Week 6.3 - Using Regression Analysis to Test the Hot Hand

Using Regression Analysis to Test the "Hot Hand"

In this section, we will use regression analysis to test for the "hot hand."

Import useful libraries and the shot log data

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.3     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
```

```
## -- Conflicts ------------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
library(ggforce)
library(broom)


Shotlog = read.csv("~/Google Drive/Sports Analytics Moocs/MOOC 1 - Foundations of sports analytics/Week
Player_Stats = read.csv("~/Google Drive/Sports Analytics Moocs/MOOC 1 - Foundations of sports analytics,
Player_Shots = read.csv("~/Google Drive/Sports Analytics Moocs/MOOC 1 - Foundations of sports analytics,
head(Shotlog)
```

```
##   team_previous_shot player_position home_game location_x
## 1                                 SF       Yes         97
## 2            MISSED               SF       Yes        279
## 3            MISSED               SF       Yes         58
## 4            SCORED               SF       Yes        691
## 5            MISSED               SF       Yes        691
## 6            MISSED               SF       Yes        679
##   opponent_previous_shot home_team           shot_type points away_team
## 1                 SCORED       ATL    Pullup Jump Shot      2       WAS
## 2                 SCORED       ATL           Jump Shot      3       WAS
## 3                 SCORED       ATL  Cutting Layup Shot      2       WAS
## 4                 MISSED       ATL    Pullup Jump Shot      3       WAS
## 5                 MISSED       ATL    Pullup Jump Shot      2       WAS
```

```
## 6                 MISSED       ATL Step Back Jump Shot      3       WAS
##   location_y                      time       date  shoot_player
## 1        405 0 days 00:01:09.000000000 2016-10-27 Kent Bazemore
## 2        130 0 days 00:03:11.000000000 2016-10-27 Kent Bazemore
## 3        275 0 days 00:09:53.000000000 2016-10-27 Kent Bazemore
## 4        100 0 days 00:04:50.000000000 2016-10-27 Kent Bazemore
## 5        181 0 days 00:06:29.000000000 2016-10-27 Kent Bazemore
## 6        109 0 days 00:07:46.000000000 2016-10-27 Kent Bazemore
##   time_from_last_shot quarter current_shot_outcome current_shot_hit
## 1                  NA       1               MISSED                0
## 2                   4       1               MISSED                0
## 3                  30       2               MISSED                0
## 4                  39       3               SCORED                1
## 5                  20       3               MISSED                0
## 6                  21       3               MISSED                0
##   lag_shot_hit average_hit shot_count shot_per_game conse_shot_hit
## 1            0   0.4085873        722             7              0
## 2            0   0.4085873        722             7              0
## 3            0   0.4085873        722             7              0
## 4            0   0.4085873        722             7              0
## 5            1   0.4085873        722             7              0
## 6            0   0.4085873        722             7              0
```

Prediction Error

Let's create a variable that equals to the difference between the outcome of the shot and the average success rate. Since we typically use the average success rate to predict the outcome of the shot, this difference will capture the prediction error.

```
Shotlog$error = Shotlog$current_shot_hit - Shotlog$average_hit
Shotlog$lagerror = Shotlog$lag_shot_hit - Shotlog$average_hit
```

We can graph the outcome of the shots to see if there is any pattern over time in the variable.

We will look at LeBron James' performance during the regular season as an example.

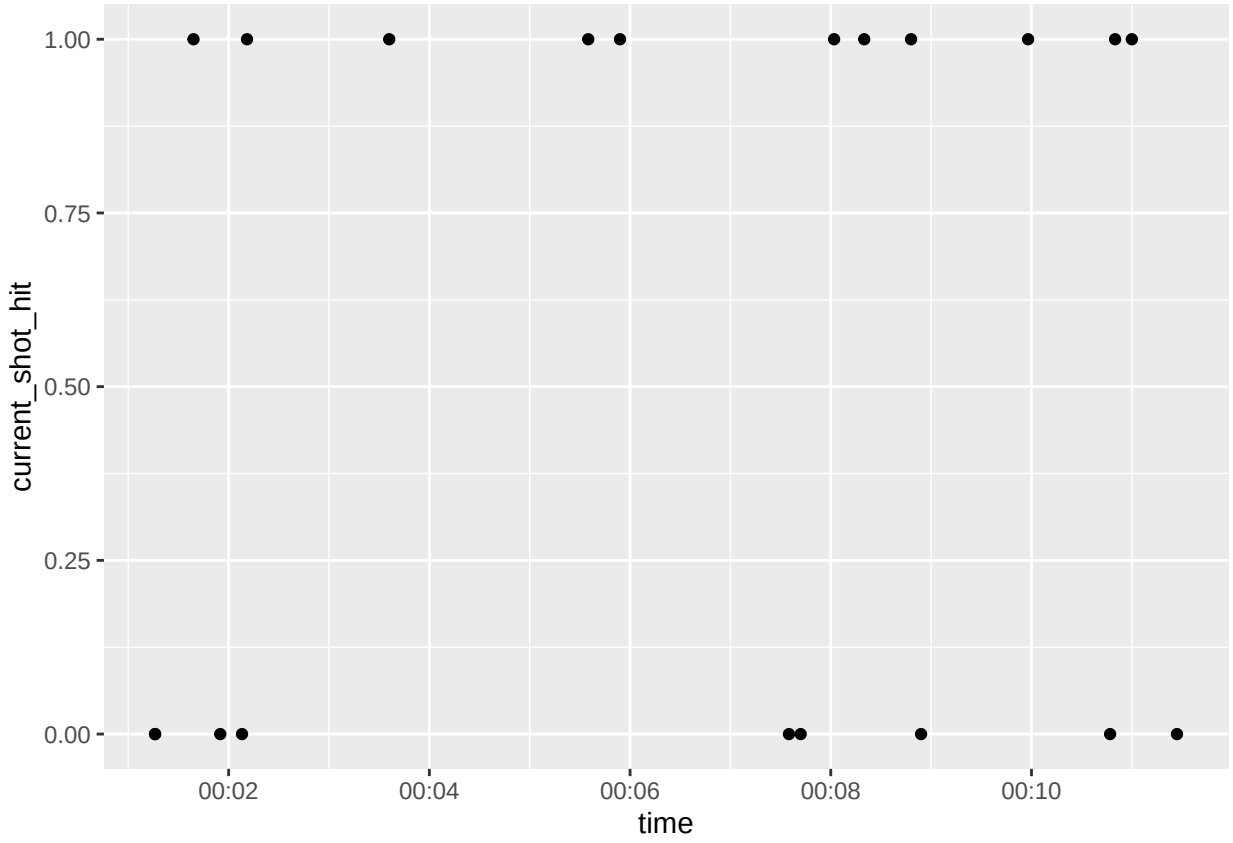
```
Shotlog$time = str_replace(Shotlog$time, '0 days ', '')

Shotlog$time = hms(Shotlog$time)
summary(Shotlog$time)
```

```
##                  Min.                1st Qu.                  Median
##                 "10S"               "3M 59S"               "6M 43S"
##                  Mean                3rd Qu.                    Max.
## "6M 41.1844940881482S"               "9M 28S"              "12M 0S"
```

We will first graph the outcome of LeBron James' shots in a single game on April 9th, 2017.

(To make this graph, we use a small trick. We will transform our time variable from a Period object into POSIXct. To convert time to a POSIXct, we will need to set an origin date in the as.POSIXct - the date used does not matter since we are only concerned with graphing the time portion.)

```
Shotlog %>% filter(shoot_player == 'LeBron James'
                   & date == '2017-04-09') %>%
  mutate(time = as.POSIXct(time, '%H:%M:%S',
                           origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point()
```



Let's create a graph of the outcomes of individual shots for Lebron James throughout the regular season. We will create a subgraph for each game he played.

We will first subset a dataset that includes only LeBron James' data.

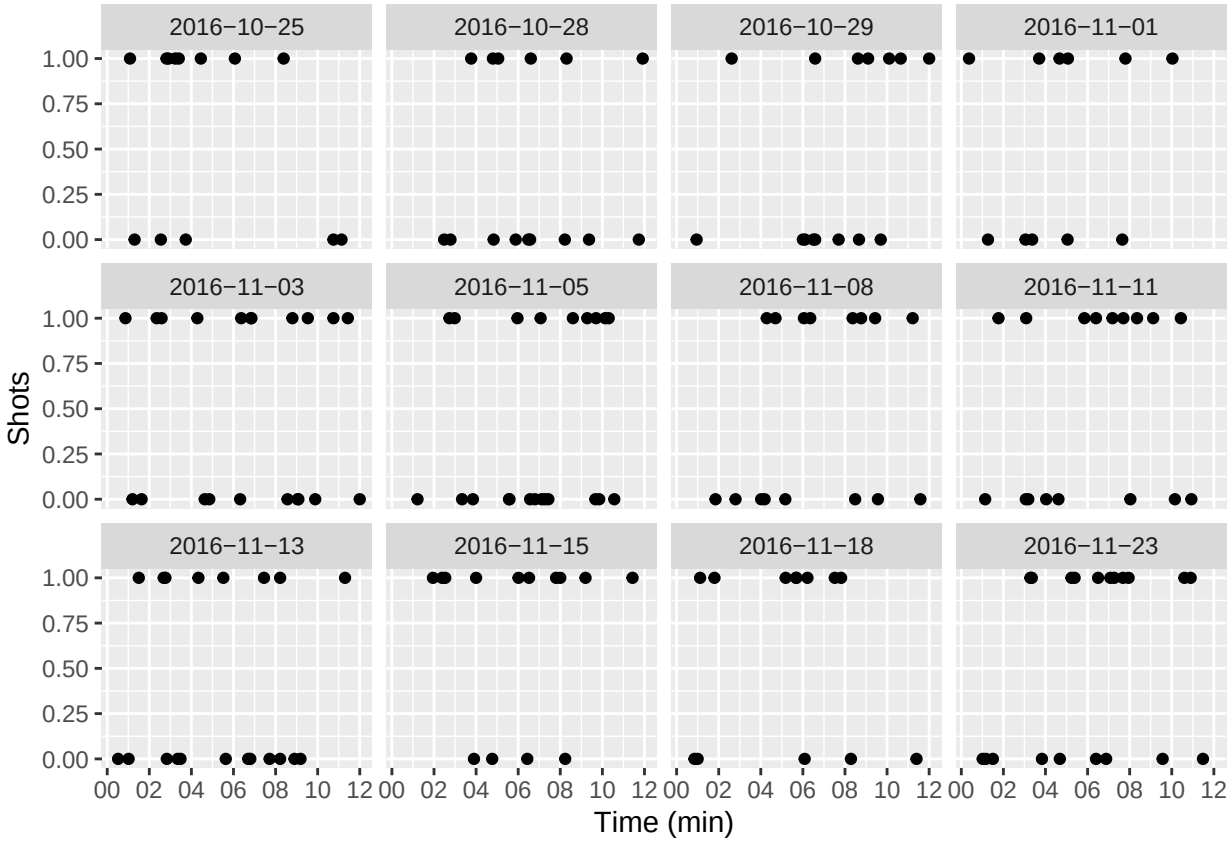
```
LeBron_James = Shotlog %>% filter(shoot_player == 'LeBron James')
head(LeBron_James)
```

```
##   team_previous_shot player_position home_game location_x
## 1             MISSED              SF       Yes        236
## 2            BLOCKED              SF       Yes        264
## 3             SCORED              SF       Yes         50
## 4             MISSED              SF       Yes         52
## 5             MISSED              SF       Yes         52
## 6             SCORED              SF       Yes         67
##   opponent_previous_shot home_team     shot_type points away_team location_y
## 1                 MISSED       CLE     Jump Shot      3       NYK         84
```

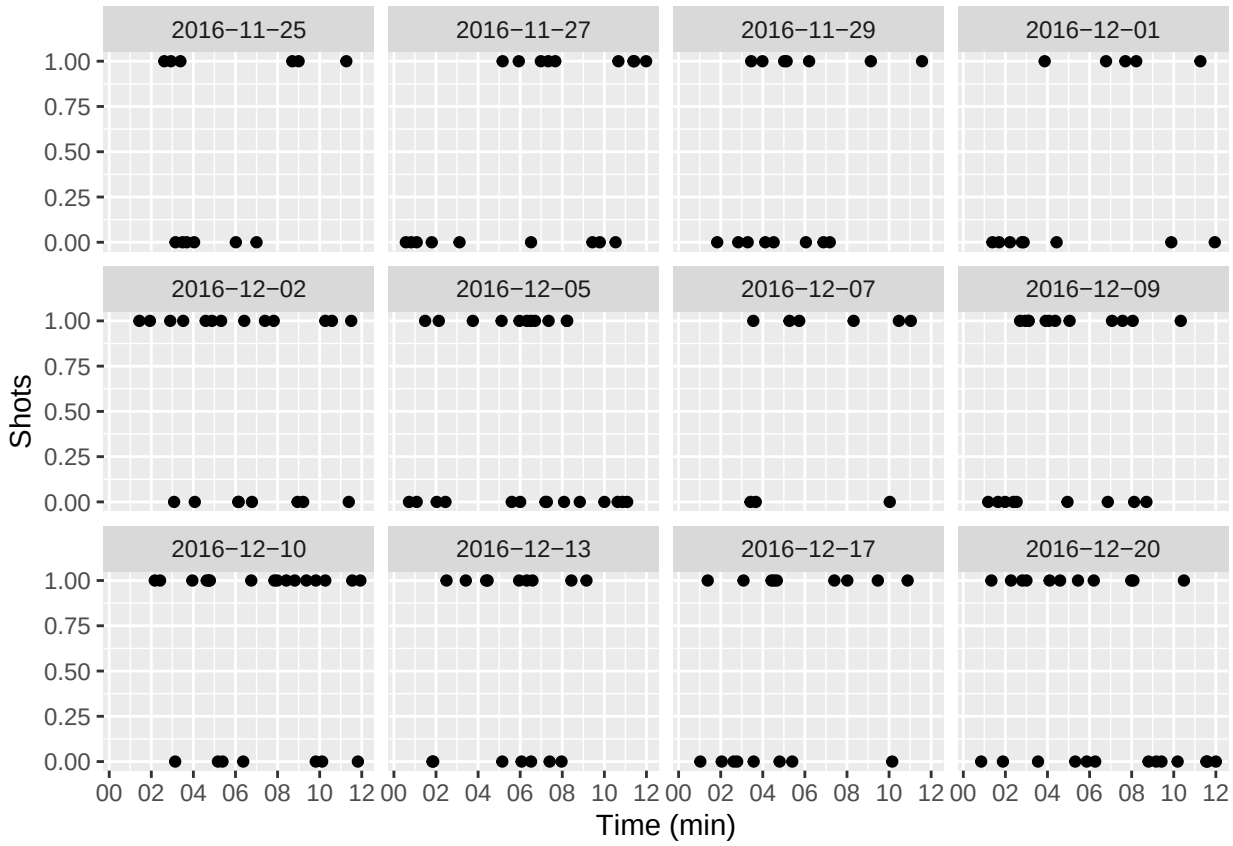
```
## 2                 SCORED       CLE     Jump Shot      3       NYK        383
## 3                 MISSED       CLE Running Layup      2       NYK        259
## 4                 SCORED       CLE  Putback Dunk      2       NYK        250
## 5                 MISSED       CLE          Dunk      2       NYK        250
## 6                 MISSED       CLE Running Layup      2       NYK        275
##     time       date shoot_player time_from_last_shot quarter
## 1 2M 33S 2016-10-25 LeBron James                   6       1
## 2 3M 44S 2016-10-25 LeBron James                  42       1
## 3  6M 4S 2016-10-25 LeBron James                  17       1
## 4 8M 23S 2016-10-25 LeBron James                   4       1
## 5  1M 5S 2016-10-25 LeBron James                  25       2
## 6 1M 18S 2016-10-25 LeBron James                  13       2
##   current_shot_outcome current_shot_hit lag_shot_hit average_hit shot_count
## 1               MISSED                0            0    0.547619       1344
## 2               MISSED                0            1    0.547619       1344
## 3               SCORED                1            1    0.547619       1344
## 4               SCORED                1            1    0.547619       1344
## 5               SCORED                1            1    0.547619       1344
## 6               MISSED                0            1    0.547619       1344
##   shot_per_game conse_shot_hit     error  lagerror
## 1            14              0 -0.547619 -0.547619
## 2            14              0 -0.547619  0.452381
## 3            14              1  0.452381  0.452381
## 4            14              1  0.452381  0.452381
## 5            14              1  0.452381  0.452381
## 6            14              0 -0.547619  0.452381
```

Now we can graph prediction error for LeBron James for all the games separately in the season. - We will be using the function "facet_wrap_paginate" from the library "ggforce" in order to print the graphs over multiple pages for viewing. If you were to use the regular "facet_wrap" function from ggplot2, the graph output will appear squished because the large number of graphs produced.

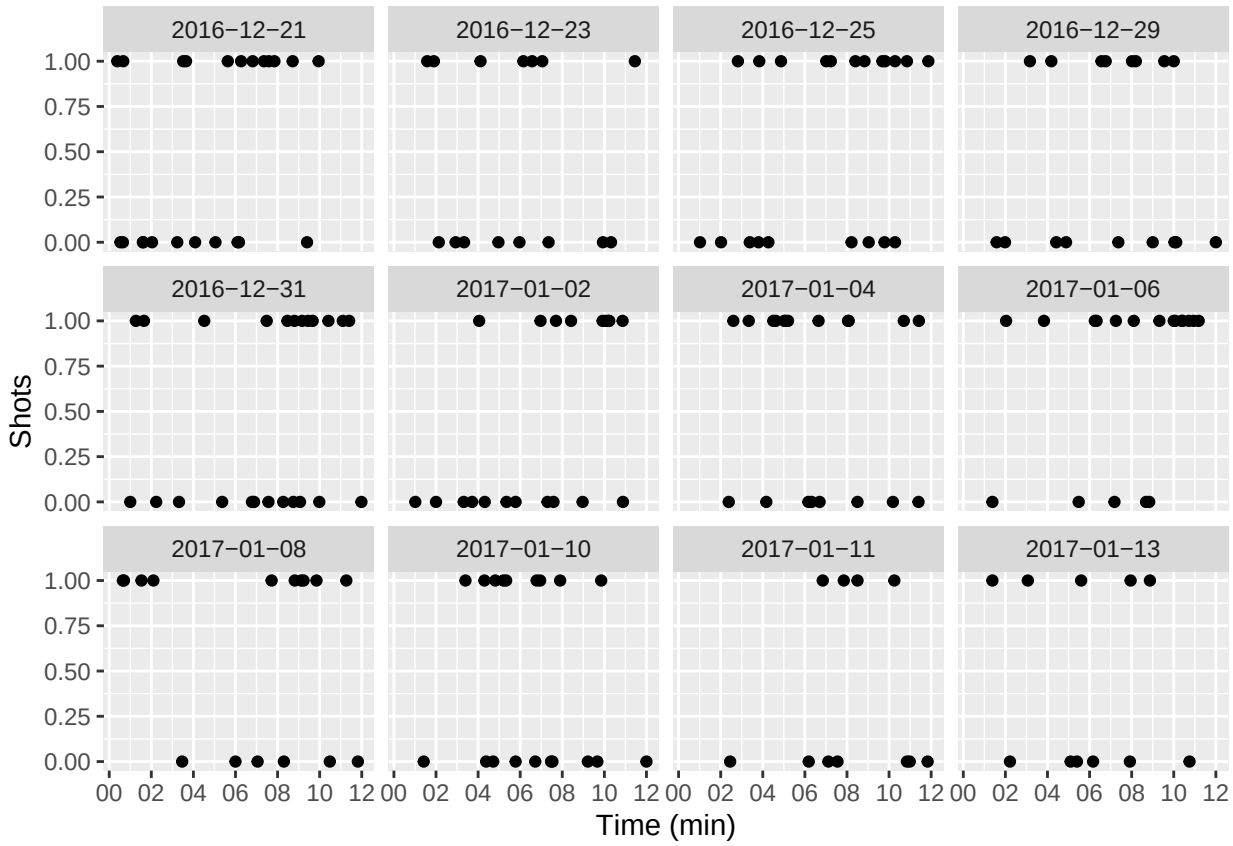
```
LeBron_James %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                                origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap_paginate(~date, ncol = 4, nrow = 3, page = 1)
```



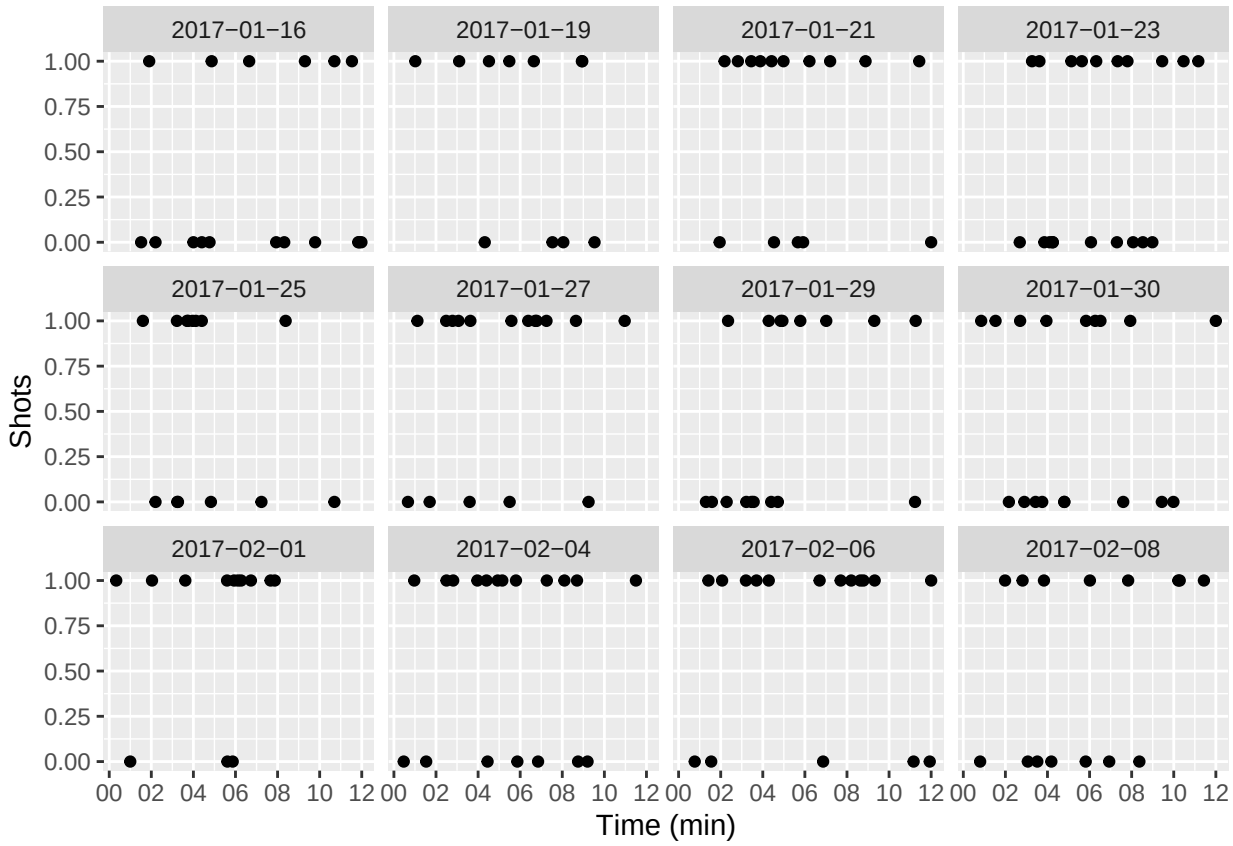

```
LeBron_James %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                                origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap_paginate(~date, ncol = 4, nrow = 3, page = 2)
```



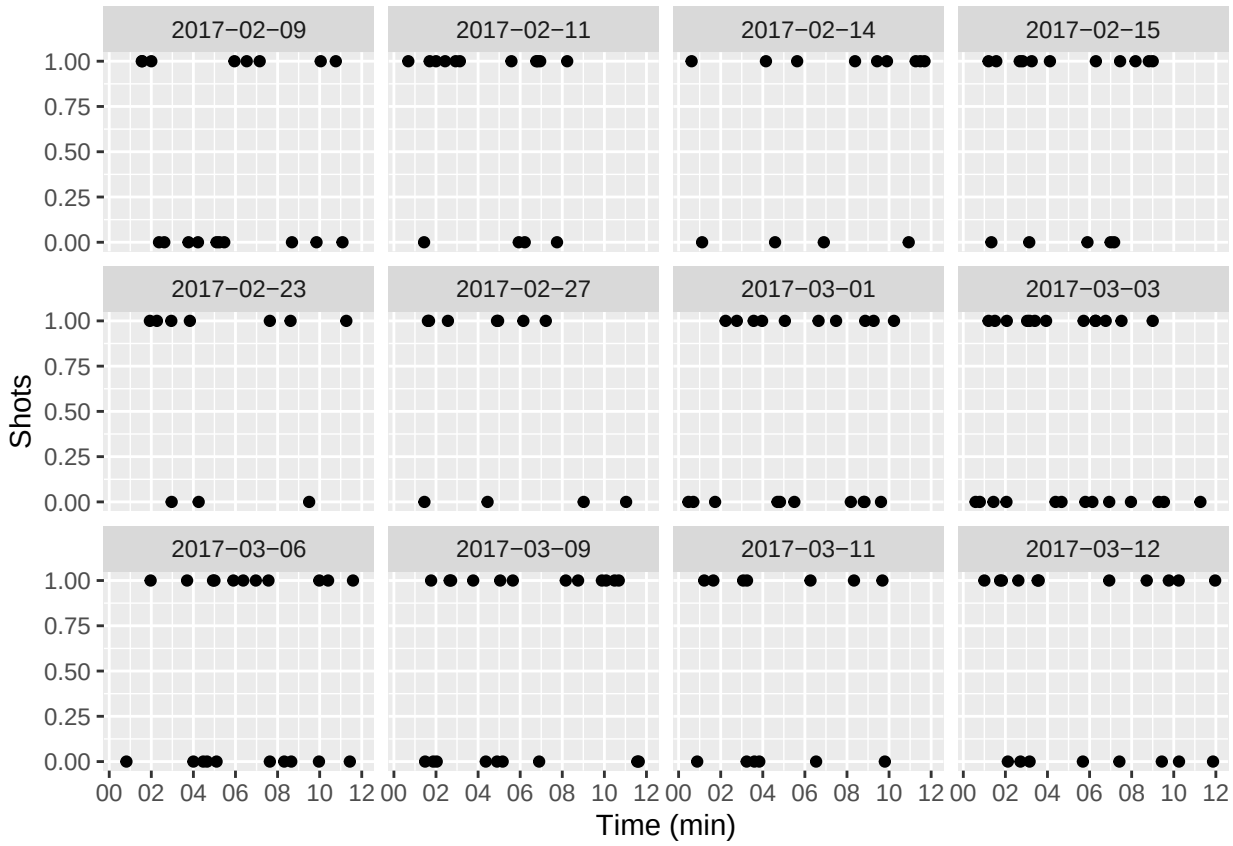

```
LeBron_James %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                            origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap_paginate(~date, ncol = 4, nrow = 3, page = 3)
```



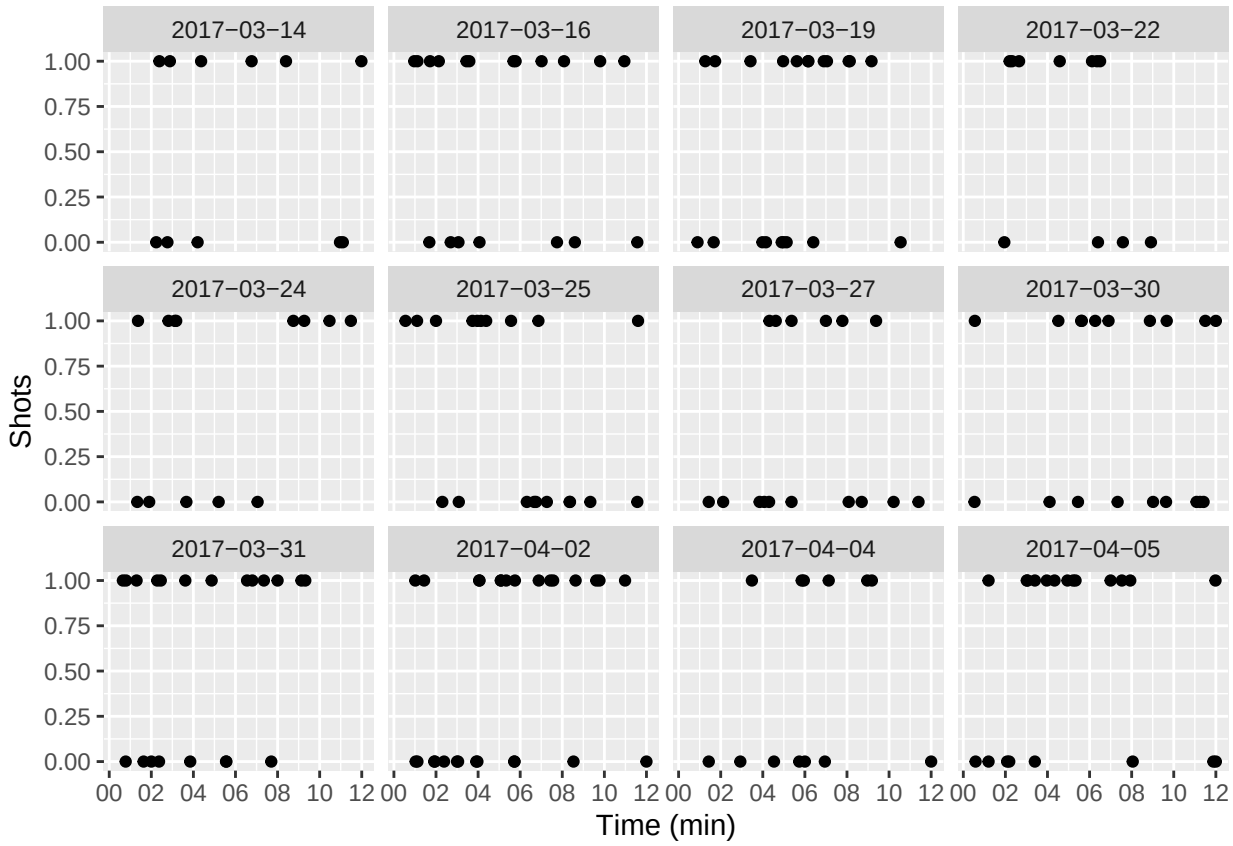

```
LeBron_James %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                                origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap_paginate(~date, ncol = 4, nrow = 3, page = 4)
```




```
LeBron_James %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                            origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap_paginate(~date, ncol = 4, nrow = 3, page = 5)
```

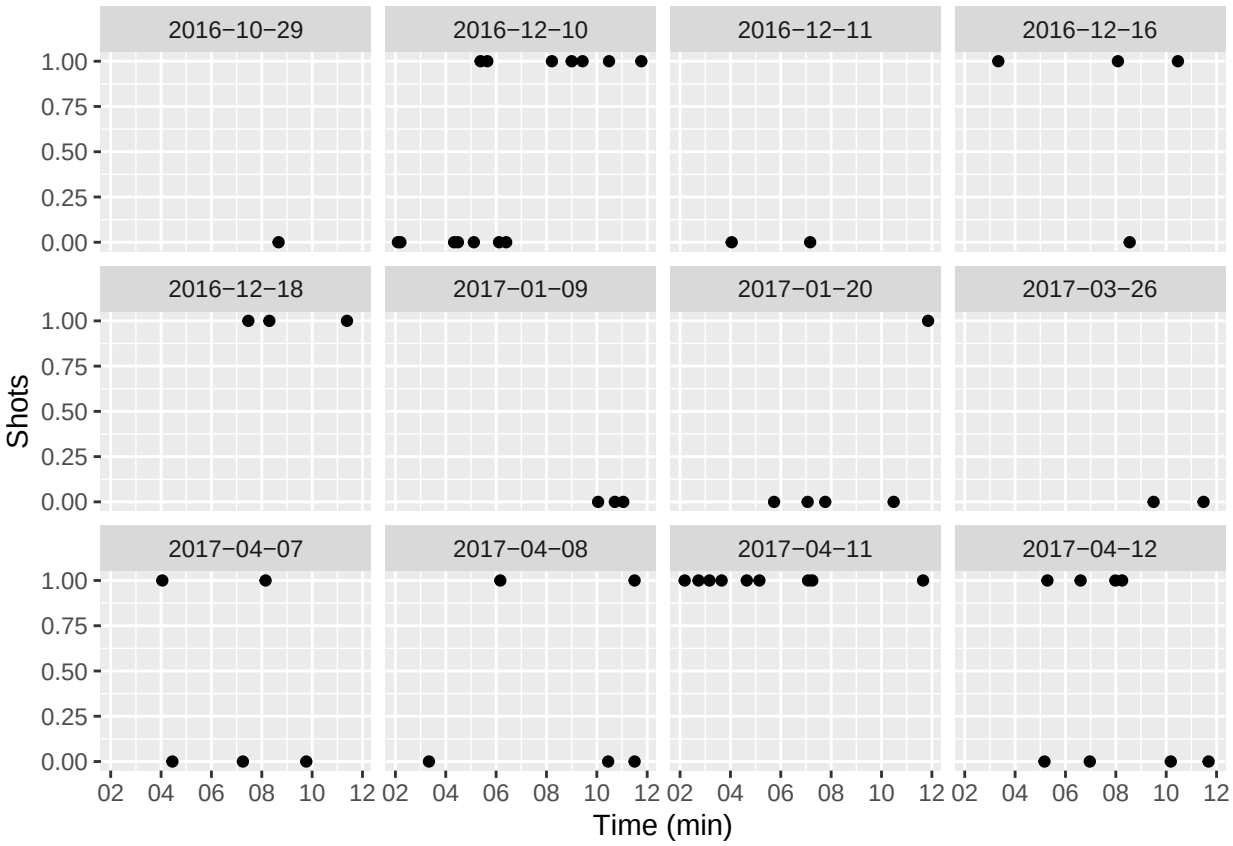



```
LeBron_James %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                            origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap_paginate(~date, ncol = 4, nrow = 3, page = 6)
```



We will do a similar exercise for the statistics of Cheick Diallo.

```
Cheick_Diallo = Shotlog %>% filter(shoot_player == 'Cheick Diallo')
Cheick_Diallo %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                         origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap(~date)
```

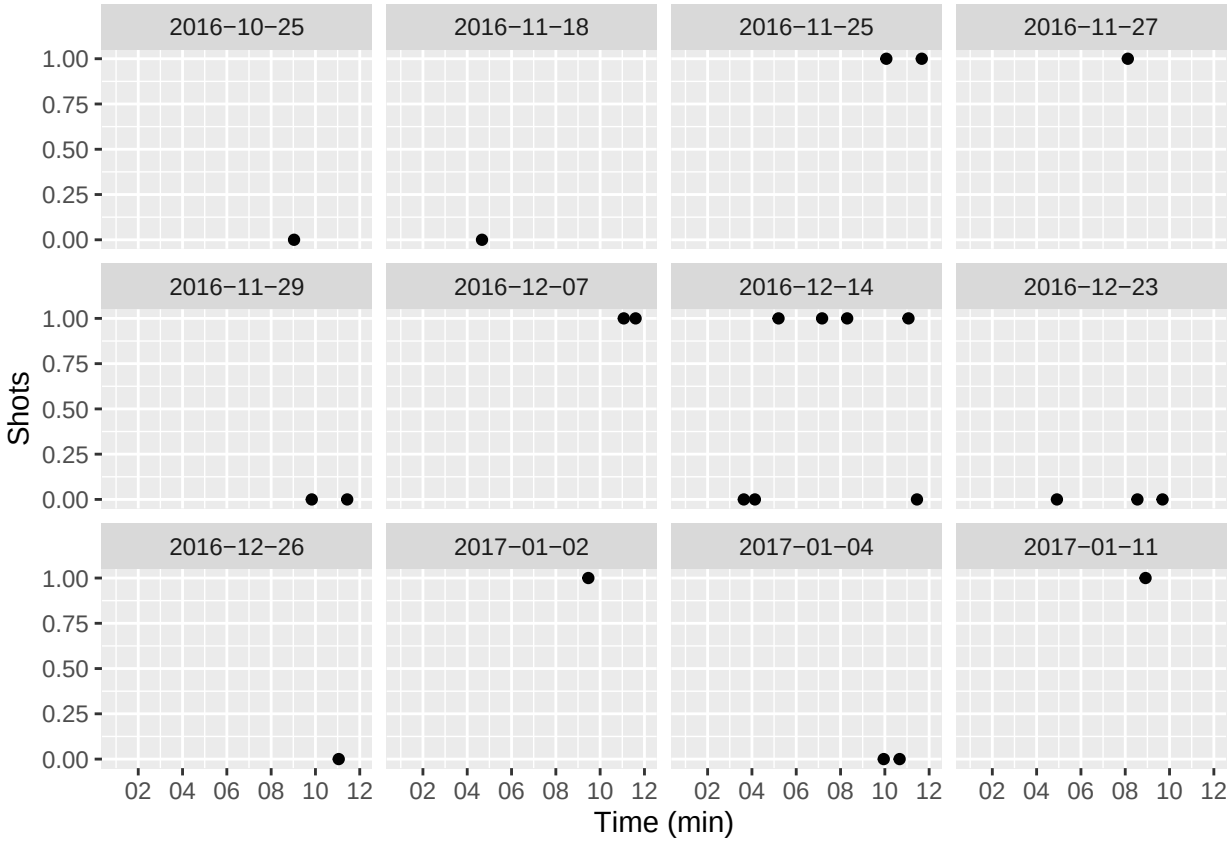

2016−10−29
2016−12−10
2016−12−11
2016−12−16
2016−12−18
2017−01−09
2017−01−20
2017−03−26
2017−04−07
2017−04−08
2017−04−11
2017−04−12
Shots
1.00
0.75
0.50
0.25
0.00
02
04
06
08
10
12
Time (min)

Self Test

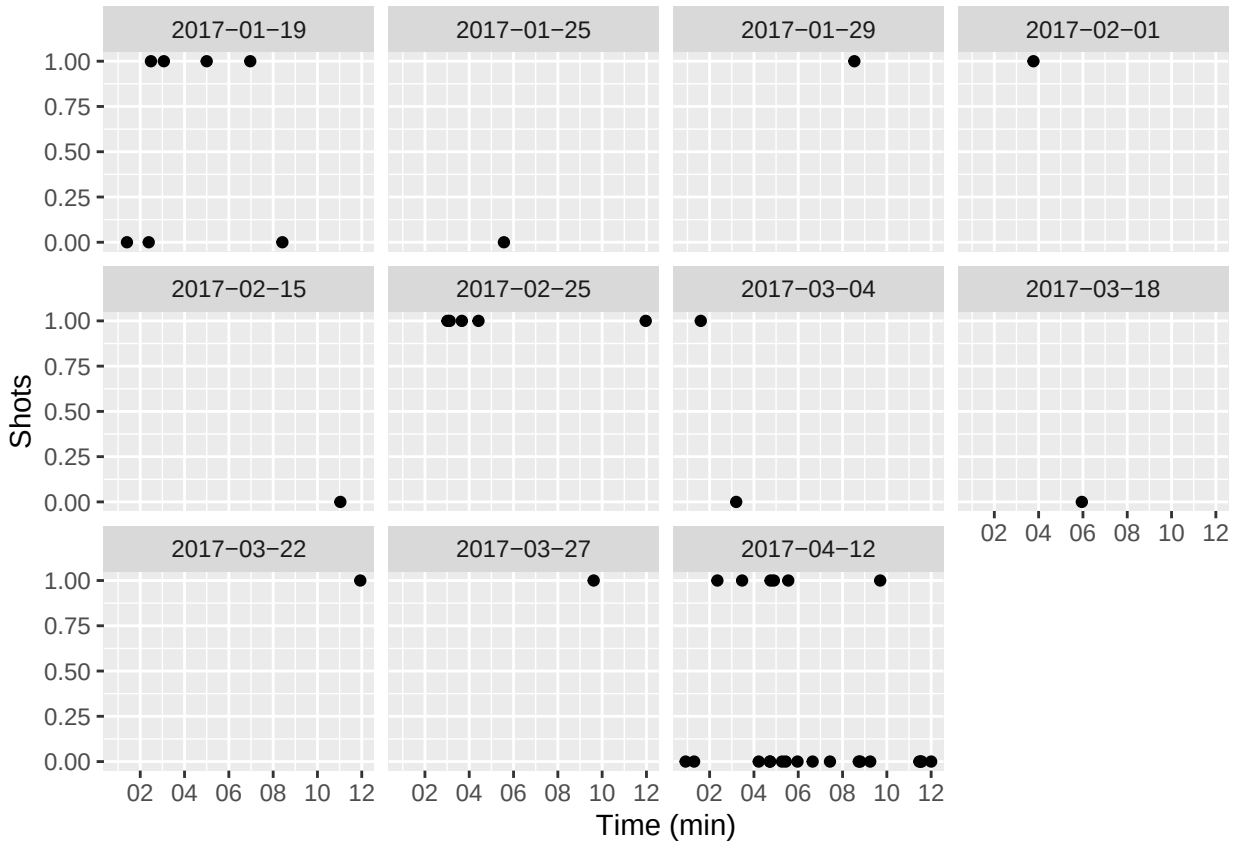
Graph the prediction error for James Jones

- Separate the shots by game
- Interpret your result

```
James_Jones = Shotlog %>% filter(shoot_player == 'James Jones')
James_Jones %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                           origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap_paginate(~date, ncol = 4, nrow = 3, page = 1)
```




```
James_Jones %>% mutate(time = as.POSIXct(time, '%H:%M:%S',
                           origin = '2020-01-20')) %>%
  ggplot(aes(x=time, y=current_shot_hit)) +
  geom_point() +
  scale_x_datetime(date_breaks = "2 min", date_labels = "%M") +
  labs(x = "Time (min)", y = "Shots") +
  facet_wrap_paginate(~date, ncol = 4, nrow = 3, page = 2)
```



Regression analysis on prediction error We will first run a simple regression of the prediction error of current period on the prediction error of previous period.

```
reg1 = lm(error ~ lagerror, data = Shotlog)
summary(reg1)
```

```
##
## Call:
## lm(formula = error ~ lagerror, data = Shotlog)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7985 -0.4526 -0.3862  0.5381  0.8978
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.0009415  0.0011507   0.818    0.413
## lagerror    -0.0121757  0.0023207  -5.247 1.55e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.495 on 185050 degrees of freedom
## Multiple R-squared:  0.0001487,  Adjusted R-squared:  0.0001433
## F-statistic: 27.53 on 1 and 185050 DF,  p-value: 1.552e-07
```

The estimated coefficient of the lagged error is statistically significant. However, the R-Squared for this regression is also zero. This means that our specified linear model is not a good fit for our data at all!

There are a lot of factors that may influence the success of shot, for example, the player's own skill as a

shooter, the type of the shot, the atmosphere of the stadium (whether it is home or away game), and whether it is at the beginning or towards the end of the game. Let's add these control variables in our regression.

```
reg2 = lm(error ~ lagerror+player_position+home_game+opponent_previous_shot
          +factor(points)+time_from_last_shot+factor(quarter), data = Shotlog)
summary(reg2)
```

```
##
## Call:
## lm(formula = error ~ lagerror + player_position + home_game +
##     opponent_previous_shot + factor(points) + time_from_last_shot +
##     factor(quarter), data = Shotlog)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8025 -0.4678 -0.3211  0.5063  0.9209
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -2.316e-02  3.896e-02  -0.594 0.552219
## lagerror                      -8.282e-03  2.320e-03  -3.570 0.000356 ***
## player_positionF               2.554e-02  6.407e-03   3.987 6.70e-05 ***
## player_positionG               2.724e-02  6.348e-03   4.291 1.78e-05 ***
## player_positionPF              1.917e-02  4.086e-03   4.693 2.70e-06 ***
## player_positionPG              2.652e-02  3.934e-03   6.742 1.57e-11 ***
## player_positionSF              3.283e-02  4.091e-03   8.027 1.01e-15 ***
## player_positionSG              3.594e-02  3.865e-03   9.297  < 2e-16 ***
## home_gameYes                   1.209e-02  2.297e-03   5.265 1.41e-07 ***
## opponent_previous_shotBLOCKED  8.080e-02  3.931e-02   2.055 0.039833 *
## opponent_previous_shotMISSED   6.520e-02  3.895e-02   1.674 0.094143 .
## opponent_previous_shotSCORED   5.294e-02  3.895e-02   1.359 0.174066
## factor(points)3               -1.207e-01  2.522e-03 -47.857  < 2e-16 ***
## time_from_last_shot           -7.412e-04  5.812e-05 -12.753  < 2e-16 ***
## factor(quarter)2               1.931e-03  3.213e-03   0.601 0.547914
## factor(quarter)3              -6.013e-03  3.217e-03  -1.869 0.061649 .
## factor(quarter)4              -3.876e-03  3.283e-03  -1.180 0.237808
## factor(quarter)5              -6.254e-02  1.569e-02  -3.986 6.73e-05 ***
## factor(quarter)6              -2.823e-02  4.530e-02  -0.623 0.533238
## factor(quarter)7              -1.727e-01  1.269e-01  -1.360 0.173746
## factor(quarter)8               6.163e-04  1.269e-01   0.005 0.996126
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4915 on 183153 degrees of freedom
##   (1878 observations deleted due to missingness)
## Multiple R-squared:  0.01423,    Adjusted R-squared:  0.01413
## F-statistic: 132.2 on 20 and 183153 DF,  p-value: < 2.2e-16
```

We can see that the R-squared is now increased to 0.014 which is still very small. The estimate on lagerror becomes statistically significant, but the magnitude of the estimate is 0.0082 which is still very small. And it is negative, meaning that the success of the previous shot would hurt the chance of the subsequent shot. This is contrary to what the hot hand predicts.

Weighted least squares regression

As we have seen, some players had a lot of shot per game while some just had a few. Different players may have different variations in their success rate in the shots. We can run a weighted least squared regression to address this problem.

Weighted least squares estimation weights the observations proportional to the reciprocal of the error variance of the observation. Thus weighted least squares can overcome the issue of non-constant variance.

We can use the "weights" paramter in the "lm" function to run the weighted least square regression weighted by the number of shot per game (weight=1/shot_per_game).

```
reg3 = lm(error ~ lagerror+player_position+home_game+
            opponent_previous_shot+points+time_from_last_shot+quarter,
          weights=1/Shotlog$shot_per_game,
          data = Shotlog)
summary(reg3)
```

```
##
## Call:
## lm(formula = error ~ lagerror + player_position + home_game +
##     opponent_previous_shot + points + time_from_last_shot + quarter,
##     data = Shotlog, weights = 1/Shotlog$shot_per_game)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -0.53664 -0.13119 -0.08548  0.14694  0.60436
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    2.144e-01  4.261e-02   5.030 4.90e-07 ***
## lagerror                      -6.821e-03  2.318e-03  -2.943 0.003250 **
## player_positionF               2.135e-02  5.909e-03   3.612 0.000303 ***
## player_positionG               2.649e-02  5.567e-03   4.758 1.95e-06 ***
## player_positionPF              1.763e-02  3.911e-03   4.508 6.55e-06 ***
## player_positionPG              2.238e-02  3.980e-03   5.624 1.87e-08 ***
## player_positionSF              3.295e-02  4.006e-03   8.225  < 2e-16 ***
## player_positionSG              3.358e-02  3.853e-03   8.714  < 2e-16 ***
## home_gameYes                   1.242e-02  2.287e-03   5.428 5.70e-08 ***
## opponent_previous_shotBLOCKED  8.588e-02  4.261e-02   2.016 0.043846 *
## opponent_previous_shotMISSED   7.148e-02  4.228e-02   1.691 0.090928 .
## opponent_previous_shotSCORED   5.813e-02  4.228e-02   1.375 0.169141
## points                        -1.231e-01  2.520e-03 -48.842  < 2e-16 ***
## time_from_last_shot           -7.467e-04  5.692e-05 -13.120  < 2e-16 ***
## quarter                       -7.554e-04  1.015e-03  -0.744 0.456887
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1593 on 183159 degrees of freedom
##   (1878 observations deleted due to missingness)
## Multiple R-squared:  0.01477,    Adjusted R-squared:  0.01469
## F-statistic: 196.1 on 14 and 183159 DF,  p-value: < 2.2e-16
```

From our summary statistics, some players exhibit a stream of the success while some don't. In our previous regressions, we are grouping all the players together. Let's see if we can find any effect if we look at individual

players.

Regression analysis on individual players

Run a regression of current error on lagged error for LeBron James.

```
reg_LeBron = lm(error ~ lagerror+home_game+opponent_previous_shot+factor(points)+
                  time_from_last_shot+factor(quarter),
          data = LeBron_James)
summary(reg_LeBron)
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     factor(points) + time_from_last_shot + factor(quarter), data = LeBron_James)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6822 -0.5736  0.3478  0.3931  0.8207
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   0.0614946  0.0863163   0.712   0.4763
## lagerror                     -0.0109936  0.0279322  -0.394   0.6940
## home_gameYes                  0.0290511  0.0278209   1.044   0.2966
## opponent_previous_shotMISSED  0.0013579  0.0810384   0.017   0.9866
## opponent_previous_shotSCORED  0.0068858  0.0811043   0.085   0.9324
## factor(points)3              -0.2395987  0.0322810  -7.422 2.13e-13 ***
## time_from_last_shot          -0.0012196  0.0006787  -1.797   0.0726 .
## factor(quarter)2              0.0378994  0.0391222   0.969   0.3329
## factor(quarter)3              0.0281348  0.0389585   0.722   0.4703
## factor(quarter)4              0.0415432  0.0408641   1.017   0.3095
## factor(quarter)5             -0.1573362  0.1866123  -0.843   0.3993
## factor(quarter)6             -0.1104128  0.2837551  -0.389   0.6973
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4867 on 1238 degrees of freedom
##   (20 observations deleted due to missingness)
## Multiple R-squared:  0.05224,    Adjusted R-squared:  0.04382
## F-statistic: 6.204 on 11 and 1238 DF,  p-value: 5.23e-10
```

Similarly, we can run a weighted least squares estimation on LeBron James' prediction error, weighted by the number of shot he made in each game.

```
reg_LeBron_wls = lm(error ~ lagerror+home_game+opponent_previous_shot+
                      points+time_from_last_shot+quarter,
                    weights = 1/LeBron_James$shot_per_game,
                    data = LeBron_James)
summary(reg_LeBron_wls)
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     points + time_from_last_shot + quarter, data = LeBron_James,
##     weights = 1/LeBron_James$shot_per_game)
```

```
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -0.19214 -0.12490  0.07443  0.09215  0.21606
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   0.6149328  0.1068745   5.754 1.10e-08 ***
## lagerror                     -0.0210241  0.0277531  -0.758   0.4489
## home_gameYes                  0.0266769  0.0276399   0.965   0.3347
## opponent_previous_shotMISSED -0.0298902  0.0791244  -0.378   0.7057
## opponent_previous_shotSCORED -0.0272411  0.0792220  -0.344   0.7310
## points                       -0.2577920  0.0320760  -8.037 2.13e-15 ***
## time_from_last_shot          -0.0013299  0.0006667  -1.995   0.0463 *
## quarter                       0.0126667  0.0126494   1.001   0.3168
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1134 on 1242 degrees of freedom
##   (20 observations deleted due to missingness)
## Multiple R-squared:  0.05777,    Adjusted R-squared:  0.05246
## F-statistic: 10.88 on 7 and 1242 DF,  p-value: 2.231e-13
```

We can also take a look back at LeBron James' autocorrelation coefficient.

```
Shotlog %>% filter(shoot_player == 'LeBron James') %>%
  summarise(auto_corr = cor(current_shot_hit, lag_shot_hit))
```

```
##     auto_corr
## 1 -0.02075217
```

The autocorrelation coefficient between the outcomes of the current shot and the previous shot for LeBron James is very small. And the autocorrelation coefficient is indeed very close to our estimates on the lagged error in the weighted least squares estimation.

We can do a similar exercise for James Jones. We will start with an ordinary least square regression.

```
reg_Jones = lm(error ~ lagerror+home_game+opponent_previous_shot+factor(points)+
                     time_from_last_shot+factor(quarter),
                   data = James_Jones)
summary(reg_Jones)
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     factor(points) + time_from_last_shot + factor(quarter), data = James_Jones)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8170 -0.3967 -0.1789  0.4849  0.8402
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   0.4777085  0.3876796   1.232    0.223
## lagerror                      0.1791685  0.1338054   1.339    0.186
## home_gameYes                 -0.1674516  0.1454617  -1.151    0.254
## opponent_previous_shotMISSED  0.0756719  0.2764125   0.274    0.785
```

```
## opponent_previous_shotSCORED -0.0689440  0.2712602  -0.254    0.800
## factor(points)3              -0.1525209  0.1336025  -1.142    0.258
## time_from_last_shot          -0.0005465  0.0047930  -0.114    0.910
## factor(quarter)2             -0.0975403  0.2763865  -0.353    0.725
## factor(quarter)3             -0.2724270  0.2644636  -1.030    0.307
## factor(quarter)4             -0.2934602  0.2389885  -1.228    0.224
##
## Residual standard error: 0.5076 on 58 degrees of freedom
## Multiple R-squared:  0.1141, Adjusted R-squared:  -0.02337
## F-statistic:  0.83 on 9 and 58 DF,  p-value: 0.5914
```

We will also run a weighted least squares estimation on Jones' statistics. Weight=1/shot_per_game.

```
reg_Jones_wls = lm(error ~ lagerror+home_game+opponent_previous_shot+points+
                     time_from_last_shot+quarter,
                weights=1/James_Jones$shot_per_game,
                data = James_Jones)
summary(reg_Jones_wls)
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     points + time_from_last_shot + quarter, data = James_Jones,
##     weights = 1/James_Jones$shot_per_game)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -0.41301 -0.13555 -0.04261  0.16307  0.43604
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   0.793210   0.616284   1.287   0.2030
## lagerror                      0.066854   0.127903   0.523   0.6031
## home_gameYes                 -0.158556   0.128613  -1.233   0.2225
## opponent_previous_shotMISSED  0.372607   0.288281   1.293   0.2011
## opponent_previous_shotSCORED  0.064940   0.283548   0.229   0.8196
## points                       -0.069878   0.133845  -0.522   0.6035
## time_from_last_shot          -0.001366   0.004644  -0.294   0.7696
## quarter                      -0.176049   0.073458  -2.397   0.0197 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2206 on 60 degrees of freedom
## Multiple R-squared:  0.1725, Adjusted R-squared:  0.07597
## F-statistic: 1.787 on 7 and 60 DF,  p-value: 0.1066
```

Self Test

Use regression analysis to test "hot hand" for Cheick Diallo 1. Run an ordinary least square regression of current error on lagged error for Cheick Diallo. 2. Run a weighted least sqaure regression of current error on lagged error for Cheick Diallo, weight=1/shot_per_game. 3. Interpret your regression results.

```
reg_Diallo = lm(error ~ lagerror+home_game+opponent_previous_shot+
                  points+time_from_last_shot+quarter,
          data = Cheick_Diallo)
summary(reg_Diallo)
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     points + time_from_last_shot + quarter, data = Cheick_Diallo)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7446 -0.4368  0.2526  0.3764  0.7875
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   1.246310   1.222600   1.019    0.313
## lagerror                      0.217853   0.147744   1.475    0.146
## home_gameYes                 -0.228570   0.258978  -0.883    0.381
## opponent_previous_shotMISSED  0.467097   0.552457   0.845    0.402
## opponent_previous_shotSCORED  0.348938   0.553808   0.630    0.531
## points                       -0.751518   0.532462  -1.411    0.164
## time_from_last_shot          -0.001420   0.003453  -0.411    0.683
## quarter                      -0.004344   0.080917  -0.054    0.957
##
## Residual standard error: 0.5015 on 53 degrees of freedom
## Multiple R-squared:  0.1256, Adjusted R-squared:  0.01006
## F-statistic: 1.087 on 7 and 53 DF,  p-value: 0.3848
```

```
reg_Diallo_wls = lm(error ~ lagerror+home_game+opponent_previous_shot+
                      points+time_from_last_shot+quarter,
         weights=1/Cheick_Diallo$shot_per_game,
         data = Cheick_Diallo)
summary(reg_Diallo_wls)
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     points + time_from_last_shot + quarter, data = Cheick_Diallo,
##     weights = 1/Cheick_Diallo$shot_per_game)
##
## Weighted Residuals:
##     Min      1Q  Median      3Q     Max
## -0.2748 -0.1567  0.0819  0.1584  0.3249
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  0.7288090  1.1306237   0.645    0.522
## lagerror                     0.2464435  0.1467754   1.679    0.099 .
## home_gameYes                -0.2325878  0.2506144  -0.928    0.358
## opponent_previous_shotMISSED 0.2073172  0.7965132   0.260    0.796
## opponent_previous_shotSCORED 0.3113405  0.7978365   0.390    0.698
## points                      -0.5705573  0.3789959  -1.505    0.138
## time_from_last_shot         -0.0004838  0.0038621  -0.125    0.901
## quarter                      0.0455784  0.0837867   0.544    0.589
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1951 on 53 degrees of freedom
## Multiple R-squared:  0.119,  Adjusted R-squared:  0.002681
```

```
## F-statistic: 1.023 on 7 and 53 DF,  p-value: 0.4261
```

More generally, we can define functions to run regressions for each individual player.

- Define a function to run ordinary least square regression by player.

```
reg_player = function(player) {
  Shotlog_player = Shotlog %>% filter(shoot_player==player)
  reg_player = lm(error ~ lagerror+home_game+opponent_previous_shot+
                    points+time_from_last_shot+quarter,
                  data = Shotlog_player)
    return(summary(reg_player))
}
```

```
reg_player('Russell Westbrook')
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     points + time_from_last_shot + quarter, data = Shotlog_player)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.4923 -0.4519 -0.3325  0.5418  0.7330
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -0.0628930  0.4982701  -0.126    0.900
## lagerror                        0.0086559  0.0231425   0.374    0.708
## home_gameYes                    0.0206659  0.0228645   0.904    0.366
## opponent_previous_shotBLOCKED   0.2993806  0.4953945   0.604    0.546
## opponent_previous_shotMISSED    0.3414694  0.4925939   0.693    0.488
## opponent_previous_shotSCORED    0.3382484  0.4925408   0.687    0.492
## points                         -0.1220405  0.0249171  -4.898 1.05e-06 ***
## time_from_last_shot            -0.0003947  0.0005906  -0.668    0.504
## quarter                         0.0022827  0.0100793   0.226    0.821
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4914 on 1846 degrees of freedom
##   (4 observations deleted due to missingness)
## Multiple R-squared:  0.0145, Adjusted R-squared:  0.01023
## F-statistic: 3.394 on 8 and 1846 DF,  p-value: 0.0007071
```

- Define a function to run weighted least square regression by player.

```
reg_wls_player = function(player) {
  Shotlog_player = Shotlog %>% filter(shoot_player==player)
  reg_wls_player = lm(error ~ lagerror+home_game+opponent_previous_shot+
                    points+time_from_last_shot+quarter,
                  weights=1/Shotlog_player$shot_per_game,
                  data = Shotlog_player)
    return(summary(reg_wls_player))
}
```

```
reg_wls_player('Russell Westbrook')
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     points + time_from_last_shot + quarter, data = Shotlog_player,
##     weights = 1/Shotlog_player$shot_per_game)
##
## Weighted Residuals:
##      Min       1Q   Median       3Q      Max
## -0.14880 -0.08684 -0.06322  0.10862  0.20837
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  -0.0745934  0.5175378  -0.144    0.885
## lagerror                      0.0252187  0.0231492   1.089    0.276
## home_gameYes                  0.0295759  0.0228642   1.294    0.196
## opponent_previous_shotBLOCKED  0.3577094  0.5147352   0.695    0.487
## opponent_previous_shotMISSED   0.3444312  0.5120742   0.673    0.501
## opponent_previous_shotSCORED   0.3479418  0.5120348   0.680    0.497
## points                       -0.1200846  0.0249449  -4.814  1.6e-06 ***
## time_from_last_shot          -0.0001410  0.0005854  -0.241    0.810
## quarter                      -0.0034019  0.0101946  -0.334    0.739
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1002 on 1846 degrees of freedom
##   (4 observations deleted due to missingness)
## Multiple R-squared:  0.01477,    Adjusted R-squared:  0.0105
## F-statistic: 3.459 on 8 and 1846 DF,  p-value: 0.0005771
```

We can extract estimated coefficient on the lagged error for each player.

- Create a list of unique player names

```
player_list = unique(Shotlog$shoot_player)
```

```
player_list[1]
```

```
## [1] "Kent Bazemore"
```

- Run weighted least squares regression for each player by specifying "shoot_play==player_list[index]"

```
Shotlog_player=Shotlog %>% filter(shoot_player==player_list[1])

reg_player=lm(error ~ lagerror+home_game+opponent_previous_shot+points+
                time_from_last_shot+quarter,
              weights=1/Shotlog_player$shot_per_game,
              data= Shotlog_player)
summary(reg_player)
```

```
##
## Call:
## lm(formula = error ~ lagerror + home_game + opponent_previous_shot +
##     points + time_from_last_shot + quarter, data = Shotlog_player,
##     weights = 1/Shotlog_player$shot_per_game)
##
## Weighted Residuals:
```

```
##      Min       1Q   Median       3Q      Max
## -0.27075 -0.12395 -0.08697  0.16842  0.30533
##
## Coefficients:
##                                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -2.171e-02  5.641e-01  -0.038  0.96931
## lagerror                       2.709e-05  3.933e-02   0.001  0.99945
## home_gameYes                   1.469e-02  3.848e-02   0.382  0.70269
## opponent_previous_shotBLOCKED  3.784e-01  5.599e-01   0.676  0.49935
## opponent_previous_shotMISSED   3.482e-01  5.520e-01   0.631  0.52841
## opponent_previous_shotSCORED   2.883e-01  5.522e-01   0.522  0.60172
## points                        -1.283e-01  3.929e-02  -3.266  0.00115 **
## time_from_last_shot           -1.235e-03  8.794e-04  -1.405  0.16066
## quarter                        8.080e-03  1.741e-02   0.464  0.64267
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1524 on 632 degrees of freedom
##   (8 observations deleted due to missingness)
## Multiple R-squared:  0.02366,    Adjusted R-squared:  0.0113
## F-statistic: 1.915 on 8 and 632 DF,  p-value: 0.05527
```

- Extract the estimated coefficients, along with the p-value and t-statistics of the estimates and store them in a dataframe
- We can do this using the "tidy" function from the "broom" package to get a tabular data representation

```
tidy(reg_player)
```

```
## # A tibble: 9 x 5
##   term                           estimate std.error statistic p.value
##   <chr>                             <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)                    -0.0217     0.564    -0.0385   0.969
## 2 lagerror                        0.0000271  0.0393    0.000689 0.999
## 3 home_gameYes                    0.0147     0.0385    0.382    0.703
## 4 opponent_previous_shotBLOCKED   0.378      0.560     0.676    0.499
## 5 opponent_previous_shotMISSED    0.348      0.552     0.631    0.528
## 6 opponent_previous_shotSCORED    0.288      0.552     0.522    0.602
## 7 points                         -0.128      0.0393   -3.27     0.00115
## 8 time_from_last_shot            -0.00124    0.000879 -1.40     0.161
## 9 quarter                         0.00808    0.0174    0.464    0.643
```

- Write a loop to extract regression outputs for each player and store them in a dataframe. We need to exclude several players who we are unable to run the regression for.

```
i = 1
Player_Results = data.frame(shoot_player = character(),
                            Coef = double(),
                            T_Statistics = double(),
                            P_Value = double())

players_exc = c('Demetrius Jackson', 'Walter Tavares', 'Mike Tobey',
                'Larry Sanders', 'Ben Bentil', 'Alonzo Gee',
                'Michael Gbinije', 'Chinanu Onuaku',
                'Isaiah Taylor', 'Brice Johnson', 'Axel Toupane',
                'John Lucas III', 'Jarrett Jack', 'Josh Huestis',
                'Josh Huestis', 'Arinze Onuaku', 'Patricio Garino',
```

```
                'Jerryd Bayless', 'Justin Harper', 'John Jenkins',
                'Jarell Eddie','Jordan Farmar')
player_list = player_list[!(player_list %in% players_exc)]

while(i <= length(player_list)){
  skip_to_next <- FALSE
  Shotlog_player = Shotlog %>% filter(shoot_player==player_list[i])

  reg_player = lm(error ~ lagerror+home_game+opponent_previous_shot+
    points+time_from_last_shot+quarter,
    weights = 1/Shotlog_player$shot_per_game,
    data = Shotlog_player)
  RegOutput = tidy(reg_player)

  LagErr = RegOutput %>% filter(term == 'lagerror') %>%
    mutate(shoot_player = player_list[i]) %>%
    rename(Coef = estimate, T_Statistics = statistic, P_Value = p.value) %>%
    select(shoot_player, Coef, T_Statistics, P_Value)
  Player_Results = rbind(Player_Results, LagErr)
  i = i + 1
}
```

```
## Warning in summary.lm(x): essentially perfect fit: summary may be unreliable

## Warning in summary.lm(x): essentially perfect fit: summary may be unreliable
```

```
RegPlayer = Player_Results %>% arrange(shoot_player)
head(Player_Results)
```

```
## # A tibble: 6 x 4
##   shoot_player           Coef T_Statistics P_Value
##   <chr>                 <dbl>        <dbl>   <dbl>
## 1 Kent Bazemore     0.0000271     0.000689   0.999
## 2 Dwight Howard     0.0193        0.443      0.658
## 3 Kyle Korver      -0.0489       -1.02       0.307
## 4 Dennis Schroder  -0.00985      -0.334      0.739
## 5 Paul Millsap     -0.0537       -1.62       0.105
## 6 Tim Hardaway Jr. -0.0559       -1.62       0.106
```

```
tail(Player_Results)
```

```
## # A tibble: 6 x 4
##   shoot_player         Coef T_Statistics P_Value
##   <chr>               <dbl>        <dbl>   <dbl>
## 1 Marcus Thornton   -0.0555      -0.735    0.463
## 2 Jason Smith        0.0125       0.195    0.845
## 3 Daniel Ochefu     -0.0113      -0.0382   0.971
## 4 Sheldon McClellan -0.0373      -0.299    0.766
## 5 Tomas Satoransky   0.0113       0.107    0.915
## 6 Ian Mahinmi        0.0916       0.822    0.413
```

- Merge the total number of shots captured in "Player_Shots" to the regression result dataframe. This total number of shots represents the sample size of each regression

```
RegPlayer = left_join(RegPlayer, Player_Shots, by = 'shoot_player')
head(RegPlayer)
```

```
## # A tibble: 6 x 6
##   shoot_player     Coef T_Statistics P_Value shot_count avg_shot_game
##   <chr>           <dbl>        <dbl>   <dbl>      <int>         <dbl>
## 1 A.J. Hammons   0.199         1.03  0.317           42          2.8
## 2 Aaron Brooks   0.0163        0.239 0.812          300          4.84
## 3 Aaron Gordon   0.0268        0.771 0.441          864         10.8
## 4 Adreian Payne -0.231        -1.46  0.154           54          3.6
## 5 Al Horford    -0.0267       -0.728 0.467          801         11.8
## 6 Al Jefferson  -0.133        -2.64  0.00866        471          7.25
```

- Display players with statistically significant estimates on the lagged error variable

```
RegPlayer %>% filter(P_Value <= 0.05)
```

```
## # A tibble: 38 x 6
##    shoot_player            Coef T_Statistics P_Value shot_count avg_shot_game
##    <chr>                  <dbl>        <dbl>   <dbl>      <int>         <dbl>
##  1 Al Jefferson         -0.133         -2.64 0.00866        471          7.25
##  2 Avery Bradley        -0.0802        -2.14 0.0324         775         14.1
##  3 Boris Diaw            0.152          2.51 0.0127         327          4.67
##  4 Christian Wood       -0.603         -3.43 0.0140          23          2.56
##  5 Cole Aldrich         -0.430         -2.54 0.0158          86          1.95
##  6 Dario Saric          -0.0716        -2.10 0.0356         927         11.4
##  7 Darren Collison       0.101          2.58 0.0100         713         10.5
##  8 Jameer Nelson        -0.0929        -2.13 0.0340         604          8.05
##  9 James Michael McAdoo  0.308          2.72 0.00837        118          2.81
## 10 Jeremy Lin           -0.147         -2.76 0.00600        400         11.1
## # ... with 28 more rows
```

There are a total of 38 players with statistically significant estimates on the lagged error variable, that is, the success of their previous shots impact the success rate of their current shot. Interestingly, more than half of these estimates are negative, which means that a success in the previous shot actually hurts the chance of scoring in the current shot. This is the opposite of a "hot hand."

Overall from our regression analyses, 13 players, Boris Diaw, Darren Collison, James Michael McAdoo, Joe Young, Kentavious Caldwell-Pope, Kyle Wiltjer, Malcolm Brogdon, Miles Plumlee, Robert Covington, Skal Labissiere, TJ Warren, Timofey Mozgov, and Tony Parker have positive and statistically significant estimate on the lagged error variable. Thus, these players may have "hot hand." Note that the estimate for Kyle Wiltjer is 1 and there are only a total of 14 observations for him. We need to interpret his result with caution.

```
#Save updated data to csv file
write.csv(Shotlog, 'Shotlog3.csv', row.names=FALSE)
write.csv(Player_Stats, 'Player_Stats3.csv', row.names=FALSE)
write.csv(Player_Shots, 'Player_Shots3.csv', row.names=FALSE)
```