

# HW02 - GSW Shot Charts

Stat 133, Spring 2018, Prof. Sanchez

*Due date: Fri Mar-09 (before midnight)*

From the logistical point of view, the purpose of this assignment is twofold. On one hand, we want you to keep working with `data.frames/tibbles` and producing plots but now using the packages "dplyr" and "ggplot2". On the other hand, we want you to start working with a more complex file structure.

From the analytics point of view, this HW involves visualizing shot data of NBA players. More specifically, you will be producing so-called "shot charts" for a handful of players from the Golden State Warriors, as well as other summaries and visualizations.

## 1) File Structure (10 pts)

After completing this assignment, the file structure of your project should look like this:

```
hw02/
  README.md
  data/
    andre-iguodala.csv
    draymond-green.csv
    kevin-durant.csv
    klay-thompson.csv
    stephen-curry.csv
    shots-data.csv
    data-dictionary.md
  code/
    make-shots-data-script.R
    make-shot-charts-script.R
  output/
    shots-summary.txt
    ... # other summary.txt files
  images/
    nba-court.jpg
    ... # pdf images of shot charts
  report/
    hw02-first-last.Rmd
    hw02-first-last.md
```

## General Instructions

- Create a folder (i.e. subdirectory) `hw02` in your github classroom repository.
  - Create a `README.md` file and include a description of what the HW is about.
  - Create a folder `data` which will contain the data files.
  - Create a folder `code` which will contain an R script file.
  - Create a folder `output` which will contain some R outputs.
  - Create a folder `images` which will contain some secondary plot images.
  - Create a folder `report` which will contain the files for your dynamic document (e.g. `Rmd` and derived files).
  - In the yaml header of the `Rmd` file, set the `output` field as `output: github_document` (Do NOT use the default `"output: html_document"`).
  - Name your `Rmd` file as `hw02-first-last.Rmd`, where `first` and `last` are your first and last names (e.g. `hw02-gaston-sanchez.Rmd`).
  - Please do not use code chunk options such as: `echo = FALSE`, `eval = FALSE`, `results = 'hide'`. All chunks must be visible and evaluated.
  - Use Git to *add* and *commit* the changes as you progress with your HW.
  - And don't forget to *push* your commits to your github repository; you should push the `Rmd` and `md` files, as well as the generated folder and files containing the plot images and other outputs.
  - Submit the link of your repository to bCourses. Do NOT submit any files (we will actually turn off the uploading files option).
  - No html files will be taken into account (no exceptions).
  - If you have questions/problems, don't hesitate to ask us for help in OH or in Piazza.
- 

## 2) Data (10 pts)

The data for this assignment involves the following five CSV files available inside the `data/` folder of the course github repo `stat133-spring-2018`.

- `andre-iguodala.csv`
- `draymond-green.csv`
- `kevin-durant.csv`
- `klay-thompson.csv`
- `stephen-curry.csv`

### 2.1) Download the data

- You will need to get a copy of the data files to your local repository.
- The instructions to download the data file should NOT be part of your report.

## 2.2) Data Dictionary (10 pts)

Create a data dictionary—using markdown syntax—in a separate text file: `data-dictionary.md`. Include names of the variables, and a short description. Most of the variable names are self-descriptive: e.g. `team_name`, `game_date`. However, depending on how much you know about basketball, some variables may be a bit cryptic. So here's a description for some of them:

- `period`: an NBA game is divided in 4 periods of 12 mins each. For example, a value for `period = 1` refers to the first period (the first 12 mins of the game).
- `minutes_remaining` and `seconds_remaining` have to do with the amount of time in minutes and seconds, respectively, that remained to be played in a given period.
- `shot_made_flag` indicates whether a shot was made (y) or missed (n).
- `action_type` has to do with the basketball moves used by players, either to pass by defenders to gain access to the basket, or to get a clean pass to a teammate to score a two pointer or three pointer.
- `shot_type` indicates whether a shot is a 2-point field goal, or a 3-point field goal.
- `shot_distance`: distance to the basket (measured in feet).
- `x` and `y` refer to the court coordinates (measured in inches) where a shot occurred .

If you are interested about the dimensions of an NBA basketball court visit these links:

<https://www.sportsknowhow.com/basketball/dimensions/nba-basketball-court-dimensions.html>

[http://www.sportscourtdimensions.com/wp-content/uploads/2015/02/nba\\_court\\_dimensions\\_h.png](http://www.sportscourtdimensions.com/wp-content/uploads/2015/02/nba_court_dimensions_h.png)

## 3) Data Preparation (20 pts)

The first stage of the assignment has to do with the so-called *data preparation* phase. The primary goal of this stage is to create a csv data file `shots-data.csv` that will contain the required variables to be used in the visualization phase.

All the R code to complete the data preparation stage must be written in an `.R` script file (do NOT confuse with an `Rmd` file). Name the R script file as `make-shots-data-script.R` and save it inside the `code/` folder. Include a header (but NOT a yaml header) in the file containing:

- title: short title
- description: a short description of what the script is about
- input(s): what are the inputs required by the script?
- output(s): what are the outputs created when running the script?

As mentioned above, the *raw* data for this assignment consists of five data CSV files (one for each player). To create a main/global table, write code in your R script to carry out these steps:

- Read in the five data sets, using relative file paths; the decision of the data types for each column is up to you.

```
# example: reading file with relative path
curry <- read.csv("../data/stephen-curry.csv", stringsAsFactors = FALSE)
```

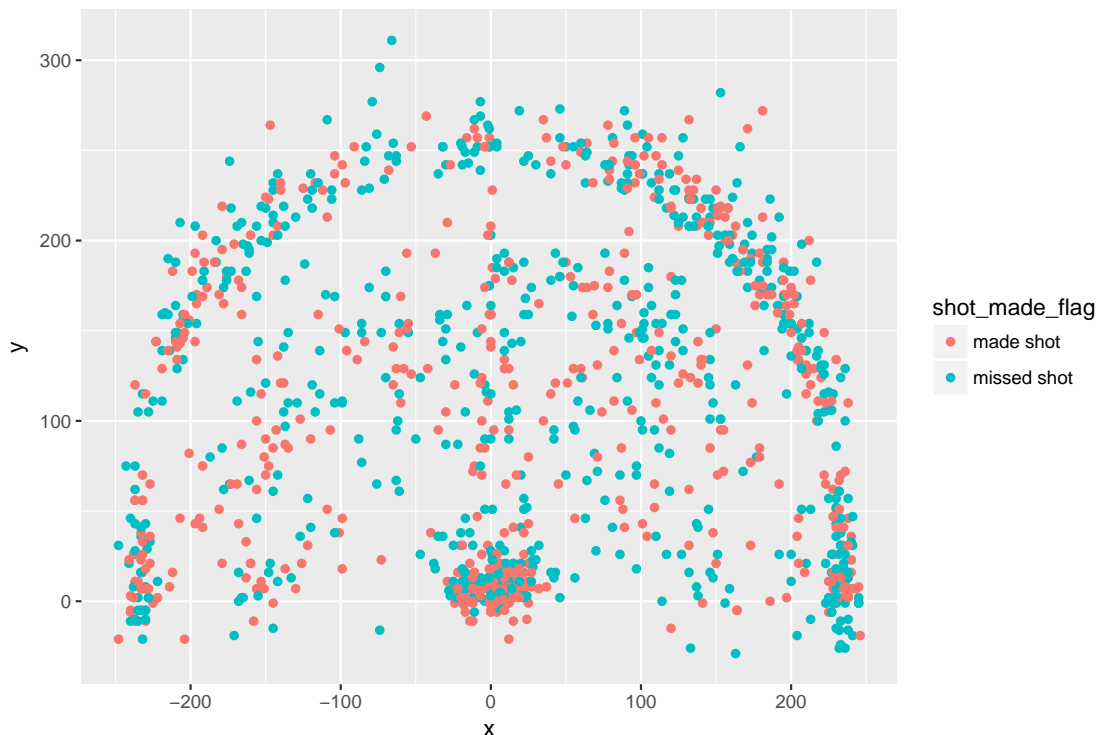
- Add a column `name` to each imported data frame, that contains the name of the corresponding player:
  - Andre Iguodala
  - Graymond Green
  - Kevin Durant
  - Klay Thompson
  - Stephen Curry
- Change the original values of `shot_made_flag` to more descriptive values: replace "n" with "missed shot", and "y" with "made shot". *Hint: you can use logical subsetting for this operation (no need to use programming structures that we haven't covered yet).*
- Add a column `minute` that contains the minute number where a shot occurred. For instance, if a shot took place during `period = 1` and `minutes_remaining = 8`, then this should correspond to a value `minute = 4`. Likewise, if a shot took place during `period = 4` and `minutes_remaining = 2` then this should correspond to a value `minute = 46`. *Hint: you can use logical subsetting for these operations (no need to use programming structures that we haven't covered yet).*
- Use `sink()` to send the `summary()` output of each imported data frame into individuals text files: `andre-iguodala-summary.txt`, `draymond-green-summary.txt`, etc. During each *sinking* operation, the produced summaries should be sent to the `output/` folder using relative paths.
- Use the row binding function `rbind()` to *stack* the tables into one single data frame (or tibble object).
- Export (i.e. write) the assembled table as a CSV file `shots-data.csv` inside the folder `data/`. Use a relative path for this operation.
- Use `sink()` to send the `summary()` output of the assembled table. Send this output to a text file named `shots-data-summary.txt` inside the `output/` folder. Use a relative path when exporting the R output.

## 4) Shot Charts (20 pts)

This part of the assignment has to do with the creation of shot charts. Write the code in an R script called `make-shot-charts-script.R`, and save it in the `code/` folder. Make sure you include a header with fields about title, description, inputs, and outputs.

Let's begin by describing some preliminary steps to make this type of charts. In this example, I will use the data for Klay Thompson. Suppose we have a `data.frame` (or `tibble`) `klay` with Klay Thompson's data. We can get a scatterplot of the shots using the `x` and `y` variables (i.e. coordinates of the shots):

```
# scatterplot
klay_scatterplot <- ggplot(data = klay) +
  geom_point(aes(x = x, y = y, color = shot_made_flag))
```



To make the shot chart more meaningful, and visually appealing, we will add a background with the image of an NBA basketball court. The JPG file with this image is available in the course github repository, inside the folder `images/`. You need to import the file `nba-court.jpg` to the `images/` folder of your HW directory. Likewise, you will need the following R packages: `"jpeg"` and `"grid"`.

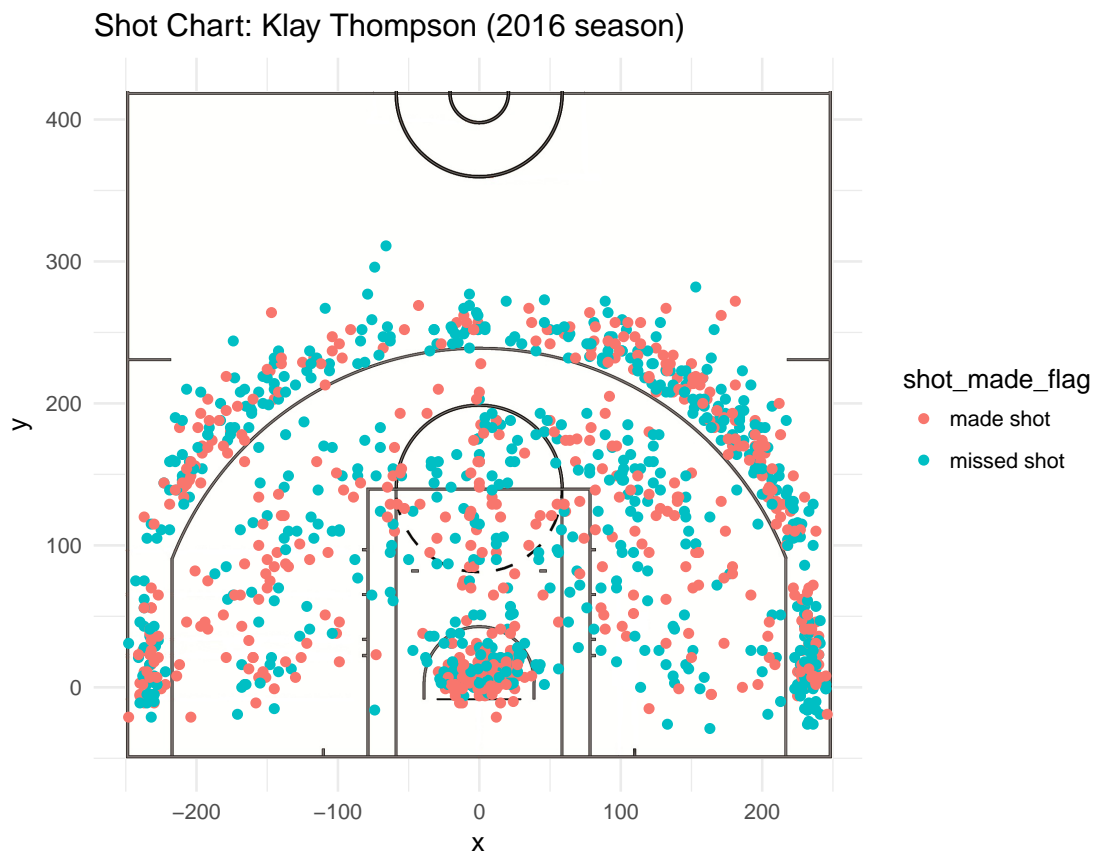
```
# court image (to be used as background of plot)
court_file <- "../images/nba-court.jpg"

# create raster object
court_image <- rasterGrob(
```

```
readJPEG(court_file),
width = unit(1, "npc"),
height = unit(1, "npc"))
```

What's happening? The function `readJPEG()` allows you to import the `.jpg` file. In turn, `rasterGrob()` takes the `.jpg` file and converts it into a raster graphical object. This object will be used as the background for the ggplot graphic:

```
# shot chart with court background
klay_shot_chart <- ggplot(data = klay) +
  annotation_custom(court_image, -250, 250, -50, 420) +
  geom_point(aes(x = x, y = y, color = shot_made_flag)) +
  ylim(-50, 420) +
  ggtitle('Shot Chart: Klay Thompson (2016 season)') +
  theme_minimal()
```



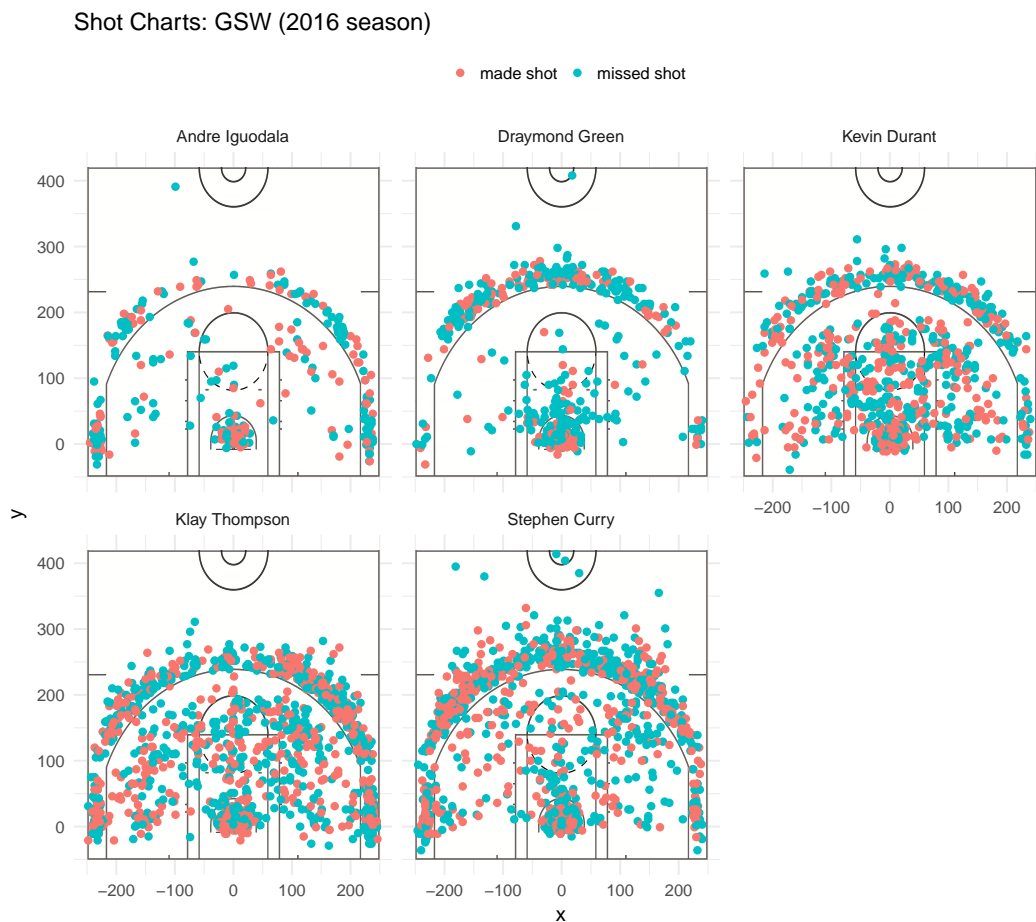
#### 4.1) Shot charts of each player (10 pts)

Create shot charts (with court backgrounds) for each player, and save the plots in PDF format, with dimensions `width = 6.5` and `height = 5` inches, inside the folder `images/`:

- `andre-iguodala-shot-chart.pdf`
- `draymond-green-shot-chart.pdf`
- `kevin-durant-shot-chart.pdf`
- `klay-thompson-shot-chart.pdf`
- `stephen-curry-shot-chart.pdf`

#### 4.2) Facetted Shot Chart (10 pts)

Create one graph, using facetting, to show all the shot charts in one image, similar to the one below. Save this image in PDF format as `gsw-shot-charts.pdf`, inside the folder `images/`. Specify image dimensions `width = 8` and `height = 7` inches.



## 5) Summary Tables (20 pts)

This and subsequent parts of the assignment should be included in your Rmd file (in the `report/` subdirectory). In your Rmd file include a global chunk option to specify the location of plots and graphics. This is done by setting the `fig.path` argument inside the `knitr::opts_chunk$set()` function. See image below:

```
1 ---
2 title: "HW 02 - Shot Charts"
3 author: "Your Name"
4 output: github_document
5 ---
6
7 ```{r setup, include=FALSE}
8 knitr::opts_chunk$set(echo = TRUE, fig.path = '../images/')
9 ```
10
```

Use "dplyr" functions to compute the following tables:

### 5.1) Total Shots by Player (10 pts)

Total number of shots (2PT and 3PT, both made and missed) by player, arranged in descending order. This tibble should have the format depicted in the diagram below.


player name	total shots
name	total
D	$X_D$
C	$X_C$
E	$X_E$
B	$X_B$
A	$X_A$

Values arranged by total shots  
(descending order)

### 5.2) Effective Shooting Percentage (10 pts)

Create the three tables listed below to summarize *Effective Shooting* percentages by player. All three tibbles should have the format of the following depicted diagram:





name	total	made	perc_made
A	$X_A$	$Y_A$	$\text{perc}_A$
D	$X_D$	$Y_D$	$\text{perc}_D$
E	$X_E$	$Y_E$	$\text{perc}_E$
B	$X_B$	$Y_B$	$\text{perc}_B$
C	$X_C$	$Y_C$	$\text{perc}_C$

*Values arranged by percentage (descending order)*

**Effective Shooting % by Player:** Overall (i.e. including 2PT and 3PT Field Goals) effective shooting percentage by player, arranged in descending order by percentage.

**2PT Effective Shooting % by Player:** 2 PT Field Goal effective shooting percentage by player, arranged in descending order by percentage.

**3PT Effective Shooting % by Player:** 3 PT Field Goal effective shooting percentage by player, arranged in descending order by percentage.

## 6) Shooting Distance (20 pts)

Consider the following question: the shorter the shooting distance, the higher the chance to successfully make a shot? Intuition and experience will suggest that YES. To confirm this, you will have to calculate, for each distance value, the proportion of made shots.

### 6.1) dplyr table (10 pts)

More precisely, use dplyr operations to obtain a tibble with two columns: `shot_distance` and `made_shot_prop`. The first row of the tibble should contain the value of distance = 0 ft, and the associated proportion of made shots (of all the five analyzed players). The second row should contain the value of distance = 1 ft, and the corresponding proportion of made shots; and so on.

Shot distance values		Proportion of made shots	
shot_distance		made_shot_prop	
0		$p_0$	
1		$p_1$	
2		$p_2$	
3		$p_3$	
...		...	

## 6.2) ggplot (10 pts)

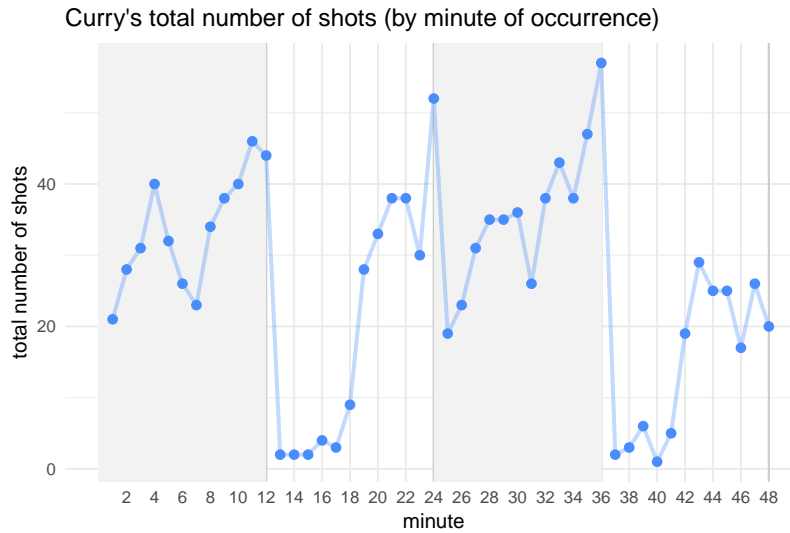
With the tibble created in 6.1), use `ggplot()` to make a scatterplot with the variables `shot_distance` and `made_shot_prop`. Use the x-axis for the shot distance, and the y-axis for the proportion of made shots.

- What do you observe?
- Can you confirm that the shorter the distance, the more effective the shots?
- Can you guesstimate a distance threshold beyond which the chance of making a successful shot is basically null?
- What distances tend to have a percentage (of making a shot) of 50% or more?

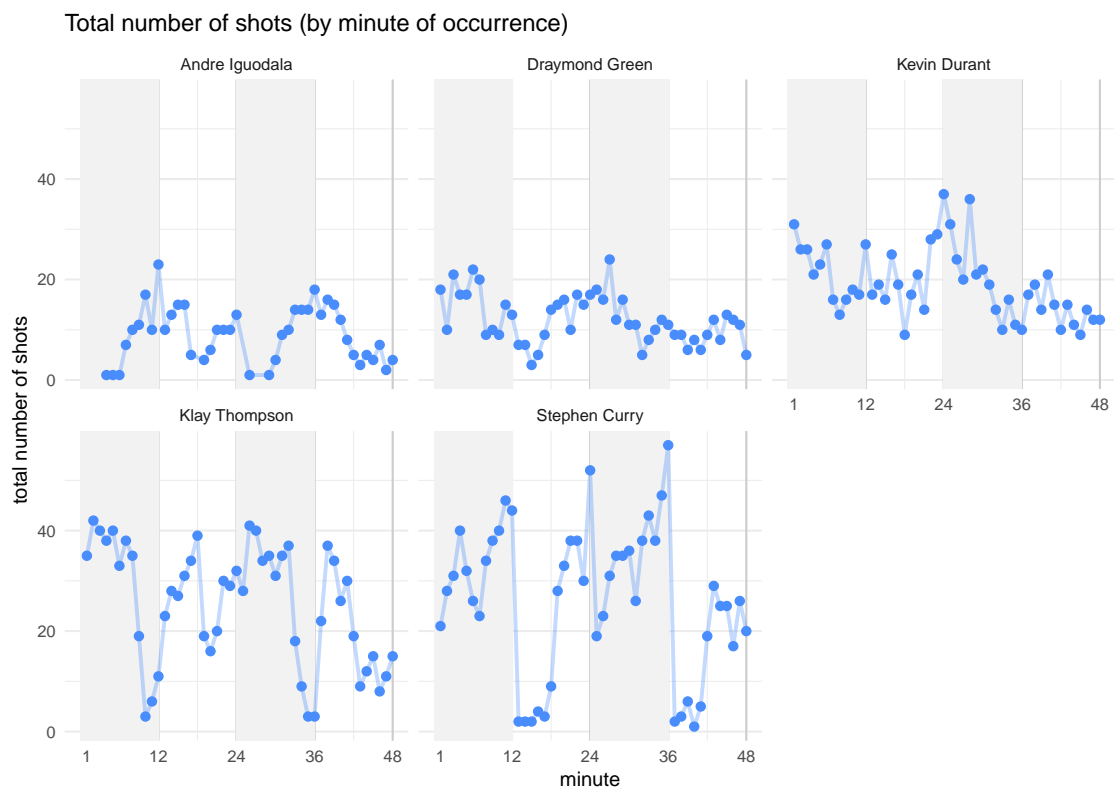
## 7) Total number of shots by minute of occurrence (10 pts)

The last part of the assignment involves looking at the *total number of shots (made and missed) by minute of occurrence*.

The following graph—just for illustration purposes—depicts Stephen Curry’s total number of shots (by minute of occurrence). The background of the plot has been adapted to reflect the four periods, each of 12 minutes. Notice how Curry’s shot activity peaks at the end of each period. Likewise, we can observe that Stephen Curry tends to rest at the beginning of the second and fourth periods.



Create a faceted graph of the total number of shots by minute of occurrence, depicting a facet per player, similar to the plot below. Try to write code that produce a plot as similar as possible.



You may need to use the following functions: `scale_x_continuous()` to modify scale of x-axis, `geom_rect()` to add background rectangles, `geom_path()` to connect the dots, `theme_minimal()` to get a minimal theme.