Model Evaluation: Classification

Data Mining & Machine Learning I

(from original lab created by Dr. Simon Caton and Dr. Cristian Rusu)

Data Preparation

Because Naive Bayes uses frequency tables for learning the data, each feature must be categorical in order to create the combinations of class and feature values comprising the matrix. Thus this time we need to transform our numeric features to categorical features, i.e., discretize numeric features. This simply means to bin numeric values, correspondingly discretization is also sometimes called binning. There are several different ways to discretize a numeric feature. Perhaps the most common is to explore the data for natural categories or cut points in the distribution of data.

Inspecting our data (note this is the same preprocessed data as we used in the previous lab):

```
str(titanicData)
```

```
## 'data.frame':    891 obs. of  12 variables:
##  $ Survived: Factor w/ 2 levels "No","Yes": 1 2 2 2 1 1 1 1 2 2 ...
##  $ Pclass  : Factor w/ 3 levels "1","2","3": 3 1 3 1 3 3 1 3 3 2 ...
##  $ Sex     : Factor w/ 2 levels "female","male": 2 1 1 1 2 2 2 2 1 1 ...
##  $ Age     : num  22 38 26 35 35 21 54 2 27 14 ...
##  $ SibSp   : int  1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch   : int  0 0 0 0 0 0 0 1 2 0 ...
##  $ Fare    : num  7.25 71.28 7.92 53.1 8.05 ...
##  $ Embarked: Factor w/ 3 levels "C","Q","S": 3 1 3 3 3 2 3 3 3 1 ...
##  $ Child   : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 2 1 2 ...
##  $ Title   : Factor w/ 5 levels "Master","Miss",..: 3 4 2 4 3 3 3 1 4 4 ...
##  $ Fsize   : num  2 2 1 2 1 1 1 5 3 2 ...
##  $ FsizeD  : Factor w/ 3 levels "large","singleton",..: 3 3 2 3 2 2 2 1 3 3 ...
```

We can actually just remove Fsize, as FsizeD is essentially contains the same information. Thus we will need to discretize Age, SibSp, Parch and Fare. Let's start with the simpler numeric values:

```
table(titanicData$SibSp)
```

```
##
##   0   1   2   3   4   5   8
## 608 209  28  16  18   5   7
```

```
table(titanicData$Parch)
```

```
##
##   0   1   2   3   4   5   6
## 678 118  80   5   4   5   1
```

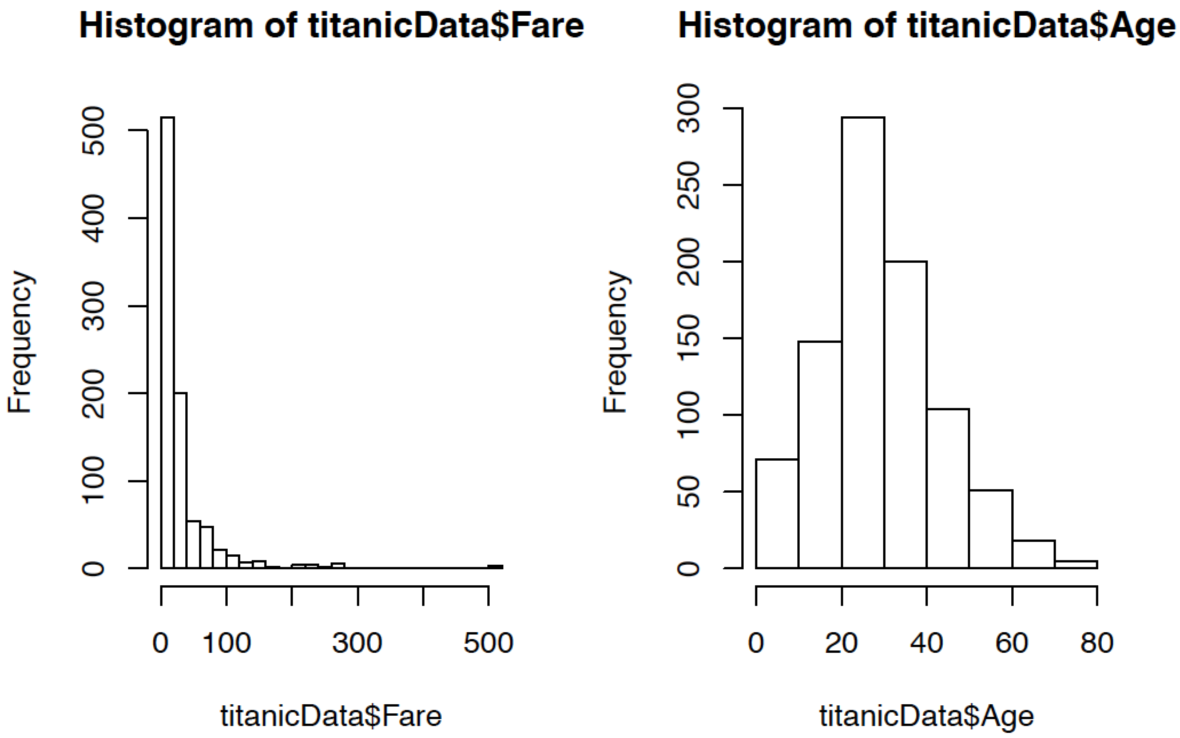
As we can see, there are only a small number of values, so in this case, we could just convert the attibute to a factor, like we did with Pclass.

Task:

Transform SibSp and Parch to factors.

Now for our two remaining numerics, let's check their distributions.

```r
par(mfrow=c(1,2))
hist(titanicData$Fare, breaks = 30)
hist(titanicData$Age)
```



So Fare is very skewed, but Age is not far off of a normal distribution. For fare, we need to make decisions concerning where to put the boundaries. We can do this somewhat based on the visual interpretation of the plot: It seems that most tickets cost 10 or less, then another group around 10 - 50, and then a small number of significantly more expensive tickets:

```r
titanicData$FareBinned <- cut(titanicData$Fare,
  breaks = c(0,10,50,max(titanicData$Fare)),
  labels=c("low", "middle", "high"))
```

This somewhat captures the class of the tickets:

```r
table(titanicData$FareBinned, titanicData$Pclass)
```

```
##
##             1   2   3
##   low       1   0 320
##   middle   71 171 153
##   high    139   7  14
```

It seems that we are reasonably good at the ends of the scale, but the middle is a little messy. We could also check some simple descriptives of Fare against class.

```r
aggregate(Fare ~ Pclass, data=titanicData, FUN=summary)
```

```
##   Pclass Fare.Min. Fare.1st Qu. Fare.Median Fare.Mean Fare.3rd Qu.
## 1      1   0.00000     30.92395    60.28750  84.15469     93.50000
## 2      2   0.00000     13.00000    14.25000  20.66218     26.00000
## 3      3   0.00000      7.75000     8.05000  13.67555     15.50000
##    Fare.Max.
## 1 512.32920
## 2  73.50000
## 3  69.55000
```

2nd and 3rd class don't appear to be that distinguishable in terms of price in terms of binning the Fare attribute. So let's leave it as is for now, and you can revisit this decision later to check how it affects model performance (if at all).
Moving on to Age, there are a lot of possible options available to us. We'll stick to something simple.

```
titanicData$AgeBinned <- cut(titanicData$Age,
  breaks = c(0,10,20,30,40,50,60,70,max(titanicData$Age)),
  labels=c("0-10", "10-20", "20-30", "30-40", "40-50", "50-60", "60-70", "70+"))
table(titanicData$AgeBinned)
```

```
##
##  0-10 10-20 20-30 30-40 40-50 50-60 60-70   70+
##    71   148   294   200   104    51    18     5
```

Now, we remove our numeric attributes, and train the model.

Task:

Remove all numeric features.

Model Creation

Now to train a model you need to use the naiveBayes function in the e1071 library. The function call will look something like this:

```
nb <- naiveBayes(training, training$Survived)
```

Task:

Appropriately evaluate the performance of Naïve Bayes using the *naiveBayes* function.

Check if changing the decisions concerning the bins for *Age* and *Fare* impact performance.

For a 75/25 holdout, you should expect to see something like this, which is actually not much different to our womenModel.

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  115  23
##        Yes  25  60
##
##                Accuracy : 0.7848
##                  95% CI : (0.7249, 0.8368)
##     No Information Rate : 0.6278
##     P-Value [Acc > NIR] : 3.381e-07
##
##                   Kappa : 0.5417
##  Mcnemar's Test P-Value : 0.8852
```

```
##
##             Sensitivity : 0.7229
##             Specificity : 0.8214
##          Pos Pred Value : 0.7059
##          Neg Pred Value : 0.8333
##              Prevalence : 0.3722
##          Detection Rate : 0.2691
##    Detection Prevalence : 0.3812
##       Balanced Accuracy : 0.7722
##
##        'Positive' Class : Yes
##
```