Econ 148, Midterm

Spring 2023

Name: _____

Email: _____@berkeley.edu

Student ID: _____

Name and SID of left neighbor: _____

Name and SID of right neighbor: _____

Instructions:

This midterm exam consists of **40 points** spread out over **3 sections** and the Honor Code and must be completed in the **50 minute** time period ending at **2:00 PM**, unless you have accommodations supported by a DSP letter.

Note that some questions have circular bubbles to select a choice. This means that you should only **select one choice**. Other questions have boxes. This means you should **select all that apply**. Please shade in the box/circle to mark your answer.

Honor Code [1 Pt]:

As a member of the UC Berkeley community, I act with honesty, integrity, and respect for others. I am the person whose name is on the exam and I completed this exam in accordance with the Honor Code.

Signature: _____

1 Hawthorne Effects [7 Pts]

Does completing a household survey change the later behavior of those surveyed? In the Waterguard study, researchers provide evidence from a variety of settings that the act of being surveyed can affect behavior and confound estimates of parameters that initially motivated the data collection.

(a) [2 Pts] In 3 sentences or less, describe a randomized controlled trial research experiment to quantify the Hawthorne measurement effect.
Hint: Do you want to split people into different groups?

Solution:

Student acknowledges the importance of a control and treatment group.
Student describes their control and treatment arms (control = less frequent surveying, treatment = frequent surveying).

(b) [3 Pts] In the project in Kenya in Lab 3, the Hawthorne effect was observed to affect two different outcome variables, and in different directions. What were the outcomes and what was the direction of the Hawthorne effect?

Solution:

For the diarrhea measurement, the Hawthorne effect caused it to go down; for the water treatment, the Hawthorne effect caused it to go up. In general, Hawthorne effect describes that if you thought you were being observed (e.g. by frequent surveys), you could be influenced to

adjust your behaviors if you didn't want to be bothered by more questions (survey fatigue) or if you feel you are under social pressure (social desirability).

(c) [2 Pts] In Lab 3 there were two related datasets with different dimensions, can you describe how the dimensions (or granularity) of the datasets relates to the sample design of the study?

Solution:

One dataset was at the household level - with household level characteristics (like water treatment); the other dataset was at the child level - with child health outcomes (like diarrhea prevalence).
The survey sample was for a set of households at springs, and for the children under 5 at each household.

2 An Intergalactic Analysis [16 Pts]

A new season of "The Mandalorian" has aired recently (no spoiler in this question) and as an ultimate Star Wars fan, Leon has embarked on a journey to dig deeper into the backstory of the show. He found several secretly hidden datasets online that contain information about the planets, ships, and other intriguing local records.
Throughout this question, we are dealing with pandas DataFrame and Series objects. All code for this question, where applicable, must be written in Python, unless explicitly stated otherwise. You may assume that pandas has been imported as `pd`.

(a) [1 Pt] Leon wants to do his analysis in the Python Jupyter Notebook. But when he tries to import the dataset he found online, it returns some weird error saying that "utf-8 cannot decode ...". He realizes that the datasets are actually coded in Galactic Basic (with a codex called `sw-gbc`). Help Leon to import one of the dataset `farm.csv` into the notebook.

Fill in the code below.

```
# load in `farm.csv` that is encoded in `sw-gbc`
nevarro_farm = _____
```

Solution:

```
nevarro_farm = pd.read_csv("farm.csv", \
encoding="sw-gbc")
```

(b) [1 Pt] Now we got the data of fruits and vegetables on planet Nevarro. Leon is only interested in the fruits of Nevarro and the fruits must be red. Filter the dataframe so that it contains only data on fruits that are red.

	Crop	Type	Price	Is red?
0	Starfruit	Fruit	750	False
1	Parsnip	Vegetable	35	True
2	Sweet Gem Berry	Fruit	3000	True
3	Red Cabbage	Vegetable	260	True

Fill in the code below.

```
nevarro_farm_red_fruits_only = \
nevarro_farm[_____]
```

Solution:

```
nevarro_farm_red_fruits_only = nevarro_farm[\
(nevarro_farm["Type"] == "Fruit") \
& (nevarro_farm["Is red?"] == True)]
```

(c) [2 Pts] Leon did something else with this dataframe, but unfortunately his kernel was dead on datahub and lost all his code. But he remembered that the output was "True False True False". Which of the following code can produce this output? Select all that apply.

	Crop	Type	Price	Is red?
0	Starfruit	Fruit	750	False
1	Parsnip	Vegetable	35	True
2	Sweet Gem Berry	Fruit	3000	True
3	Red Cabbage	Vegetable	260	True

→

```
0     True
1    False
2     True
3    False
```

☐ **`nevarro_farm["Crop"].str.startswith("S")`**

☐ `nevarro_farm["Is red?"]`

☐ `nevarro_farm[nevarro_farm["Type"] == "Fruit"]`

☐ **`nevarro_farm["Type"] == "Fruit"`**

☐ None of the above

Solution:

Note that choice c is incorrect as it outputs the dataframe filtered by the boolean conditions, not the boolean condition itself.

(d) [4 Pts] Leon is interested in smuggling some of those Red Star Fruits through the blockade. Leon found another dataset called `ships`, that describes all the types of ships in the universe.

	name	model	manufacturer	cost_in_credits	length	max_atmosphering_speed	cre
0	CR90 corvette	CR90 corvette	Corellian Engineering Corporation	3500000	150	950	30-16
1	Star Destroyer	Imperial I-class Star Destroyer	Kuat Drive Yards	150000000	1,600	975	47,06
2	Sentinel-class landing craft	Sentinel-class landing craft	Sienar Fleet Systems, Cyngus Spaceworks	240000	38	1000	
3	Death Star	DS-1 Orbital Battle Station	Imperial Department of Military Research, Sien...	1000000000000	120000	n/a	342,95

```
print (ships.info())

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36 entries, 0 to 35
Data columns (total 18 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   name                    36 non-null     object
 1   model                   36 non-null     object
 2   manufacturer            36 non-null     object
 3   cost_in_credits         36 non-null     object
 4   length                  36 non-null     object
 5   max_atmosphering_speed  36 non-null     object
 6   crew                    36 non-null     object
 7   passengers              36 non-null     object
 8   cargo_capacity          36 non-null     object
 9   consumables             36 non-null     object
 10  hyperdrive_rating       36 non-null     object
 11  MGLT                    36 non-null     object
 12  starship_class          36 non-null     object
 13  pilots                  36 non-null     object
 14  films                   36 non-null     object
 15  created                 36 non-null     object
 16  edited                  36 non-null     object
 17  url                     36 non-null     object
```

Leon wants to do some analysis of the ships. Write some Pandas commands to do the following:

A Order the ships from biggest to smallest based on cargo capacity. Return all columns.

Solution:

```
ships.sort_values("cargo_capacity", \
ascending=False)
```

B Find the average cost by manufacturer. Return the manufacturer names and the average costs.

Solution:

```
ships.groupby("manufacturer")\
["cost_in_credits"].mean()
```

C Find the lowest-cost ship with hyperdrive rating of 4. Return all columns for that ship (if there are multiple ships with the lowest cost, you can return whichever one)

Solution:

```
ships[ships["hyperdrive_rating"] == 4]\
.sort_values("cost_in_credits").iloc[0]
```

D Find any ship that can carry a cargo of at least 1 million kilos but with a length less than 200. Return all columns for those ships.

Solution:

```
ships[(ships["cargo_capacity"] >= 1000000) \
& (ships["length"] < 200)]
```

(e) [4 Pts] There's another database called `ships` in SQL with the same data structure (or schema) as the one above! Now do some additional analyses with SQL.

	name	model	manufacturer	cost_in_credits	length	max_atmosphering_speed	cre
0	CR90 corvette	CR90 corvette	Corellian Engineering Corporation	3500000	150	950	30-16
1	Star Destroyer	Imperial I-class Star Destroyer	Kuat Drive Yards	150000000	1,600	975	47,06
2	Sentinel-class landing craft	Sentinel-class landing craft	Sienar Fleet Systems, Cyngus Spaceworks	240000	38	1000	
3	Death Star	DS-1 Orbital Battle Station	Imperial Department of Military Research, Sien...	1000000000000	120000	n/a	342,95

```
print (ships.info())
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 36 entries, 0 to 35
Data columns (total 18 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   name                  36 non-null     object
 1   model                 36 non-null     object
 2   manufacturer          36 non-null     object
 3   cost_in_credits       36 non-null     object
 4   length                36 non-null     object
 5   max_atmosphering_speed 36 non-null    object
 6   crew                  36 non-null     object
 7   passengers            36 non-null     object
 8   cargo_capacity        36 non-null     object
 9   consumables           36 non-null     object
 10  hyperdrive_rating     36 non-null     object
 11  MGLT                  36 non-null     object
 12  starship_class        36 non-null     object
 13  pilots                36 non-null     object
 14  films                 36 non-null     object
 15  created               36 non-null     object
 16  edited                36 non-null     object
 17  url                   36 non-null     object
```

Write some SQL queries to do the following:

A Order the ships from biggest to smallest based on cargo capacity. Return all columns.

Solution:

```
SELECT * FROM ships
ORDER BY cargo_capacity DESC
```

B Find the average cost by manufacturer. Return the manufacturer names and the average costs.

Solution:

```
SELECT manufacturer, AVG(cost_in_credits)
FROM ships
GROUP BY manufacturer
```

C Find the lowest-cost ship with hyperdrive rating of 4. Return all columns for that ship (if there are multiple ships with the lowest cost, you can return whichever one)

Solution:

```
SELECT * FROM ships
WHERE hyperdrive_rating = 4
ORDER BY cost_in_credits ASC
LIMIT 1
```

D Find any ship that can carry a cargo of at least 1 million kilos but with a length less than 200. Return all columns for those ships.

Solution:

```
SELECT * FROM ships
WHERE cargo_capacity >= 1000000
AND length < 200
```

Leon came across a dataset called `planets` and started to look at the description of the planets. Leon is interested in finding a suitable planet for a young padawan to train. He has a theory that planets with a temperate climate have a relatively shorter orbital period.

`planets`

	name	rotation_period	orbital_period	diameter	climate	gravity	terrain	s
0	Tatooine	23	304	10465	arid	1 standard	desert	
1	Alderaan	24	364	12500	temperate	1 standard	grasslands, mountains	
2	Yavin IV	24	4818	10200	temperate, tropical	1 standard	jungle, rainforests	
3	Hoth	23	549	7200	frozen	1.1 standard	tundra, ice caves, mountain ranges	

(f) [2 Pts] How can Leon test whether planets with a temperate climate have a different orbital period than planets with other types of climate? Answer in both words and code.

Solution:

First, we want to find out which planets have a temperate climate. To do so, we can use `planets["climate"].str.contains("temperate")`

and then assign this list of booleans to a new column `temperate_climate`. Next we want to compare the distribution of orbital periods for the planets that have a temperate climate versus those that do not. The simpliest method is to compare the mean or the median in these two groups. To do so, we can use

```
planets.groupby("temperate_climate")["orbital_period"].mean()
```

Alternatively, using barplots, histograms, etc. for these two groups is also a good way to identify the potential difference in the distribution.

(g) [2 Pts] How can Leon use a violin graph to test this hypothesis? Answer in both words and code.

Solution:

A violin graph can visualize the distributions of numerical data for multiple categories using density plots. For our purposes, we want to examine the distribution of orbital periods for planets that have a temperate climate versus those that do not.

```
sns.violinplot(data=planets, x="temperate_climate", y="orbital_period")
```

By comparing the two density plots in this violin graph, we can see if there's any difference in the distributions. Alternatively, we can also generate density plots for all types of climates.

3 Phillips Curve [16 Pts]

Two officials at the New York Fed – J Bow and G Pow – are having a heated debate over what policy they should set to combat the soaring inflation. First they want to look at several macro indicators that can summarize the current state of the economy.

All code for this question, where applicable, must be written in Python, unless explicitly stated otherwise. You may assume that pandas has been imported as `pd`.

(a) [1 Pt] Name one macro indicator that can represent the inflation rate?

Solution:

CPI or PCE or any other relevant indicator.

(b) [2 Pts] Which two economic variables does the Phillips Curve capture? Draw a classical Phillips curve (label your plot properly).

Solution:

Inflation rate and unemployment rate. Plot omitted.

(c) [2 Pts] In 3 sentences or less, describe the underlying mechanisms of the hypothesized Phillips curve.

Solution:

The underlying mechanism can be explained by the relationship between a tightening labor pool and the increased marginal costs associated with hiring another worker and its effect on overall costs and inflation rate. To explain it in more detail: when the opportunities are good, firms will decide to hire more labor to increase their output. This will result in lower unemployment. As the firms are hiring, the labor pool (people willing to work) dwindled, and in order to attract talents remaining, firms will raise wages. At the same time, higher wages will drive up the production costs for the firms, and they will pass that increased costs to consumers in forms of price increases. Therefore, the overall price level in the economy rises, and there's inflation! The reverse is also true.

Other reasonable explanations without referencing the labor market can also be awarded credit.

J Bow and G Pow are suspicious of the theory presented in the textbook. They want to see if the Phillips Curve is actually true using empirical evidence – with some datasets.

(d) [2 Pts] In 2 sentences or less, describe your optimal dataset to test the hypothesized Phillips curve (also note the time frame and frequency). Name one data source that you can think of that contains the dataset of your interest.

Solution:

Inflation and unemployment rate data needs to be collected. The optimal timeframe could be the last 50 years (since it contains data for both high and low inflation and unemployment

rate). The frequency could be quarterly. In this case, FRED can be a good data source.

(e) [2 Pts] You discovered that there is an API from Econ148.org that contains the relevant data series for the Phillips curve. In this API, the relevant data series id are `x-series` and `y-series` respectively; and there's a valid API key called `DEMO_KEY`.

Look at the API documentation provided below, and fetch relevant data for `x-series` from Jan. 1st, 1960 to Mar. 10th, 2023.

The url is `https://econ148.org`.
The endpoint is `series/observations`.

Parameters

1. api_key: The API Keys for this data source.
2. series_id: The id for a series.
3. observation_start: The start of the observation period as YYYY-MM-DD formatted string, optional, default: 1776-07-04 (earliest available)
4. observation_end: The end of the observation period as YYYY-MM-DD formatted string, optional, default: 9999-12-31 (latest available)

Fill in the code below.

```
import requests
from urllib.parse import urlencode

base_url = _____________________
endpoint = _____________________
params = {
_________________ : _________________,
_________________ : _________________,
_________________ : _________________,
_________________ : _________________
}

url_params = urlencode(params)
url = _____________________

# fires off the request
res = requests.get(url)

# return the content of the response
return res.json()

(additional data processing omitted)
```

Solution:

```
base_url = "https://econ148.org"
endpoint = "/series/observations"
params = {
"api_key" : "DEMO_KEY",
"series_id" : "x-series",
"observation_start" : "1960-01-01",
"observation_end" : "2023-03-10"
}

url_params = urlencode(params)
url = f"{base_url}{endpoint}?{url_params}"
```

(f) [2 Pts] You have collected data for J Bow and G Pow, and now you want to produce a visualization of the hypothesized Phillips Curve. Which type of visualization would you choose to

use (line plot, histogram, scatterplot, barchart, etc.) and why?

Solution:

A scatter plot is overall the best choice. A scatter plot can help us to visualize the relationship between two numerical variables.

(g) [3 Pts] Given two datasets (`x.csv` and `y.csv`) that you have collected and stored on your laptop, fill out the following code in Python to generate a plot you describe above. In the final plot, only include points where there is x and y data for the corresponding date in both of the original dataframes. You may use any methods in Pandas and plotting libraries in Python.

The dataframes look like the following:

dataframe x

	DATE	x
0	1948-01-01	3.4
1	1948-02-01	3.8
2	1948-03-01	4.0
3	1948-04-01	3.9
4	1948-05-01	3.5
...	...	...
895	2022-08-01	3.7
896	2022-09-01	3.5
897	2022-10-01	3.7
898	2022-11-01	3.6
899	2022-12-01	3.5

900 rows × 2 columns

dataframe y

	DATE	y
0	1958-01-01	2.79720
1	1958-04-01	2.42775
2	1958-07-01	2.06659
3	1958-10-01	1.82232
4	1959-01-01	1.81406
...	...	...
255	2021-10-01	5.00808
256	2022-01-01	6.29779
257	2022-04-01	6.01857
258	2022-07-01	6.29602
259	2022-10-01	5.98561

260 rows × 2 columns

Assume dataframe y is the inflation rate, and dataframe x is for the other variable in the Phillips Curve.

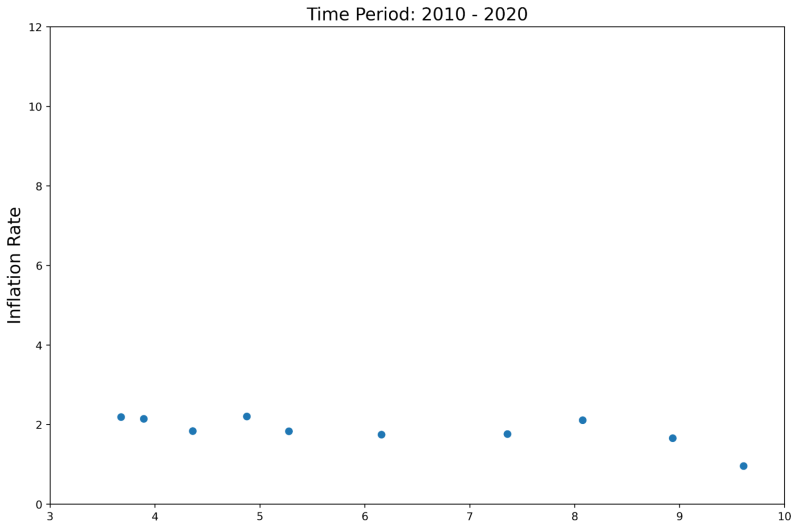
```
# load in the datasets
x = _____________________
y = _____________________

# merge the datasets
pc_df = _____________________

# make a visualization of the Phillips curve
(label both axes and title properly)
_____________________________________________
_____________________________________________
_____________________________________________
_____________________________________________
_____________________________________________
_____________________________________________
_____________________________________________
(You may not need all the lines)
```

Solution:

```
# load in the datasets
x = pd.read_csv("x.csv")
y = pd.read_csv("y.csv")

# merge the datasets
pc_df = pd.merge(x, y, on="DATE", how="inner")

# make a visualization of the Phillips curve
(label both axes and title properly)

# Note: order doesn't matter as long as they
# label it correctly
plt.scatter(pc_df["x"], pc_df["y"])
plt.xlabel("Unemployment rate")
plt.ylabel("Inflation rate")
plt.title("Pillips Curve")
```

(h) [2 Pts] Below is a graph that G Pow found online for data in 2010-2020. Does this piece of

empirical data support the hypothesized Phillips curve? If not, explain one potential way to reconcile this difference.



Solution:

No, the empirical data doesn't seem to fully support the classical Phillips curve. We could take into account people's inflation expectations in the model.