Mining Association Rules Relating to Demographics and Voting Preference

Ian Jeffries

5/16/2019

The following R Markdown outlines the steps used by Ian Jeffries to mine interesting association rules based on demographics from the 2016 election results in the USA. This is an unsupervised machine learning problem, and the steps used to clean the data are recreated from the classification problem done earlier in this project.

Install necessary packages

The following code will install the packages used in the project:

```
#list of packages used
packages <- c("dplyr", "tidyr", "ggplot2", "class", "rpart", "rpart.plot", "neuralnet",
              "arules", "plyr", "mltools", "arulesViz", "plotly", "RCurl")

#check to see if package is already installed, if not, install
for(p in packages){
  if(!require(p, character.only = TRUE)) {
    install.packages(p)
    library(p, character.only = TRUE)
  }
}
```

Import the Demographics Dataset

Import the data from my github page

```
#import our county features dataset
path <- "https://raw.githubusercontent.com/ianjeffries/election-predictions/master/data/
county_facts.csv"

demographics <- read.csv(path, header=TRUE)

#find the state summary info and remove
#(want only counties, this dataset includes state summaries)
demographics <- demographics[-which(demographics$state_abbreviation == ""), ]
```

Add in Dictionary information

Add in the dictionary file to replace columm headers with readable values.

```
#pull in dictionary to get true names of the variables
path <- "https://raw.githubusercontent.com/ianjeffries/election-predictions/master/data/
county_facts_dictionary.csv"
```

```
var_names <- read.csv(path, header = TRUE)
```

Pull in 2016 Election Results and Clean the Data

As of now, the US primarily operates in a two-party system. Because of this, only the democratic and republican candidates are of interest.

```
#add in the election results
path <- "https://raw.githubusercontent.com/ianjeffries/
election-predictions/master/data/pres16results.csv"

election_results <- read.csv(path, header = TRUE)

#drop columns we don't need
election_results <- election_results[-c(2, 5, 8)]

#filter by the republican and democratic candidates & remove null values
ER_clean <- election_results %>%
              filter(cand %in% c("Donald Trump", "Hillary Clinton") & county != "<NA>")

#change to wide dataset to merge the election results with the demographics dataframe
ER_clean <- spread(ER_clean, key = cand, value = votes)

#add in % won by
ER_clean$Percent_Rep <- round(ER_clean$`Donald Trump` / ER_clean$total_votes, 2)
ER_clean$Percent_Dem <- round(ER_clean$`Hillary Clinton` / ER_clean$total_votes, 2)
```

Join the Demographic and Election Results Datasets

Now that cleanup is complete, the two datasets can be joined.

```
#join our election results with the demographics table
final_dataset <- left_join(demographics,
                           select(ER_clean, Percent_Dem, Percent_Rep, st, county),
                           by = c("area_name" = "county", "state_abbreviation" = "st"))

#find number of counties with N/A election results
print(paste0("Number of counties with null election results: ",
             nrow(final_dataset[which(complete.cases(final_dataset) == FALSE), ])))
```

```
## [1] "Number of counties with null election results: 33"
```

```
#looks like these aren't counties (the majority)
#and the N is small enough I feel comfortable removing them
final_dataset <- na.omit(final_dataset)
```

Final Cleanup

The final cleanup steps are to remove irrelevant values and reassign column names for analysis. The data will also be normalized, with 0 or 1 values assigned to the target variable.

```r
#change any column in dataset based on % of the population to true percentage
#(basically already normalized between 0 and 1)
final_dataset[,c(8:24, 28:29, 35, 41:46)] <- final_dataset[,c(8:24, 28:29, 35, 41:46)]/100

#drop clearly irrelevant values
final_dataset <- final_dataset[ , -c(5:8, 38)]

#assign headers to something more understandable, based on the dictionary dataframe
names(final_dataset) <- var_names$description[match(names(final_dataset),
                                                    var_names$column_name)]

#assign labels that weren't in the dictionary
names(final_dataset[ , c(1, 2, 3, 50, 51)]) <- c("fips", "area_name", "state_abbreviation",
                                                 "Percent_Dem", "Percent_Rep")

#normalize any data not described as a %
for (i in c(4, 21, 22, 23, 26, 27, 28, 29, 30, 32, 33, 34, 35, 42, 43, 44,
            45, 46, 47, 48, 49)) {
  final_dataset[,i] <- ((final_dataset[,i] - min(final_dataset[,i])) /
                          (max(final_dataset[,i]) - min(final_dataset[,i])) *
                          (1 - 0) + 0)
}

#add in 1 or 0 if they voted democratic or republican
final_dataset$Winner <- 0

for (i in 1:nrow(final_dataset)) {
    if (final_dataset[i, "percent_dem"] < .50) {
      final_dataset[i, "Winner"] <- 0
      } else {
        final_dataset[i, "Winner"] <- 1
      }}

#set winner column to factor for classification
final_dataset$Winner <- as.factor(final_dataset$Winner)

#remove columns that don't seem to have a correlation to voting preference
final_dataset <- final_dataset[ , -c(1,5,7,10,12,13,16,28,30,34,35,37,39,42,43,44,
                                     45,47,48)]

#create dataset for association rules
association_data <- final_dataset[ , c(1:30,33)]

#change winner to party name for readability
association_data$Winner <- factor(association_data$Winner, labels = c("Republican",
                                                                      "Democrat"))
```

Break Data into 5 Bins to Enable Market Basket Analysis

The data needs to be separated into 5 equal bins to create a format for market basket analysis.

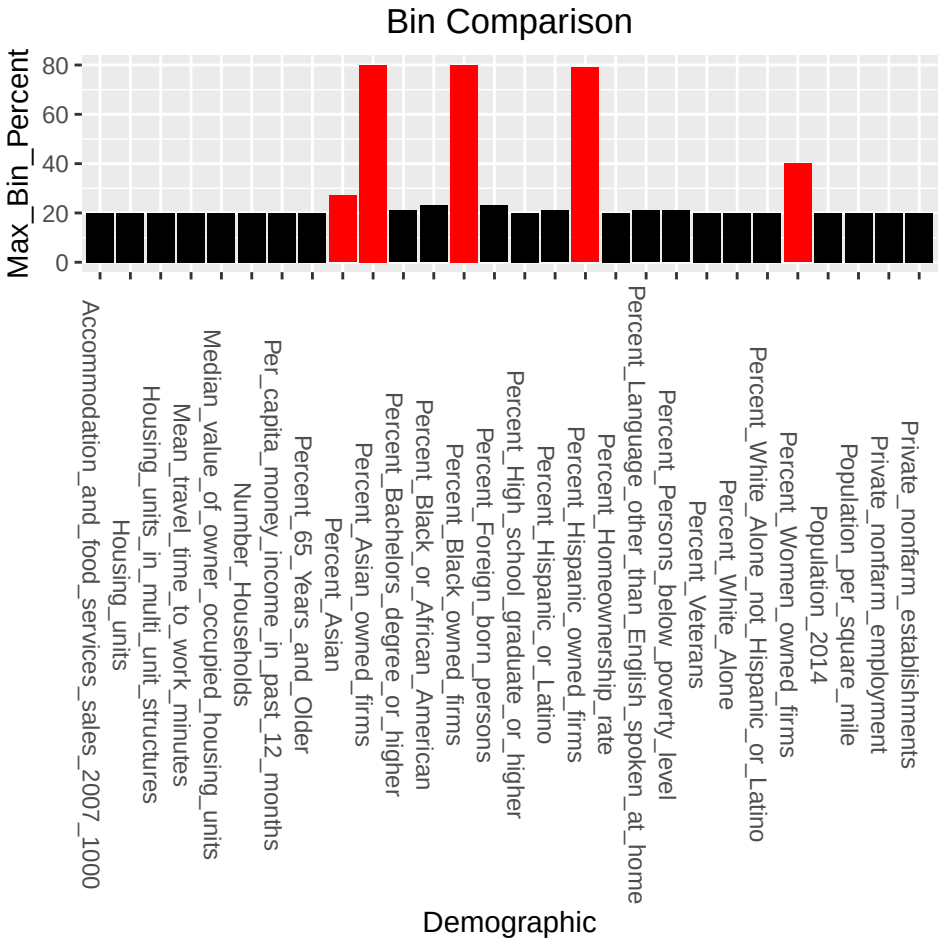
```r
#create new variables to house our binning data
association_data_bin <- association_data

#change to association data by binning 5 equal bins
#Resulting bins have an equal number of observations in each group
for (i in 3:30) {
  association_data_bin[ ,i] <- bin_data(association_data_bin[ ,i], bins=5,
                                        binType = "quantile")
}

#change values to more readable format
#(this can obviously be done a better way but I was short on time)
for (i in 3:30) {
  association_data_bin[,i] <- revalue(association_data_bin[,i], c(
      "[0, 0.000894409486015041)"="0_to_0.0008",
      "[0.000894409486015041, 0.0018724239788016 1)"="0.0009_to_0.0018",
      "[0.00187242397880161, 0.00364677171296063)"="0.0019_to_0.0035",
      "[0.00364677171296063, 0.00919441564419892)"="0.0036_to_0.0091",
      "[0.00919441564419892, 1]"="0.0092_to_1", "[0.041, 0.1418)"="0.041_to_0.1417",
      "[0.1418, 0.163)"="0.1418_to_0.1629", "[0.163, 0.183)"="0.163_to_0.1829",
       "[0.183, 0.208)"="0.183_to_0.2079", "[0.208, 0.529]"="0.208_to_0.529",
      "[0.108, 0.759)"="0.108_to_0.7589", "[0.759, 0.8916)"="0.759_to_0.8915",
      "[0.8916, 0.942)"="0.8916_to_0.9419", "[0.942, 0.9652)"="0.942_to_0.9651",
      "[0.9652, 0.993]"="0.9652_to_0.993", "[0, 0.006)"="0_to_0.0059",
      "[0.006, 0.014)"="0.006_to_0.0139", "[0.014, 0.043)"="0.014_to_0.0429",
      "[0.043, 0.152)"="0.043_to_0.1519", "[0.152, 0.851]"="0.152_to_0.851",
      "[0, 0.004)"="0_to_0.0039", "[0.004, 0.005)"="0.004_to_0.0049",
      "[0.005, 0.008)"="0.005_to_0.0079", "[0.008, 0.015)"="0.008_to_0.0149",
      "[0.015, 0.424]"="0.015_to_0.424", "[0.002, 0.018)"="0.002_to_0.0179",
      "[0.018, 0.029)"="0.018_to_0.0289", "[0.029, 0.052)"="0.029_to_0.0519",
      "[0.052, 0.1182)"="0.052_to_0.1181", "[0.1182, 0.958]"="0.1182_to_0.958",
      "[0.031, 0.6148)"="0.031_to_0.6147", "[0.6148, 0.783)"="0.6148_to_0.7829",
      "[0.783, 0.888)"="0.783_to_0.8879", "[0.888, 0.942)"="0.888_to_0.9419",
      "[0.942, 0.986]"="0.942_to_0.986", "[0, 0.01)"="0_to_0.0099",
      "[0.01, 0.019)"="0.01_to_0.0189", "[0.019, 0.034)"="0.019_to_0.0339",
      "[0.034, 0.066)"="0.034_to_0.0659", "[0.066, 0.513]"="0.066_to_0.513",
      "[0, 0.025)"="0_to_0.0249", "[0.025, 0.04)"="0.025_to_0.0399",
      "[0.04, 0.0634000000000003)"="0.04_to_0.0633",
      "[0.0634000000000003, 0.125)"="0.0634_to_0.1249",
      "[0.125, 0.956]"="0.125_to_0.956", "[0.45, 0.786)"="0.45_to_0.7859",
      "[0.786, 0.84)"="0.786_to_0.8399", "[0.84, 0.876)"="0.84_to_0.8759",
      "[0.876, 0.905)"="0.876_to_0.9049", "[0.905, 0.99]"="0.905_to_0.99",
      "[0.032, 0.129)"="0.032_to_0.1289", "[0.129, 0.16)"="0.129_to_0.1599",
      "[0.16, 0.194)"="0.16_to_0.1939", "[0.194, 0.254)"="0.194_to_0.2539",
      "[0.254, 0.744]"="0.254_to_0.744", "[0, 0.00221686165721867)"="0_to_0.0021",
      "[0.00221686165721867, 0.00447955614521771)"="0.0022_to_0.0044",
      "[0.00447955614521771, 0.00892835604872754)"="0.0045_to_0.0088",
      "[0.00892835604872754, 0.0221197684235918)"="0.0089_to_0.022",
      "[0.0221197684235918, 1]"="0.0221_to_1", "[0, 0.286111111111111)"="0_to_0.286",
      "[0.286111111111111, 0.369444444444444)"="0.2861_to_0.3693",
      "[0.369444444444444, 0.438888888888889)"="0.3694_to_0.4388",
      "[0.438888888888889, 0.536111111111111)"="0.4389_to_0.536",
      "[0.536111111111111, 1]"="0.5361_to_1", "[0, 0.00131027840616391)"="0_to_0.0012",
```

```
      "[0.00131027840616391, 0.00260217903060647)"="0.0013_to_0.0025",
      "[0.00260217903060647, 0.00484742708184373)"="0.0026_to_0.0047",
      "[0.00484742708184373, 0.0116445070820505)"="0.0048_to_0.0115",
      "[0.0116445070820505, 1]"="0.0116_to_1", "[0.194, 0.668)"="0.194_to_0.6679",
      "[0.668, 0.718)"="0.668_to_0.7179", "[0.718, 0.752)"="0.718_to_0.7519",
      "[0.752, 0.784)"="0.752_to_0.7839", "[0.784, 0.938]"="0.784_to_0.938",
      "[0, 0.055)"="0_to_0.0549", "[0.055, 0.082)"="0.055_to_0.0819",
      "[0.082, 0.116)"="0.082_to_0.1159", "[0.116, 0.178)"="0.116_to_0.1779",
      "[0.178, 0.985]"="0.178_to_0.985", "[0, 0.0496032189560747)"="0_to_0.0495",
      "[0.0496032189560747, 0.0691852017436012)"="0.0496_to_0.0691",
      "[0.0691852017436012, 0.097619313736448)"="0.0692_to_0.0975",
      "[0.097619313736448, 0.144517715435341)"="0.0976_to_0.1444",
      "[0.144517715435341, 1]"="0.1445_to_1", "[0, 0.00109969507948055)"="0_to_0.001",
      "[0.00109969507948055, 0.00226384137941709)"="0.0011_to_0.0022",
      "[0.00226384137941709, 0.00440775767331714)"="0.0023_to_0.0043",
      "[0.00440775767331714, 0.0107369789651276)"="0.0044_to_0.0106",
      "[0.0107369789651276, 1]"="0.0107_to_1", "[0, 0.192791783380019)"="0_to_0.1927",
      "[0.192791783380019, 0.238834733893557)"="0.1928_to_0.2387",
      "[0.238834733893557, 0.282603174603175)"="0.2388_to_0.2825",
      "[0.282603174603175, 0.337456582633053)"="0.2826_to_0.3374",
      "[0.337456582633053, 1]"="0.3375_to_1", "[0.009, 0.112)"="0.009_to_0.1119",
      "[0.112, 0.144)"="0.112_to_0.1439", "[0.144, 0.176)"="0.144_to_0.1759",
      "[0.176, 0.216)"="0.176_to_0.2159", "[0.216, 0.48]"="0.216_to_0.48",
      "[0, 0.000742416882875839)"="0_to_0.0006",
      "[0.000742416882875839, 0.0015243240254791 2)"="0.0007_to_0.0014",
      "[0.0015243240254791 2, 0.00306839318082195)"="0.0015_to_0.003",
      "[0.00306839318082195, 0.00793359317924235)"="0.0031_to_0.0078",
      "[0.00793359317924235, 1]"="0.0079_to_1", "[0, 0.000441914390403152)"="0_to_0.0003",
      "[0.000441914390403152, 0.00113226088212871)"="0.0004_to_0.001",
      "[0.00113226088212871, 0.00257237756100206)"="0.0011_to_0.0025",
      "[0.00257237756100206, 0.00759333770370314)"="0.0026_to_0.0075",
      "[0.00759333770370314, 1]"="0.0076_to_1", "[0, 0.016)"="0_to_0.0159",
      "[0.016, 0.667]"="0.016_to_0.667", "[0, 0.011)"="0_to_0.0109",
      "[0.011, 0.566]"="0.011_to_0.566", "[0, 0.012)"="0_to_0.0119",
      "[0.012, 0.78]"="0.012_to_0.78", "[0, 0.203)"="0_to_0.2029",
      "[0.203, 0.253)"="0.203_to_0.2529", "[0.253, 0.286)"="0.253_to_0.2859",
      "[0.286, 0.562]"="0.286_to_0.562", "[0, 0.000159321994078648)"="0_to_0.0001",
      "[0.000159321994078648, 0.000587838780495615)"="0.0002_to_0.0005",
      "[0.000587838780495615, 0.00160802412567208)"="0.0006_to_0.0015",
      "[0.00160802412567208, 0.00566270531352 77)"="0.0016_to_0.0056",
      "[0.00566270531352 77, 1]"="0.0057_to_1", "[0, 0.000182819567163878)"="0_to_0.0001",
      "[0.000182819567163878, 0.000471012302173394)"="0.0002_to_0.0004",
      "[0.000471012302173394, 0.000906900215064908)"="0.0005_to_0.0008",
      "[0.000906900215064908, 0.00230554188007612)"="0.0009_to_0.0022",
      "[0.00230554188007612, 1]"="0.0023_to_1"))
}

#drop county and state information
association_data_bin <- association_data_bin[,-c(1,2)]

#create matrix to store max bin distribution (see if bins have equal percentages)
binmax <- matrix(rep(0, 56), ncol = 2, dimnames = list(1:28, c("Demographic",
                                                               "Max_Bin_Percent")))
```

```r
binmax[ ,1] <- names(association_data_bin[,1:28])
binmax <- as.data.frame(binmax, stringsAsFactors = FALSE)

#calculate if bins are skewed in one category
#should be 20% of values in each bin, since there are 5 bins
for (i in 1:28) {
  binmax2 <- (count(association_data_bin[,i]))
  binmax[i,2] <- round(max(binmax2$freq) / nrow(association_data_bin), 2) * 100
}

#create numeric field
binmax$Max_Bin_Percent <- as.numeric(as.character(binmax$Max_Bin_Percent))
str(binmax)
```

```
## 'data.frame':    28 obs. of  2 variables:
##  $ Demographic    : chr  "Population_2014" "Percent_65_Years_and_Older" "Percent_White_Alone" "Percen
##  $ Max_Bin_Percent: num  20 20 20 23 27 21 20 23 21 20 ...
```

```r
#create an threshold amount (don't want anything over 25%)
binthresh <- binmax %>%
            filter(Max_Bin_Percent > 25)

#plot the results (highlight problem bins in red)
theme_update(plot.title = element_text(hjust = 0.5))
ggplot(as.data.frame(binmax), aes(x = Demographic, y = Max_Bin_Percent)) +
  geom_bar(stat = "identity", fill = "black") +
  ggtitle("Bin Comparison") +
  theme(axis.text.x = element_text(angle = -90)) +
  geom_bar(aes(Demographic, Max_Bin_Percent), data = binthresh,
           stat= "identity", fill = "red")
```



Mine data for interesting association rules

```
#remove bins that don't have 5 equal labels, will skew rules results
final_assoc_data <- subset(association_data_bin, select = -c(Percent_Black_owned_firms,
                                                             Percent_Asian_owned_firms,
                                                             Percent_Hispanic_owned_firms,
                                                             Percent_Women_owned_firms,
                                                             Percent_Asian))

#create rules using the apriori function for the republican party
#played with many confidence and support parameters to get perfect balance
rules1 <- apriori(final_assoc_data, parameter = list(minlen=2, maxlen=5,
                                                     conf = .923, supp = .189),
                  appearance= list(rhs=c("Winner=Republican"),default="lhs"))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##       0.923    0.1    1 none FALSE            TRUE       5   0.189      2
```

```
##  maxlen target   ext
##       5  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 587
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[117 item(s), 3110 transaction(s)] done [0.00s].
## sorting and recoding items ... [114 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 done [0.00s].
## writing ... [15 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
#inspect the Republican rules
summary(rules1)
```

```
## set of 15 rules
##
## rule length distribution (lhs + rhs):sizes
##  2
## 15
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##       2       2       2       2       2       2
##
## summary of quality measures:
##     support         confidence          lift           count
##  Min.   :0.1891   Min.   :0.9379   Min.   :1.084   Min.   :588.0
##  1st Qu.:0.1913   1st Qu.:0.9486   1st Qu.:1.097   1st Qu.:595.0
##  Median :0.1923   Median :0.9550   Median :1.104   Median :598.0
##  Mean   :0.1938   Mean   :0.9562   Mean   :1.106   Mean   :602.9
##  3rd Qu.:0.1942   3rd Qu.:0.9593   3rd Qu.:1.109   3rd Qu.:604.0
##  Max.   :0.2135   Max.   :0.9840   Max.   :1.138   Max.   :664.0
##
## mining info:
##              data ntransactions support confidence
##  final_assoc_data          3110   0.189      0.923
```

```
inspect(rules1)
```

```
##      lhs                                                          rhs                     suppor
## [1]  {Housing_units_in_multi_unit_structures=0.055_to_0.0819}        => {Winner=Republican} 0.189067
## [2]  {Percent_White_Alone_not_Hispanic_or_Latino=0.888_to_0.9419}    => {Winner=Republican} 0.192283
## [3]  {Per_capita_money_income_in_past_12_months=0.1928_to_0.2387}    => {Winner=Republican} 0.189710
## [4]  {Per_capita_money_income_in_past_12_months=0.2388_to_0.2825}    => {Winner=Republican} 0.190996
## [5]  {Percent_White_Alone=0.9652_to_0.993}                           => {Winner=Republican} 0.195498
## [6]  {Percent_White_Alone_not_Hispanic_or_Latino=0.942_to_0.986}     => {Winner=Republican} 0.197427
## [7]  {Median_value_of_owner_occupied_housing_units=0.0692_to_0.0975} => {Winner=Republican} 0.191961
## [8]  {Percent_Homeownership_rate=0.752_to_0.7839}                    => {Winner=Republican} 0.192283
```

```
## [9]  {Percent_65_Years_and_Older=0.183_to_0.2079}                  => {Winner=Republican} 0.192283
## [10] {Percent_65_Years_and_Older=0.208_to_0.529}                   => {Winner=Republican} 0.193247
## [11] {Percent_Homeownership_rate=0.784_to_0.938}                   => {Winner=Republican} 0.193569
## [12] {Percent_White_Alone=0.942_to_0.9651}                         => {Winner=Republican} 0.191639
## [13] {Percent_Homeownership_rate=0.718_to_0.7519}                  => {Winner=Republican} 0.189389
## [14] {Percent_Bachelors_degree_or_higher=0.16_to_0.1939}           => {Winner=Republican} 0.194855
## [15] {Percent_Foreign_born_persons=0.01_to_0.0189}                 => {Winner=Republican} 0.213504
```

```
#create rules using the apriori function for the democratic party
#need much lower support, since fewer counties voted democratic
rules2 <- apriori(final_assoc_data, parameter = list(minlen=2, maxlen=5,
                                                     conf = .817, supp = .031),
                  appearance= list(rhs=c("Winner=Democrat"),default="lhs"))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##       0.817    0.1    1 none FALSE            TRUE       5   0.031      2
##  maxlen target   ext
##       5  rules FALSE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 96
##
## set item appearances ...[1 item(s)] done [0.00s].
## set transactions ...[117 item(s), 3110 transaction(s)] done [0.00s].
## sorting and recoding items ... [117 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5

## Warning in apriori(final_assoc_data, parameter = list(minlen = 2, maxlen
## = 5, : Mining stopped (maxlen reached). Only patterns up to a length of 5
## returned!

##  done [0.11s].
## writing ... [16 rule(s)] done [0.00s].
## creating S4 object  ... done [0.02s].
```

```
#inspect the democratic rules
summary(rules2)
```

```
## set of 16 rules
##
## rule length distribution (lhs + rhs):sizes
##  4  5
##  4 12
##
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
```

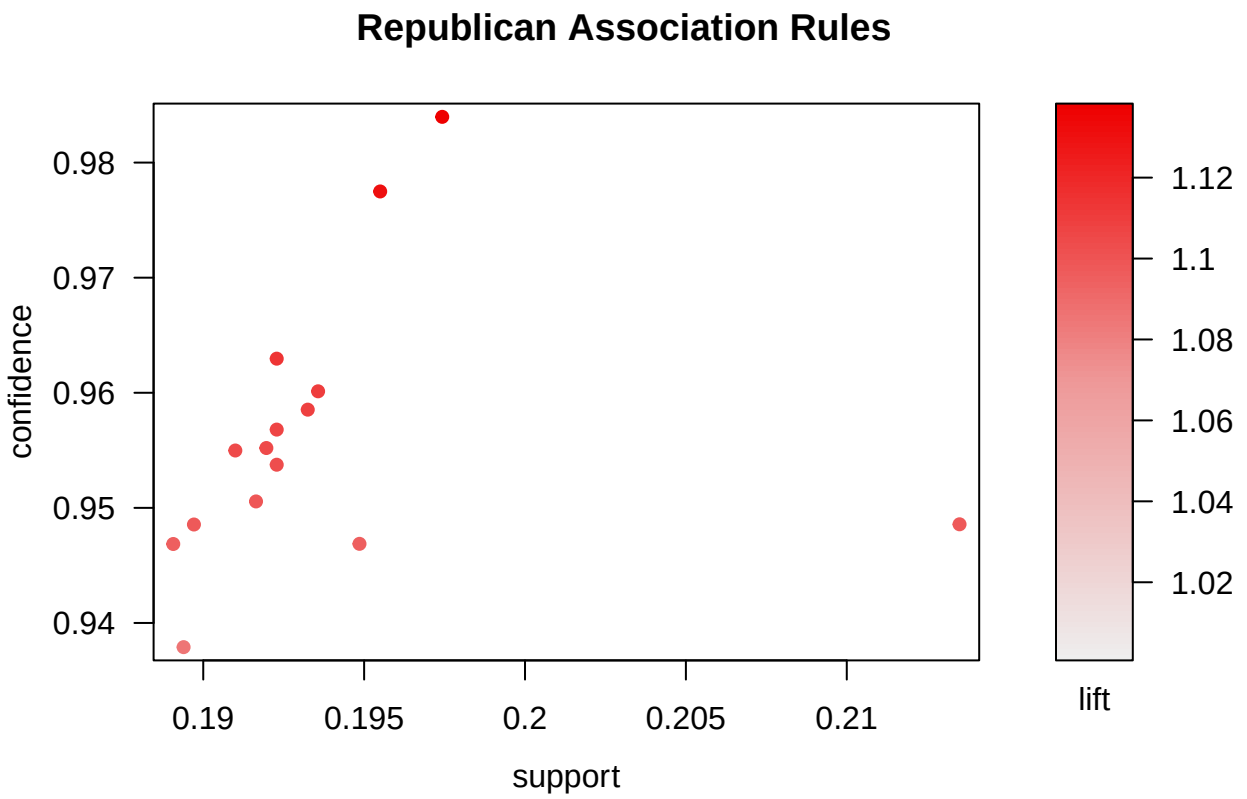
```
##    4.00    4.75    5.00    4.75    5.00    5.00
##
## summary of quality measures:
##     support          confidence          lift           count
##  Min.   :0.03119   Min.   :0.8175   Min.   :6.053   Min.   : 97.0
##  1st Qu.:0.03215   1st Qu.:0.8218   1st Qu.:6.085   1st Qu.:100.0
##  Median :0.03264   Median :0.8235   Median :6.098   Median :101.5
##  Mean   :0.03308   Mean   :0.8287   Mean   :6.136   Mean   :102.9
##  3rd Qu.:0.03288   3rd Qu.:0.8303   3rd Qu.:6.148   3rd Qu.:102.2
##  Max.   :0.03698   Max.   :0.8772   Max.   :6.495   Max.   :115.0
##
## mining info:
##              data ntransactions support confidence
##  final_assoc_data          3110   0.031      0.817
```

```
inspect(rules2)
```

```
##      lhs                                                                rhs                  support c
## [1]  {Percent_White_Alone=0.108_to_0.7589,
##       Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Population_per_square_mile=0.0023_to_1}                        => {Winner=Democrat} 0.03601286  0
## [2]  {Percent_White_Alone=0.108_to_0.7589,
##       Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Percent_Veterans=0.0221_to_1}                                  => {Winner=Democrat} 0.03151125  0
## [3]  {Percent_White_Alone=0.108_to_0.7589,
##       Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985}         => {Winner=Democrat} 0.03633441  0
## [4]  {Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Population_per_square_mile=0.0023_to_1}                        => {Winner=Democrat} 0.03697749  0
## [5]  {Percent_White_Alone=0.108_to_0.7589,
##       Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Percent_Homeownership_rate=0.194_to_0.6679,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985}         => {Winner=Democrat} 0.03247588  0
## [6]  {Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Percent_Homeownership_rate=0.194_to_0.6679,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Population_per_square_mile=0.0023_to_1}                        => {Winner=Democrat} 0.03215434  0
## [7]  {Percent_Foreign_born_persons=0.066_to_0.513,
##       Percent_Homeownership_rate=0.194_to_0.6679,
##       Median_value_of_owner_occupied_housing_units=0.1445_to_1,
##       Accommodation_and_food_services_sales_2007_1000=0.0057_to_1}   => {Winner=Democrat} 0.03118971  0
## [8]  {Percent_Bachelors_degree_or_higher=0.254_to_0.744,
##       Percent_Homeownership_rate=0.194_to_0.6679,
##       Median_value_of_owner_occupied_housing_units=0.1445_to_1,
##       Accommodation_and_food_services_sales_2007_1000=0.0057_to_1}   => {Winner=Democrat} 0.03151125  0
## [9]  {Percent_White_Alone=0.108_to_0.7589,
##       Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Population_per_square_mile=0.0023_to_1}                        => {Winner=Democrat} 0.03215434  0
## [10] {Population_2014=0.0092_to_1,
##       Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Population_per_square_mile=0.0023_to_1}                        => {Winner=Democrat} 0.03247588  0
```

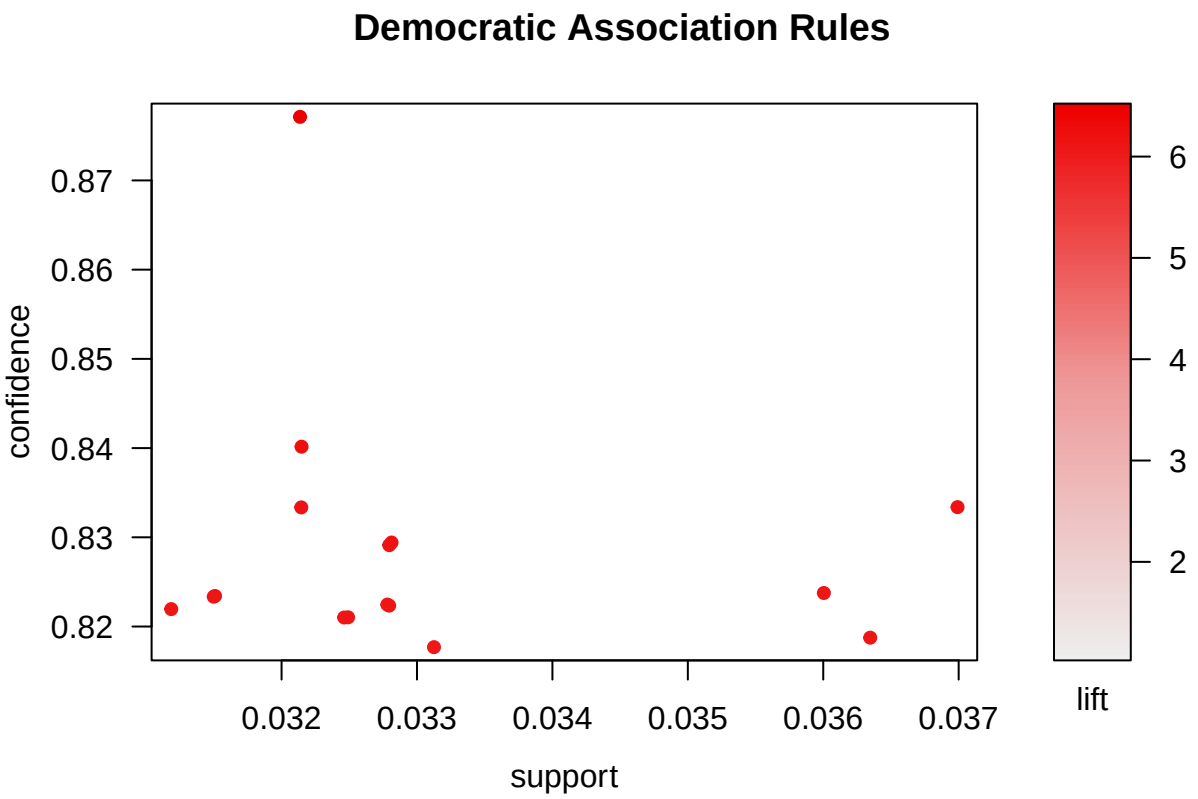
```
## [11] {Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Percent_Veterans=0.0221_to_1,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Population_per_square_mile=0.0023_to_1}                    => {Winner=Democrat} 0.03215434  0
## [12] {Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units=0.0116_to_1,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Population_per_square_mile=0.0023_to_1}                    => {Winner=Democrat} 0.03279743  0
## [13] {Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Number_Households=0.0107_to_1,
##       Population_per_square_mile=0.0023_to_1}                    => {Winner=Democrat} 0.03279743  0
## [14] {Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Private_nonfarm_establishments=0.0079_to_1,
##       Population_per_square_mile=0.0023_to_1}                    => {Winner=Democrat} 0.03279743  0
## [15] {Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Private_nonfarm_employment=0.0076_to_1,
##       Population_per_square_mile=0.0023_to_1}                    => {Winner=Democrat} 0.03311897  0
## [16] {Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,
##       Housing_units_in_multi_unit_structures=0.178_to_0.985,
##       Accommodation_and_food_services_sales_2007_1000=0.0057_to_1,
##       Population_per_square_mile=0.0023_to_1}                    => {Winner=Democrat} 0.03279743  0
```

```
#plot the rules to sees support, confidence, and lift
plot(rules1, main = "Republican Association Rules")
```



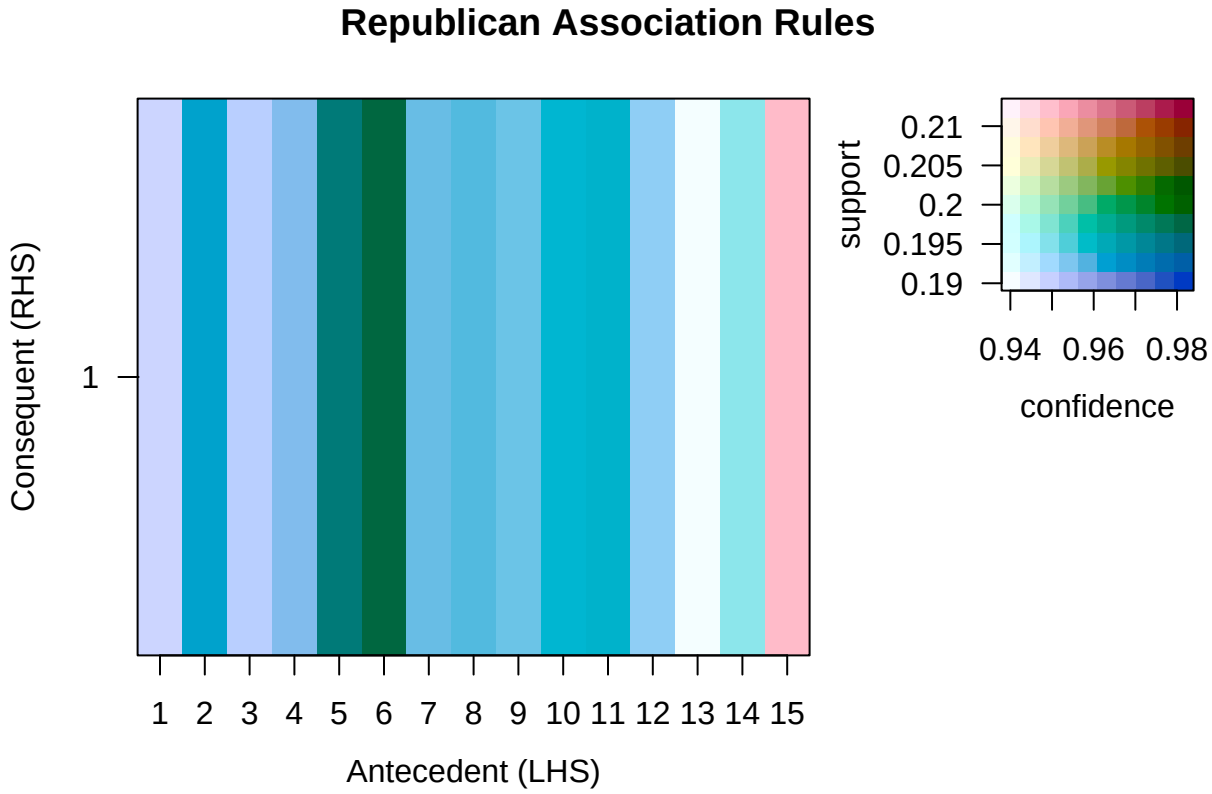

```
plot(rules2, main = "Democratic Association Rules")
```

```
## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
```



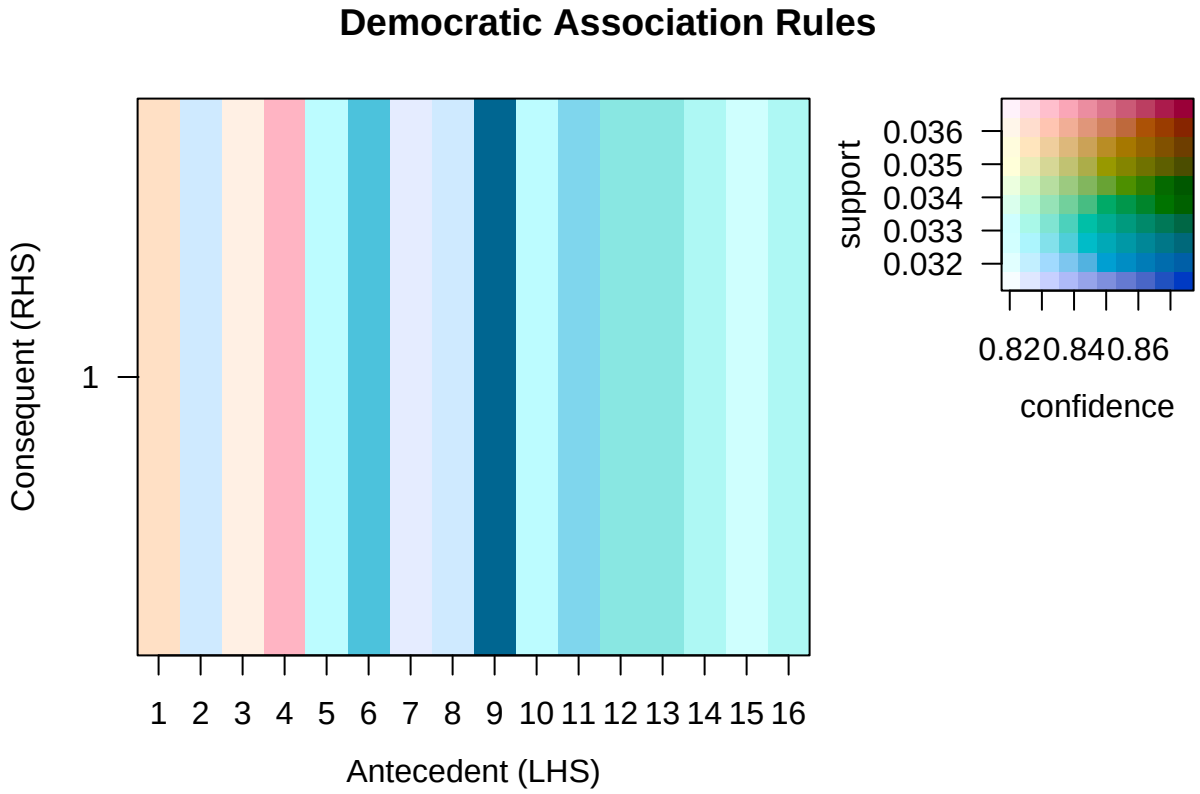

```
#plot shading matrix for rules
plot(rules1, method="matrix", shading=c("support", "confidence"),
     main = "Republican Association Rules")
```

```
## Itemsets in Antecedent (LHS)
##  [1] "{Housing_units_in_multi_unit_structures=0.055_to_0.0819}"
##  [2] "{Percent_White_Alone_not_Hispanic_or_Latino=0.888_to_0.9419}"
##  [3] "{Per_capita_money_income_in_past_12_months=0.1928_to_0.2387}"
##  [4] "{Per_capita_money_income_in_past_12_months=0.2388_to_0.2825}"
##  [5] "{Percent_White_Alone=0.9652_to_0.993}"
##  [6] "{Percent_White_Alone_not_Hispanic_or_Latino=0.942_to_0.986}"
##  [7] "{Median_value_of_owner_occupied_housing_units=0.0692_to_0.0975}"
##  [8] "{Percent_Homeownership_rate=0.752_to_0.7839}"
##  [9] "{Percent_65_Years_and_Older=0.183_to_0.2079}"
## [10] "{Percent_65_Years_and_Older=0.208_to_0.529}"
## [11] "{Percent_Homeownership_rate=0.784_to_0.938}"
## [12] "{Percent_White_Alone=0.942_to_0.9651}"
## [13] "{Percent_Homeownership_rate=0.718_to_0.7519}"
## [14] "{Percent_Bachelors_degree_or_higher=0.16_to_0.1939}"
## [15] "{Percent_Foreign_born_persons=0.01_to_0.0189}"
## Itemsets in Consequent (RHS)
## [1] "{Winner=Republican}"
```




```
plot(rules2, method="matrix", shading=c("support", "confidence"),
     main = "Democratic Association Rules")
```

```
## Itemsets in Antecedent (LHS)
##  [1] "{Percent_White_Alone=0.108_to_0.7589,Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.614
##  [2] "{Percent_White_Alone=0.108_to_0.7589,Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.614
##  [3] "{Percent_White_Alone=0.108_to_0.7589,Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.614
##  [4] "{Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Housing_units_in_multi_unit_structu
##  [5] "{Percent_White_Alone=0.108_to_0.7589,Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.614
##  [6] "{Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Percent_Homeownership_rate=0.194_to
##  [7] "{Percent_Foreign_born_persons=0.066_to_0.513,Percent_Homeownership_rate=0.194_to_0.6679,Median_
##  [8] "{Percent_Bachelors_degree_or_higher=0.254_to_0.744,Percent_Homeownership_rate=0.194_to_0.6679,H
##  [9] "{Percent_White_Alone=0.108_to_0.7589,Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.614
## [10] "{Population_2014=0.0092_to_1,Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Housing
## [11] "{Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Percent_Veterans=0.0221_to_1,Housi
## [12] "{Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Housing_units=0.0116_to_1,Housing_u
## [13] "{Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Housing_units_in_multi_unit_structu
## [14] "{Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Housing_units_in_multi_unit_structu
## [15] "{Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Housing_units_in_multi_unit_structu
## [16] "{Percent_White_Alone_not_Hispanic_or_Latino=0.031_to_0.6147,Housing_units_in_multi_unit_structu
## Itemsets in Consequent (RHS)
## [1] "{Winner=Democrat}"
```



Conclusion

In conclusion, using market basket analysis allowed for the mining of interesting rules that outline voting preference by demographics. A full write-up of the results can be found on Ian Jeffries' github page.