Dataset:

- "NBA Players" by Justinas Cirtautas (updated 2 years ago)
- Summary: Biometric, biographic, and basic box score stats from 1996 to the 2022 season
- Link: https://www.kaggle.com/datasets/justinas/nba-players-data
- Report at the end of this document

Summary:

1. Installing and importing packages

```
R 4.5.1 · ~/Desktop/Projects/Class Projects/CUS 610/Assignment5_Project/
> install.packages("arules")
trying URL 'https://cran.rstudio.com/bin/macosx/big-sur-arm64/contrib/4.5/arules_1.7-11.tgz'
Content type 'application/x-gzip' length 2736021 bytes (2.6 MB)
==================================================
downloaded 2.6 MB


The downloaded binary packages are in
	/var/folders/v2/j9qyr40n4p1_tf97m0r491lr0000gn/T//RtmppJA6zy/downloaded_packages
> library(arules)
Loading required package: Matrix

Attaching package: ‘arules’

The following objects are masked from ‘package:base’:

    abbreviate, write
```

```
> library(arulesViz)
```

2. Reading the data set as a data frame, performing initial summary statistics

```
> bbData <- read.csv(file="/Users/themi/Downloads/all_seasons.csv")
>
```

```
> head(bbData)
  X       player_name team_abbreviation age player_height player_weight                college country draft_year draft_round
1 0 Randy Livingston               HOU  22        193.04      94.80073        Louisiana State     USA       1996           2
2 1 Gaylon Nickerson               WAS  28        190.50      86.18248 Northwestern Oklahoma     USA       1994           2
3 2     George Lynch               VAN  26        203.20     103.41898         North Carolina     USA       1993           1
4 3   George McCloud               LAL  30        203.20     102.05820          Florida State     USA       1989           1
5 4     George Zidek               DEN  23        213.36     119.74829                   UCLA     USA       1995           1
6 5   Gerald Wilkins               ORL  33        198.12     102.05820 Tennessee-Chattanooga     USA       1985           2
  draft_number gp  pts reb ast net_rating oreb_pct dreb_pct usg_pct ts_pct ast_pct  season
1           42 64  3.9 1.5 2.4        0.3    0.042    0.071   0.169  0.487   0.248 1996-97
2           34  4  3.8 1.3 0.3        8.9    0.030    0.111   0.174  0.497   0.043 1996-97
3           12 41  8.3 6.4 1.9       -8.2    0.106    0.185   0.175  0.512   0.125 1996-97
4            7 64 10.2 2.8 1.7       -2.7    0.027    0.111   0.206  0.527   0.125 1996-97
5           22 52  2.8 1.7 0.3      -14.1    0.102    0.169   0.195  0.500   0.064 1996-97
6           47 80 10.6 2.2 2.2       -5.8    0.031    0.064   0.203  0.503   0.143 1996-97
```

```
> str(bbData)
'data.frame':	12844 obs. of  22 variables:
 $ X                : int  0 1 2 3 4 5 6 7 8 9 ...
 $ player_name      : chr  "Randy Livingston" "Gaylon Nickerson" "George Lynch" "George McCloud" ...
 $ team_abbreviation: chr  "HOU" "WAS" "VAN" "LAL" ...
 $ age              : num  22 28 26 30 23 33 26 30 24 24 ...
 $ player_height    : num  193 190 203 203 213 ...
 $ player_weight    : num  94.8 86.2 103.4 102.1 119.7 ...
 $ college          : chr  "Louisiana State" "Northwestern Oklahoma" "North Carolina" "Florida State" ...
 $ country          : chr  "USA" "USA" "USA" "USA" ...
 $ draft_year       : chr  "1996" "1994" "1993" "1989" ...
 $ draft_round      : chr  "2" "2" "1" "1" ...
 $ draft_number     : chr  "42" "34" "12" "7" ...
 $ gp               : int  64 4 41 64 52 80 73 79 80 80 ...
 $ pts              : num  3.9 3.8 8.3 10.2 2.8 10.6 10.6 26.8 21.1 21.4 ...
 $ reb              : num  1.5 1.3 6.4 2.8 1.7 2.2 6.6 4 6.3 9 ...
 $ ast              : num  2.4 0.3 1.9 1.7 0.3 2.2 0.4 2 3.1 7.3 ...
 $ net_rating       : num  0.3 8.9 -8.2 -2.7 -14.1 -5.8 6.9 3.2 -2.9 6.9 ...
 $ oreb_pct         : num  0.042 0.03 0.106 0.027 0.102 0.031 0.098 0.025 0.051 0.049 ...
 $ dreb_pct         : num  0.071 0.111 0.185 0.111 0.169 0.064 0.217 0.087 0.144 0.232 ...
 $ usg_pct          : num  0.169 0.174 0.175 0.206 0.195 0.203 0.185 0.272 0.278 0.283 ...
 $ ts_pct           : num  0.487 0.497 0.512 0.527 0.5 0.503 0.618 0.605 0.528 0.556 ...
 $ ast_pct          : num  0.248 0.043 0.125 0.125 0.064 0.143 0.024 0.088 0.146 0.356 ...
 $ season           : chr  "1996-97" "1996-97" "1996-97" "1996-97" ...
```

```
> summary(bbData)
       X          player_name        team_abbreviation       age        player_height   player_weight       college
 Min.   :    0   Length:12844       Length:12844       Min.   :18.00   Min.   :160.0   Min.   : 60.33   Length:12844
 1st Qu.: 3211   Class :character   Class :character   1st Qu.:24.00   1st Qu.:193.0   1st Qu.: 90.72   Class :character
 Median : 6422   Mode  :character   Mode  :character   Median :26.00   Median :200.7   Median : 99.79   Mode  :character
 Mean   : 6422                                         Mean   :27.05   Mean   :200.6   Mean   :100.26
 3rd Qu.: 9632                                         3rd Qu.:30.00   3rd Qu.:208.3   3rd Qu.:108.86
 Max.   :12843                                         Max.   :44.00   Max.   :231.1   Max.   :163.29
   country           draft_year         draft_round        draft_number             gp             pts
 Length:12844       Length:12844       Length:12844       Length:12844       Min.   : 1.00   Min.   : 0.000
 Class :character   Class :character   Class :character   Class :character   1st Qu.:31.00   1st Qu.: 3.600
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Median :57.00   Median : 6.700
                                                                             Mean   :51.15   Mean   : 8.213
                                                                             3rd Qu.:73.00   3rd Qu.:11.500
                                                                             Max.   :85.00   Max.   :36.100
      reb              ast           net_rating          oreb_pct          dreb_pct          usg_pct           ts_pct
 Min.   : 0.000   Min.   : 0.000   Min.   :-250.000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
 1st Qu.: 1.800   1st Qu.: 0.600   1st Qu.:   -6.400   1st Qu.:0.02100   1st Qu.:0.0960   1st Qu.:0.1490   1st Qu.:0.4820
 Median : 3.000   Median : 1.200   Median :   -1.300   Median :0.04000   Median :0.1305   Median :0.1810   Median :0.5250
 Mean   : 3.558   Mean   : 1.825   Mean   :   -2.226   Mean   :0.05407   Mean   :0.1406   Mean   :0.1846   Mean   :0.5131
 3rd Qu.: 4.700   3rd Qu.: 2.400   3rd Qu.:    3.200   3rd Qu.:0.08300   3rd Qu.:0.1790   3rd Qu.:0.2170   3rd Qu.:0.5630
 Max.   :16.300   Max.   :11.700   Max.   : 300.000   Max.   :1.00000   Max.   :1.0000   Max.   :1.0000   Max.   :1.5000
    ast_pct          season
 Min.   :0.0000   Length:12844
 1st Qu.:0.0660   Class :character
 Median :0.1030   Mode  :character
 Mean   :0.1316
 3rd Qu.:0.1790
 Max.   :1.0000
```

3. Clearing NAs, renaming columns

```
> bbData <- na.omit(bbData)
```

```
> #Using dplyr to rename columns
> library(dplyr)

Attaching package: ‘dplyr’

The following objects are masked from ‘package:arules’:

    intersect, recode, setdiff, setequal, union

The following objects are masked from ‘package:stats’:

    filter, lag

The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union

> bbData <- bbData %>%
+     rename(team = team_abbreviation)
```

```
> bbData <- bbData %>%
+     rename(team = team_abbreviation)
> bbData <- bbData %>%
+     rename(height = player_height, weight = player_weight)
>
```

4. Creating a new data frame with the most essential columns, filtering the dataset by games played ensures outliers are removed, and data cleaning (clearing NAs, NULLs, and empty strings)

```
> # Discretizing the most important columns
> bbDataImportant <- bbData[, c("team", "age", "height", "weight", "gp", "pts", "reb", "ast", "net_rating", "oreb_pct", "dreb_pct",
"usg_pct", "ts_pct", "ast_pct")]
```

```
> # Keep players that have at least played 20 games (~1/4 of the 82 game season in the NBA)
> bbDataImportant <- subset(bbDataImportant, gp >= 20)
```

```
> sum(is.na(bbDataImportant))
[1] 42
```

```
> bbDataImportant <- na.omit(bbDataImportant)
> sum(is.na(bbDataImportant))
[1] 0
```

5. Discretizing columns for Apriori

```
> bbDataImportant$age <-
+     cut(bbDataImportant$age, breaks = c(17, 24, 30, 45), labels = c("Young", "Prime", "Veteran"), include.lowest = TRUE)
> bbDataImportant$height <-
+     cut(bbDataImportant$height, breaks = 3, labels = c("Short", "Medium", "Tall"))
> bbDataImportant$weight <-
+     cut(bbDataImportant$weight, breaks = 3, labels = c("Light", "Medium", "Heavy"))
> bbDataImportant$pts <-
+     cut(bbDataImportant$pts, breaks = 3, labels = c("Low", "Medium", "High"))
> bbDataImportant$ast <-
+     cut(bbDataImportant$ast, breaks = 3, labels = c("Low", "Medium", "High"))
> bbDataImportant$reb <-
+     cut(bbDataImportant$reb, breaks = 3, labels = c("Low", "Medium", "High"))
```

```
> # Discretizing Rate(%) Stats
> bbDataImportant$net_rating <- cut(bbDataImportant$net_rating,
+                                    breaks = c(-50, -5, 5, 50),
+                                    labels = c("Negative", "Average", "Positive"),
+                                    include.lowest = TRUE)
>
> bbDataImportant$oreb_pct <- cut(bbDataImportant$oreb_pct,
+                                  breaks = c(0, 0.05, 0.10, 0.25),
+                                  labels = c("Low", "Medium", "High"),
+                                  include.lowest = TRUE)
>
> bbDataImportant$dreb_pct <- cut(bbDataImportant$dreb_pct,
+                                  breaks = c(0, 0.10, 0.20, 0.40),
+                                  labels = c("Low", "Medium", "High"),
+                                  include.lowest = TRUE)
>
> bbDataImportant$usg_pct <- cut(bbDataImportant$usg_pct,
+                                 breaks = c(0, 0.18, 0.25, 0.40),
+                                 labels = c("Low", "Medium", "High"),
+                                 include.lowest = TRUE)
>
> bbDataImportant$ts_pct <- cut(bbDataImportant$ts_pct,
+                                breaks = c(0, 0.52, 0.60, 0.70),
+                                labels = c("LowEff", "Solid", "Elite"),
+                                include.lowest = TRUE)
>
> bbDataImportant$ast_pct <- cut(bbDataImportant$ast_pct,
+                                 breaks = c(0, 0.10, 0.25, 0.50),
+                                 labels = c("Low", "Medium", "High"),
+                                 include.lowest = TRUE)
>
```

```
> # Preparing data for arules library and Apriori
> bbDataImportant <- lapply(bbDataImportant, as.factor)
> bbDataImportant <- as.data.frame(bbDataImportant)
> transactions <- as(bbDataImportant, "transactions")
```

6. Running and tuning Apriori, displaying the top 10 rules by confidence and lift
    a. First run: 5% support, 60% confidence, minimum length of rule = 2

```
> # Running Apriori, starting with 5% support and 60% confidence as the baseline
> rules <- apriori(transactions, parameter = list(supp=0.05, conf=0.6, minlen=2))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.6    0.1    1 none FALSE            TRUE       5    0.05      2     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 533

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[138 item(s), 10679 transaction(s)] done [0.01s].
sorting and recoding items ... [32 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 9 done [0.03s].
writing ... [26236 rule(s)] done [0.01s].
creating S4 object  ... done [0.00s].
```

```
> summary(rules)
set of 26236 rules

rule length distribution (lhs + rhs):sizes
   2    3    4    5    6    7    8    9 
 162 1489 5457 8986 7134 2631  369    8 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  2.000   4.000   5.000   5.176   6.000   9.000 

summary of quality measures:
    support          confidence        coverage            lift            count     
 Min.   :0.05000   Min.   :0.6000   Min.   :0.05000   Min.   :0.6978   Min.   : 534  
 1st Qu.:0.05843   1st Qu.:0.7060   1st Qu.:0.07089   1st Qu.:1.0892   1st Qu.: 624  
 Median :0.07154   Median :0.8522   Median :0.08980   Median :1.2055   Median : 764  
 Mean   :0.08887   Mean   :0.8263   Mean   :0.11050   Mean   :1.2564   Mean   : 949  
 3rd Qu.:0.09926   3rd Qu.:0.9464   3rd Qu.:0.12482   3rd Qu.:1.3344   3rd Qu.:1060  
 Max.   :0.69445   Max.   :1.0000   Max.   :0.86431   Max.   :5.6332   Max.   :7416  

mining info:
         data ntransactions support confidence
 transactions         10679    0.05        0.6
                                                                                  call
 apriori(data = transactions, parameter = list(supp = 0.05, conf = 0.6, minlen = 2))
```

```
> # Inspecting top 10 rules by confidence and lift
> inspect(sort(rules, by = "confidence")[1:10])
     lhs                                 rhs         support    confidence coverage   lift     count
[1]  {ast_pct=Low}                    => {ast=Low} 0.46596123 1          0.46596123 1.156988 4976
[2]  {ts_pct=Elite, ast_pct=Low}      => {ast=Low} 0.05300122 1          0.05300122 1.156988  566
[3]  {ast=Medium, dreb_pct=Low}       => {reb=Low} 0.05543590 1          0.05543590 1.271007  592
[4]  {dreb_pct=Low, ast_pct=High}     => {reb=Low} 0.07369604 1          0.07369604 1.271007  787
[5]  {oreb_pct=High, ast_pct=Low}     => {ast=Low} 0.12716546 1          0.12716546 1.156988 1358
[6]  {reb=Low, oreb_pct=High}         => {ast=Low} 0.07772263 1          0.07772263 1.156988  830
[7]  {dreb_pct=High, ast_pct=Low}     => {ast=Low} 0.10974810 1          0.10974810 1.156988 1172
[8]  {reb=Low, dreb_pct=High}         => {pts=Low} 0.05515498 1          0.05515498 1.364205  589
[9]  {reb=Low, dreb_pct=High}         => {ast=Low} 0.05515498 1          0.05515498 1.156988  589
[10] {net_rating=Positive, ast_pct=Low} => {ast=Low} 0.06938852 1          0.06938852 1.156988  741
```

```
> # Inspecting top 10 rules by confidence and lift
> inspect(sort(rules, by = "lift")[1:10])
     lhs                                                rhs             support    confidence coverage   lift     count
[1]  {weight=Light, reb=Low, ast=Medium}             => {ast_pct=High} 0.05637232 0.7395577  0.07622437 5.633193 602
[2]  {weight=Light, reb=Low, ast=Medium, oreb_pct=Low} => {ast_pct=High} 0.05590411 0.7379481  0.07575616 5.620933 597
[3]  {weight=Light, ast=Medium}                      => {ast_pct=High} 0.05880700 0.7353630  0.07997003 5.601242 628
[4]  {weight=Light, ast=Medium, oreb_pct=Low}        => {ast_pct=High} 0.05796423 0.7325444  0.07912726 5.579773 619
[5]  {reb=Low, ast=Medium, oreb_pct=Low}             => {ast_pct=High} 0.06358273 0.6893401  0.09223710 5.250687 679
[6]  {reb=Low, ast=Medium}                           => {ast_pct=High} 0.06414458 0.6843157  0.09373537 5.212416 685
[7]  {height=Medium, reb=Low, ast=Medium, oreb_pct=Low} => {ast_pct=High} 0.05206480 0.6674670  0.07800356 5.084080 556
[8]  {height=Medium, reb=Low, ast=Medium}            => {ast_pct=High} 0.05253301 0.6631206  0.07922090 5.050973 561
[9]  {ast=Medium, oreb_pct=Low}                      => {ast_pct=High} 0.07135500 0.6620330  0.10778163 5.042689 762
[10] {height=Medium, ast=Medium, oreb_pct=Low}       => {ast_pct=High} 0.05890065 0.6438076  0.09148797 4.903867 629
> 
```

b. Second run: 10% support, 70% confidence, minimum length of rule = 2

```
> # Running Apriori (x2),10% support and 70% confidence as the baseline
> rules <- apriori(transactions, parameter = list(supp=0.1, conf=0.7, minlen=2))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.7    0.1    1 none FALSE            TRUE       5     0.1      2     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 1067 

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[138 item(s), 10679 transaction(s)] done [0.01s].
sorting and recoding items ... [30 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 done [0.01s].
writing ... [5038 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

```
> summary(rules)
set of 5038 rules

rule length distribution (lhs + rhs):sizes
   2    3    4    5    6    7 
  92  690 1733 1734  706   83 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
    2.0     4.0     5.0     4.5     5.0     7.0 

summary of quality measures:
    support         confidence        coverage           lift            count     
 Min.   :0.1001   Min.   :0.7002   Min.   :0.1001   Min.   :0.8129   Min.   :1069  
 1st Qu.:0.1122   1st Qu.:0.8006   1st Qu.:0.1279   1st Qu.:1.0886   1st Qu.:1198  
 Median :0.1305   Median :0.8854   Median :0.1517   Median :1.1693   Median :1394  
 Mean   :0.1549   Mean   :0.8749   Mean   :0.1796   Mean   :1.2141   Mean   :1654  
 3rd Qu.:0.1716   3rd Qu.:0.9581   3rd Qu.:0.2016   3rd Qu.:1.2987   3rd Qu.:1833  
 Max.   :0.6944   Max.   :1.0000   Max.   :0.8643   Max.   :2.7569   Max.   :7416  

mining info:
         data ntransactions support confidence
 transactions         10679     0.1        0.7
                                                                                   call
 apriori(data = transactions, parameter = list(supp = 0.1, conf = 0.7, minlen = 2))
```

```
> # Inspecting top 10 rules by confidence and lift
> inspect(sort(rules, by = "confidence")[1:10])
     lhs                                rhs       support   confidence coverage  lift     count
[1]  {ast_pct=Low}                   => {ast=Low} 0.4659612 1          0.4659612 1.156988 4976 
[2]  {oreb_pct=High, ast_pct=Low}    => {ast=Low} 0.1271655 1          0.1271655 1.156988 1358 
[3]  {dreb_pct=High, ast_pct=Low}    => {ast=Low} 0.1097481 1          0.1097481 1.156988 1172 
[4]  {reb=Medium, ast_pct=Low}       => {ast=Low} 0.1030995 1          0.1030995 1.156988 1101 
[5]  {net_rating=Negative, ast_pct=Low} => {ast=Low} 0.1345632 1          0.1345632 1.156988 1437 
[6]  {height=Tall, ast_pct=Low}      => {ast=Low} 0.1951494 1          0.1951494 1.156988 2084 
[7]  {weight=Light, dreb_pct=Low}    => {reb=Low} 0.2016106 1          0.2016106 1.271007 2153 
[8]  {dreb_pct=Low, usg_pct=Medium}  => {reb=Low} 0.1308175 1          0.1308175 1.271007 1397 
[9]  {dreb_pct=Low, ts_pct=LowEff}   => {reb=Low} 0.1299747 1          0.1299747 1.271007 1388 
[10] {age=Prime, dreb_pct=Low}       => {reb=Low} 0.1262290 1          0.1262290 1.271007 1348 
```

```
> # Inspecting top 10 rules by confidence and lift
> inspect(sort(rules, by = "lift")[1:10])
     lhs                                                 rhs            support   confidence coverage  lift     count
[1]  {reb=Low, oreb_pct=Low, ast_pct=High}            => {weight=Light} 0.1033805 0.9012245  0.1147111 2.756854 1104 
[2]  {reb=Low, ast_pct=High}                          => {weight=Light} 0.1047851 0.8995177  0.1164903 2.751633 1119 
[3]  {oreb_pct=Low, ast_pct=High}                     => {weight=Light} 0.1065643 0.8460967  0.1259481 2.588217 1138 
[4]  {ast_pct=High}                                   => {weight=Light} 0.1085308 0.8266762  0.1312857 2.528810 1159 
[5]  {oreb_pct=Low, dreb_pct=Low, usg_pct=Medium}     => {weight=Light} 0.1005712 0.8069121  0.1246371 2.468351 1074 
[6]  {reb=Low, oreb_pct=Low, dreb_pct=Low, usg_pct=Medium} => {weight=Light} 0.1005712 0.8069121  0.1246371 2.468351 1074 
[7]  {dreb_pct=Low, usg_pct=Medium}                   => {weight=Light} 0.1025377 0.7838225  0.1308175 2.397720 1095 
[8]  {reb=Low, dreb_pct=Low, usg_pct=Medium}          => {weight=Light} 0.1025377 0.7838225  0.1308175 2.397720 1095 
[9]  {oreb_pct=Low, dreb_pct=Low, ast_pct=Medium}     => {weight=Light} 0.1067516 0.7760381  0.1375597 2.373908 1140 
[10] {reb=Low, oreb_pct=Low, dreb_pct=Low, ast_pct=Medium} => {weight=Light} 0.1067516 0.7760381  0.1375597 2.373908 1140 
```

c. Third run: 7% support, 65% confidence, minimum length of rule = 3

```
> # Running Apriori (x3),7% support and 65% confidence and minlen of 3
> rules <- apriori(transactions, parameter = list(supp=0.07, conf=0.65, minlen=3))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
       0.65    0.1    1 none FALSE            TRUE       5    0.07      3     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 747

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[138 item(s), 10679 transaction(s)] done [0.01s].
sorting and recoding items ... [31 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 8 done [0.02s].
writing ... [12037 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
>
```

```
> summary(rules)
set of 12037 rules

rule length distribution (lhs + rhs):sizes
   3    4    5    6    7    8 
1057 3244 4384 2671  641   40 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  3.000   4.000   5.000   4.893   6.000   8.000 

summary of quality measures:
    support          confidence        coverage            lift            count     
 Min.   :0.07004   Min.   :0.6500   Min.   :0.07004   Min.   :0.7527   Min.   : 748  
 1st Qu.:0.08053   1st Qu.:0.7439   1st Qu.:0.09523   1st Qu.:1.0945   1st Qu.: 860  
 Median :0.09729   Median :0.8718   Median :0.11724   Median :1.1966   Median :1039  
 Mean   :0.11489   Mean   :0.8514   Mean   :0.13760   Mean   :1.2366   Mean   :1227  
 3rd Qu.:0.12717   3rd Qu.:0.9534   3rd Qu.:0.15423   3rd Qu.:1.3213   3rd Qu.:1358  
 Max.   :0.60268   Max.   :1.0000   Max.   :0.69445   Max.   :5.0427   Max.   :6436  

mining info:
         data ntransactions support confidence
 transactions         10679    0.07       0.65
                                                                                  call
 apriori(data = transactions, parameter = list(supp = 0.07, conf = 0.65, minlen = 3))
>
```

```
> # Inspecting top 10 rules by confidence (7% sup, 65% conf, minlen of 3)
> inspect(sort(rules, by = "confidence")[1:10])
     lhs                                  rhs        support    confidence coverage   lift     count
[1]  {dreb_pct=Low, ast_pct=High}        => {reb=Low} 0.07369604 1          0.07369604 1.271007  787
[2]  {oreb_pct=High, ast_pct=Low}        => {ast=Low} 0.12716546 1          0.12716546 1.156988 1358
[3]  {reb=Low, oreb_pct=High}            => {ast=Low} 0.07772263 1          0.07772263 1.156988  830
[4]  {dreb_pct=High, ast_pct=Low}        => {ast=Low} 0.10974810 1          0.10974810 1.156988 1172
[5]  {reb=Medium, ast_pct=Low}           => {ast=Low} 0.10309954 1          0.10309954 1.156988 1101
[6]  {age=Veteran, ast_pct=Low}          => {ast=Low} 0.09635734 1          0.09635734 1.156988 1029
[7]  {pts=Medium, dreb_pct=Low}          => {reb=Low} 0.07154228 1          0.07154228 1.271007  764
[8]  {net_rating=Negative, dreb_pct=Low} => {reb=Low} 0.07819084 1          0.07819084 1.271007  835
[9]  {net_rating=Negative, ast_pct=Low}  => {ast=Low} 0.13456316 1          0.13456316 1.156988 1437
[10] {height=Tall, ast_pct=Low}          => {ast=Low} 0.19514936 1          0.19514936 1.156988 2084
```

```
> # Inspecting top 10 rules by lift (7% sup, 65% conf, minlen of 3)
> inspect(sort(rules, by = "lift")[1:10])
     lhs                                                  rhs             support    confidence coverage   lift     count
[1]  {ast=Medium, oreb_pct=Low}                        => {ast_pct=High}  0.07135500 0.6620330  0.10778163 5.042689  762
[2]  {reb=Low, oreb_pct=Low, ast_pct=High}             => {weight=Light} 0.10338047 0.9012245  0.11471112 2.756854 1104
[3]  {reb=Low, ast_pct=High}                           => {weight=Light} 0.10478509 0.8995177  0.11649031 2.751633 1119
[4]  {height=Medium, reb=Low, oreb_pct=Low, ast_pct=High} => {weight=Light} 0.07865905 0.8750000  0.08989606 2.676633  840
[5]  {height=Medium, reb=Low, ast_pct=High}            => {weight=Light} 0.07987639 0.8730809  0.09148797 2.670762  853
[6]  {ast=Low, dreb_pct=High}                          => {height=Tall}  0.10899897 0.6995192  0.15581983 2.624795 1164
[7]  {weight=Medium, ast=Low, dreb_pct=High}           => {height=Tall}  0.10300590 0.6957622  0.14804757 2.610697 1100
[8]  {dreb_pct=High, ast_pct=Low}                      => {height=Tall}  0.07631801 0.6953925  0.10974810 2.609310  815
[9]  {ast=Low, dreb_pct=High, ast_pct=Low}             => {height=Tall}  0.07631801 0.6953925  0.10974810 2.609310  815
[10] {weight=Medium, dreb_pct=High, ast_pct=Low}       => {height=Tall}  0.07332147 0.6941489  0.10562787 2.604644  783
> 
```

d. Fourth run: 10% support, 70% confidence, minimum length of rule = 3

```
> # Running Apriori (x4),10% support and 70% confidence and minlen of 3
> rules <- apriori(transactions, parameter = list(supp=0.1, conf=0.70, minlen=3))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.7    0.1    1 none FALSE            TRUE       5     0.1      3     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 1067

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[138 item(s), 10679 transaction(s)] done [0.00s].
sorting and recoding items ... [30 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 done [0.01s].
writing ... [4946 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

```
> summary(rules)
set of 4946 rules

rule length distribution (lhs + rhs):sizes
   3    4    5    6    7 
 690 1733 1734  706   83 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  3.000   4.000   5.000   4.547   5.000   7.000 

summary of quality measures:
    support         confidence        coverage           lift            count     
 Min.   :0.1001   Min.   :0.7002   Min.   :0.1001   Min.   :0.8129   Min.   :1069  
 1st Qu.:0.1119   1st Qu.:0.8017   1st Qu.:0.1272   1st Qu.:1.0900   1st Qu.:1195  
 Median :0.1299   Median :0.8862   Median :0.1508   Median :1.1716   Median :1387  
 Mean   :0.1514   Mean   :0.8756   Mean   :0.1754   Mean   :1.2151   Mean   :1617  
 3rd Qu.:0.1684   3rd Qu.:0.9584   3rd Qu.:0.1987   3rd Qu.:1.2998   3rd Qu.:1798  
 Max.   :0.6027   Max.   :1.0000   Max.   :0.6944   Max.   :2.7569   Max.   :6436  

mining info:
         data ntransactions support confidence
 transactions         10679     0.1        0.7
                                                                             call
 apriori(data = transactions, parameter = list(supp = 0.1, conf = 0.7, minlen = 3))
```

e. Fifth (and final run): 15% support, 80% confidence, minimum length of rule = 2

```
> # Running Apriori (x5),15% support and 80% confidence and minlen of 2
> rules <- apriori(transactions, parameter = list(supp=0.15, conf=0.80, minlen=2))
Apriori

Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen maxlen target  ext
        0.8    0.1    1 none FALSE            TRUE       5    0.15      2     10  rules TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

Absolute minimum support count: 1601

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[138 item(s), 10679 transaction(s)] done [0.01s].
sorting and recoding items ... [26 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 5 6 7 done [0.01s].
writing ... [1291 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
```

```
> summary(rules)
set of 1291 rules

rule length distribution (lhs + rhs):sizes
  2   3   4   5   6   7 
 46 294 537 336  74   4 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  2.000   3.000   4.000   4.085   5.000   7.000 

summary of quality measures:
    support         confidence        coverage            lift            count     
 Min.   :0.1500   Min.   :0.8000   Min.   :0.1502   Min.   :0.9282   Min.   :1602  
 1st Qu.:0.1676   1st Qu.:0.8694   1st Qu.:0.1854   1st Qu.:1.1005   1st Qu.:1790  
 Median :0.1968   Median :0.9130   Median :0.2128   Median :1.1570   Median :2102  
 Mean   :0.2198   Mean   :0.9130   Mean   :0.2422   Mean   :1.1949   Mean   :2348  
 3rd Qu.:0.2421   3rd Qu.:0.9664   3rd Qu.:0.2666   3rd Qu.:1.2704   3rd Qu.:2585  
 Max.   :0.6944   Max.   :1.0000   Max.   :0.8643   Max.   :1.7696   Max.   :7416  

mining info:
         data ntransactions support confidence
 transactions         10679    0.15        0.8
                                                                             call
 apriori(data = transactions, parameter = list(supp = 0.15, conf = 0.8, minlen = 2))
```

```
> # Inspecting top 10 rules by lift (15% sup, 85% conf, minlen of 2)
> inspect(sort(rules, by = "lift")[1:10])
     lhs                                               rhs             support   confidence coverage  lift     count
[1]  {height=Tall, weight=Medium, pts=Low, ast=Low}       => {ast_pct=Low}  0.1597528 0.8245529  0.1937447 1.769574 1706
[2]  {height=Tall, pts=Low, ast=Low}                      => {ast_pct=Low}  0.1652776 0.8232276  0.2007679 1.766730 1765
[3]  {height=Tall, weight=Medium, pts=Low}                => {ast_pct=Low}  0.1597528 0.8213770  0.1944939 1.762758 1706
[4]  {height=Tall, pts=Low}                               => {ast_pct=Low}  0.1652776 0.8201673  0.2015170 1.760162 1765
[5]  {weight=Light, dreb_pct=Low}                         => {oreb_pct=Low} 0.1977713 0.9809568  0.2016106 1.709750 2112
[6]  {weight=Light, reb=Low, dreb_pct=Low}                => {oreb_pct=Low} 0.1977713 0.9809568  0.2016106 1.709750 2112
[7]  {height=Medium, weight=Light, dreb_pct=Low}          => {oreb_pct=Low} 0.1672441 0.9786301  0.1708962 1.705695 1786
[8]  {height=Medium, weight=Light, reb=Low, dreb_pct=Low} => {oreb_pct=Low} 0.1672441 0.9786301  0.1708962 1.705695 1786
[9]  {weight=Light, reb=Low, ast_pct=Medium}              => {oreb_pct=Low} 0.1624684 0.9681920  0.1678060 1.687502 1735
[10] {weight=Light, ast_pct=Medium}                       => {oreb_pct=Low} 0.1646222 0.9675289  0.1701470 1.686346 1758
```

```
> # Inspecting top 10 rules by confidence (15% sup, 85% conf, minlen of 2)
> inspect(sort(rules, by = "confidence")[1:10])
     lhs                                  rhs        support   confidence coverage  lift     count
[1]  {ast_pct=Low}                     => {ast=Low} 0.4659612 1          0.4659612 1.156988 4976
[2]  {height=Tall, ast_pct=Low}        => {ast=Low} 0.1951494 1          0.1951494 1.156988 2084
[3]  {weight=Light, dreb_pct=Low}      => {reb=Low} 0.2016106 1          0.2016106 1.271007 2153
[4]  {net_rating=Average, dreb_pct=Low} => {reb=Low} 0.1581609 1          0.1581609 1.271007 1689
[5]  {oreb_pct=Low, dreb_pct=Low}      => {reb=Low} 0.2608859 1          0.2608859 1.271007 2786
[6]  {pts=Low, dreb_pct=Low}           => {reb=Low} 0.2000187 1          0.2000187 1.271007 2136
[7]  {oreb_pct=Medium, ast_pct=Low}    => {ast=Low} 0.1785748 1          0.1785748 1.156988 1907
[8]  {age=Young, ast_pct=Low}          => {ast=Low} 0.1619065 1          0.1619065 1.156988 1729
[9]  {usg_pct=Medium, ast_pct=Low}     => {ast=Low} 0.1566626 1          0.1566626 1.156988 1673
[10] {ts_pct=LowEff, ast_pct=Low}      => {ast=Low} 0.1911228 1          0.1911228 1.156988 2041
>
```

Report:

For this Apriori analysis, the "NBA Players" dataset from Kaggle was used. Created by Justinas Cirtautas in 2020 and updated in 2022, this dataset includes basic box score statistics for NBA players who played between the 1996-97 and 2021-2022 seasons. These stats include points, rebounds, assists, and advanced efficiency metrics (e.g., true shooting percentage, usage rate, and net rating). The goal of this mini project was to utilize the Apriori algorithm to uncover frequently co-occurring relationships between player statistics that reflect patterns found in the real-world NBA.

Before applying the Apriori algorithm to the dataset, the dataset was cleaned and preprocessed. Players who played less than 20 games (~1/4th of the 82-game regular season) were filtered out and removed from the dataset to eliminate outlier statistics from players with small sample sizes. Additional cleaning steps were taken, including removing nulls, empty strings, and empty numbers. Post-cleaning, the dataset contained 10,679 records. The next step was to choose the essential columns. The columns chosen were team, age, height, weight, games played ("gp"), points ("pts"), rebounds ("reb"), assists ("ast"), net rating, offensive rebound percentage ("oreb_pct"), defensive rebound percentage ("dreb_pct"), usage rate/percentage ("usg_pct"), true shooting percentage ("ts_pct"), and assist percentage ("ast_pct"). These columns were chosen for their importance to the game of basketball and give the most complete picture of any given basketball player. After all of this, the data frame was saved as "bbDataImportant."

Next, continuous statistics such as true shooting percentage, usage rate, offensive rebound percentage, and net rating were binned and discretized into three categories (Low, Medium, and High). Basketball-specific thresholds were applied to each, such as a true shooting percentage above 60% being considered "elite" and a usage percentage above 25% denoted as high usage. All numeric attributes were then converted to factors, and the dataset was transformed into a "transactions" object through the arules package. Each player represented one transaction, and each categorical variable was treated as an item. This format enables Apriori to operate appropriately and analyze the co-occurring relationships among stats.

The Apriori algorithm was implemented through a series of runs, each with a summary of each and the top ten rules by confidence and lift. Starting with a support threshold of 5%, a confidence threshold of 60%, and a minimum rule length of 2, the first run generated over 26,000 rules. To improve the interpretability of the rules, the thresholds were raised to 15% support, 80% confidence, and a minimum length of 2. This resulted in fewer than 2,000 rules (1,291 rules total).

After raising the thresholds, the final Apriori model generated 1,291 rules. The top ten rules, ranked by confidence and lift, were generated for this model. The top five rules, ranked by confidence and lift, are presented in the tables below for readability and convenience, along with explanations of their meaning and application in the real-world NBA.

Top Ten Rules by Confidence:

Rule	Support	Confidence	Lift	Interpretation
{ast_pct=Low} => {ast=Low}	0.4659612	1	1.156988	Players with a low assist percentage consistently have low total assists.
{height=Tall, ast_pct=Low} => {ast=Low}	0.1951494	1	1.156988	Tall players with low assist percentages also have low assist totals. This is typical of big men throughout the last 30 years, who are more focused on rebounding and defense.
{weight=Light, dreb_pct=Low} => {reb=Low}	0.2016106	1	1.271007	Lightweight players (who are typically guards/shorter players) with low defensive rebound percentages also record low total rebounds, which is consistent with how NBA guards play.
{net_rating=Average, dreb_pct=Low} => {reb=Low}	0.1581609	1	1.271007	Players with average net ratings but low defensive rebounding rates tend to have low total rebounds. This seems to suggest that position rather than

				performance drives rebounding.
{oreb_pct=Low, dreb_pct=Low} => {reb=Low}	0.2608859	1	1.271007	Players who struggle on both offensive and defensive boards predictably have low rebound totals.

Top Ten Rules by Lift:

Rule	Support	Confidence	Lift	Interpretation
{height=Tall, weight=Medium, pts=Low, ast=Low} => {ast_pct=Low}	0.1597528	0.8245529	1.769574	Tall, medium-weight players with low scoring and assist counts almost always have low assist percentages. This is very typical of interior players, who are usually power forwards or centers.
{height=Tall, pts=Low, ast=Low} => {ast_pct=Low}	0.1652776	0.8232276	1.766730	Taller players who score and assist less frequently tend to have low assist percentages, which aligns with the above rule as well and is typical of power forwards and centers.
{height=Tall, weight=Medium, pts=Low} => {ast_pct=Low}	0.1597528	0.8213770	1.762758	Similar to the two rules above. Tall, moderately weighted players with low scoring

				numbers often show low assist percentages, consistent with the play of power forwards and centers.
{height=Tall, pts=Low} => {ast_pct=Low}	0.1652776	0.8201673	1.760162	Similar to the previous rules. Tall players with low scoring output typically have low assist percentages, which shows the limited playmaking responsibilities of frontcourt players.
{weight=Light, dreb_pct=Low} => {oreb_pct=Low}	0.1977713	0.9809568	1.709750	Lightweight players with low defensive rebounding percentages also have low offensive rebounding percentages, showing that guards typically rebound less than frontcourt players.

Given the above rules that rank in the top ten in confidence and lift, the following assessments of this dataset post-Apriori can be made:

- Guards (PG & SG), who are typically light and short players, have lower rebounding numbers but higher assist numbers.
- Big men (PF and C), who generally are heavier and taller players, demonstrate low assist rates but higher rebounding numbers.

- Rules with 100% confidence confirm consistency between the total metrics, while those with high lift highlight distinct positional roles and tendencies among players with different heights and weights.

To conclude, the Apriori algorithm successfully identified the most meaningful co-occurring relationships among NBA player stats and attributes. The discussed rules closely align with how basketball is played in the NBA today and throughout its nearly 80-year history. This mini project demonstrates the value of association rule mining, particularly the Apriori algorithm, in the context of sports analytics. These analytics provide NBA teams, players, front offices, fans, and the media with interpretable insights that complement watching basketball, offering these groups a different angle and an edge on how to approach the game. In the future, this mini project can be extended by incorporating datasets with per-minute, lineup-adjusted, and era-adjusted data to refine the discussed relationships further, as well as provide people with a bird’s-eye view of how the game has evolved throughout its history.