Lecture 14 – Midterm Review

DSC 10, Spring 2024

Announcements

- Discussion section is today. Problems are here.
- The Midterm Exam is **this Friday during lecture**. See this post on Ed for lots of details, including what is covered, what to bring, and how to study.
- Make a 1-on-1 appointment with a tutor as you prepare for the exam. Sign up here!
- The Midterm Project is due on **Tuesday, May 7th at 11:59PM**. Only one partner needs to submit.

Agenda

- We'll work through selected problems from the Winter 2024 Midterm and this quarter's Quiz 2.
- We won't write any code, since you can't run code during the exam. Instead, we'll try to think like the computer ourselves.
- These annotated slides will be posted after lecture is over.
- **Try the problems with us!**

Winter 2024 Midterm

Access the exam here. Make sure to read the data info sheet at the top before starting.

🔍 *Clue*: The Murder Mystery Game 🔍

Clue is a murder mystery game where players use the process of elimination to figure out the details of a crime. The premise is that a murder was committed inside a large home, by one of **6 suspects**, with one of **7 weapons**, and in one of **9 rooms**.

The game comes with **22 cards**, one for each of the 6 suspects, 7 weapons, and 9 rooms. To set up the game, one suspect card, one weapon card, and one room card are chosen randomly, without being looked at, and placed aside in an envelope. The cards in the envelope represent the details of the murder: who did it, with what weapon, and in what room.

The remaining 19 cards are randomly shuffled and dealt out to the players (as equally as possible). Players then look at the cards they were dealt and can conclude that any cards they see were **not** involved in the murder. In the gameplay, players take turns moving around to different rooms of the house on the gameboard, which gives them opportunities to see cards in other players' hands and further eliminate suspects, weapons, and rooms. The first player to narrow it down to one suspect, with one weapon, and in one room can make an accusation and win the game!

Suppose Janine, Henry, and Paige are playing a game of Clue. Janine and Paige are each dealt 6 cards, and Henry is dealt 7. The DataFrame `clue` has 22 rows, one for each card in the game. `clue` represents **Janine's knowledge** of who is holding each card. `clue` is indexed by `"Card"`, which contains the name of each suspect, weapon, and room in the game. The `"Category"` column contains `"suspect"`, `"weapon"`, or `"room"`. The `"Cardholder"` column contains `"Janine"`, `"Henry"`, `"Paige"`, or `"Unknown"`.

Card	Category	Cardholder
Col. Mustard	suspect	Unknown
Dr. Orchid	suspect	Henry
Miss Scarlett	suspect	Henry
Mr. Green	suspect	Paige
Mrs. Peacock	suspect	Unknown
...	...	...
hall	room	Janine
kitchen	room	Janine
library	room	Unknown
lounge	room	Janine
study	room	Unknown

22 rows × 2 columns

Since Janine's knowledge is changing throughout the game, the `"Cardholder"` column needs to be updated frequently. At the beginning of the game, the `"Cardholder"` column contains only `"Janine"` and `"Unknown"` values. We'll assume throughout this exam that `clue` contains Janine's current knowledge at an arbitrary point in time, not necessarily at the beginning of the game. For example, `clue` may look like the DataFrame at right.

Note: Throughout the exam, assume we have already run `import babypandas as bpd` and `import numpy as np`.

About the Game

- 22 cards, in three categories
  - 6 suspects
  - 7 weapons
  - 9 rooms
- Envelope: one card of each category
- Janine: 6 cards
- Paige: 6 cards
- Henry: 7 cards

22 - 3 = 19

Henry	?
Janine	6
Paige	?
Unknown	?

Question 1 (18 pts)

• index pulls out labels

Each of the following expressions evaluates to an integer. Determine the value of that integer, if possible, or circle "not enough information."

a) `(clue.get("Cardholder") == "Janine").sum()`

T = 1
F = 0

6

not enough information

b) `np.count_nonzero(clue.get("Category").str.contains("p"))`

suspect + weapon

13

not enough information

c) `clue[(clue.get("Category") == "suspect") & (clue.get("Cardholder") == "Janine")].shape[0]`

not enough information

suspect cards Janine is dealt (random)

d) `len(clue.take(np.arange(5, 20, 3)).index)`

(5, 8, 11, 14, 17)

5 row df

5

not enough information

e) `len(clue[clue.get("Category") >= "this"].index)`

"weapon" ≥ "this" abcs

7

not enough information

f) `clue.groupby("Cardholder").count().get("Category").sum()`

22

not enough information

Question 3 (8 pts)

An important part of the game is knowing when you've narrowed it down to just one suspect with one weapon in one room. Then you can make your accusation and win the game!

Suppose the DataFrames `grouped` and `filtered` are defined as follows.

```
grouped = (clue.reset_index()
               .groupby(["Category", "Cardholder"])
               .count()
               .reset_index())
filtered = grouped[grouped.get("Cardholder") == "Unknown"]
```

still unknown

a) (4 pts) Fill in the blank below so that `"Ready to accuse"` is printed when Janine has enough information to make an accusation and win the game.

```
if filtered.get("Card").______ == 3:
    print("Ready to accuse")
```

What goes in the blank?

○ `count()` ● `sum()` ○ `max()` ○ `min()` ○ `shape[0]`

b) (4 pts) Now, let's look at a different way to do the same thing. Fill in the blank below so that `"Ready to accuse"` is printed when Janine has enough information to make an accusation and win the game.

```
if filtered.get("Card").______ == 1:
    print("Ready to accuse")
```

What goes in the blank?

○ `count()` ○ `sum()` ● `max()` ○ `min()` ○ `shape[0]`

groupby aggregation method - must come after groupby

```
1 grouped = clue.reset_index().groupby(['Category', 'Cardholder']).count().reset_index()
```

	Card	Category	Cardholder
0	Col. Mustard	suspect	Unknown
1	Dr. Orchid	suspect	Henry
2	Miss Scarlett	suspect	Henry
3	Mr. Green	suspect	Paige
4	Mrs. Peacock	suspect	Unknown
...	...	...	...
17	hall	room	Janine
18	kitchen	room	Janine
19	library	room	Unknown
20	lounge	room	Janine
21	study	room	Unknown

Category	Cardholder	Card
room	Janine	4
	Unknown	5
suspect	Henry	2
	Paige	1
	Unknown	3
weapon	Janine	2
	Paige	1
	Unknown	4

all must be == 1 to accuse

	Category	Cardholder	Card
0	room	Janine	4
1	room	Unknown	5
2	suspect	Henry	2
3	suspect	Paige	1
4	suspect	Unknown	3
5	weapon	Janine	2
6	weapon	Paige	1
7	weapon	Unknown	4

← grouped

```
2 filtered = grouped[grouped.get('Cardholder') == 'Unknown']
```

	Category	Cardholder	Card
1	room	Unknown	5
4	suspect	Unknown	3
7	weapon	Unknown	4

filtered

4
4
1,

sum() == 3

Question 4 (7 pts)

When someone is ready to make an accusation, they make a statement such as:

"It was Miss Scarlett with the dagger in the study"

While the suspect, weapon, and room may be different, an accusation will always have this form:

"It was _____ with the _____ in the _____"

Suppose the array `words` is defined as follows (note the spaces).

```
words = np.array(["It was ", " with the ", " in the "])
```

Suppose another array called `answers` has been defined. `answers` contains three elements: the name of the suspect, weapon, and room that we would like to use in our accusation, in that order. Using `words` and `answers`, complete the `for`-loop below so that `accusation` is a string, formatted as above, that represents our accusation.

```
accusation = ""
for i in ___(a)___:
    accusation = ___(b)___
```

a) (3 pts) What goes in blank (a)?

[0, 1, 2]

b) (4 pts) What goes in blank (b)?

accusation + words[i] + answers[i]

accusation = words[0] + answers[0] + words[1] + answers[1] + words[2] + answers[2]

Question 5 (12 pts)

Recall that the game *Clue* comes with 22 cards, one for each of the 6 suspects, 7 weapons, and 9 rooms. One suspect card, one weapon card, and one room card are chosen randomly, without being looked at, and placed aside in an envelope. The remaining 19 cards (5 suspects, 6 weapons, 8 rooms) are randomly shuffled and dealt out, splitting them as evenly as possible among the players. Suppose in a three-player game, Janine gets 6 cards, which are dealt one at a time.

Answer the probability questions that follow. Leave your answers **unsimplified**.

a) (4 pts) Cards are dealt one at a time. What is the probability that the first card Janine is dealt is a weapon card?

$6/19$

b) (4 pts) What is the probability that all 6 of Janine's cards are weapon cards?

$\frac{6}{19} \cdot \frac{5}{18} \cdot \frac{4}{17} \cdot \frac{3}{16} \cdot \frac{2}{15} \cdot \frac{1}{14}$

c) (4 pts) Determine the probability that exactly one of the first two cards Janine is dealt is a weapon card. This probability can be expressed in the form

$$\frac{k \cdot (k+1)}{m \cdot (m+1)}$$

where k and m are **integers**. What are the values of k and m?

W, nW OR nW, W

$\frac{6}{19} \cdot \frac{13}{18} + \frac{13}{19} \cdot \frac{6}{18}$

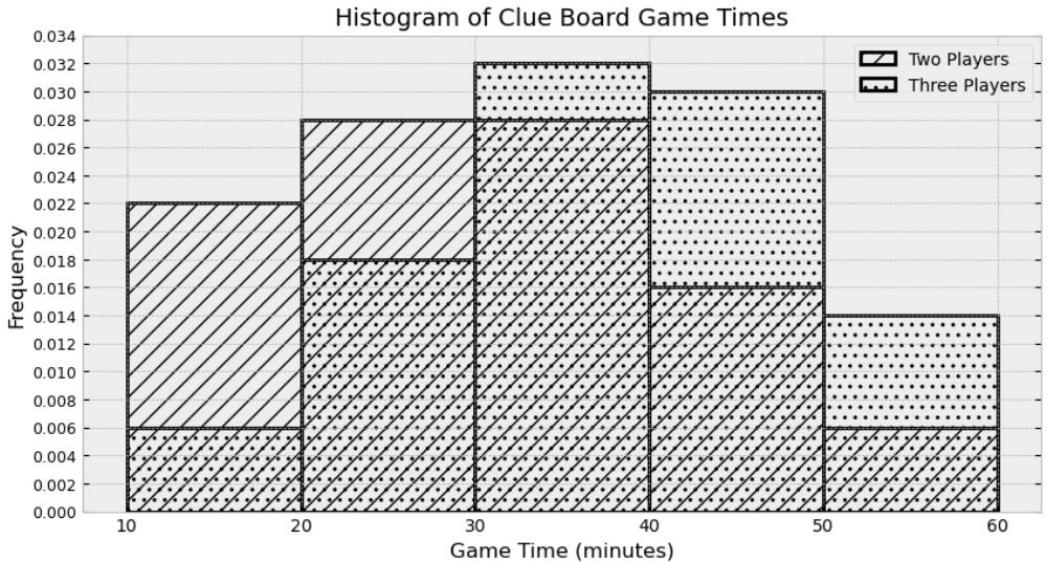
$\frac{6 \cdot 13 + 13 \cdot 6}{18 \cdot 19} = \frac{12 \cdot 13}{18 \cdot 19}$

$k = 12$, $m = 18$

Hint: There is no need for any sort of calculation that you can't do easily in your head, such as long division or multiplication.

Question 8 (8 pts)

The histogram below shows the distribution of game times in minutes for both two-player and three-player games of *Clue*, with each distribution representing 1000 games played.

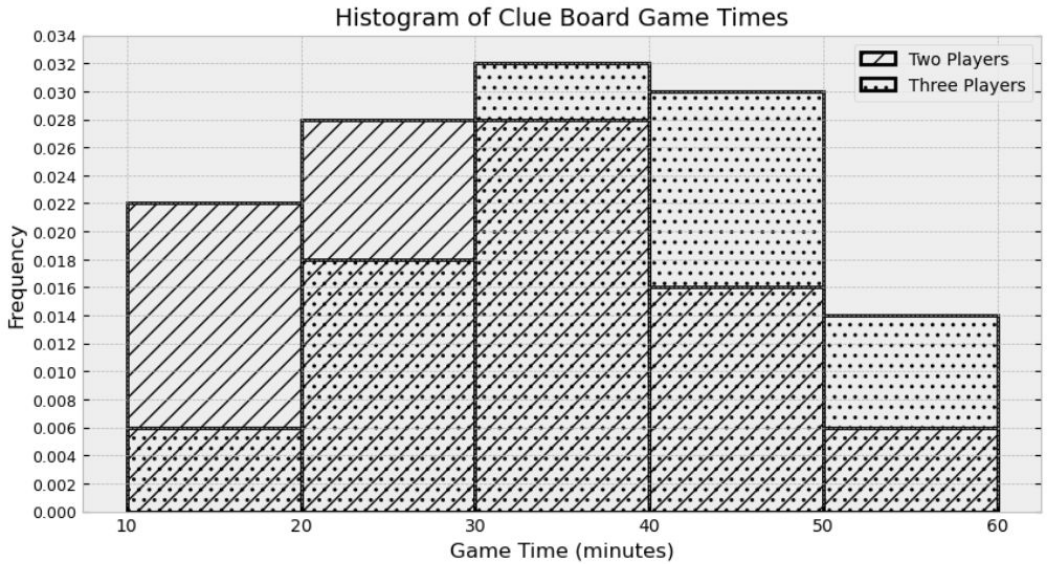


see 10 am lecture

a) (4 pts) How many **more** three-player games than two-player games took at least 50 minutes to play? Give your answer as an **integer, rounded to the nearest multiple of 10**.

Question 8 (8 pts)

The histogram below shows the distribution of game times in minutes for both two-player and three-player games of *Clue*, with each distribution representing 1000 games played.



b) (4 pts) Calculate the approximate area of overlap of the two histograms. Give your answer as a **proportion between 0 and 1, rounded to two decimal places.**

see 10am lecture

Quiz 2

Access the quiz here.

An art museum records information about its collection in a DataFrame called `art`. The columns of `art` are as follows:

- `"title"` `(str)`: the name of the art piece.
- `"artist"` `(str)`: the name of the artist.
- `"year"` `(int)`: the year the art piece was produced.
- `"price"` `(float)`: the selling price of the art piece in dollars.

Question 2

b) Fill in the blanks in the code below to find the name of the artist in `art` who made the most art pieces in a single year.

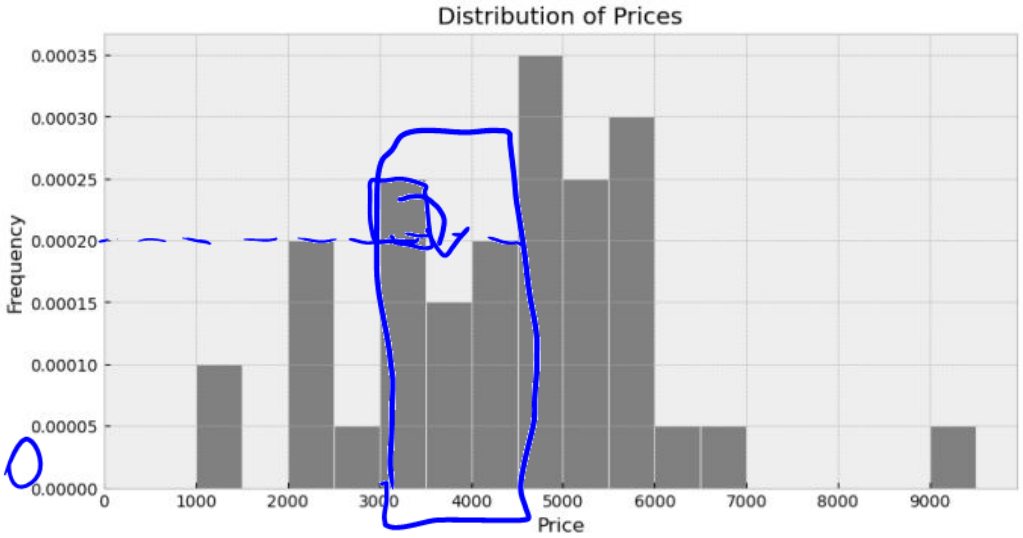
```
(art.groupby(___(x)___).___(y)___.reset_index()
    .sort_values(by="title", ascending=False)
    .get("artist").iloc[0])
```

see 11 am lecture

(x): _____ , (y): _____

Question 3

b) The ~~density~~ histogram below shows the distribution of `"price"` in `art`. If the museum has 100 art pieces in total, how many pieces cost at least \$3,000 but less than \$4,500?



30% of 100

30

3 bars

1500 × 0.0002

0.3 = 1.5 * 0.2

Question 4

a) Fill in the return statement of the function `is_expensive`, which takes as input the price of an art piece (as a float, in dollars) and returns `True` if the price is more than 20 million dollars. Otherwise, it returns `False`.

```
def is_expensive(price):
    return ___(a)___
```

(a):

b) Write one line of code to add a new column called `exp` to the `art` DataFrame, which categorizes if each art piece is worth more than 20 million dollars, using Boolean values. You must use the `is_expensive` function you wrote above. Make sure to modify `art`!

c) Next, we make a new DataFrame called `expensive` as follows.

```
expensive = art[art.get(exp)]
merged = art.merge(expensive, on="artist")
```

Van Gogh is one artist represented in `art` and exactly half of his pieces in `art` are worth over 20 million dollars. If Van Gogh's art appears in 72 rows of the `merged` DataFrame, how many rows does Van Gogh actually have in the original `art` DataFrame?

see llam lecture