	1	17.37	6	4	6	0	
15	1	15.37	6	4	1	0	

6 rows | 1-8 of 21 columns

Hide

```
# extract raw probabilities
probas <- predict(nb_mod, newdata=test, type="raw")
head(probas)
```

```
                0         1
[1,] 0.0359130124 0.9640870
[2,] 0.0006408067 0.9993592
[3,] 0.0004902167 0.9995098
[4,] 0.0009128645 0.9990871
[5,] 0.1058760509 0.8941239
[6,] 0.0412088032 0.9587912
```

For the first 6 observations we can see by looking at the test set, those are poisonous mushrooms. The probability estimates from naive bayes shows ta high probability for guessing 1 for those observations. The accuracy score of the prediction is still slightly lower than that of logistic regression by around 4%.