Math 5365

Data Mining 1

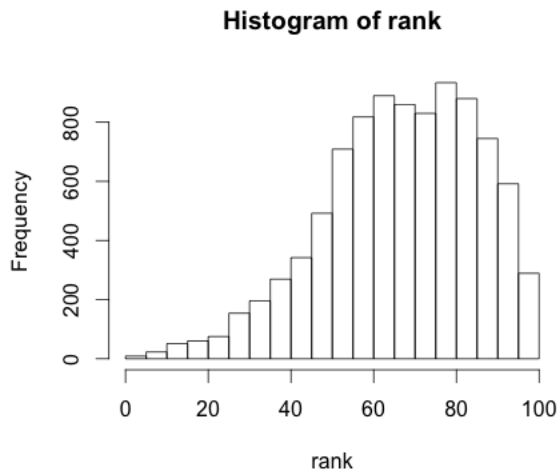
Homework 1

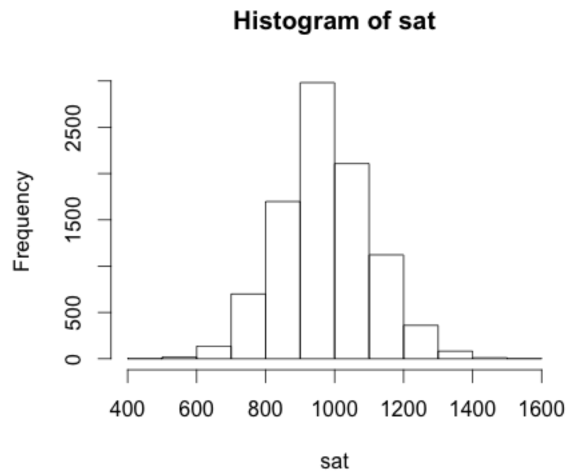
Mary Barker

The purpose of this project is to learn R through replicating a previous project that examines the interrelations between freshmen student admissions requirements and retention rates for 9,218 first time college freshmen admitted between 2004 and 2011. The variables considered are retention, which is taken as the number of students retained until the 2nd fall, rank, which is the student percentile rank, and sat, the students' SAT scores.

The data file is in the format of a comma separated table of values. The first step is to read in all of the values and store the required values in vectors titled 'retention', 'sat' and 'rank.'

One of the advantages of using R as a programming language is that it has a wide range of supporting packages that are easy to implement. An example of this is the `hist()` command. Calling `hist()` with a vector produces a histogram plot of its entries that is easy to export and use. Two histograms of the percentile rank and SAT scores of the students are shown below.





In addition, calling `summary()` and `mean()` produces information about the value distribution in the vectors. `mean()` gives the mean value for each vector. The values computed in this example for sat and rank were 978.21 and 67.04133 respectively. The summary for each of the vectors is shown below.

```
> summary(sat)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  460.0   890.0   980.0   978.2  1070.0  1530.0 
```

Figure 1: Summary of SAT scores

```
> summary(rank)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
   3.00   55.00   68.00   67.04   82.00  100.00 
```

Figure 2: Summary of student percentile rank

The values of retention are stored as 1 if the student stays until the second semester and 0 if not. Using the `mean()` function on the vector retention gives the retention rate. In this case the retention rate was computed as 66.847%.

The retention rate is assumed correlated to the admissions requirements for SAT scores and student rank. The motivation for this study is to find admission requirements that maximize the retention rate while maintaining high enrollment. In order to improve admissions and retention statistics, the interrelations between these quantities should be considered. The relation between retention, rank and sat can be studied using yet more built in R functionality. A basic plot of the two is created using the command `plot(rank, sat)`.

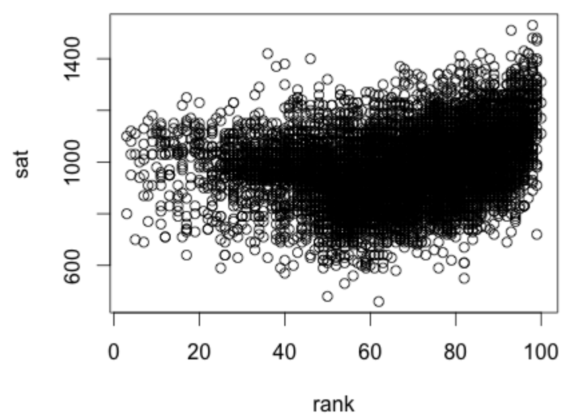


Figure 3: Plot of rank and sat

A plot of rank with respect to retention can be created using the plot command. However, since retention has values 0 and 1, R treats the vector as a quantitative variable. Plotting it as a factor can be done by running `plot(as.factor(retention), rank)`.

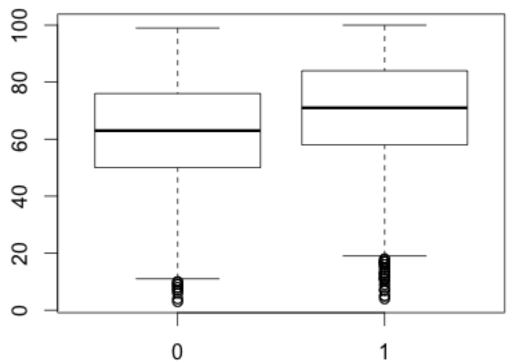


Figure 4: Plot of retention and rank

The `lm(v1 ~ v2)` function creates a linear regression model for predicting `v1` using `v2`. With this and the `summary()` command which is overloaded to accept a model, the statistical significance of rank and sat in predicting retention can be better examined.

The summary for `summary`(lm(sat∼rank)) is shown below.

```
lm(formula = sat ~ rank)

Residuals:
    Min      1Q  Median      3Q     Max
-508.03  -87.93    0.15   82.68  504.50

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 842.78097    4.94282  170.51   <2e-16 ***
rank          2.02008    0.07104   28.43   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 126.9 on 9216 degrees of freedom
Multiple R-squared:  0.08066,	Adjusted R-squared:  0.08056
F-statistic: 808.5 on 1 and 9216 DF,  p-value: < 2.2e-16
```

The command `glm()` creates a logistic regression model. The summaries for the models for retention and sat and for retention and rank are shown below.

```
glm(formula = retention ~ sat, family = "binomial")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6729  -1.4494   0.8646   0.9100   1.0897

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.2290226  0.1655133  -1.384    0.166
sat          0.0009538  0.0001686   5.656 1.55e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 11712  on 9217  degrees of freedom
Residual deviance: 11679  on 9216  degrees of freedom
AIC: 11683

Number of Fisher Scoring iterations: 4
```

Figure 5: linear regression model for predicting retention with SAT score

```
glm(formula = retention ~ rank, family = "binomial")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7981  -1.3367   0.7645   0.9106   1.4398

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.682445   0.081821  -8.341   <2e-16 ***
rank         0.020987   0.001211  17.332   <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: 11712  on 9217  degrees of freedom
Residual deviance: 11402  on 9216  degrees of freedom
AIC: 11406

Number of Fisher Scoring iterations: 4
```

Figure 6: linear regression model for predicting retention with student rank

Finally, a logistic regression model that predicts retention based on the combination of sat and rank is shown.

```
glm(formula = retention ~ rank + sat, family = "binomial")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8199  -1.3354   0.7656   0.9113   1.4446

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.8451714  0.1733283  -4.876 1.08e-06 ***
rank         0.0206538  0.0012507  16.514  < 2e-16 ***
sat          0.0001897  0.0001780   1.066    0.287
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 11712  on 9217  degrees of freedom
Residual deviance: 11401  on 9215  degrees of freedom
AIC: 11407

Number of Fisher Scoring iterations: 4
```

Figure 7: linear regression model for predicting retention with student rank and SAT scores combined

The admissions requirements are shown in the table below. Those with rank over 89 are automatically admitted, so there is no SAT score reqirement for those with rank of 90 or above.

Percentile Rank	1-24	25-49	50 - 89
SAT Requirement	1030	950	400

A function was created to calculate retention rate based upon the admissions requirements. For the original set of requirements, the retention rate for these requirements is 67.4%, which is slightly higher than the retention rate with the admitted students. In addition, the enrollment loss (number of potential students not accepted based upon their rank and score) is also computed. For this set of requirements, 547 of the 9,218 students fail to meet the requirements. Therefore the total enrollment loss is 547.

In order to improve retention, an alternative set of thresholds were proposed for admissions. Note that those with rank of 90 or above are always admitted, so the only changes are for those in the lower 90% of the class.

Percentile Rank	1-32	33-49	50 - 89
SAT Requirement	1610	950	400

These changes resulted in a better retention rate than the previous case, and a far worse enrollment loss. The retention rate computed for this example is 68.1%, while the enrollment loss is 825. In order to see whether there is a set of admissions thresholds that maximizes retention rate without exceeding the enrollment loss of 825, a brute force method was implemented that computed the enrollment loss and retention rate for all of the possible combinations of rank and SAT requirements.

In order to increase efficiency for this case, I used values 1, 5, 10, ..., 85 as possible values for rank cutoff values and values 400, 410, ..., 1610 for SAT scores. The code for this project is attached. The brute force search did not result in a better retention rate than 68.1% without simultaneously having an enrollment loss less than 825.

```r
# This line replaces using the 'import dataset' option
# because I prefer writing a script over writing a
# series of commands in console however saveable

# for this project, the dataset in FTIC.csv is an array 9128 x 6
# of stats about freshmen admitted to TSU.
# The 6 columns are TERM, QUARTER, PERCENTILE, SAT, X1st_Spring and X2nd_Fall
FTIC <- read.table("~/Dropbox/data_mining/hw01/FTIC.csv", header=TRUE, sep=",")

# pick out the 'X2nd_Fall' column from the dataset.
# this gives the retention rate for this set of students
retention=FTIC$X2nd_Fall
# The two factors which we are going to consider as
# predictors of retention are precentile rank and sat scores
rank=FTIC$PERCENTILE
sat=FTIC$SAT

# create histograms for rank and sat and compute
# values related to distribution for later use.
hist(rank)
summary(rank)
mean(rank)
hist(sat)
summary(sat)
mean(sat)

#make retention boolean
```

```
retention=(retention=="Y")*1

#linear regression of sat and rank
model=lm(sat˜rank)
summary(model)

plot(as.factor(retention), rank)

rankmodel=glm(retention˜rank, family=’binomial’)
summary(rankmodel)

satmodel=glm(retention˜sat, family=’binomial’)
summary(satmodel)

model=glm(retention˜rank+sat, family=’binomial’)
summary(model)

index = ( (rank < 25)  & (sat >= 1030) )|
        ( (rank >= 25) & (rank < 50) & (sat >= 950) ) |
        ( (rank >= 50) & (rank < 90) & (sat >= 400) ) |
        (rank >= 90)
table(index)
retention[index]
mean(retention[index])


# fun example of a function with a list
```

```
mysumanddiff=function(x, y){
 L = list(sum = x + y, diff = x - y)
 return(L)
}

mylist=mysumanddiff(8, 3)

mylist$sum
mylist$diff

# this function evaluates retention based on threshold requirements
# for sat and rank that are passed in. Return value is a list that
# contains enrollment loss and retention rate based on sat and rank
evalthresholds=function(rank, sat, retention, rankthresholds,
                        satthresholds){
#index[i] = 1 if student [i] fulfills admission requirements
  index=( (rank<rankthresholds[1]) &
          (sat>=satthresholds[1]) ) |
        ( (rank>=rankthresholds[1]) &
          (rank<rankthresholds[2]) &
          (sat>=satthresholds[2]) ) |
        ( (rank>=rankthresholds[2]) &
          (rank<rankthresholds[3]) &
          (sat>=satthresholds[3]) ) |
        ( (rank>=rankthresholds[3]) )

  x1 = sum( (index==FALSE)*1)
```

```
  x2 = mean(retention[index])
  L = list(enrollmentloss = x1, newretention = x2)
  return(L)
}
# test this function with solution in homework to see if it works.
rankthresholds=c(25,50)
satthresholds=c(1030,950,400)
mylist=evalthresholds(rank,sat,retention,rankthresholds,satthresholds)
mylist$enrollmentloss
mylist$newretention

# now compute enrollmentloss and retention
# with a different set of threshold criteria
rankthresholds=c(33,50, 90)
satthresholds=c(1610,950,400)

mylist=evalthresholds(rank,sat,retention,rankthresholds,satthresholds)
mylist$enrollmentloss
mylist$newretention

finalrankthreshold=rankthresholds
finalsatthreshold=satthresholds

# brute force method to compute better possible threshold requirements
# for admissions based on sat and rank
for(i in 10:78){
  for(j in 20:89){
```

```
    if(i < j){
      print(paste(i,j))
      rankthresholds=c(i,j,90)
      for(k in 95:160){
        print(k)
        kk = 10*k
        for(m in 50:95){
          mm = 10*m
          for(n in 40:50){
            nn = 10*n
            satthresholds=c(nn, mm, kk)
            newlist = evalthresholds(rank,sat,retention,rankthresholds,satthresholds)
            if( (newlist$enrollmentloss<=mylist$enrollmentloss) &
                (newlist$newretention>mylist$newretention)){
                mylist=newlist
                finalrankthreshold=rankthresholds
                finalsatthreshold=satthresholds
                print(paste("changed values", i,j,k,m,n))
            }
          }
        }
      }
    }
  }
}
print("==============================================")
print("=========Final Enrollment Statistics==========")
```

```r
print("==============================================")
print(paste("Enrollment loss:",mylist$enrollmentloss))
print(paste("Enrollment loss:",mylist$newretention))

print("The final rank thresholds are: ")
print(finalrankthreshold)
print(finalsatthreshold)

#expand.grid(1:5,10:13)
```