Data8: Foundations of Data Science — Study Guide

Oliver Muellerklein
PhD candidate and Tutor

omuellerklein@berkeley.edu
OliverTutor.me

1 UC Berkeley and Beyond

The questions in this study guide are modified versions of final exam questions from the UC Berkeley course **Data 8: Foundations of Data Science**. However, the material in this study guide should be appropriate for anyone taking an introduction to data science at many universities and colleges.

If you find this material useful and would like to enhance your skills, feel free to reach out to me for 1:1 tutoring. You can email me at **omuellerklein@berkeley.edu** and find me at **OliverTutor.me**.

2 Overview of Questions

This study guide contains questions inspired from midterms and finals spanning Spring 2022 to Spring 2024 in Data 8. The questions I used for this study guide are ones that I helped many Data8 students with during 1:1 tutoring sessions, especially those that most students find tricky or non-intuitive. I have recreated the same type of questions that are generally asked on the Data 8 exams to help students prepare themselves.

These questions cover the following topics:

- Python Programming
- Hypothesis Testing
- Standard Units and Data Transformations
- Correlation Coefficients and Linear Regression
- P-values and Confidence Intervals
- Probability

3 Hypothesis Questions: Part 1 of 2

Charlie and Lee are using daily viewer count data to predict the highest trending viral videos on social media. The dataset, named `video_insights`, consists of data from 200 popular videos released in the past week. The dataset includes the following columns:

- **Title**: A string, the name of the video.
- **Creator**: A string, the name of the content creator.
- **Views**: An integer representing the number of views a video received within the first 24 hours of its release.
- **Trend_Rank**: An integer representing the highest trending position the video achieved (1 is the highest).

(a) (2 points) Fill in blank (*a*) such that `calc_su` returns the input array in standard units.

```
def calc_su(array):
    mean_a = np.mean(array)
    sd = np.std(array)
    ...(a)...
```

(b) Fill in the code to calculate the correlation coefficient between the number of views on release day and the top trending rank the video achieved.

```
views_in_su = compute_su(...(a)...)
trends_in_su = compute_su(...(b)...)
corr_value = ...(c)...
```

(i) **(1 point)** Fill in blank (*a*).

(ii) **(1 point)** Fill in blank (*b*).

(iii) **(2 points)** Fill in blank (*c*) to finalize the correlation calculation.

3.1 Solutions: Part 1 of 2

1. Background and Context

Before we getting to the solutions, here's a quick overview of what's going on:

- We have a function called `calc_su` that is supposed to transform an array into *standard units* ("su").
- "Standard units" is just another way of saying we subtract the mean and then divide by the standard deviation. Outside of data8, this is most often done in Python with SciKit-learn's (`Sklearn`) `StandardScaler`.
- Data8 at UC Berkeley uses a custom Python library (called "datascience") that has a custom `Table` object at the core of most operations.
- In case you have tried searching online for solutions, 99% of data science in Python use the pandas library `DataFrame` object and NOT a `Table` object. But the idea is basically the same: consists of rows and columns and each column can be modified with different functions.
- Our two relevant numeric columns are `Views` and `Trend_Rank` (in the `video_insights` table).

Something I see time and time again with the students I have tutored over the years in Data8 is confusion between the custom Python library you use in this course and what the rest of the world is doing.

I am here to tell you: **YES - IT IS CONFUSING!**

Data8 at UC Berkeley uses a custom Python library that is inspired by a combination of the widely used **Numpy** and **Pandas** libraries. However, it is even more confusing that this library is used with Numpy as well. This course can be frustrating if you get stuck because searching for many of the questions online will give you different solutions to the same problems. That is because this course uses a library that you will never see or use again...

This is exactly why I was inspired to create this study guide. I want to help clarify and explain concepts to current and future Data8 students.

We need to finish the function `calc_su`, which takes an array of numbers (like a column from a table) and returns a new array where each value has been converted to standard units.

2. Understanding Standard Units

- The phrases "standardize data," "put data in standard units," or "apply a standard scaler," all mean the same thing!
- Mathematically, if x is your array of data, then:

  $$x_{\text{su}} = \frac{x - \bar{x}}{\sigma},$$

  where $\bar{x}$ is the mean of x and σ is the standard deviation of x.
- After this transformation, the new x_{su} will have mean 0 and standard deviation 1.
- This is incredibly helpful for comparing variables on wildly different scales.

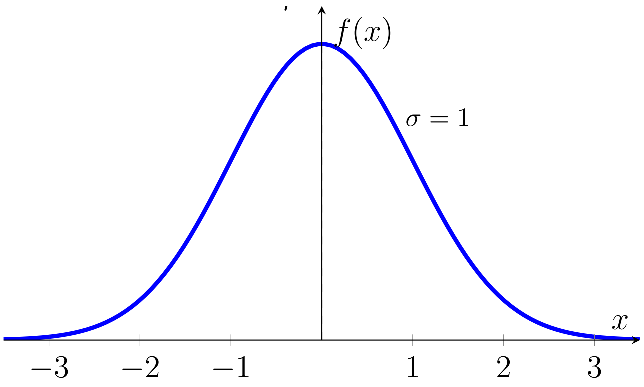

Figure 1: A standard normal (bell) curve with mean 0 and standard deviation 1 .

3. Solution to (a): Completing `calc_su`

- The function with the solution:

```
def calc_su(array):
    mean_a = np.mean(array)
    sd = np.std(array)
    return (array - mean_a) / sd    # return array in standard units
```

- We grab the mean (`mean_a`) and standard deviation (`sd`), and do exactly the formula above.

- Note: this is what sklearn's `StandardScaler` function does for you.

`(a) = return (array - mean_a) / sd`

4. Solution to (a): Key Tips and Reminders

- Always ensure your column is numeric! If it's a string column (like titles or names), you won't be able to do math on it.

- Watch out for parentheses in Python! If you omit them, you might accidentally do integer division or run into operator precedence issues.

- Double-check that you're using the correct column from the table. A simple typo can derail your entire analysis.

5. Solution to (b): Applying `calc_su`

Now that `calc_su` is defined, we want to get the columns `Views` and `Trend_Rank` in standard units and compute their correlation (often denoted as **r**).

- We have a `Table` called `video_insights` and we want to get two of the columns.
- Python code to get columns as arrays:

```
views_array = video_insights.column("Views")
trend_array = video_insights.column("Trend_Rank")
```

- Then pass these arrays to the `calc_su` function:

```
views_in_su = calc_su(views_array)
trend_in_su = calc_su(trend_array)
```

- **Blank (a)** = `video_insights.column("Views")`
- **Blank (b)** = `video_insights.column("Trend_Rank")`
- **Blank (c)** = `np.mean(views_in_su * trend_in_su)`

When both arrays are in standard units, their correlation is just the average of the product of the corresponding elements.

6. Deep Dive: Why Standard Units Make Correlation Easy

- This section is for those who like to understand the math behind what is happening.
- Normally, we'd do a bit more math to find correlation:

$$r = \frac{\text{cov}(X, Y)}{\sigma_X \sigma_Y}.$$

- But once each variable is already scaled so that $\sigma_X = \sigma_Y = 1$, it becomes $\text{mean}(X_{\text{su}} \times Y_{\text{su}})$.

7. Key Takeaways

- Converting to standard units is straightforward but super powerful for comparing variables on different scales.
- Once in standard units, correlation is just the average of the product (no need to worry about each variable's separate standard deviation).

- You've seen how to define and use a custom standard units function (`calc_su`).
- You've also seen how to compute correlation when both variables are in standard units.
- Remember: outside of Data 8, you'll likely see pandas `DataFrame` objects or even use sklearn's `StandardScaler`, but the basic concepts remain exactly the same.
- Also, outside this class, in regular Python + pandas world, you might grab a column of data by doing `video_insights['Views']`. But here, the syntax is often `video_insights.column("Views")`.

4 Hypothesis Questions: Part 2 of 2

(c) (2 points)
Assume that Charlie correctly calculated $r = -0.85$. Given that the standard deviation for Views is 120,000 and for Trend_Rank is 250, fill in blank (a) to compute the slope of the simple linear regression line in original units.

(Reminder: Charlie is using the number of Views in the first 24 hours to predict the Trend Rank)

$$\text{slope} = \_\_\_\_(a)\_\_\_\_$$

(d) Charlie wants to know if there is a linear relationship between Views and Trend_Rank.

(i) **(2 points)** Which of the following are valid hypotheses? Select all that apply.

□ The true correlation coefficient between Views and Trend_Rank is 0.

□ The true correlation coefficient between Views and Trend_Rank is not 0.

□ The true slope between Views and Trend_Rank is 0.

□ The true slope between Views and Trend_Rank is not 0.

□ None of the above.

(ii) **(2 points)** Which of the following are valid alternative hypotheses for this hypothesis test? Select all that apply.

□ There is a positive linear relationship between Views and Trend_Rank.

□ There is a negative linear relationship between Views and Trend_Rank.

□ The correlation coefficient between Views and Trend_Rank is zero.

□ The correlation coefficient between Views and Trend_Rank is not zero.

(iii) **(2 points)** Charlie constructs a 90% confidence interval for the correlation between Views and Trend_Rank, which is $[-0.97, -0.65]$. Which of the following is true? Select all that apply.

□ 90% of the data we observe can be explained by the regression line.

□ The data supports the null hypothesis.

□ The data supports the alternative hypothesis.

□ There's a 90% chance that the true correlation coefficient is in the interval $[-0.97, -0.65]$.

□ If we run this process of collecting data and running a hypothesis test 100 times, we would expect around 90 of the confidence intervals to contain the true correlation coefficient.

4.1 Solutions: Part 2 of 2

1. (c) Finding the Slope in Original Units

Restate the Question: We know Charlie has calculated the correlation coefficient r. Now, we want the slope of the regression line that predicts `Trend_Rank` (y) from `Views` (x). Remember:

$$\text{slope} = r \times \frac{\text{std of y}}{\text{std of x}}.$$

- But why multiply by $\frac{\text{std y}}{\text{std x}}$?
- When data are in standard units, the slope in the "standardized" world is just r.
- But in the original units (i.e., actual number of views, actual trend rank), we have to *rescale.*
- That means we take r (the slope in standard units) and multiply by the ratio of the standard deviations of y and x.

Setting Up the Calculation

- We're told:

$$r = -0.85, \quad \text{std of y (Trend\_Rank)} = 250, \quad \text{std of x (Views)} = 120{,}000.$$

- Hence,

$$\text{slope} = -0.85 \times \frac{250}{120{,}000}.$$

2. Solution to (c): Finding Slope

```
-0.85 * 250 / 120000
```

- This means that for every additional 1 view (in the original scale), the trend rank is expected to decrease by that fraction—indicating that more views *lower* the rank number (which is good, since a "lower" rank means a higher position).
- Think of `Views` on the x-axis and `Trend_Rank` on the y-axis. A negative r means as `Views` go up, the `Trend_Rank` value tends to go down.

3. Background and Context: Why Hypothesis Testing?

- Hypothesis testing is a classic strategy in statistics: we define a "null" scenario where there's no special effect or relationship, and an "alternative" scenario where there is some effect or pattern.
- We collect data, compute some statistic (like a correlation, r), and see whether that data is surprising under the null.
- The **alternative hypothesis** (i.e., "There is a linear relationship") is usually stated in the question: "Charlie is interested in whether there is a linear relationship between `Views` and `Trend_Rank`."
- So, the **null hypothesis** becomes: "There is no linear relationship between `Views` and `Trend_Rank`."
- Statistically, "no linear relationship" means the correlation coefficient is 0 and the slope of the linear regression line is 0.

4. Solution to (d): Part i.

"The true correlation coefficient between Views and Trend_Rank is 0."

and

"The true slope between Views and Trend_Rank is 0."

- These statements capture the null hypothesis.
- They're essentially the same statement, just phrased differently (because correlation and slope both reflect a linear relationship).

5. Solution to (d): Part ii.

"The correlation coefficient between Views and Trend_Rank is not zero."

- The alternative says "There *is* a linear relationship," which means correlation $\neq 0$ (or slope $\neq 0$).
- This covers both positive or negative relationships. We're not specifying which direction—just that it's not zero.
- The correlation (or slope) "would not be zero."
- That's how we represent that a relationship exists in either direction.

6. Question (d)(iii): Confidence Intervals and P-values

We have a 90% confidence interval for the correlation coefficient between `Views` and `Trend_Rank`, and we want to interpret what it means in the context of hypothesis testing. In particular, we're looking at which statements correctly reflect the confidence interval concept and its relationship to the alternative hypothesis.

I. Understanding P-values in a Nutshell

- A **p-value** is the probability of seeing data at least as extreme as what we observed, *if the null hypothesis were true.*
- For example, if the p-value is 0.06 and our chosen cutoff (α) is 0.10, that means:

  > "There's about a 6% chance we'd see our observed data (or something more extreme) if there really was no relationship between `Views` and `Trend_Rank`."

- Since 0.06 is smaller than 0.10, we say, "wow, that's quite unlikely," and we *reject the null hypothesis* at the 10% significance level.
- However, if we had used $\alpha = 0.05$ (the classic 5% standard), 0.06 would be larger than 0.05, so we'd *fail to reject the null.*
- This is why the choice of α matters! The same p-value can lead to different decisions depending on the cutoff.

II. Binary World of Null vs. Alternative

- In the hypothesis testing framework, we're given only two sides to the story: the **null hypothesis** and the **alternative hypothesis**.
- If we *reject one*, we're effectively *accepting the other* in this binary setup. (If this feels limiting, note that there are indeed more nuanced approaches in statistics—feel free to chat with me for a more advanced perspective!)

III. Confidence Intervals and What They Mean

- A **90% confidence interval** for the correlation is, in a loose sense, an interval that would contain the true correlation coefficient in about 90 out of 100 hypothetical experiments of the same kind.
- Be cautious: it's easy to misinterpret CIs. The standard textbook phrase is:

  > "If we repeat this entire process (collect data, compute correlation, build the CI) many times, about 90% of the resulting confidence intervals would capture the true correlation coefficient."

- This is *not* the same as saying there's a 90% probability that the true correlation is in this specific interval (although many students initially think that way).

7. Solution to Question (d): Part iii

- $\boxed{\text{“The data supports the alternative hypothesis.”}}$
  - If our p-value is below our chosen significance level (like 0.10 for a 90% confidence approach), we reject the null, meaning the data leans in favor of an actual linear relationship.
- $\boxed{\text{“If we run this process 100 times, ... contain the true correlation.”}}$
  - This is the standard interpretation of what a 90% confidence interval means. It's a probability statement *about the process*, not about one single interval.

7. Key Takeaways

- Hypothesis tests help us determine whether our observed data is consistent with the null.
- If it's very unlikely under the null, we have evidence for the alternative. If not, we fail to reject the null.
- Remember: it's not about "proving" the null false, just looking at the likelihood of our data if the null were true.
- For a linear relationship question:

  $$\text{Null} = (\rho = 0) \quad \text{vs.} \quad \text{Alternative} = (\rho \neq 0),$$

  where ρ is the *true* correlation in the population.
- Equivalently for slope:

  $$\text{Null} = (\text{slope} = 0) \quad \text{vs.} \quad \text{Alternative} = (\text{slope} \neq 0).$$
- P-values and confidence intervals often go hand in hand. A CI that excludes 0 correlates with a p-value that suggests rejecting the null of "no relationship."
- If you ever feel confused, remember: p-values measure how surprising your data is under the null, and confidence intervals show you a plausible range for the true value.
- Practice is key here. The more you use these concepts, the more comfortable you'll become with them. And if you'd like deeper insight or alternative statistical approaches, I'm always open to 1:1 sessions.

5 Probability Questions

Each robot model at the end of the San Jose Robotics Fair is chosen from a collection of 20 robots: 10 AlphaBots, 9 BetaBots, and 1 OmegaBot.

For each event below, choose the Python expression that evaluates to the probability of that event.

(a) (8.0 points)

(i) **(2.0 pt)** When one robot is chosen at random, the probability that it is either an AlphaBot or an OmegaBot.

□ `(9/20) ** 2`

□ `(10/20) * (1/20)`

□ `(10/20) + (1/20)`

□ `1 - (9/20) ** 2`

□ `1 - (10/20) * (1/20)`

□ `1 - ((10/20) + (1/20))`

(ii) **(2.0 pt)** When two robots are chosen at random with replacement, the probability that they are both BetaBots.

□ `(9/20) ** 2`

□ `(10/20) * (1/20)`

□ `(10/20) + (1/20)`

□ `1 - (9/20) ** 2`

□ `1 - (10/20) * (1/20)`

□ `1 - (10/20) + (1/20)`

(iii) **(2.0 pt)** When two robots are chosen at random with replacement, the probability that the first is an AlphaBot and the second is not.

□ `10/20 + 10/20`

□ `(10/20) * (10/20)`

□ `(10/20) * (9/20) * (1/20)`

□ `1 - (10/20) * (10/20)`

□ `1 - (10/20 + 10/20)`

□ `1 - (10/20) * (9/20) * (1/20)`

(iv) **(2.0 pt)** When two robots are chosen at random with replacement, the probability that the first can defeat the second.

(Assume BetaBots can only defeat AlphaBots, AlphaBots can only defeat OmegaBots, and OmegaBots cannot defeat either - OmegaBots can only defend.)

□ (10/20) * (10/20)

□ (19/20) * (10/20)

□ (10/20) * (1/20) + (9/20) * (10/20)

□ 1 - ((9/20) * (1/20) + (10/20) * (9/20))

□ 1 - ((10/20) ** 2 + (9/20) ** 2 + (1/20) ** 2)

□ 1 - ((10/20) ** 2 * 2 + (9/20) ** 2 * 2 + (1/20))

5.1 Solutions: Probability

Context Reminder:
There is a collection of 20 robot models:

10 AlphaBots
9 BetaBots
1 OmegaBot

A single robot or multiple robots may be selected *with replacement* (meaning after you pick a robot, you record that choice but then put it "back" so it could be chosen again). We want to compute various probabilities related to choosing these robots.

Question (i)
When one robot is chosen at random, what is the probability that it is either an AlphaBot or an OmegaBot?

Step-by-Step Solution

- One of the most fundamental rules in probability is that **"and"** translates to multiplication while **"or"** translates to addition (provided the events are mutually exclusive).
- In this question, the event is: *"the robot is AlphaBot" OR "the robot is OmegaBot."*
- We can denote the probability of selecting an AlphaBot as $P(\text{AlphaBot})$ and the probability of selecting an OmegaBot as $P(\text{OmegaBot})$.
- Since there are 10 AlphaBots out of 20 and 1 OmegaBot out of 20:

$$P(\text{AlphaBot}) = \frac{10}{20}, \quad P(\text{OmegaBot}) = \frac{1}{20}.$$

- Using *addition* of these probabilities ("or"):

$$P(\text{AlphaBot or OmegaBot}) \;=\; \frac{10}{20} \;+\; \frac{1}{20}.$$

- This simplifies to:

$$\frac{10}{20} + \frac{1}{20}$$

- **Extra Hint:** Be mindful in Python or any programming language to use parentheses where needed, for instance: `(10/20) + (1/20)`.

Python answer for (i):

$$\boxed{(10/20) + (1/20)}$$

Question (ii)

When two robots are chosen at random with replacement, what is the probability that they are both BetaBots?

Step-by-Step Solution

- Here, the problem involves two draws *with replacement*, meaning the probability of picking a BetaBot on the second draw does not change due to the first draw.
- We use the rule that **"and"** translates to multiplication (for independent events).
- Let $P(\text{BetaBot})$ be the probability of selecting a BetaBot on a single draw. Since there are 9 BetaBots out of 20:
$$P(\text{BetaBot}) = \frac{9}{20}.$$
- The probability that the first chosen is BetaBot *and* the second chosen is BetaBot is:
$$P(\text{BetaBot}) \times P(\text{BetaBot}) = \left(\frac{9}{20}\right) \times \left(\frac{9}{20}\right).$$
- We can write this as:
$$\left(\frac{9}{20}\right)^2$$
- **Note:** Because of *replacement*, the denominator remains the same for each draw.

Python answer for (ii):

$$\boxed{\texttt{(9/20)**2}}$$

Question (iii)

When two robots are chosen at random with replacement, the probability that the first is an AlphaBot and the second is not an AlphaBot.

Step-by-Step Solution

- Again, we have two draws *with replacement.* The question is actually *"AlphaBot AND not AlphaBot"*.
- Break the problem into two smaller parts:
$$P(\text{AlphaBot on first draw}) = \frac{10}{20}.$$
- Next, $P(\text{not AlphaBot on second draw})$. There are two ways to think about "not AlphaBot":

1. **1 minus** the probability of AlphaBot:

$$1 - \frac{10}{20} = \frac{10}{20}.$$

2. **Sum** of BetaBots and OmegaBots:

$$\frac{9}{20} + \frac{1}{20} = \frac{10}{20}.$$

Either method gives the same result: $\frac{10}{20}$.

- Since these two events (first draw = AlphaBot, second draw = not AlphaBot) happen consecutively, we multiply:

$$P(\text{AlphaBot then not AlphaBot}) = P(\text{AlphaBot}) \times P(\text{not AlphaBot}) = \left(\frac{10}{20}\right) \times \left(\frac{10}{20}\right)$$

Python answer for (iii):

$$\boxed{(10/20) * (10/20)}$$

Question (iv)

When two robots are chosen at random with replacement, what is the probability that the first can defeat the second?
(Assume BetaBots can only defeat AlphaBots, AlphaBots can only defeat OmegaBots, and OmegaBots cannot defeat either.)

Detailed Solution and Tips

- We have two specific scenarios in which the first robot can defeat the second:

$$\text{Scenario 1: First} = \text{BetaBot, Second} = \text{AlphaBot}$$

$$\text{Scenario 2: First} = \text{AlphaBot, Second} = \text{OmegaBot}$$

- Let $P(\text{BetaBot}) = \frac{9}{20}$, $P(\text{AlphaBot}) = \frac{10}{20}$, and $P(\text{OmegaBot}) = \frac{1}{20}$.
- Because the selections are *with replacement*, the probability of each selection remains the same from draw to draw.
- Probability of *Scenario 1*:

$$P(\text{BetaBot first}) \times P(\text{AlphaBot second}) = \frac{9}{20} \times \frac{10}{20}.$$

- Probability of *Scenario 2*:

$$P(\text{AlphaBot first}) \times P(\text{OmegaBot second}) = \frac{10}{20} \times \frac{1}{20}.$$

- Since these two scenarios are mutually exclusive but both satisfy "first can defeat the second," we *add* their probabilities:

$$\left(\frac{9}{20} \times \frac{10}{20}\right) + \left(\frac{10}{20} \times \frac{1}{20}\right).$$

- In Python, you might write this as:

```
(9/20)*(10/20) + (10/20)*(1/20)
```

- **Alternative factoring (polynomial approach):** with some algebra, you can get

```
(10/20) * (10/20)
```

- **In Python:**

```
(9/20)*(10/20) + (10/20)*(1/20)
```

  which simplifies to:

```
(10/20)*(10/20)
```

Python answers for (iv):

```
(9/20)*(10/20) + (10/20)*(1/20)
```

which simplifies to:

```
(10/20)*(10/20)
```

Key Takeaways:

- **With replacement:** Each selection is independent. The denominator or probability structure does not change between draws.
- **"And" = Multiply, "Or" = Add:** This is fundamental for independent events and mutually exclusive events, respectively.
- **Check your code expressions in Python carefully**, especially using parentheses for addition and multiplication to enforce the correct order of operations.