Freestyle Prediction

How to be hired?

Ryan Lee

Data preparation

```
hired <- read.csv('HireTrainApr10.csv',stringsAsFactors = T)
v <- sample(1:nrow(hired))
hiredScrambled <- hired[v,]

train <- hiredScrambled[1:1900,]
valid <- hiredScrambled[(nrow(hiredScrambled)-99):nrow(hiredScrambled),]
```

- First, I imported 'HireTrainApr10.csv' as a dataset.
- Number of rows of hired dataset: 2000
- For 1-step cross validation, I separated train dataset and valid dataset.

Get to know your data

- To be familiar with my dataset, I conducted same code as textbook snippet.
- Used colnames, summary, unique, table function.

```
#Get to know your data
colnames(train)
summary(train)
unique(train$Coding)
unique(train$Impression)
unique(train$Major)
unique(train$College)
table(train$Coding,train$Hired)
table(train$Impression,train$Hired)
table(train$Major,train$Hired)
table(train$College,train$Hired)
```

Get to know your data

```
> table(train$Coding,train$Hired)

           No Yes
  Excellent  35 598
  OK         93 534
  Weak      576  64
> table(train$Impression,train$Hired)

           No Yes
  Confident 168 304
  Nerdy     177 308
  Outgoing  139 312
  Shy       220 272
> table(train$Major,train$Hired)

              No Yes
  CS          171 316
  DataScience 172 301
  IT          182 282
  Stats       179 297
> table(train$College,train$Hired)

              No Yes
  BestCollege 208 176
  BYU         145 252
  Peters      120 256
  PJIT        119 221
  Redbrick    112 291
```

```
> colnames(train)
[1] "Coding"     "Impression" "Major"      "College"    "Hired"
> summary(train)
       Coding        Impression          Major            College    Hired
 Excellent:633   Confident:472   CS         :487   BestCollege:384   No : 704
 OK       :627   Nerdy    :485   DataScience:473   BYU        :397   Yes:1196
 Weak     :640   Outgoing :451   IT         :464   Peters     :376
                 Shy      :492   Stats      :476   PJIT       :340
                                                   Redbrick   :403

> unique(train$Coding)
[1] Excellent OK        Weak
Levels: Excellent OK Weak
> unique(train$Impression)
[1] Outgoing  Nerdy     Shy       Confident
Levels: Confident Nerdy Outgoing Shy
> unique(train$Major)
[1] DataScience CS          IT          Stats
Levels: CS DataScience IT Stats
> unique(train$College)
[1] Peters      BYU         BestCollege Redbrick    PJIT
Levels: BestCollege BYU Peters PJIT Redbrick
```

Get to know your data

Bayesian

- Tried Odds of being hired when you are from Best College.

```
> #Odds of being hired when you are from BestCollege
> Prior<-nrow(hired[hired$College=='BestCollege',])/nrow(hired)
> Prior
[1] 0.201
> PriorOdds<-round(Prior/(1-Prior),2)
> PriorOdds
[1] 0.25
> TruePositive<-round(nrow(hired[hired$Hired=='Yes' & hired$College=='BestCollege',])/nrow(hired[hired$Hired=='Yes',]),2)
> TruePositive
[1] 0.15
> FalsePositive<-round(nrow(hired[hired$Hired!='Yes' & hired$College=='BestCollege',])/nrow(hired[hired$Hired!='Yes',]),2)
> FalsePositive
[1] 0.29
> LikelihoodRatio<-round(TruePositive/FalsePositive,2)
> LikelihoodRatio
[1] 0.52
> PosteriorOdds <-LikelihoodRatio * PriorOdds
> PosteriorOdds
[1] 0.13
> Posterior <-PosteriorOdds/(1+PosteriorOdds)
> Posterior
[1] 0.1150442
```

```
#Odds of being hired when you are from BestCollege
Prior<-nrow(hired[hired$College=='BestCollege',])/nrow(hired)
Prior
PriorOdds<-round(Prior/(1-Prior),2)
PriorOdds
TruePositive<-round(nrow(hired[hired$Hired=='Yes' & hired$College=='BestCollege',])/nrow(hired[hired$Hired=='Yes',]),2)
TruePositive
FalsePositive<-round(nrow(hired[hired$Hired!='Yes' & hired$College=='BestCollege',])/nrow(hired[hired$Hired!='Yes',]),2)
FalsePositive
LikelihoodRatio<-round(TruePositive/FalsePositive,2)
LikelihoodRatio
PosteriorOdds <-LikelihoodRatio * PriorOdds
PosteriorOdds
Posterior <-PosteriorOdds/(1+PosteriorOdds)
Posterior
```

Get to know your data

- From the dataset analysis, we could know the distribution of the data and type of values for each column.
- In table (train$Coding, train$Hired), I could observe that most of ‘Excellent’ or ‘Ok’ coding skill could get a job while ‘Weak’ code skill could not get a job.
- From this analysis, I tried to focus on the coding skill first.

Chisq test

- I conducted Chisq Test to check which columns are dependent from hired status.
- Coding, Impression, College columns' p-value are less than 0.05. So, we can say they are kinda dependent to the hired status.
- (But, Never say never as professor said.)

```r
#chisq test
chisq.test(train$Coding,train$Hired)
chisq.test(train$Impression,train$Hired)
chisq.test(train$Major,train$Hired)
chisq.test(train$College,train$Hired)
```

```
> #chisq test
> chisq.test(train$Coding,train$Hired)

        Pearson's Chi-squared test

data:  train$Coding and train$Hired
X-squared = 1171.7, df = 2, p-value < 2.2e-16

> chisq.test(train$Impression,train$Hired)

        Pearson's Chi-squared test

data:  train$Impression and train$Hired
X-squared = 20.393, df = 3, p-value = 0.0001407

> chisq.test(train$Major,train$Hired)

        Pearson's Chi-squared test

data:  train$Major and train$Hired
X-squared = 1.8822, df = 3, p-value = 0.5972

> chisq.test(train$College,train$Hired)

        Pearson's Chi-squared test

data:  train$College and train$Hired
X-squared = 67.958, df = 4, p-value = 6.123e-14
```

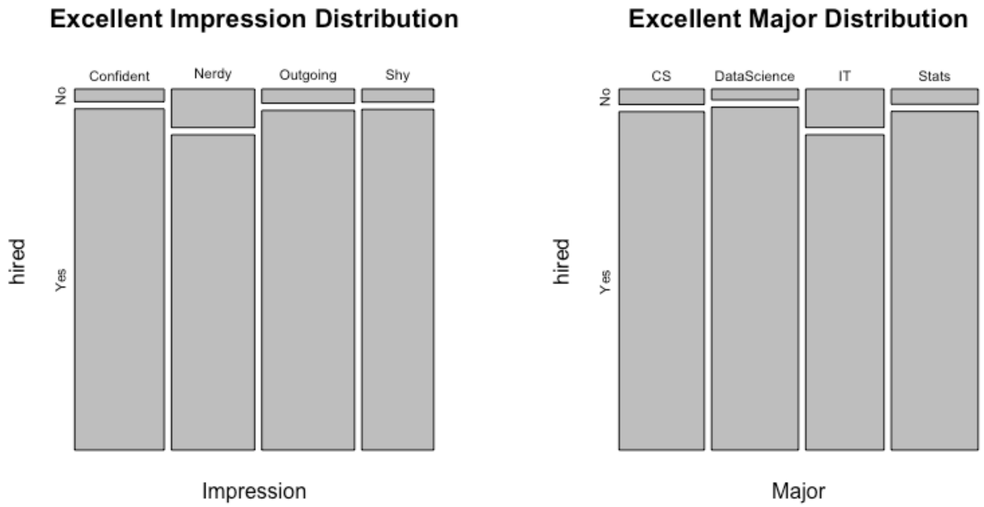
Mosaic Plots to find irregular points

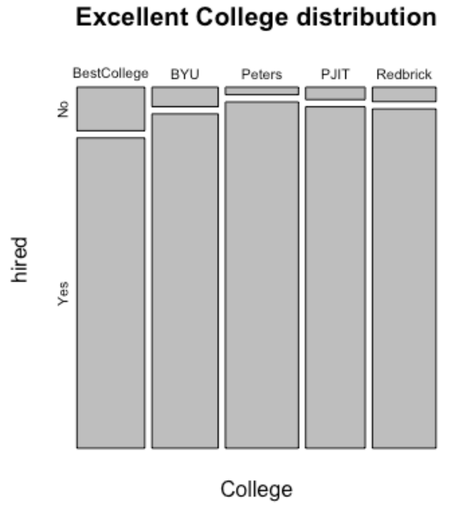
- As I mentioned before, most of ‘Excellent’ or ‘ok’ coding skills got a job, and ‘Weak’ coding skill couldn’t get a job. So, I set default value for ‘Excellent’ or ‘OK’ to ‘Yes’ and ‘Weak’ to ‘No’. But, there are still irregulars with this standard. To find irregular points, I conducted this codes.

```
#Mosaic plots to view
#coding
excellentCoding <- train[train$Coding== 'Excellent',]
mosaicplot(excellentCoding$Impression~excellentCoding$Hired,xlab='Impression',ylab='hired')
mosaicplot(excellentCoding$Major~excellentCoding$Hired,xlab='Major',ylab='hired')
mosaicplot(excellentCoding$College~excellentCoding$Hired,xlab='College',ylab='hired')
a <- excellentCoding[excellentCoding$Major=='CS',]

okCoding <- train[train$Coding== 'OK',]
mosaicplot(okCoding$Impression~okCoding$Hired,xlab='Impression',ylab='hired')
mosaicplot(okCoding$Major~okCoding$Hired,xlab='Major',ylab='hired')
mosaicplot(okCoding$College~okCoding$Hired,xlab='College',ylab='hired')
a <- okCoding[okCoding$College=='PJIT',]

weakCoding <- train[train$Coding=='Weak',]
mosaicplot(weakCoding$Impression~weakCoding$Hired,xlab='Impression',ylab='hired')
mosaicplot(weakCoding$Major~weakCoding$Hired,xlab='Major',ylab='hired')
mosaicplot(weakCoding$College~weakCoding$Hired,xlab='College',ylab='hired')
a<-weakCoding[weakCoding$Impression=='Shy' & weakCoding$Major=='Stats',]
```

From Excellent Coding Skills

- From this mosaic plots, we can observe some weird points.
- Nerdy, IT, or Best College has more ‘No’s for hired column. So, I tried to focus on that parts.





Excellent Coding

```
temp <- excellentCoding[ excellentCoding$Hired=='No',]
```

- With this code, many of Nerdy, Best College, IT students could not get a job. I also checked for other conditions like the below, irregular point is only at Nerdy IT Best college students.

```
isbyu <- excellentCoding[excellentCoding$College=='BYU' & excellentCoding$Major=='CS',]
table(isbyu$Hired)
```

- I saw what happens in this section and the result was like this. So, I added this rule.

```
hypothesis <- train[train$College=='BestCollege' & train$Impression=='Nerdy' & train$Major=='IT',]
table(hypothesis$Hired)
```

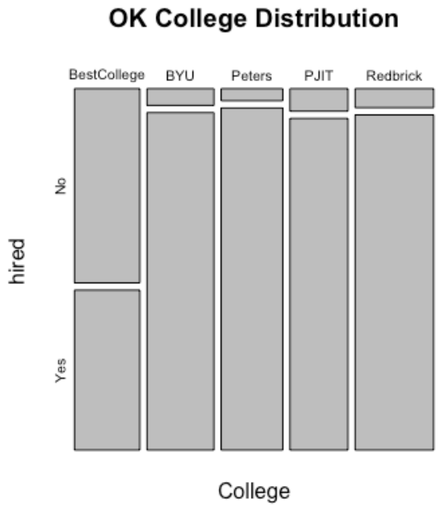
```
 No Yes
 25   0
```

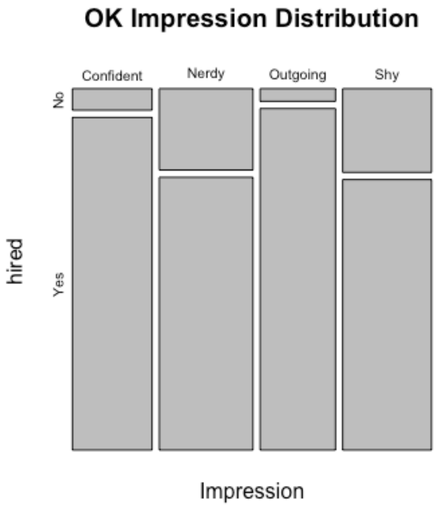
	Coding	Impression	Major	College	Hired
1462	Excellent	Outgoing	CS	BestCollege	No
144	Excellent	Confident	CS	BYU	No
1692	Excellent	Confident	CS	BYU	No
728	Excellent	Shy	CS	BYU	No
1277	Excellent	Shy	CS	BYU	No
1190	Excellent	Nerdy	CS	Redbrick	No
1561	Excellent	Outgoing	CS	Redbrick	No
1090	Excellent	Nerdy	DataScience	BestCollege	No
1762	Excellent	Outgoing	DataScience	BestCollege	No
1593	Excellent	Confident	DataScience	BestCollege	No
1517	Excellent	Nerdy	DataScience	BYU	No
1003	Excellent	Confident	DataScience	Peters	No
714	Excellent	Nerdy	IT	BestCollege	No
1661	Excellent	Nerdy	IT	BestCollege	No
1096	Excellent	Nerdy	IT	BestCollege	No
320	Excellent	Nerdy	IT	BestCollege	No
1943	Excellent	Shy	IT	BestCollege	No
1369	Excellent	Nerdy	IT	BestCollege	No
408	Excellent	Shy	IT	BestCollege	No
1302	Excellent	Nerdy	IT	BestCollege	No
299	Excellent	Nerdy	IT	BestCollege	No
808	Excellent	Nerdy	IT	BestCollege	No
134	Excellent	Nerdy	IT	BestCollege	No
61	Excellent	Nerdy	IT	BestCollege	No
1800	Excellent	Outgoing	IT	BYU	No
1599	Excellent	Confident	IT	Peters	No
1427	Excellent	Shy	IT	PJIT	No
7	Excellent	Outgoing	IT	Redbrick	No
725	Excellent	Outgoing	Stats	BYU	No

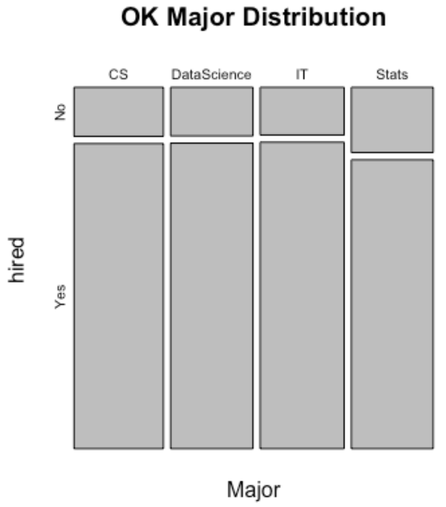
BestCollege

From OK Coding Skills

- From the mosaic plots of OK coding skills, we could find some tendency. Generally, you can get a job with OK coding skills.
- From impression plot, Nerdy and Shy personalities are more likely to get ‘No’.
- From College plot, BestCollege is more likely to get No than the others.



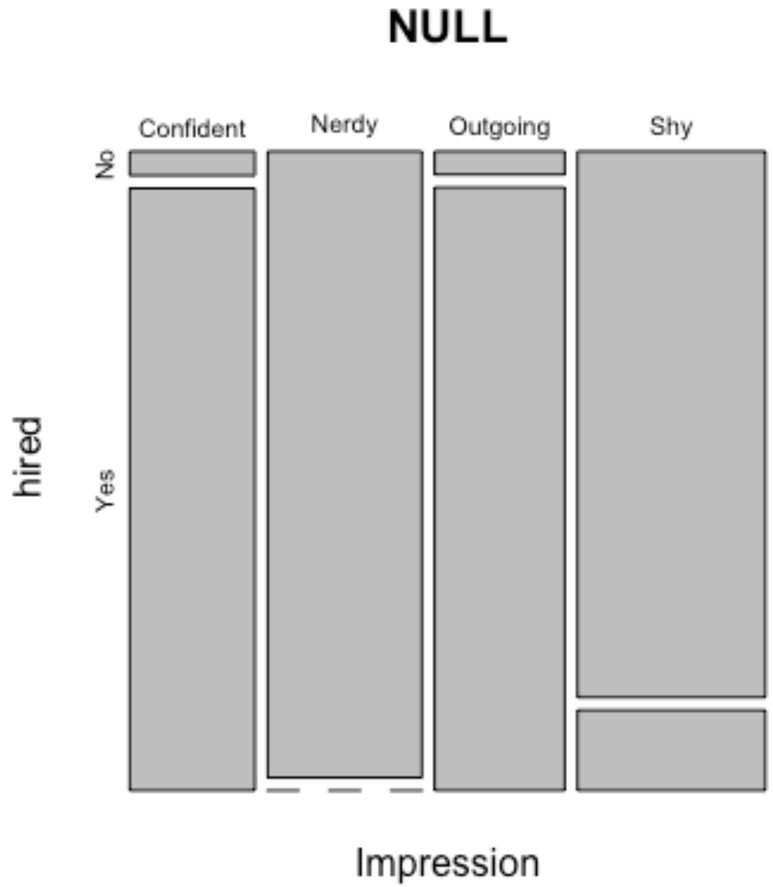




OK Coding Skills

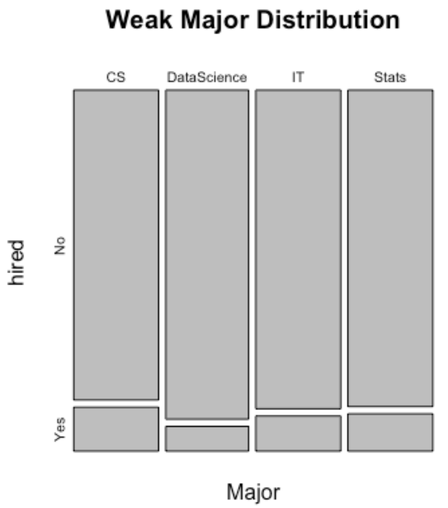
```
temp7 <- train[train$Coding=='OK' & train$College=='BestCollege',]
mosaicplot(temp7$Impression~temp7$Hired,xlab='Impression',ylab='hired')
```

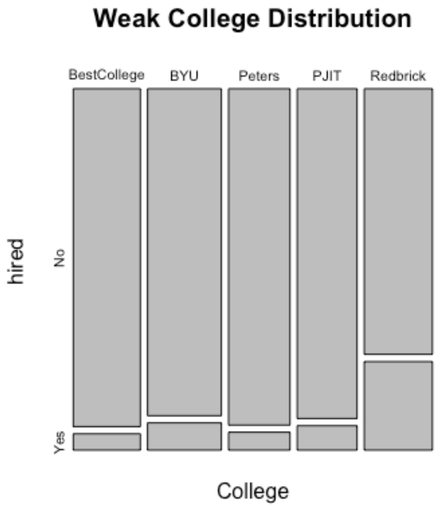
- With upper code, I made a plot.
- From OK coding skills in best college, most of students with nerdy and shy personalities couldn't get a job.

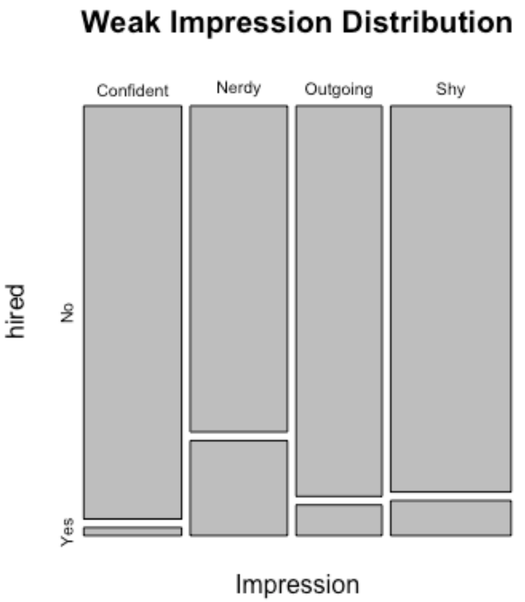


From Weak coding skills

- From the plots, we could observe most of students with weak coding skill could not get a job.
- People from Redbrick college & people with Nerdy personality are more likely to get a job.







Weak Coding Skills

- From the code below, I could find nerdy students from Redbridge could get a job even though they have weak coding skills.

```
> summary(temp)
       Coding       Impression         Major           College   Hired
 Excellent: 0   Confident: 0   CS         :11   BestCollege: 0   No : 0
 OK       : 0   Nerdy    :27   DataScience: 9   BYU        : 0   Yes:33
 Weak     :33   Outgoing : 2   IT         :10   Peters     : 0
                Shy      : 4   Stats      : 3   PJIT       : 0
                                                Redbrick   :33
> temp <- weakCoding[weakCoding$Impression=='Nerdy' & weakCoding$Hired=='Yes',]
> summary(temp)
       Coding       Impression         Major           College   Hired
 Excellent: 0   Confident: 0   CS         :12   BestCollege: 2   No : 0
 OK       : 0   Nerdy    :35   DataScience: 8   BYU        : 3   Yes:35
 Weak     :35   Outgoing : 0   IT         :10   Peters     : 1
                Shy      : 0   Stats      : 5   PJIT       : 2
                                                Redbrick   :27
> temp <- train[train$College=='Redbrick' & train$Impression=='Nerdy' & train$Cod
> table(temp$Hired)

 No Yes
  3  27
```

```
temp <- weakCoding[weakCoding$College=='Redbrick'& weakCoding$Hired=='Yes',]
summary(temp)
temp <- weakCoding[weakCoding$Impression=='Nerdy' & weakCoding$Hired=='Yes',]
summary(temp)

temp <- train[train$College=='Redbrick' & train$Impression=='Nerdy' & train$Coding=='Weak',]
table(temp$Hired)
```

Decision

```
decision <- rep('Yes',nrow(valid))
decision[valid$Coding=='Weak'] <- 'No'
decision[valid$Coding=='Weak' & valid$Impression=='Nerdy' & valid$College=='Redbrick'] <- 'Yes'
decision[valid$Major == 'IT' & valid$College=='BestCollege' & valid$Impression=='Nerdy'] <- 'No'
decision[valid$Coding=='OK' & valid$College=='BestCollege' & valid$Impression=='Nerdy'] <- 'No'
decision[valid$Coding=='OK' & valid$College=='BestCollege' & valid$Impression=='Shy'] <- 'No'
error <- mean(valid$Hired != decision)
error
```

- From my analysis, I conducted rules.
- 1. Fill all rows with ‘Yes’.
- 2. If the students have ‘Weak’ coding skills, set them to ‘No’.
- 3. But If they weak coding students come from ‘Redbrick’ and have ‘Nerdy’ personalities, set them to ‘Yes’.
- 4. If coding skill is ‘OK’ and come from ‘BestCollege’ and their Impression are ‘Nerdy or Shy’, set them to ‘No’.
- 5. Students from ‘BestCollege’ and ‘IT’ major and have ‘Nerdy’ impressions are ‘No’.

Result

- I conducted 10 tests with this rules.
- My lowest error was 0.02, highest error was 0.06. So, I think my attempt is fair.
- Image is the result from Kaggle.