Data 8
Fall 2018

Foundations of Data Science

Midterm Solutions

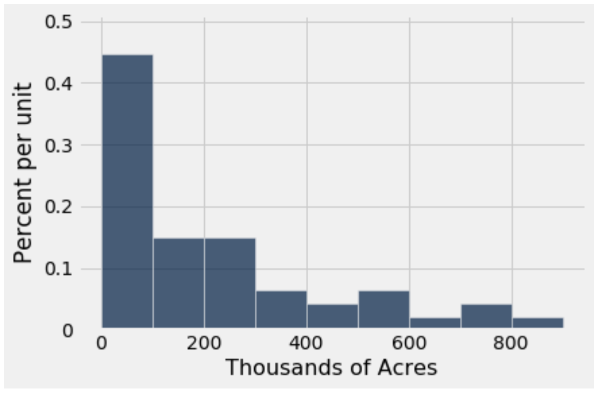
INSTRUCTIONS

- You have 80 minutes to complete the exam.
- The exam is closed book, closed notes, closed computer, closed calculator, except one hand-written 8.5” × 11” sheet of notes of your own creation and the official midterm exam reference guide provided with the exam.
- Mark your answers **on the exam itself**. We will *not* grade answers written on scratch paper.
- For all Python code, you may assume that the statements `from datascience import *` and `import numpy as np` have been executed. Do not use features of the Python language that have not been described in this course.

Last name	
First name	
Student ID number	
Calcentral email (`_@berkeley.edu`)	
Lab GSI	
Name of the person to your left	
Name of the person to your right	
All the work on this exam is my own. **(please sign)**	

1. (15 points) Histograms

We gathered a data set of US national parks, and plotted below a histogram of the size of these parks (in thousands of acres). All bars are 100 wide. The area of the visible bars sum to 100%.



Calculate each quantity described below or write *Unknown* if there is not enough information above to express the quantity as a single number (not a range). It's OK to write your answer as a Python expression or an unsimplified expression. You may need to estimate the height of bars visually; if so, make your best estimate. Don't show your work.

(a) (4 pt) The percentage of parks that are less than 100 thousand acres in size.

$0.45 \times 100 = 45\%$.

(b) (4 pt) The percentage of parks that are between 200 to 400 thousand acres in size (specifically, the range $[200, 400)$, i.e., at least 200 thousand acres and less than 400 thousand acres).

$(0.15 + 0.06) \times 100 = 21\%$.

(c) (4 pt) The percentage of parks that are less than 150 thousand acres in size.

Unknown. (We don't know the sizes of the parks in the $[100, 200)$ bin; they could all be on the left half of the bin, or in the right half, or any mixture of those two.)

(d) (3 pt) The percentage of parks that are at least 1200 thousand acres in size.

0%. (It's to the right of all the bars.)

2. (30 points) Table operations

A table `insurance` contains one row for each beneficiary that is covered by a particular insurance company:

age	bmi	smoker	region	cost
25	20.8	no	southwest	3208.79
25	30.2	yes	southwest	33900.71
62	32.1	no	northeast	1355.50

```
... (20198 rows omitted)
```

The table contains five columns:

- **age**: an int, the age of the beneficiary
- **bmi**: a float, the Body Mass Index (BMI) of the beneficiary
- **smoker**: a string, which indicates whether the beneficiary smokes
- **region**: a string, the region of the United States where the beneficiary lives
- **paid**: a float, the total amount of medical costs that the insurance company paid for this beneficiary last year

In each part below, fill in the blanks of the Python expression to compute the described value. **You must use ONLY the lines provided.** Do not write any code outside the blanks provided. The code in the line should evaluate to the value described.

(a) **(3 pt)** The largest amount paid by the insurance company last year to any beneficiary.

```
max(insurance.column('paid'))
```

(b) **(3 pt)** A table similar to `insurance`, but with only the rows for smokers, and without the **smoker** column.

```
insurance.where('smoker', are.equal_to('yes')).drop('smoker').
```

Also valid: `insurance.where('smoker', 'yes').drop('smoker')`

(c) **(4 pt)** A table that is the same as `insurance`, but with an added column named `predicted_paid` that shows the amount we predict to pay to that beneficiary this year, if we predict that their cost this year will be the cost last year increased by 5%.

```
a = insurance.column('paid') * 1.05
insurance.with_column('predicted_paid', a)
```

(d) **(3 pt)** A table containing one row per region, listing for each region the total amount paid by the insurance company last year to all beneficiaries in that region.

```
insurance.group('region', sum).select('region', 'paid sum')
```

(e) **(4 pt)** The name of the region with the smallest number of beneficiaries.

```
by_region = insurance.group('region')
t = by_region.sort('count')
t.column('region').item(0)
```

(f) **(3 pt)** A scatter plot comparing BMI vs amount paid last year for only the beneficiaries whose cost exceeded $25,000 (i.e., with one dot per beneficiary whose cost last year exceeded $25,000).

```
high_cost = insurance.where('paid', are.above(25000))
high_cost.scatter('bmi', 'paid')
```

(g) **(4 pt)** Write a function that takes an age as a parameter, and returns the average BMI of all beneficiaries of that age.

```
def average_bmi(age):
    right_age = insurance.where('age', are.equal_to(age))
    bmis = right_age.column('bmi')
    avg = sum(bmis) / len(bmis)
    return avg
```

3. (30 points) Hypothesis testing

A scientist claims that in a particular region, 60% of bees are honey bees, 30% are bumblebees, and 10% are carpenter bees. You are suspicious of this claim, so you take a uniform random sample of 100 bees in the area. You get the following results from your random sample:

Bee Type	**Number of bees**
Honey bee	72
Bumblebee	18
Carpenter bee	10

You'll use the following answer bank in parts (a)–(b) of this question:

⟨A⟩ Each bee is chosen randomly, with a 60% chance of being a honey bee, 30% chance of bumblebee, and 10% of carpenter bee.

⟨B⟩ 72 of the bees in the observed sample were honey bees, 18 were bumblebees, and 10 were carpenter bees.

⟨C⟩ Bees are distributed according to the probability distribution specified by the scientist.

⟨D⟩ Bees are not distributed according to the probability distribution specified by the scientist.

⟨E⟩ The sample was not chosen randomly, because of a bias in how bees were selected.

⟨F⟩ The observed distribution in the sample is not consistent with the scientist's claim. The difference between the observed vs. claimed distribution is highly statistically significant ($p < 0.01$).

(a) **(4 pt)** In order to determine whether the scientist's claim is actually true, you want to perform a hypothesis test. Which of the following could be a reasonable null hypothesis? Fill in the oval next to **all** that are reasonable possibilities.

[✓ ⟨A⟩] ⟨B⟩ [✓ ⟨C⟩] ⟨D⟩ ⟨E⟩ ⟨F⟩

(b) **(3 pt)** Which of the following could be a reasonable alternative hypothesis? Fill in the oval next to **all** that are reasonable possibilities.

⟨A⟩ ⟨B⟩ ⟨C⟩ [✓ ⟨D⟩] ⟨E⟩ ⟨F⟩

⟨E⟩ is not a suitable alternative hypothesis. The question says that we are testing the scientist's claim. We are the ones who control how the sampling was done, so we're not testing whether it was done randomly (the question tells us it was random).

(c) **(4 pt)** You use the null and alternative hypothesis from parts (a)–(b) in a hypothesis test. You decide to use the total variation distance (TVD) between the empirical distribution (in the sample) and the probability distribution specified by the scientist as your test statistic. Fill in the code below so it computes this value and assigns it to `observed_tvd`.

```
def tvd(pred, obs):
    return sum(np.abs(pred - obs)) / 2

predicted = make_array(0.60, 0.30, 0.10)
observed = make_array(0.72, 0.18, 0.10)
observed_tvd = tvd(predicted, observed)
```

(d) **(8 pt)** Fill in the following code, so it will correctly perform a hypothesis test using the TVD as test statistic and compute the empirical p-value. Assume the code in part (c) correctly computes the TVD. You may use variables/names defined in part (c).

```
simulated_tvds = make_array()
for i in np.arange(10000):
    simulated_sample = sample_proportions(100, predicted)
    statistic = tvd(predicted, simulated_sample)
    simulated_tvds = np.append(simulated_tvds, statistic)
empirical_pval = np.count_nonzero(simulated_tvds >= observed_tvd)/10000
empirical_pval
```

(e) **(5 pt)** Suppose that the result of your computation is `empirical_pval = 0.0258`. Which of the following conclusions would be justified? Fill in the oval next to **all** that apply.

- ● If we use a p-value cut-off of 5%, we should reject the null hypothesis.
- ○ If we use a p-value cut-off of 1%, we should reject the null hypothesis.
- ○ Bees are distributed according to the probability distribution specified by the scientist.
- ○ The scientist's claim is wrong; honey bees occur in this region with a higher probability than the scientist claimed.
- ○ The sample was not chosen randomly, because of a bias in how bees were selected.
- ● The observed distribution in the sample is not consistent with the scientist's claim. The difference between the observed vs. claimed distribution is statistically significant ($p < 0.05$).

The 4th option is not correct. The above code tests whether there is a difference in the overall distribution; it doesn't test anything about honeybees specifically. Thus, we are not entitled to conclude (based on the fact that `empirical_pval = 0.0258`) anything about honeybees specifically. If we wanted to draw a conclusion about honeybees, we'd need to do a hypothesis test with a different test statistic that looks solely at the number of honeybees (rather than the TVD).

The 5th option is not correct. As mentioned in part (b), we are testing the scientist's claim, not whether the sample was taken randomly. Also, even if we didn't know whether the sample was taken randomly or not, we aren't entitled to conclude (from the fact that `empirical_pval = 0.0258`) that the sample was non-random; it could be that the actual distribution of bees differs from the scientist's claim, and the sample was chosen randomly.

(f) **(2 pt)** Suppose you rerun the hypothesis testing code above, but this time you replace both instances of `10000` with `100000` (thus, you do 10× as many iterations of the loop). Which of the following describes what we should expect to happen to `empirical_pval`? Fill in the oval next to **one** answer.

- ○ It should be about 10× larger (i.e., about `0.258`, or a little bigger or a little smaller).
- ○ It should be about 10× smaller (i.e., about `0.00258`, or a little bigger or a little smaller).
- ● It should be about the same (i.e., it remains at about `0.0258`, or a little bigger or a little smaller).

This follows from the law of averages. `empirical_pval` is a proportion of the time that something happens in many (identical, independent) iterations a random process; doing more iterations will give you approximately the same proportion.

(g) **(4 pt)** Suppose we wanted to test only whether bumblebees appear to be as common as the scientist claimed, and we don't care about honey bees or carpenter bees. In particular, suppose our null hypothesis is that each bee has a 30% probability of being a bumblebee, and our alternative hypothesis is that bumblebees are more common than that. Which of the following would be a good choice of test statistic, for this null and alternative hypothesis? Assume that the observed sample has 100 bees and `numbumbles` is the number of bumblebees in the observed sample. Fill in the oval next to **all** choices that are good.

- ● `numbumbles`
- ● `numbumbles/100`
- ○ `abs(numbumbles - 10)`
- ○ `abs((numbumbles/100) - 0.10)`

- ○ `tvd(make_array(numbumbles/100, 1 - (numbumbles/100)), make_array(0.10, 0.90))`
- ○ The total variation distance between the observed distribution of bees and the probability distribution specified by the scientist.
- ○ The number of bees in the observed sample.
- ○ The empirical distribution of bees in the observed sample.

Comment: This problem was easier than anticipated. Because of an oversight when writing the problem, the numbers `10` and `0.10` above were intended to be `30` and `0.30`. Thus, in their current form, the other options are certainly incorrect. Option 6 is incorrect because we want to test something about bumblebees specifically, and the TVD takes into account all three types of bees.

Even if we had changed `10` and `0.10` to `30` and `0.30`, options 3–7 would still be incorrect. The 3rd and 4th options would be reasonable if our alternative hypothesis said "the null hypothesis is not correct", but that wasn't our alternative hypothesis; our alternative more specifically says that bumblebees are more common than the scientist's model, and the absolute value isn't appropriate for that.

4. (25 points) Probability

I go to Monterey Market to pick out some pumpkins to decorate my house for Halloween. They have a special October deal where you get 3 randomly chosen gourds for $10. Each gourd can be either an orange pumpkin, green pumpkin, or squash. The probability for each is shown in the table below:

Type	**Probability**
Orange Pumpkin	0.6
Green Pumpkin	0.3
Squash	0.1

I decide to get the special deal!

Answer the following questions. It's OK to write your answer as a Python expression or an unsimplified expression—you do not need to evaluate it to get a number.

(a) (4 pt) What is the probability that the first gourd is an orange pumpkin, and the second gourd is a green pumpkin, and the third gourd is a squash?

$0.6 \times 0.3 \times 0.1$ (multiplication rule)

(b) (2 pt) What is the probability that I get three orange pumpkins?

$0.6 \times 0.6 \times 0.6$ (multiplication rule again)

(c) (4 pt) What is the probability that I get no squashes?

$0.9 \times 0.9 \times 0.9$ (use the addition rule to note that the probability of getting a squash the first time is $0.6 + 0.3 = 0.9$, then use the multiplication rule)

(d) (3 pt) What is the probability that I get at least one orange pumpkin?

$1 - 0.4 \times 0.4 \times 0.4$ ($1-$ the probability of getting no orange pumpkins)

It turns out this special October deal spans back one hundred years! Monterey Market has kept track of the price of the deal for each year from 1919 to today (2018). We are given the data in the form of an array of 100 integers named `prices`. The first item in the array is the price of the deal in 1919: $1.50. The last item in the array is the price of the deal today: $10.00.

```
array([ 1.50 , 3.99 , 2.75 , ... , 8.99 , 12.50 , 10.00])
```

For each of the following quantities, write a Python expression that computes it.

(e) (3 pt) Whether or not the price has ever been $1.00 or less (your code should evaluate to `True` or `False`).

`min(prices) <= 1`
Also valid: `np.count_nonzero(prices <= 1) > 0` or `np.average(prices <= 1) > 0` or `sum(prices <= 1) > 0`

(f) (3 pt) The number of times the price increased by more than $2.00 from one year to the next. (Hint: use `np.diff`.)

`np.count_nonzero(np.diff(prices) > 2)`
Also valid: `sum(np.diff(prices) > 2)` and other similar possibilities as well.

(g) (3 pt) The probability of choosing a price greater than $7.00, if I pick one of the prices uniformly at random.

`np.count_nonzero(prices > 7) / len(prices)`
Also valid: `np.count_nonzero(prices > 7) / 100` or `np.average(prices > 7)`, and other similar possibilities as well.

5. (0 points) Write your name in the space provided on one side of every page of the exam. You're done!