Problem 9 - Occupation Prediction

Henry Neeb, Christopher Kurrus, Tyler Chase, and Yash Vyas

May 22, 2016

Libraries

```
library(gbm)
library(ggplot2)
```

File Parameters

```
setwd("Z:/Acads/3spr_2016/Stats315b/")
OccData.table = read.table(file = 'Occupation_Data.txt', header = F, sep =
',')
```

Read in Data

We name the column headers and convert the values into categorical variables. The columns that have unordered categorical variables are: * Occupation * House Type * Sex * Marital status * Dual income status * Whether you rent or own a house * Type of house residency * Ethnicity * Language spoken

While the columns that have categorical values with an ordered relationship are:

- Age
- Education
- Income
- Number of years of residency in the bay area
- Number of people in the house
- Number of people in house below 18 years

```
# Name of the variables imported
colnames(OccData.table) = c("occu","HouseType", "sex", "marStatus", "age",
                            "education",  "income",
                            "ResYears", "dualInc", "housePeople",
                            "Below18Peop", "HouseResType",
                            "Ethini", "Lingo")

# Converting to unordered categorical variable
OccData.table$occu = factor(OccData.table$occu, levels = c(1:9))
OccData.table$HouseType = factor(OccData.table$HouseType, levels = c(1:5))
OccData.table$sex = factor(OccData.table$sex, levels = c(1:2))
OccData.table$marStatus = factor(OccData.table$marStatus, levels = c(1:5))
OccData.table$dualInc = factor(OccData.table$dualInc, levels = c(1:3))
```

```
OccData.table$HouseResType =factor(OccData.table$HouseResType, levels =
c(1:3))
OccData.table$Ethini = factor(OccData.table$Ethini, levels = c(1:8))
OccData.table$Lingo = factor(OccData.table$Lingo, levels = c(1:3))

# Converting to ordered categorical variable
OccData.table$age = ordered(OccData.table$age, levels = c(1:7))
OccData.table$education = ordered(OccData.table$education, levels = c(1:6))
OccData.table$income = ordered(OccData.table$income, levels = c(1:9))
OccData.table$ResYears = ordered(OccData.table$ResYears, levels = c(1:5))
OccData.table$housePeople = ordered(OccData.table$housePeople, levels =
c(1:9))
OccData.table$Below18Peop = ordered(OccData.table$Below18Peop, levels =
c(0:9))
```

Randomize the Dat

We shuffle the data in order to ensure that the gbm choose does sequencial rows in test or train datasets.

```
# Randomize the data
set.seed(1)
u = runif(nrow(OccData.table))
OccData.tableShuffle = OccData.table[order(u),]


# Split 70% training and 30% test
train = sample(1:nrow(OccData.tableShuffle),
70/100*nrow(OccData.tableShuffle))
```

Fitting the GBM

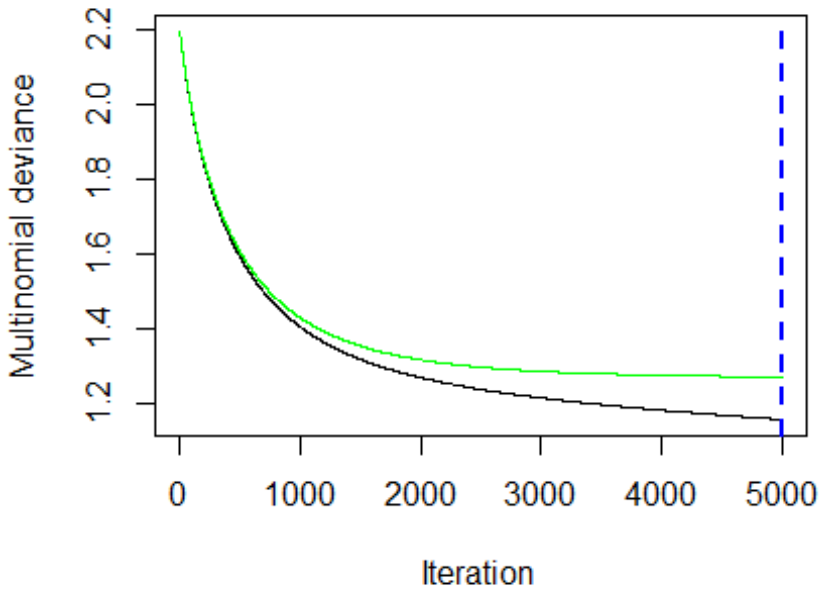
We now fit our first model. We choose 2500 randomly as values for number of trees.

```
# Model
boost.occ = gbm(occu ~., data= OccData.tableShuffle[train,],
                distribution = "multinomial",n.trees =5000,
                interaction.depth =4, cv.folds = 5, verbose = TRUE)

# Find best iteration by 5 fold crossvalidation
gbm.perf(object = boost.occ, method = "cv")

## [1] 4997

bestIter_cv <- gbm.perf(object = boost.occ, method = "cv")
```




```
bestIter_cv
```

```
## [1] 4997
```

We use 5 fold cross validation to pick our best number of trees. We will then use this amount (4997) to predict on our holdout testing data.

Determine Misclassification Rate on Test data

We now apply the predictions to our test data and generate the misclassification error.

```
# Determine Predictions based on best CV iteration
occ.pred = max.col(as.data.frame
                   (predict(boost.occ, newdata = OccData.tableShuffle[-
train,],
                            n.trees = bestIter_cv)))

misclass.matrix = table(occ.pred, OccData.tableShuffle[-train,1])
misclass.rate = 1 - sum(diag(misclass.matrix))/nrow(OccData.tableShuffle[-
train,])

#Obtaining the misclassification rate for each class

err.class = NA

for (i in 1:9){
  err.class[i] = 1- misclass.matrix[i,i]/sum(misclass.matrix[,i])
```

```
}
err.class
```

```
## [1] 0.1975758 0.9398148 0.6387665 0.7262248 0.4202899 0.1834452 0.7600000
## [8] 0.1881188 0.7946429
```

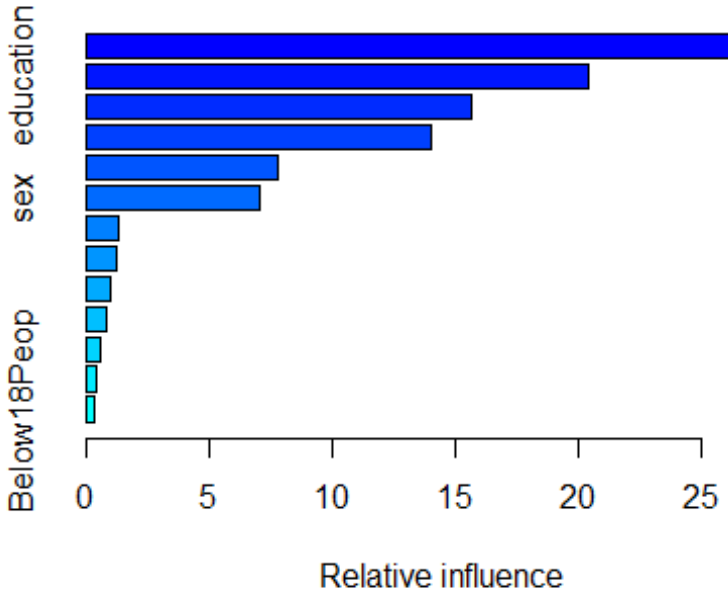
Occupation Prediction

We get `0.4206` as our overall boosted tree misclassification error. Our optimal number of trees that we fit is 4997. We selected this based on a 5-fold cross validation.

We have fit trees with depth of 4. And used the default shrinkage parameter as 0.001.

The most important predictors for income in order are:

```
summary(boost.occ)
```




```
##                        var    rel.inf
## age                    age 29.2317564
## education        education 20.4035857
## income              income 15.6619919
## HouseResType HouseResType 14.0238070
## dualInc            dualInc  7.8191364
## sex                    sex  7.0419062
## housePeople    housePeople  1.3595753
## HouseType        HouseType  1.2137822
## Ethini              Ethini  1.0384646
## marStatus        marStatus  0.8101742
```

```
## ResYears          ResYears  0.5922543
## Lingo                Lingo  0.4388237
## Below18Peop    Below18Peop  0.3647421
```

Conclusions

We notice that the most important variable in order to predict the occupation of the person is his age. Education level, income and the type of house a person lives in are also among the top variables that help in predicting the occupation of a person. The importance of these variables corroborates our beliefs. Age clearly determines the type of work that one pursues. Also, occupations are highly associated with education levels. Rarely would one find a person with low education level to be in a professional role. and all occupation levels do not offer same pay.

We also noticed that the type of occupation of a person also depends on whether there is a dual source of income in the house and the sex of the person.

Besides, we find that the misclassification rate for a person in one of the following classes is highly accurate:

- Professional/Managerial
- Homemaker
- Student,
- HS or College
- Retired

as compared to predicting the occupation of a person belonging to the following class:

- Sales Worker
- Factory Worker/Laborer/Driver
- Clerical/Service Worker
- Military
- Unemployed