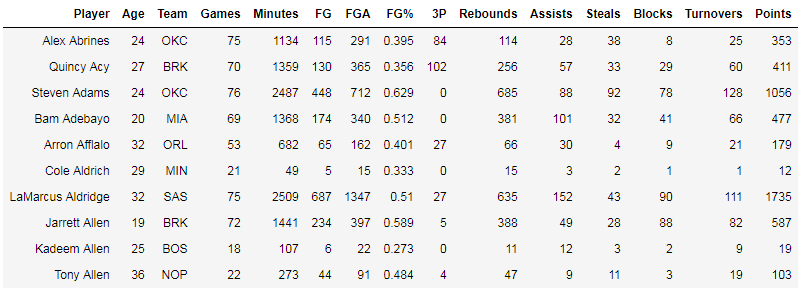
Week 8: Midterm Review

Data 8 Tutoring

# Midterm Review

**1. Table Practice**

The nba table contains data from the 2017-2018 season for every active player. Each row represents statistics over the whole season. In this table, percentages are expressed as decimals.



a) eFG% is an advanced statistic commonly used over FG% (field goal percentage). Calculate eFG% using the formula (FG + 0.5\*3P)/FGA. Make sure to use FG and not FG%.

Once it’s been calculated, append the values as the column eFG% to this table.

numerator = nba.\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_) + \_\_\_\_\_\*nba.\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_)

denominator = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

nba\_efg = nba.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_, numerator / denominator)

b) Find the team with the lowest average eFG% (return the name only)

by\_team = nba\_efg.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

by\_team.sort(\_\_\_\_\_\_\_\_\_\_\_\_\_).\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_).\_\_\_\_\_\_\_\_(\_\_\_\_\_)

c) What proportion of points scored were by players who had an eFG% above 60%? The variable answer should be your final proportion.

more\_than\_sixty = \_\_\_\_\_\_\_\_\_\_(nba\_efg.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(eFG%,

\_\_\_\_\_\_\_\_\_.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_)).column(\_\_\_\_\_\_\_\_\_\_))

total = \_\_\_\_\_\_\_\_\_\_\_(nba\_efg.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(“Points”))

answer = more\_than\_sixty / total

**2. Coding Practice**

a) Suppose a broken candy machine dispenses sweet candy 99% of the time and sour candy otherwise. What is the chance you find at least 1 sour candy in 50 candies dispensed randomly from the machine? Let candies be an array with the first element containing the probability of picking a sweet candy and the second element being the probability of picking sour candy. Use a simulation to estimate the probability of finding at least 1 sour candy in 50 candies dispensed.

candies = make\_array(0.99, 0.01)

sour = \_\_\_\_\_\_\_\_\_\_\_

for i in \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(5000):

chosen\_candies\_prop = sample\_proportions(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

sour\_prop = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

if \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

sour = sour + 1

chance\_of\_at\_least\_one = \_\_\_\_\_\_\_\_\_\_/5000

b) What is the exact probability of finding at least 1 sour candy in 50 candies dispensed from the broken candy machine?

Hint: think about the complement rule!

**3. Defining Functions**

Suppose you have a table dinners, which contains a row for every dinner eaten at a given restaurant for a week. dinners has a column “Day” of strings corresponding to which day of the week the dinner was eaten and a column “Subtotal” of integers containing costs without tips.

a) Define a function called compute\_tip that given a total bill as an integer, returns a 20% tip calculated from the bill.

def compute\_tip(bill):

return \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b) Add a new column to the dinners table called “Tip” that is the 20% tip for each bill in the “Subtotal” column and name the resulting table dinners\_with\_tip.

dinners\_with\_tip = dinners.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(“Tip”,

dinners.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_))

c) Write code to calculate the day of the week with the lowest average cost (not including tip).

by\_day = dinners.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

by\_day.sort(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_).\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_).item(0)

**4. Probability**

Your good friend Gary is playing with a fair 6-sided die.

a) Gary rolls the die 10 times and rolls a 6 every time. What is the probability of that event occurring?

b) Then, Gary rolls the die twice. What is the probability he rolls a 1 on the first roll and a 2 on the second roll?

c) Suppose Gary rolls the die 5 times - what is the probability he rolls at least one 3?

d) If you are considering whether the die is fair, do the results of the throws represent numerical or categorical outcomes?

**5. Hypothesis Testing**

Gary rolls the die another 10 times and rolls 7 sixes, 2 fours, and 1 two. I suspect Gary is using an unfair die and I want to do a hypothesis test to check this.

a) Specify a null and alternative hypothesis.

b) What test statistic would you choose to compare the null distribution above with what you simulate repeatedly to test whether the die is fair? Explain.

c) Calculate the obs\_test\_stat.

fair\_die = make\_array(1/6, 1/6, 1/6, 1/6, 1/6, 1/6)

gary\_die = \_\_\_\_\_\_\_\_\_\_(\_\_\_, \_\_\_, \_\_\_, \_\_\_, \_\_\_, \_\_\_)

obs\_test\_stat = 0.5 \* \_\_\_\_\_(\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ - \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_))

d) Fill in the code below to simulate the distribution of faces if we roll a fair die 10 times and calculate one test statistic.

def calculate\_test\_stat():

fair\_die = make\_array(1/6, 1/6, 1/6, 1/6, 1/6, 1/6)

simulated\_dist = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_,\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)

test\_stat = 0.5 \* \_\_\_\_\_(\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ - \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_))

return test\_stat

e) Fill in the code below to simulate 10000 test statistics and generate the following histogram.

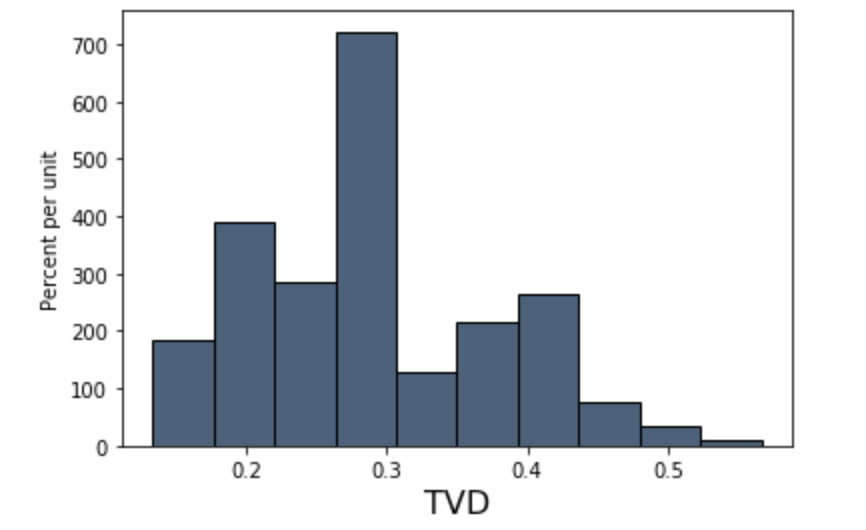
test\_stats = \_\_\_\_\_\_\_\_\_\_\_\_\_

for i in \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_:

one\_test\_stat = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

test\_stats = \_\_\_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_\_)

Table().with\_column(\_\_\_\_\_, \_\_\_\_\_).hist(\_\_\_\_\_)



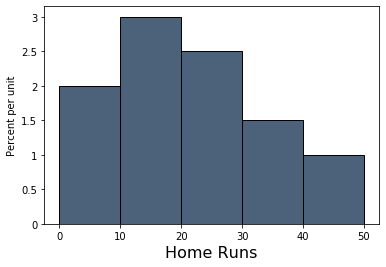
f) Fill in the code below to calculate the p-value.

p\_value = \_\_\_\_\_\_\_\_(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_)/\_\_\_\_(\_\_\_\_\_\_\_\_)

g) If p\_value turns out to be 0.004 and we use a p-value cutoff of 0.05, what conclusion do we draw from this test?

**6. Histograms**

The following is a histogram of the number of home runs hit by MLB players in the 2019 season.



Answer the following questions using the histogram above. If it is not possible to compute the answer, write “Not Possible” and explain why you cannot calculate the answer.

a) Find the percent of players who hit between 10 and 20 (not inclusive) home runs in 2019.

b) Find the number of players who hit more than 30 home runs in 2019.

c) What percent of players hit at least 20 home runs?

d) How many players hit between 25 and 30 home runs?

e) We decide that we want more information about the players that hit between 10 and 20 home runs, so we split the [10,20) bin into two bins: a [10, 15) bin and a [15, 20) bin. We find that 20% of players hit between 10 and 15 home runs. What is the height of this new [15, 20) bin?