Problem Set 9

UGBA 96: Data and Decisions, Fall 2018

In [1]:
```
from datascience import *
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
plt.style.use('fivethirtyeight')
import gsExport
```

Deadline: This assignment is due **Friday, November 30th at 11:59pm**. Late work will not be accepted.

You will submit your solutions using both OKpy and Gradescope. You will find detailed submission instructions at the bottom of this notebook and on bCourses (here (https://bcourses.berkeley.edu/files/73630077/download?download_frd=1)). **Please do not remove or add cells and please ignore the '#newpage' cells** (these are here to facilitate Gradescope submission).

You should start early so that you have time to get help if you're stuck. Post questions on Piazza. Check the syllabus for the office hours schedule. Remember that Connector Assistant office hours are for *coding questions only*.

(60 points)

Background

In this problem set you will replicate the main results from the paper *Learning from the Crowd: Regression Discontinuity Estimates of the Effects of an Online Review Database*, written by Michael Anderson and Jeremy Magruder. Both authors are on the faculty at UC Berkeley's department of Agricultural and Resource Economics. You can find the paper itself on bCourses. **You will need to reference the paper to complete this problem set.**

The authors study the following research question: do online reviews drive consumer demand? More specifically, the authors estimate the causal effect of positive Yelp ratings on restaurant reservation availability.

Data

Prof. Anderson and Prof. Magruder have graciously provided us with the data they collected for their paper.

The authors collected two data sets and merged them. The first data set is the full history of reviews for each San Francisco restaurant on Yelp.com as of February 2011. Using this database, the authors reconstructed the average rating and total number of reviews for each restaurant at every point in time.

The authors combine these data with reservation availability data from a large online reservation website (e.g., Open Table or Reserve). This website lists real-time reservation availability for a subset of the restaurants in the Yelp data. From July 21st to October 29th 2010, the authors recorded reservation availability for a party of four on Thursday, Friday and Saturday evenings. They checked the availability at 7 and 8 pm. Availability was measured approximately 36 hours prior to the time of the desired reservation.

We will work with a subset of their data that will be sufficient for replicating the main results. Each row of data corresponds to a restaurant at a given night of reservation availability. Here is a description of each column in the dataset:

- `restaurant_id` : ID for restaurant
- `restaurant` : Name of restaurant
- `date` : Night of reservation
- `wk_id` : Week of reservation
- `total_reviews` : Total reviews on Yelp
- `avg_rating` : Average review rating of restaurant on Yelp
- `display_rating` : Restaurant rating *displayed* to Yelp visitors (in half star increments)
- `neighborhood` : San Francisco neighborhood of restaurant
- `yelp_category` : Cuisine type
- `delivers` : Indicator for whether restaurant delivers
- `takeout` : Indicator for whether restaurant sells take out food
- `price` : price category of restaurant (`$`, `$$`, `$$$`, `$$$$`)
- `avail_7pm` : Indicator for whether restaurant is available at 7pm on `date`
- `avail_8pm` : Indicator for whether restaurant is available at 8pm on `date`

In [2]:
```
#run this cell to load the data

#read in data
yelp_data = Table.read_table("yelp_data.csv")

delivers = np.int32(yelp_data.column('delivers') == 'Yes')
takeout = np.int32(yelp_data.column('takeout') == 'Yes')

yelp_data = yelp_data.with_column('delivers', delivers).with_column('takeout', takeout)

yelp_data.show(5)
```

restaurant_id	restaurant	date	wk_id	total_reviews	avg_rating	display_rating	neighborhood	yelp_category	delivers	tak
2	thirsty-bear-brewing-company-san-francisco	30oct2010	44	795	3.23145	3	SOMA	Breweries	0	
2	thirsty-bear-brewing-company-san-francisco	29oct2010	44	794	3.23048	3	SOMA	Breweries	0	
2	thirsty-bear-brewing-company-san-francisco	28oct2010	43	793	3.22965	3	SOMA	Breweries	0	
2	thirsty-bear-brewing-company-san-francisco	23oct2010	43	791	3.2263	3	SOMA	Breweries	0	
2	thirsty-bear-brewing-company-san-francisco	22oct2010	43	791	3.2263	3	SOMA	Breweries	0	

... (14235 rows omitted)

Part I: Descriptive Statistics

Below are some basic descriptive statistics from the paper that provide useful context. Note: the corresponding values in your data will vary somewhat from the numbers in this table, but not by much.

Table 1

Summary Statistics

	All restaurants		Restaurants with reservation data	
	Mean (SD)	Range	Mean (SD)	Range
Reviewer's rating	3.69 (1.13)	1–5	3.68 (1.11)	1–5
Reviews per restaurant	167.9 (248.5)	1–2,929	452.1 (344.4)	28–2,236
Restaurant's average rating	3.50 (0.68)	1–5	3.64 (0.34)	2.68–4.41
Monthly reviews per restaurant (September 2010)	5.48 (6.76)	0–101	9.21 (7.21)	1–45
Restaurants	3,953		328	
Unique reviews	663,790		148,281	
Unique reviewers	108,547		50,409	
Reservation availability at 6 PM			0.74 (0.44)	
Reservation availability at 7 PM			0.59 (0.49)	
Reservation availability at 8 PM			0.68 (0.47)	

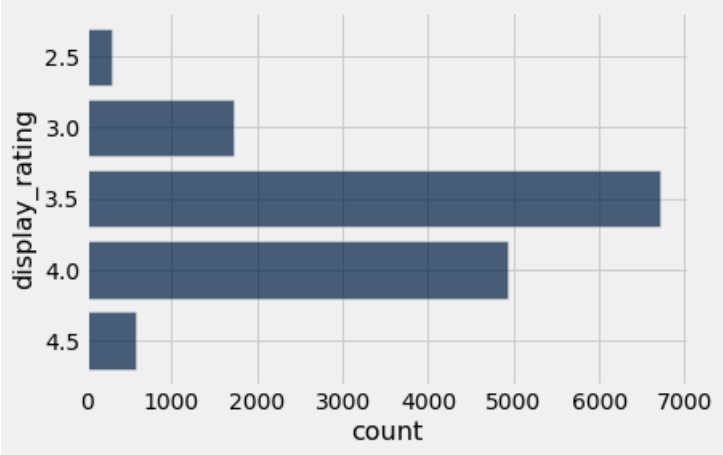
Notes. Availability measures indicate whether the reservations were available at that time on Thursday, Friday or Saturday when queried 36 hours in advance.

There are about 325 restaurants represented in our data. The average review rating for a restaurant is about 3.6 stars. Restaurants are available for a 7pm reservation about 59% of the time.

newpage

a. (2 points) Plot a barplot of `display_rating`. This should display the frequency of each value of `display_rating` in the data.

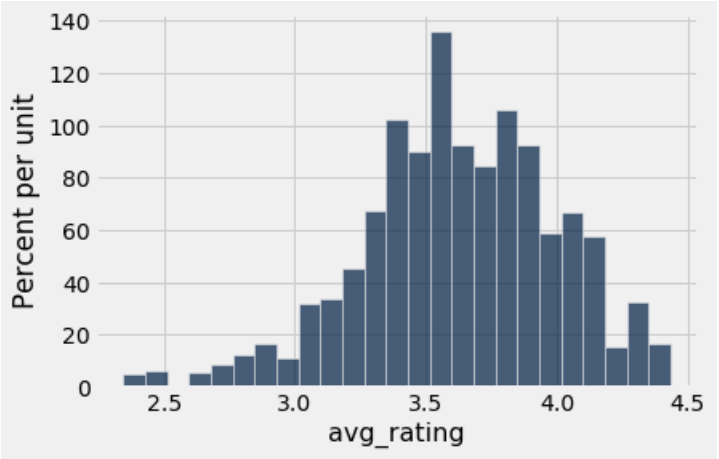
```
In [3]: yelp_data.group('display_rating').barh('display_rating')
```



`display_rating` takes a discrete set of values, 1 to 5 in half star increments (though our sample takes a more limited range). This masks substantial variation in `avg_rating` across restaurants, which is a simple average of all past Yelp reviews for the restaurant.

b. (2 points) Plot a histogram of `avg_rating` .

```
In [4]: yelp_data.hist('avg_rating', bins = 25)
```



Even among restaurants with the same `display_rating` , there is substantial variation in `avg_rating` .

newpage

Part II: Regression Approach

Our objective is to measure the average causal effect of Yelp star rating on reservation availability.

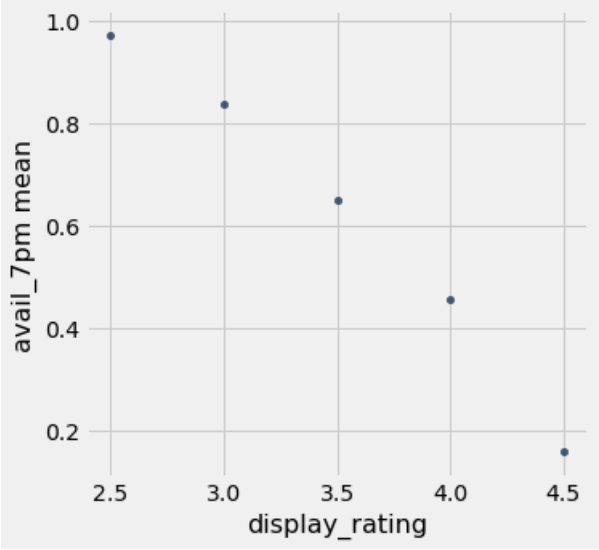
c. (4 points) Describe a hypothetical experiment that would identify the causal effect of Yelp star ratings on reservation availability.

A: In an ideal experiment, we could randomly assign Yelp star ratings to restaurants and compare reservation availability across restaurants with different star ratings.

In the absence of an experiment, a natural approach to use here is **regression**. In this section we will estimate the relationship between Yelp star ratings and availability.

d. (2 points) Create a scatter plot with `display_rating` on the horizontal axis and the average value of `avail_7pm` by display rating on the vertical axis. (Hint: your plot should have 5 points total.)

In [5]:
```
yelp_data.group('display_rating', collect = np.mean).scatter('display_rating', 'avail_7pm mean'
)
```



e. (3 points) Based on your plot from **part (d)**, describe the relationship between `display_rating` and `avail_7pm` .

A: Availability declines sharply with the display rating.

f. (5 points) Estimate a regression model with `avail_7pm` as the dependent variable and `display_rating` as the explanatory variable. Report the coefficients. Interpret the coefficent on `display_rating` in a full sentence. (Be sure to mention the *magnitude* of the estimate and not just the *sign*.)

In [6]:
```
#write your code here: define MSE function
def avail7_mse(slope, intercept):
    x = yelp_data.column('display_rating')
    y = yelp_data.column('avail_7pm')
    fitted = intercept + slope*x
    return np.mean((y - fitted) ** 2)
```

In [7]:
```
#write any additional code here
coefficients = minimize(avail7_mse)
coefficients
```

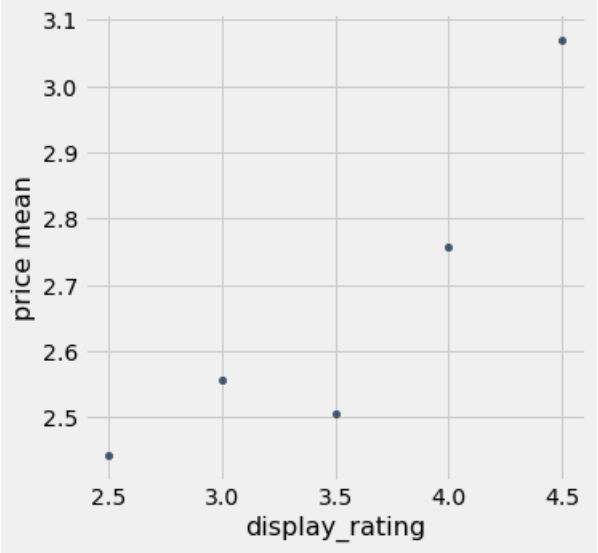
Out[7]:
```
array([-0.39682288,  2.03257346])
```

A: An increase in the display rating by 1 star is associated with a 39.7 percentage point decrease in availability for 7pm reservations.

This coefficient may not have a causal interpretation. Restaurants with higher display ratings may have less availability for reasons unrelated to the causal effect of the display rating. Next, we investigate `price` as a potential confounding variable. [We will assume here that restaurants do not change their pricing in response to their Yelp rating.]

g. (2 points) Create a scatter plot with `display_rating` on the horizontal axis and average `price` by display rating on the vertical axis.

In [8]:
```
#Write code here
yelp_data.group('display_rating', collect = np.mean).scatter('display_rating', 'price mean')
```



h. (3 points) Describe the pattern you see. What issue does this present for estimating the causal effect of `display_rating` on reservation availability?

A: Restaurant price is increasing in display rating. This suggests price may introduce omitted variable bias. That is, the relationship identified above may in part reflect the fact that high-priced restaurants have higher display ratings, and high-priced restaurants are also more difficult to reserve. The relationship between `display_rating` and `avail_7pm` may not reflect the causal effect of `display_rating` .

Of course, we can always include `price` as a control in our regression model.

i. (3 points) Estimate the following regression model:

$$\text{avail\_7pm}_i = \alpha + \beta \text{display\_rating}_i + \gamma \text{price}_i + e_i$$

Be sure to report the coefficients.

In [9]:
```
#write your code here: define MSE function
def avail7_mse(slope1, slope2, intercept):
    x1 = yelp_data.column('display_rating')
    x2 = yelp_data.column('price')
    y = yelp_data.column('avail_7pm')
    fitted = intercept + slope1*x1 + slope2*x2
    return np.mean((y - fitted) ** 2)
```

In [10]:
```
#write any additional code here
coefficients = minimize(avail7_mse)
coefficients
```

Out[10]:
```
array([-0.34399066, -0.17427984,  2.29759474])
```

j. (3 points) Interpret the estimated coefficient β in a full sentence. (Be sure to mention the *magnitude* of the estimate and not just the *sign*.)

A: An increase in the display rating by 1 star is associated with a 34.4 percentage point decrease in availability for 7pm reservations.

However, even after controlling for price, there may be other confounding factors that differ between restaurants with high and low ratings that produce an omitted variable bias.

k. (5 points) Describe a potential confounding factor that would generate an **omitted variable bias** in our regression of of restaurant availability on `display_rating` (with `price` already included as a control). What sign would you anticipate for this omitted variable bias, and why?

A: Even after controlling for price, restaurant food quality may generate selection bias, for example. Restaurants with better food may be more difficult to reserve *and* have higher Yelp ratings. This will lead to a positive relationship between Yelp ratings and reservation availability even in the absence of a causal relationship between `display_rating` and `meanavail_7` .

newpage

Part III: Regression Discontinuity Approach

Though we can control for *observable* differences across restaurants using regression, *unobservable* differences remain a major concern. Fortunately, Yelp's review system provides a natural regression discontinuity research design that may allow us to account for *unobservable* differences across restaurants. `avg_rating` will serve as the running variable.

Here is an explanation provided in the text of Anderson and Magruder (2012):

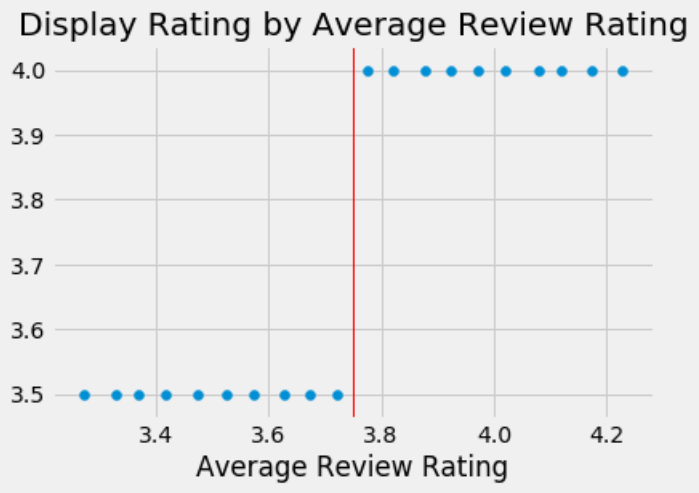
"When leaving a review on Yelp, a user must assign a rating from 1 to 5 stars in whole-star increments. Yelp aggregates all reviews for a given business and displays the average rating prominently. However, when Yelp computes the average rating they round off to the nearest half-star. Two restaurants that have similar average ratings can thus appear to be of very different quality. For example, a restaurant with an average rating of 3.24 displays a 3-star average rating while a restaurant with an average rating of 3.26 displays a 3.5-star average rating."

In this section we will implement this idea. First, we will confirm that `display_rating` changes sharply around specific cutoffs in `avg_rating` . We will do this by creating a *bin scatter* plot as we saw in lecture. We will bin the data by `avg_rating` , then plot the average `display_rating` for each bin around the cutoff. We will use a bandwidth of 0.5. That is, we will focus on restaurants with value `avg_rating` within 0.5 of the cutoff.

In [11]:
```
#create grid for avg_rating bins
break_points = np.arange(2.75,4.80,0.05)

#create column of bins
bins = pd.cut(yelp_data.column('avg_rating'), break_points, right = False)

#add bins to data
yelp_data = yelp_data.with_column('bin', bins)
```

In [12]:
```
#create table of bin means around 4 star cutoff
#bandwidth = 0.5
disc4_means = yelp_data.where('avg_rating', are.between_or_equal_to(3.25, 4.25)).group('bin', c
ollect = np.mean)

plt.scatter(disc4_means.column('avg_rating mean'), disc4_means.column('display_rating mean'))
plt.title('Display Rating by Average Review Rating')
plt.xlabel('Average Review Rating')
plt.axvline(x=3.75, linewidth=1, color = 'red')
```

Out[12]: <matplotlib.lines.Line2D at 0x7fb26c25d240>



The display rating changes sharply once `avg_rating` reaches 3.75. [The paper exploits similar cutoffs at 3.25 (moving from 3 stars to 3.5 stars) and 4.25 (moving from 4 stars to 4.5 stars).]

l. (3 points) Compared to the regression approach taken in **Part II**, what are the advantages of this approach to estimating the causal effect of a Yelp star rating?

A: The advantage of the regression discontinuity design is that, under the continuity assumption, it ensures balance. Restaurants just below and just above the discontinuity will be comparable except for the difference in `display_rating`. This approach provides more compelling estimates for causal effects.

m. (3 points) We can use this regression discontinuity approach to estimate the causal effect of a Yelp star rating on availability. If the standard regression discontinuity assumptions are met, we will recover this treatment effect *for what set of restaurants*?

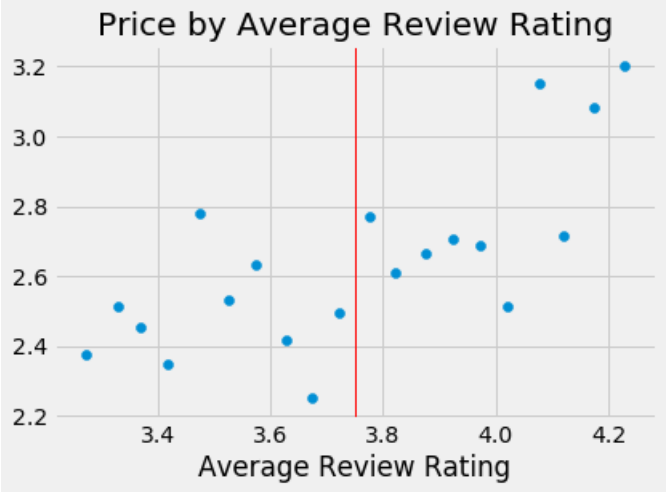
A: We will recover the treatment effect for restaurants at the cutoff--those with an `avg_rating` value of 3.75.

An important identifying assumption for the RD approach is that other restaurant characteristics vary *smoothly* or *continuously* in the running variable at the cutoff. In other words, we should not see a discontinuity in these other covariates.

Below we check whether `price` varies continuously at the cutoff. This is the same plot as above with `display_rating` replaced by `price`.

In [13]:
```
#run this cell to generate price plot

plt.scatter(disc4_means.column('avg_rating mean'), disc4_means.column('price mean'))
plt.title('Price by Average Review Rating')
plt.xlabel('Average Review Rating')
plt.axvline(x=3.75, linewidth=1, color = 'red')
```

Out[13]: <matplotlib.lines.Line2D at 0x7fb26c3a0ac8>



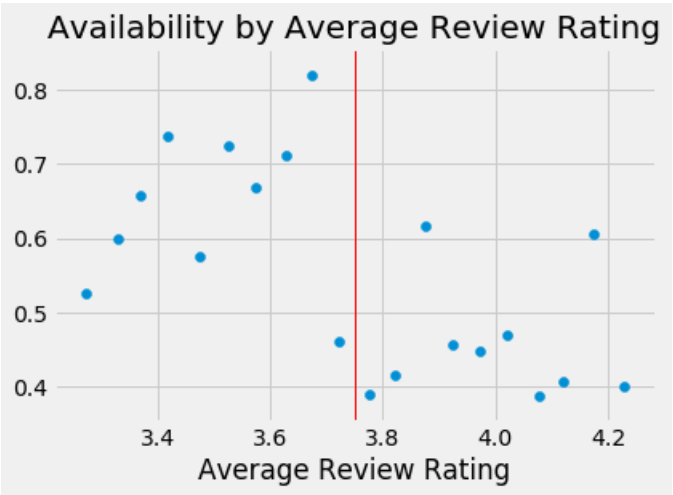
By contrast to `display_rating`, there is no clear discontinuity in `price` at the cutoff of 3.75. This is consistent with the continuity assumption from lecture (and the text).

Next, you'll make the main set of RD plots: plots of the outcome against the running variable.

n. (6 points) Replicate Figure 2(b) from the paper. [Your figure should be close, but not quite identical.]

In [14]:
```
#Write code for figure here
disc4_means = yelp_data.where('avg_rating', are.between_or_equal_to(3.25, 4.25)).group('bin', c
ollect = np.mean)

plt.scatter(disc4_means.column('avg_rating mean'), disc4_means.column('avail_7pm mean'))
plt.title('Availability by Average Review Rating')
plt.xlabel('Average Review Rating')
plt.axvline(x=3.75, linewidth=1, color = 'red')
```

Out[14]: <matplotlib.lines.Line2D at 0x7fb26c22ddd8>



o. (5 points) Estimate the regression discontinuity model for the figure above. Using a bandwidth of 0.5 (meaning restricting the data to observations with values of `avg_rating` within 0.5 of the cutoff, 3.75), estimate the following regression model:

$$\text{avail\_7pm}_i = \alpha + \beta \mathbb{1}_{\text{4 stars}} + \gamma(\text{avg\_rating}_i - c) + \delta(\text{avg\_rating}_i - c) \times \mathbb{1}_{\text{4 stars}} + e_i$$

where $c = 3.75$, the cutoff for a `display_rating` of 4. Note that $\mathbb{1}_{\text{4 stars}}$ is an indicator for a `display_rating` of 4.

Be sure to report the coefficients.

In [15]:
```
#create indicator for a display_rating value of 4
disp_4 = np.int32(yelp_data.column('display_rating') == 4)

yelp_data = yelp_data.with_column('disp_4', disp_4)

disc4_data = yelp_data.where('avg_rating', are.between_or_equal_to(3.25, 4.25))

dist_4 = disc4_data.column('avg_rating') - 3.75
dist_4xdisp_4 = dist_4*disc4_data.column('disp_4')

disc4_data = disc4_data.with_column('dist_4', dist_4).with_column('dist_4xdisp_4', dist_4xdisp_
4)
```

In [16]:
```
#write your code here: define MSE function
def disc4_mse(slope1, slope2, slope3, intercept):
    x1 = disc4_data.column('disp_4')
    x2 = disc4_data.column('dist_4')
    x3 = disc4_data.column('dist_4xdisp_4')
    y = disc4_data.column('avail_7pm')
    fitted = intercept + slope1*x1 + slope2*x2 + slope3*x3
    return np.mean((y - fitted) ** 2)
```

In [17]:
```
#write any additional code here
coefficients = minimize(disc4_mse)
coefficients
```

Out[17]:
```
array([-0.21249474,  0.06181229, -0.03734092,  0.66348119])
```

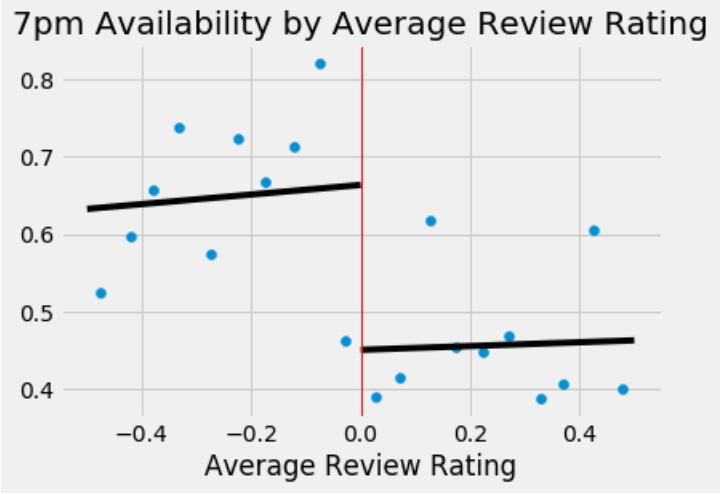
p. (3 points) Based on this model, what do you estimate is the causal effect of a *half star* increase in Yelp rating on reservation availability?

A: The estimate for the coefficient on `disp_4` is -0.212. Hence, our estimate for the causal effect of a half star increase in Yelp rating--from 3.5 to 4--is a 21.2 percentage point decrease in availability for 7pm reservations.

q. (3 points) Run the cell below to overlay your regression line over the bin scatter plot. Replace the values `alpha` (α), `beta` (β), `gamma` (γ), and `delta` (δ) with your coefficient estimates.

In [18]:
```
alpha = coefficients[3]
beta = coefficients[0]
gamma = coefficients[1]
delta = coefficients[2]

#plot regression lines
X_plot_pre = np.linspace(-0.5, 0, 10)
plt.plot(X_plot_pre, alpha + X_plot_pre*gamma, color = 'black')

X_plot_post = np.linspace(0, 0.5, 10)
plt.plot(X_plot_post, alpha + beta + X_plot_post*(gamma + delta), color = 'black')

dist_4 = yelp_data.where('avg_rating', are.between_or_equal_to(3.25, 4.25)).column('avg_rating'
) - 3.75
disc4_means = yelp_data.where('avg_rating', are.between_or_equal_to(3.25, 4.25)).with_column('d
ist_4', dist_4).group('bin', collect = np.mean)


#generate bin scatter plot
plt.scatter(disc4_means.column('dist_4 mean'), disc4_means.column('avail_7pm mean'))
plt.title('7pm Availability by Average Review Rating')
plt.xlabel('Average Review Rating')
plt.axvline(x=0, linewidth=1, color = 'red')
```

Out[18]:
```
<matplotlib.lines.Line2D at 0x7fb269d63c50>
```



r. (3 points) Describe how this figure provides visual evidence for the average causal effect of a restaurant's `dispay_rating` on that restaurant's reservation availability.

A: This figure illustrates a clear discontinuity in restaurant availability at the `avg_rating` cutoff of 3.75. Under the necessary regression disconinuity assumptions, this jump is equal to the average causal effect of an increase in a restaurant's `display_rating` (from 3.5 to 4) among restaurants at the cutoff.

newpage

Submission

Before submitting, please click "Kernel" above and click "Restart & Run All" to ensure all of your code is working as expected. This is important. Code that does not run cannot be graded. After confirming that all of your work looks and runs as you'd like it to, run **BOTH** of the below cells to submit your work. Failure to submit correctly may result in a 10% point penalty.

First, make sure that the following runs successfully for submission to OKpy.

In [ ]:
```
from client.api.notebook import Notebook
ok = Notebook('pset9.ok')
_ = ok.auth(inline=True)
_ = ok.submit()
```

Then, make sure that the following runs successfully to generate a PDF to upload to Gradescope. **Do not upload any other PDF to Gradescope other than the one generated by the below code.** If you have difficulty downloading the PDF, please see Piazza for troubleshooting steps.

In [ ]:
```
gsExport.generateSubmission('pset9.ipynb')
```