

Stats 506, F20, Problem Set 1

Zhihao Xu, xuzhihao@umich.edu

September 27, 2020

In this assignment, `ps1_q1_ohxdn.sh` is the bash file for Question 1 Part A, `ps1_q1_demo.sh` is the bash file for Part B, `xpt2csv.R` is the Rcode used to transfer the XPT file to csv file, and `ps1_q2.R` is the R code for Question 2.

Question 1

Part A

Here is the solution text for `ps1_q1_ohxdn.sh`. In the resulting datasets, there are totally 35909 observations and 62 variables.

```
#!/usr/bin/env bash

# Stats 506, Fall 2020 Homework 1 Question 1
#
# Author: Zhihao Xu
# Updated: September 22, 2020
# 79: -----

files=("OHXDEN_J.XPT" "OHXDEN_I.XPT" "OHXDEN_H.XPT" "OHXDEN_G.XPT")
urls=(
  "https://wwwn.cdc.gov/Nchs/Nhanes/2017-2018/OHXDEN_J.XPT"
  "https://wwwn.cdc.gov/Nchs/Nhanes/2015-2016/OHXDEN_I.XPT"
  "https://wwwn.cdc.gov/Nchs/Nhanes/2013-2014/OHXDEN_H.XPT"
  "https://wwwn.cdc.gov/Nchs/Nhanes/2011-2012/OHXDEN_G.XPT"
)
csv_files=("OHXDEN_J.csv" "OHXDEN_I.csv" "OHXDEN_H.csv" "OHXDEN_G.csv")

# download data for the problem set
for i in 0 1 2 3
do
  if [ ! -f "${files[i]}" ]; then
    wget ${urls[i]}
  fi
done

# transfer the XPT file to csv by R
for i in 0 1 2 3
do
  if [ ! -f "${csv_files[i]}" ]; then
    Rscript ./xpt2csv.R "${files[i]}"
  fi
done
```

```

file="nhanes_ohxden.csv"
# cut selected column
if [ ! -f "$file" ]; then
    head -n +1 OHXDEN_J.csv |
    cut -d "," -f2,4,6-65>> $file
    for i in 3 2 1 0
    do
        tail -n +2 "${csv_files[i]}" |
        cut -d "," -f2,4,6-65>> $file
    done
fi

# check duplicate lines
uniq_lines=$(< $file sort | uniq | wc -l)
n_lines=$(< $file wc -l)
if [ $uniq_lines == $n_lines ]; then
    echo "No duplicates in $file. Part (a) Finished."
else
    echo "Total lines in $file:" $n_lines, Unique lines: $uniq_lines.
fi

# 79: -----

```

Part B

Here is the solution text for `ps1_q1_demo.sh`. In the resulting datasets, there are totally 39156 observations and 10 variables.

```

#!/usr/bin/env bash

# Stats 506, Fall 2020 Homework 1 Question 1
#
# Author: Zhihao Xu
# Updated: September 22, 2020
# 79: -----

files=("DEMO_J.XPT" "DEMO_I.XPT" "DEMO_H.XPT" "DEMO_G.XPT")
urls=(
    "https://wwwn.cdc.gov/Nchs/Nhanes/2017-2018/DEMO_J.XPT"
    "https://wwwn.cdc.gov/Nchs/Nhanes/2015-2016/DEMO_I.XPT"
    "https://wwwn.cdc.gov/Nchs/Nhanes/2013-2014/DEMO_H.XPT"
    "https://wwwn.cdc.gov/Nchs/Nhanes/2011-2012/DEMO_G.XPT"
)
csv_files=("DEMO_J.csv" "DEMO_I.csv" "DEMO_H.csv" "DEMO_G.csv")

# download data for the problem set
for i in 0 1 2 3
do
    if [ ! -f "${files[i]}" ]; then
        wget ${urls[i]}
    fi
done

# transfer the XPT file to csv by R

```

```

for i in 0 1 2 3
do
    if [ ! -f "${csv_files[i]}" ]; then
        Rscript ./xpt2csv.R "${files[i]}"
    fi
done

file="nhanes_demo.csv"
# cut selected column
if [ ! -f "nhanes_demo.csv" ]; then
    head -n +1 DEMO_J.csv |
    cut -d "," -f2,6,9,18-19,4,44-45,43,42>> $file
    for i in 3 2 1 0
    do
        tail -n +2 "${csv_files[i]}" |
        cut -d "," -f2,6,9,18-19,4,44-45,43,42>> $file
    done
fi

# check duplicate lines
uniq_lines=$(< $file sort | uniq | wc -l)
n_lines=$(< $file wc -l)
if [ $uniq_lines == $n_lines ]; then
    echo "No duplicates in $file. Part (b) Finished."
else
    echo "Total lines in $file:" $n_lines, Unique lines: $uniq_lines.
fi

# 79: -----

```

Question 2

Part A and B

```
## Stats 506, Fall 2020 Homework 1 Question 2
##
## Author: Zhihao Xu, xuzhihao@umich.edu
## Updated: September 24, 2020 - Last modified date
# 79: -----

# libraries: -----
library(ggplot2)
# data: -----
data = read.csv("isolet_results.csv")

# Compute the TP, FP, TN, FN, sensitivity and specificity used in part (a)
# Input:
#   -tau: given threshold
#   -y,yhat: true and predict value
# Output:
#   -a vector of tp, fp, tn, fn, sensitivity and specificity
compute_tf1 = function(tau,y,yhat){
  tp = sum((yhat>=tau)&(y==1))
  fp = sum((yhat>=tau)&(y==0))
  tn = sum((yhat<tau)&(y==0))
  fn = sum((yhat<tau)&(y==1))
  se = tp/(tp+fn)
  sp = tn/(fp+tn)
  return(c(tp, fp, tn, fn, se, sp))
}

# Compute the area under the curve using trapezoidal rule.
# Input:
#   - x,y: values on x-axis and y-axis
# Output:
#   - the area under the curve
compute_area = function(x,y){
  delta = abs(x[2:length(x)] - x[1:(length(x)-1)])
  val = (y[2:length(y)] + y[1:(length(y)-1)])/2
  return(sum(delta*val))
}

# Compute the table of TP, FP, TN, FN, Sensitivity and Specifity,
# Compute the area under the ROC curve,
# Plot the ROC curve if needed.
# Input:
#   -y,yhat: true and predict value
#   -plot: indicating no plot or plot by base R graphics or plot by ggplot2.
# Output:
#   plot of ROC curve if base or ggplot2 is chosen for plot
#   A list with 2 elements:
#     - df: the target data frame with 7 columns
#     - area_roc: the area under the ROC curve
perf_roc = function(y, yhat, plot = c("none", "base", "ggplot2")){
```

```

tau = sort(unique(yhat))
df = matrix(rep(NA,7*length(tau)),ncol=7)
colnames(df) = c("Tau","TP","FP","TN","FN","Sensitivity","Specificity")
df[,1] = tau
df[,2:7] = t(sapply(tau, compute_tf1,y,yhat))
# add 0,1 to avoid just compute area from min to max,
# which can make this estimate more accurate
fpr = c(1,1- df[,7],0)
tpr = c(df[,6],df[,6],0)
area_roc = compute_area(fpr,tpr)
# Plot of ROC Curve
plot = match.arg(plot)
switch (plot,
        none = cat("No Plot"),
        base = plot(fpr, tpr, type="l",
                     xlab="False Positive Rate (FPR)",
                     ylab="True Positive Rate (TPR)",
                     ggplot2 = {
p = ggplot(data.frame(fpr,tpr),aes(x=fpr,y=tpr))+
  geom_path(size=0.6, alpha=0.7)+
  labs(x="False Positive Rate (FPR)",
       y="True Positive Rate (TPR)") +
  theme(plot.title = element_text(hjust = 0.5))
print(p)
}
)
return(list(df=df, area_roc=area_roc))
}

# Compute the TP, FP, TN, FN, recall and precision used in part (a)
# Input:
#   -tau: given threshold
#   -y,yhat: true and predict value
# Output:
#   -a vector of tp, fp, tn, fn, recall and precision

compute_tf2 = function(tau,y,yhat){
  tp = sum((yhat>=tau)&(y==1))
  fp = sum((yhat>=tau)&(y==0))
  tn = sum((yhat<tau)&(y==0))
  fn = sum((yhat<tau)&(y==1))
  re = tp/(tp+fn)
  pr = tp/(tp+fp)
  return(c(tp, fp, tn, fn, re, pr))
}

# Compute the table of TP, FP, TN, FN, Recall and Precision,
# Compute the area under the Precision-Recall Curve,
# Plot the Precision-Recall Curve if needed.
# Input:
#   -y,yhat: true and predict value
#   -plot: indicating no plot or plot by base R graphics or plot by ggplot2.
# Output:

```

```

# plot of Precision-Recall curve if base or ggplot2 is chosen for plot
# A list with 2 elements:
# - df: the target data frame with 7 columns
# - area_pr: the area under the Precision-Recall Curve
perf_pr = function(y, yhat, plot = c("none", "base", "ggplot2")){
  tau = sort(unique(yhat))
  df = matrix(rep(NA, 7*length(tau)), ncol=7)
  colnames(df) = c("Tau", "TP", "FP", "TN", "FN", "Recall", "Precision")
  df[,1] = tau
  df[,2:7] = t(sapply(tau, compute_tf2, y, yhat))
  # add 0,1 to avoid just compute area from min to max,
  # which can make this estimate more accurate
  re = c(1, df[,6], 0)
  pre = c(0, df[,7], df[dim(df)[1], 7])
  area_pr = compute_area(re, pre)
  # Plot of Precision-Recall Curve
  plot = match.arg(plot)
  switch (plot,
    none = cat("No Plot"),
    base = plot(re, pre, type="l",
               xlab="Recall",
               ylab="Precision"),
    ggplot2 = {
      p = ggplot(data.frame(re, pre), aes(x=re, y=pre)) +
        geom_path(size=0.6, alpha=0.7) +
        labs(x="Recall",
              y="Precision") +
        theme(plot.title = element_text(hjust = 0.5))
      print(p)
    }
  )
  return(list(df=df, area_pr=area_pr))
}
# 79: -----

```

Part C

The AUC-ROC is 0.9687874 and the AUC-PR is 0.8614572. Attached are both the base R and ggplot2 versions of plots showing the curves.

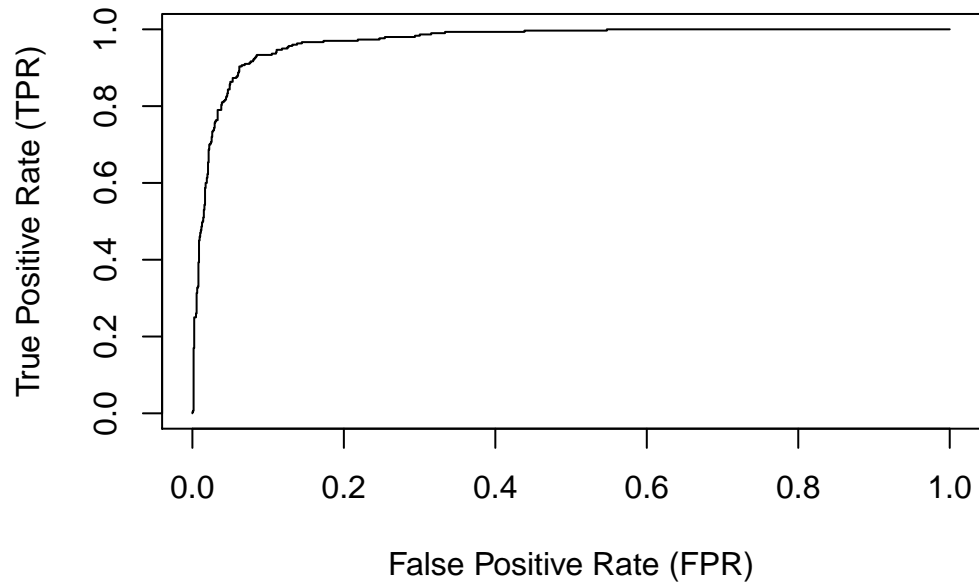


Figure 1: ROC Curve by base R Graphics

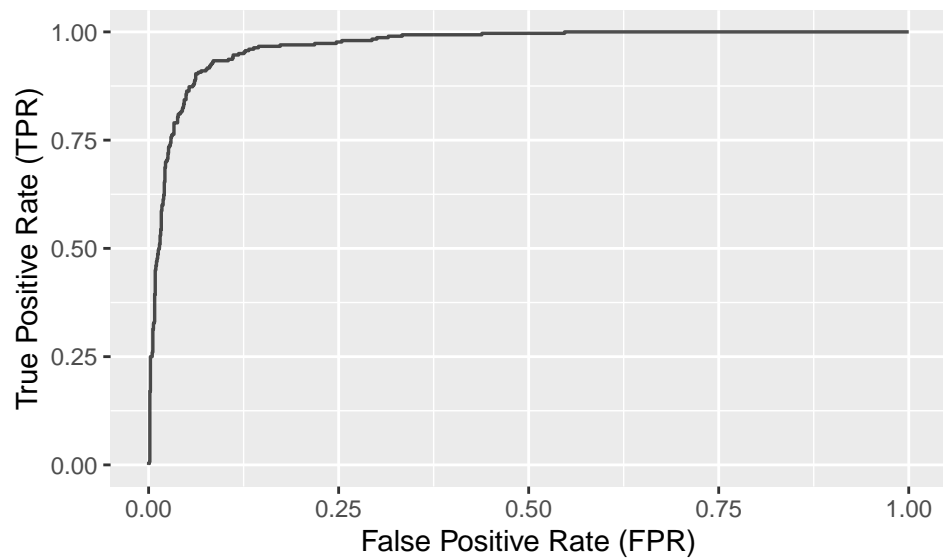


Figure 2: ROC Curve by ggplot2

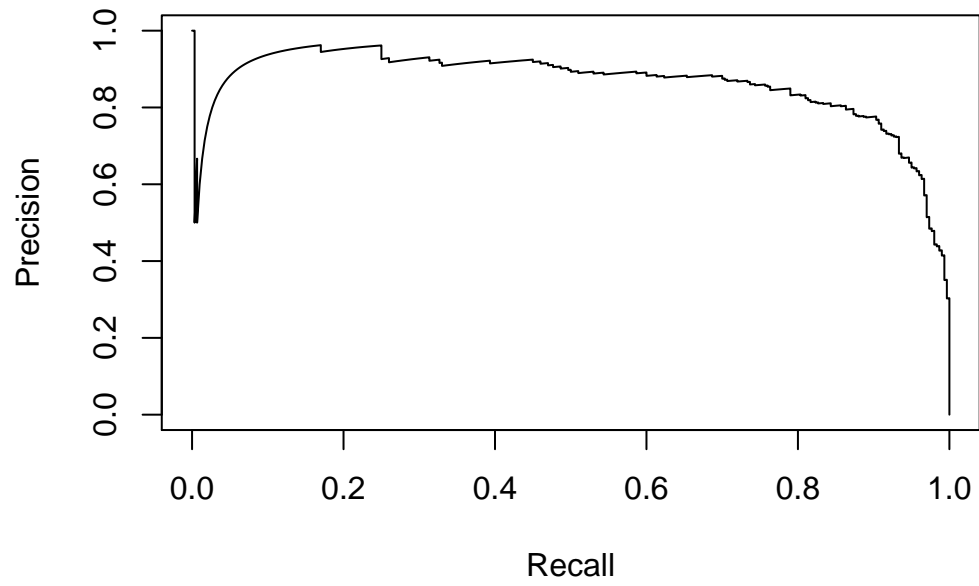


Figure 3: Prediction-Recall Curve by base R Graphics

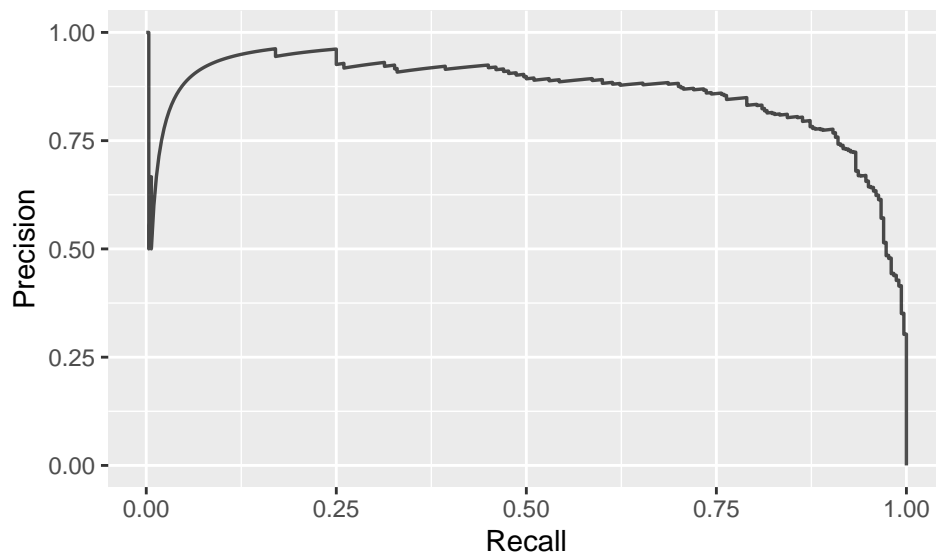


Figure 4: Prediction-Recall Curve by ggplot2