QSS20_Finalpset1_Group_Molly

April 20, 2021

1 0. Load packages and imports

[66]:
```
## basic functionality
import pandas as pd
import numpy as np
import re
import plotnine
from plotnine import *

## can add others if you need them

## repeated printouts
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

1.1 0.1: Load the data (0 points)

Load the `sentencing_asof0405.csv` data

Notes: You may receive a warning about mixed data types upon import; feel free to ignore

[67]:
```
sentencing = pd.read_csv("sentencing_asof0405.csv")
```

```
/opt/conda/lib/python3.8/site-packages/IPython/core/interactiveshell.py:3165:
DtypeWarning: Columns (10,11,14,25) have mixed types.Specify dtype option on
import or set low_memory=False.
```

1.2 0.2: Print head, dimensions, info (0 points)

[70]:
```
sentencing.head()
sentencing.info()
```

[70]:
```
        CASE_ID  CASE_PARTICIPANT_ID          RECEIVED_DATE  \
0  149765331439         175691153649  8/15/1984 12:00:00 AM
1  149765331439         175691153649  8/15/1984 12:00:00 AM
2  149765331439         175691153649  8/15/1984 12:00:00 AM
3  149765331439         175691153649  8/15/1984 12:00:00 AM
4  149765331439         175691153649  8/15/1984 12:00:00 AM
```

```
    OFFENSE_CATEGORY  PRIMARY_CHARGE_FLAG    CHARGE_ID  CHARGE_VERSION_ID  \
0  PROMIS Conversion                False  50510112469       116304211997
1  PROMIS Conversion                False  50510213021        98265074680
2  PROMIS Conversion                False  50516447217       131972895911
3  PROMIS Conversion                False  50516497493       131966356472
4  PROMIS Conversion                False  50516648320        98059642859

  DISPOSITION_CHARGED_OFFENSE_TITLE  CHARGE_COUNT       DISPOSITION_DATE  \
0              FIRST DEGREE MURDER             2  12/17/2014 12:00:00 AM
1                    HOME INVASION            14  12/17/2014 12:00:00 AM
2              FIRST DEGREE MURDER             4  12/17/2014 12:00:00 AM
3              FIRST DEGREE MURDER             5  12/17/2014 12:00:00 AM
4                    HOME INVASION            13  12/17/2014 12:00:00 AM

   … INCIDENT_CITY   INCIDENT_BEGIN_DATE INCIDENT_END_DATE  \
0  …           NaN  8/9/1984 12:00:00 AM               NaN
1  …           NaN  8/9/1984 12:00:00 AM               NaN
2  …           NaN  8/9/1984 12:00:00 AM               NaN
3  …           NaN  8/9/1984 12:00:00 AM               NaN
4  …           NaN  8/9/1984 12:00:00 AM               NaN

  LAW_ENFORCEMENT_AGENCY LAW_ENFORCEMENT_UNIT            ARREST_DATE  \
0    CHICAGO POLICE DEPT                  NaN  8/15/1984 12:00:00 AM
1    CHICAGO POLICE DEPT                  NaN  8/15/1984 12:00:00 AM
2    CHICAGO POLICE DEPT                  NaN  8/15/1984 12:00:00 AM
3    CHICAGO POLICE DEPT                  NaN  8/15/1984 12:00:00 AM
4    CHICAGO POLICE DEPT                  NaN  8/15/1984 12:00:00 AM

       FELONY_REVIEW_DATE FELONY_REVIEW_RESULT       ARRAIGNMENT_DATE  \
0  08/15/1984 12:00:00 AM   Charge(S) Approved  9/21/1984 12:00:00 AM
1  08/15/1984 12:00:00 AM   Charge(S) Approved  9/21/1984 12:00:00 AM
2  08/15/1984 12:00:00 AM   Charge(S) Approved  9/21/1984 12:00:00 AM
3  08/15/1984 12:00:00 AM   Charge(S) Approved  9/21/1984 12:00:00 AM
4  08/15/1984 12:00:00 AM   Charge(S) Approved  9/21/1984 12:00:00 AM

  UPDATED_OFFENSE_CATEGORY
0                 Homicide
1                 Homicide
2                 Homicide
3                 Homicide
4                 Homicide

[5 rows x 41 columns]

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 248146 entries, 0 to 248145
```

```
Data columns (total 41 columns):
 #   Column                            Non-Null Count   Dtype
---  ------                            --------------   -----
 0   CASE_ID                           248146 non-null  int64
 1   CASE_PARTICIPANT_ID               248146 non-null  int64
 2   RECEIVED_DATE                     248146 non-null  object
 3   OFFENSE_CATEGORY                  248146 non-null  object
 4   PRIMARY_CHARGE_FLAG               248146 non-null  bool
 5   CHARGE_ID                         248146 non-null  int64
 6   CHARGE_VERSION_ID                 248146 non-null  int64
 7   DISPOSITION_CHARGED_OFFENSE_TITLE  248146 non-null  object
 8   CHARGE_COUNT                      248146 non-null  int64
 9   DISPOSITION_DATE                  248146 non-null  object
 10  DISPOSITION_CHARGED_CHAPTER       248146 non-null  object
 11  DISPOSITION_CHARGED_ACT           242771 non-null  object
 12  DISPOSITION_CHARGED_SECTION       242771 non-null  object
 13  DISPOSITION_CHARGED_CLASS         248127 non-null  object
 14  DISPOSITION_CHARGED_AOIC          248122 non-null  object
 15  CHARGE_DISPOSITION                248146 non-null  object
 16  CHARGE_DISPOSITION_REASON         904 non-null     object
 17  SENTENCE_JUDGE                    247404 non-null  object
 18  SENTENCE_COURT_NAME               246761 non-null  object
 19  SENTENCE_COURT_FACILITY           246216 non-null  object
 20  SENTENCE_PHASE                    248146 non-null  object
 21  SENTENCE_DATE                     248146 non-null  object
 22  SENTENCE_TYPE                     248146 non-null  object
 23  CURRENT_SENTENCE_FLAG             248146 non-null  bool
 24  COMMITMENT_TYPE                   246464 non-null  object
 25  COMMITMENT_TERM                   246434 non-null  object
 26  COMMITMENT_UNIT                   246434 non-null  object
 27  LENGTH_OF_CASE_in_Days            229126 non-null  float64
 28  AGE_AT_INCIDENT                   238359 non-null  float64
 29  RACE                              246879 non-null  object
 30  GENDER                            247337 non-null  object
 31  INCIDENT_CITY                     228745 non-null  object
 32  INCIDENT_BEGIN_DATE               239122 non-null  object
 33  INCIDENT_END_DATE                 22008 non-null   object
 34  LAW_ENFORCEMENT_AGENCY            239405 non-null  object
 35  LAW_ENFORCEMENT_UNIT              76408 non-null   object
 36  ARREST_DATE                       242981 non-null  object
 37  FELONY_REVIEW_DATE                171907 non-null  object
 38  FELONY_REVIEW_RESULT              171907 non-null  object
 39  ARRAIGNMENT_DATE                  229126 non-null  object
 40  UPDATED_OFFENSE_CATEGORY          248146 non-null  object
dtypes: bool(2), float64(2), int64(5), object(32)
memory usage: 74.3+ MB
```

2 Part one: data cleaning/interpretation (group)

2.1 1.1: Understanding the unit of analysis (5 points)

- Print the number of unique values for the following columns. Do so in a way that avoids copying/pasting code for the three:
  - Cases (`CASE_ID`)
  - People in that case (`CASE_PARTICIPANT_ID`)
  - Charges (`CHARGE_ID`)
- Write a couple sentences on the following and show an example of each (e.g., a case involving multiple people):
  - Why there are more unique people than unique cases?
  - Why there are more unique charges than unique people?
- Print the mean and median number of charges per case/participant
- Print the mean and median number of participants per case
- Does the data seem to enable us to follow the same defendant across different cases they're charged in? Write 1 sentence in support of your conclusion.

[71]:
```
sentencing[["CASE_ID", 'CASE_PARTICIPANT_ID', 'CHARGE_ID']].nunique()

# Why there are more unique people than unique cases?
    # Because one case can involve multiple people.
# Why there are more unique charges than unique people?
    # Because one person can be the subject of multiple charges.

sentencing.groupby("CASE_ID")["CHARGE_ID"].nunique().agg([np.mean, np.median])
sentencing.groupby("CASE_PARTICIPANT_ID")["CHARGE_ID"].nunique().agg([np.mean,␣
↪np.median])
sentencing.groupby("CASE_ID")["CASE_PARTICIPANT_ID"].nunique().agg([np.mean, np.
↪median])
sentencing.groupby("CASE_PARTICIPANT_ID")["CASE_ID"].nunique().max()

# Does the data seem to enable us to follow the same defendant across different␣
↪cases they're charged in?
    # No. When grouping by participant id and finding the max number of cases␣
↪each participant has,
    # the max number of cases any participant is charged with is 1, so it seems␣
↪like
    # the participant ID changes between cases.
```

[71]:
```
CASE_ID                197519
CASE_PARTICIPANT_ID    211977
CHARGE_ID              229015
dtype: int64
```

```
[71]: mean      1.159458
      median    1.000000
      Name: CHARGE_ID, dtype: float64
```

```
[71]: mean      1.126146
      median    1.000000
      Name: CHARGE_ID, dtype: float64
```

```
[71]: mean      1.073198
      median    1.000000
      Name: CASE_PARTICIPANT_ID, dtype: float64
```

```
[71]: 1
```

2.2 1.2.1: Which offense is final? (3 points)

- First, read the data documentation link and summarize in your own words the differences between `OFFENSE_CATEGORY` and `UPDATED_OFFENSE_CATEGORY`
- Construct an indicator `is_changed_offense` that's True for case-participant-charge observations (rows) where there's a difference between the original charge (offense category) and the most current charge (updated offense category). What are some of the more common changed offenses? (can just print result of sort_values based on original offense category)
- Print one example of a changed offense from one of these categories and comment on what the reason may be

```
[72]: # OFFENSE_CATEGORY is the category of offense encoded before specific charges␣
       ↪are brought in a case,
      # while UPDATED_OFFENSE_CATEGORY is the category of offense encoded based on␣
       ↪the primary charge of the case.

      sentencing["is_changed_offense"] = sentencing["OFFENSE_CATEGORY"] !=␣
       ↪sentencing["UPDATED_OFFENSE_CATEGORY"]
      sentencing[sentencing["is_changed_offense"]]["OFFENSE_CATEGORY"].sort_values().
       ↪first
      sentencing[sentencing["is_changed_offense"]][["OFFENSE_CATEGORY",␣
       ↪"UPDATED_OFFENSE_CATEGORY"]].head(1)

      # The charge PROMIS Conversion has changed because the prosecution decides␣
       ↪whether to charge a
      # crime as a homicide as enough evidences are collected, and that decision is␣
       ↪reflected in UPDATED_OFFENSE_CATEGORY.
```

```
[72]: <bound method NDFrame.first of 161337         Aggravated Assault Police Officer
      231526         Aggravated Assault Police Officer
      204443         Aggravated Assault Police Officer
      204444         Aggravated Assault Police Officer
```

```
138559        Aggravated Assault Police Officer
                              …
225629    Violation of Sex Offender Registration
65257     Violation of Sex Offender Registration
114277    Violation of Sex Offender Registration
59875     Violation of Sex Offender Registration
17990     Violation of Sex Offender Registration
Name: OFFENSE_CATEGORY, Length: 35865, dtype: object>
```

[72]:

```
     OFFENSE_CATEGORY UPDATED_OFFENSE_CATEGORY
0  PROMIS Conversion                 Homicide
```

2.3 1.2.2: Simplifying the charges (5 points)

Using the field (`UPDATED_OFFENSE_CATEGORY`), create a new field, `simplified_offense_derived`, that simplifies the many offense categories into broader buckets using the following process:

First, combine all offenses beginning with "Aggravated" into a single category without that prefix (e.g., Aggravated Battery and Battery just becomes Battery)

Then: - Combine all offenses with arson into a single arson category (`Arson`) - Combine all offenses with homicide into a single homicide category (`Homicide`) - Combine all offenses with vehicle/vehicular in the name into a single vehicle category (`Vehicle-related`) - Combine all offenses with battery in the name into a single battery category (`Battery`)

Try to do so efficiently (e.g., using map and a dictionary or np.select rather than separate line for each recoded offense)

Print the difference between the # of unique offenses in the original `UPDATED_OFFENSE_CATEGORY` field and the # of unique offenses in your new `simplified_offense_derived` field

[73]:

```
sentencing["simplified_offense_derived"] =␣
↪sentencing["UPDATED_OFFENSE_CATEGORY"].str.replace("Aggravated ", "")

conditions = [sentencing["simplified_offense_derived"].str.contains("Arson"),
              sentencing["simplified_offense_derived"].str.contains("Homicide"),
              sentencing["simplified_offense_derived"].str.
↪contains("Vehicle-related"),
              sentencing["simplified_offense_derived"].str.contains("Battery")]

choices = ["Arson", "Homicide", "Vehicle-related", "Battery"]

sentencing["simplified_offense_derived"] = np.select(conditions, choices,␣
↪sentencing["simplified_offense_derived"])

sentencing.UPDATED_OFFENSE_CATEGORY.nunique() - sentencing.
↪simplified_offense_derived.nunique()
```

[73]: 11

2.4 1.3: Cleaning additional variables (10 points)

Clean the following variables; make sure to retain the original variable in data and use the derived suffix so it's easier to pull these cleaned out variables later (e.g., `age_derived`) to indicate this was a transformation

- Race: create True/false indicators for `is_black_derived` (Black only or mixed race with hispanic), Non-Black Hispanic, so either hispanic alone or white hispanic (`is_hisp_derived`), White non-hispanic (`is_white_derived`), or none of the above (`is_othereth_derived`)
- Gender: create a boolean true/false indicator for `is_male_derived` (false is female, unknown, or other)
- Age at incident: you notice outliers like 130-year olds. Winsorsize the top 0.01% of values to be equal to the 99.99th percentile value pre-winsorization. Call this `age_derived`
- Create `sentenceymd_derived` that's a version of `SENTENCING_DATE` converted to datetime format. Also create a rounded version, `sentenceym_derived`, that's rounded down to the first of the month and the year (e.g., 01-05-2016 and 01-27-2016 each become 01-01-2016)
  - Hint: all timestamps are midnight so u can strip in conversion. For full credit, before converting, you notice that some of the years have been mistranscribed (e.g., 291X or 221X instead of 201X). Programatically fix those (eg 2914 -> 2014). Even after cleaning, there will still be some that are after the year 2021 that we'll filter out later. For partial credit, you can ignore the timestamps that cause errors and set errors = "coerce" within `pd.to_datetime()` to allow the conversion to proceed.
- Sentencing judge: create an identifier (`judgeid_derived`) for each unique judge (`SENTENCE_JUDGE`) structured as judge_1, judge_2...., with the order determined by sorting the judges (will sort on fname then last). When finding unique judges, there are various duplicates we could weed out — for now, just focus on (1) the different iterations of Doug/Douglas Simpson, (2) the different iterations of Shelley Sutker (who appears both with her maiden name and her hyphenated married name).
  - Hint: due to mixed types, you may need to cast the `SENTENCE_JUDGE` var to a diff type to sort

After finishing, print a random sample of 10 rows (data.sample(n = 10)) with the original and cleaned columns for the relevant variables to validate your work

[74]:
```
sentencing["is_black_derived"] = np.where((sentencing["RACE"] == "Black") |
                                          (sentencing["RACE"] == "White/Black␣
 ↪[Hispanic or Latino]"),
                                          True, False)

sentencing["is_hisp_derived"] = np.where((sentencing["RACE"] == "HISPANIC") |
                                          (sentencing["RACE"] == "White␣
 ↪[Hispanic or Latino]"),
                                          True, False)

sentencing["is_white_derived"] = np.where((sentencing["RACE"] == "White"),
                                          True, False)
```

```
sentencing["is_othereth_derived"] = np.where((sentencing["is_black_derived"] ==␣
↪False) &
                                            (sentencing["is_hisp_derived"] ==␣
↪False) &
                                            (sentencing["is_white_derived"] ==␣
↪False),
                                            True, False)
```

[75]:
```
sentencing["is_male_derived"] = np.where(sentencing["GENDER"] == "Male", True,␣
↪False)
```

[76]:
```
sentencing.AGE_AT_INCIDENT.quantile(q = 0.9999)

sentencing["age_derived"] = np.where(sentencing["AGE_AT_INCIDENT"] > 81.0, 81.
↪0, sentencing["AGE_AT_INCIDENT"])
```

[76]: 81.0

[77]:
```
sentencing["sentenceymd_derived"] = sentencing.SENTENCE_DATE.str[:-12]

sentencing["sentenceymd_derived"] = np.where(sentencing.sentenceymd_derived.
↪str[-4:-2].astype("int") > 20,
                                           sentencing.sentenceymd_derived.str[:
↪-3] + "0" + sentencing.sentenceymd_derived.str[-2:],
                                           sentencing.sentenceymd_derived)

sentencing["sentenceymd_derived"] = pd.to_datetime(sentencing.
↪sentenceymd_derived)

sentencing["sentenceym_derived"] = sentencing.sentenceymd_derived.
↪astype('datetime64[M]')
```

[78]:
```
judges = sentencing.groupby("SENTENCE_JUDGE").CASE_ID.count().reset_index()
judges["judgeid_derived"] = "judge_" + (judges.index).astype("string")
judges = judges[["SENTENCE_JUDGE","judgeid_derived"]]
judges[(judges.SENTENCE_JUDGE.str.contains("Doug")) | (judges.SENTENCE_JUDGE.
↪str.contains("Shelley"))]
```

[78]:

	SENTENCE_JUDGE	judgeid_derived
70	Doug Simpson	judge_70
71	Douglas J Simpson	judge_71
280	Shelley Sutker	judge_280
281	Shelley Sutker-Dermer	judge_281

[79]:
```
sentencing = pd.merge(sentencing, judges)
sentencing["judgeid_derived"] = np.where(sentencing.judgeid_derived ==␣
↪"judge_71", "judge_70", sentencing.judgeid_derived)
sentencing["judgeid_derived"] = np.where(sentencing.judgeid_derived ==␣
↪"judge_281", "judge_280", sentencing.judgeid_derived)
sentencing[["SENTENCE_JUDGE","judgeid_derived"]][(sentencing.SENTENCE_JUDGE.str.
↪contains("Doug")) | (sentencing.SENTENCE_JUDGE.str.contains("Shelley"))].
↪value_counts()
```

[79]:
```
SENTENCE_JUDGE           judgeid_derived
Shelley  Sutker-Dermer   judge_280          2507
Douglas J Simpson        judge_70            101
Doug  Simpson            judge_70              6
Shelley  Sutker          judge_280             1
dtype: int64
```

[81]:
```
## print a random sample of 10 rows (data.sample(n = 10)) with the original and␣
↪cleaned columns for the
## relevant variables to validate your work


sample = sentencing[["RACE",␣
↪"is_black_derived","is_hisp_derived","is_white_derived","is_othereth_derived","GENDER","is_
                    "AGE_AT_INCIDENT","age_derived", ␣
↪"SENTENCE_JUDGE","judgeid_derived","SENTENCE_DATE",␣
↪"sentenceymd_derived","sentenceym_derived"]]
sample.sample(n = 10)
```

[81]:
```
          RACE  is_black_derived  is_hisp_derived  is_white_derived  \
74350    Black              True            False             False
66573    Black              True            False             False
61369    Black              True            False             False
154001   White             False            False              True
93001    Black              True            False             False
9507     Black              True            False             False
210060   Black              True            False             False
242051   Black              True            False             False
117399   Black              True            False             False
244912   Black              True            False             False

        is_othereth_derived  GENDER  is_male_derived  AGE_AT_INCIDENT  \
74350                 False    Male             True             30.0
66573                 False    Male             True             35.0
61369                 False    Male             True              NaN
154001                False    Male             True             41.0
93001                 False    Male             True             22.0
9507                  False    Male             True             22.0
```

```
210060                False    Male              True              34.0
242051                False    Male              True              26.0
117399                False    Male              True              24.0
244912                False  Female             False              46.0

        age_derived      SENTENCE_JUDGE judgeid_derived  \
74350          30.0    Michele M Pitman       judge_229
66573          35.0       Thomas M Davy       judge_306
61369           NaN   Vincent M Gaughan       judge_322
154001         41.0  Colleen Ann Hyland        judge_41
93001          22.0       Arthur F Hill        judge_17
9507           22.0    William J Kunkle       judge_333
210060         34.0       Byrne, Thomas        judge_22
242051         26.0    Marvin P Luckman       judge_202
117399         24.0     Erica L Reddick        judge_82
244912         46.0    Steven G Watkins       judge_288

                  SENTENCE_DATE sentenceymd_derived sentenceym_derived
74350    5/26/2016 12:00:00 AM          2016-05-26         2016-05-01
66573    6/11/2014 12:00:00 AM          2014-06-11         2014-06-01
61369    6/21/2013 12:00:00 AM          2013-06-21         2013-06-01
154001    9/9/2016 12:00:00 AM          2016-09-09         2016-09-01
93001   10/15/2015 12:00:00 AM          2015-10-15         2015-10-01
9507    12/13/2012 12:00:00 AM          2012-12-13         2012-12-01
210060    8/9/2016 12:00:00 AM          2016-08-09         2016-08-01
242051   12/5/2017 12:00:00 AM          2017-12-05         2017-12-01
117399   2/24/2015 12:00:00 AM          2015-02-24         2015-02-01
244912  10/21/2019 12:00:00 AM          2019-10-21         2019-10-01
```

2.5 1.4: Subsetting rows to analytic dataset (5 points)

You decide based on the above to simplify things in the following ways:

- Subset to cases where only one participant is charged, since cases with >1 participant might have complications like plea bargains/informing from other participants affecting the sentencing of the focal participant
- To go from a participant-case level dataset, where each participant is repeated across charges tied to the case, to a participant-level dataset, where each participant has one charge, subset to a participant's primary charge and their current sentence (`PRIMARY_CHARGE_FLAG` is True and `CURRENT_SENTENCE_FLAG` is True). Double check that this worked by confirming there are no longer multiple charges for the same case-participant
- Filter out observations where judge is nan or nonsensical (indicated by is.null or equal to FLOOD)
- Subset to sentencing date between 01-01-2012 and 04-05-2021 (inclusive)

After completing these steps, print the number of rows in the data

[82]:
```
one_participant = sentencing.groupby("CASE_ID").agg(participant_count =␣
↪("CASE_PARTICIPANT_ID", "count")).reset_index()
one_participant = one_participant[one_participant["participant_count"] == 1]

one_participant_series = one_participant.CASE_ID


sentencing = sentencing[sentencing.CASE_ID.isin(one_participant_series)]
```

[83]:
```
sentencing = sentencing[(sentencing.PRIMARY_CHARGE_FLAG == True) & (sentencing.
↪CURRENT_SENTENCE_FLAG == True)]

sentencing[["CASE_ID", 'CASE_PARTICIPANT_ID', 'CHARGE_ID']].count()
sentencing.shape
```

[83]:
```
CASE_ID                137154
CASE_PARTICIPANT_ID    137154
CHARGE_ID              137154
dtype: int64
```

[83]:
```
(137154, 52)
```

[84]:
```
sentencing = sentencing[(~sentencing.SENTENCE_JUDGE.isnull()) &
                        (sentencing.SENTENCE_JUDGE != "FLOOD")]
```

[85]:
```
sentencing = sentencing[(sentencing.sentenceymd_derived >= "01-01-2012") &
                        (sentencing.sentenceymd_derived <= "04-05-2021")]
sentencing.shape[0]
```

[85]:
```
121396
```

3 Part two: investigating Black vs. White sentencing disparities

Now that the data are cleaned, we're going to investigate different types of disparities in sentencing between Black defendants and White defendants. We're focusing on these groups for the purpose of the problem set, but the analysis could be extended to study Hispanic defendants or, in a different jurisdiction, Asian and other minoritized groups.

Details if interested in digging deeper: If interested (optional), you can read more technical coverage of how we might (1) measure disparities, and (2) what factors you want to adjust for when deciding whether two defendants are 'similarly situated' but for their race in the following sources:

- Review of sentencing disparities research
- Discussion of causal model/blinding race at charging stage of the prosecutorial process
- Discussion of measuring discrimination in policing that can generalize to the sentencing case
- General discussion of causal challenges in measuring between-group disparities

One major caveat: when investigating whether two similar defendants received different sentences, we're missing one important attribute that influences sentencing: the defendant's criminal history. This influences sentencing both through sentencing guidelines, which can prescribe longer sentences for those who have certain types of prior convictions, and through judicial discretion if judges are more lenient with first-time defendants. The above sources discuss how much we want to "control away" for this prior history, since if we think there are racial biases in which defendants, conditional on *committing* a crime, are arrested and charged, we may not want to adjust for that factor. More discussion in this article

3.1 2.0: (0 points)

First, read in the following dataset (regardless of progress on part one): `sentencing_cleaned.pkl` (if you can't read in the pkl you can read in the .csv format but may need to recast some of the datetime columns)

Note: don't worry if there are slight differences in your output from Part One and this dataset/it's not a good use of time to try to reverse engineer Part One answers from this cleaned data.

[86]:
```
sent_cleaned = pd.read_pickle("sentencing_cleaned.pkl")
```

3.2 2.1: Investigating one type of between-group difference: who reaches the sentencing stage? (5 points)

Tabulate and visualize the proportion of defendants, out of all defendants sentenced in a given month/year, who are Black and who are White (separate proportions)

- Denominator is number of unique cases that month
- Numerator for black defendants is count of is_black_derived
- Numerator for white defendants is count of is_white_derived
- Fraction of each is numerator/denominator
- Print the table
- Create a graph with two lines— one for Black defendants as fraction of total; another for White defendants. Make sure it includes a legend summarizing which color is for which group, and clean the legend so that it has informative names (e.g., Black or White rather than prop_black or prop_white)
- Use mathematical notation to write out each of the proportions using summation notation in a 1-2 sentence writeup describing trends. What seems to be going on in April and May 2020?

Optional challenge: improve the viz by shading the background of the visualization for months with fewer than 100 cases

Optional challenge: improve the viz by adding a vertical line for 12-01-2016, the month that new State's Attorney Foxx took office

[87]:
```
table = sent_cleaned.groupby("sentenceym_derived").agg(black_defendent =␣
↪("is_black_derived","sum"),
```

```
                                                       white_defendent =␣
↪("is_white_derived", "sum"),
                                                       denominator =␣
↪("CASE_ID", "nunique"))

table["black_fraction"] = table.black_defendent / table.denominator
table["white_fraction"] = table.white_defendent / table.denominator
table
```

[87]:

```
                    black_defendent  white_defendent  denominator  \
sentenceym_derived
2012-01-01                     1134              242         1674
2012-02-01                      941              233         1450
2012-03-01                     1030              255         1612
2012-04-01                     1000              211         1436
2012-05-01                     1074              223         1545
…                                …                …            …
2020-11-01                      298               76          491
2020-12-01                      347               85          537
2021-01-01                      278               50          447
2021-02-01                      320               66          491
2021-03-01                      362               79          532

                    black_fraction  white_fraction
sentenceym_derived
2012-01-01                0.677419        0.144564
2012-02-01                0.648966        0.160690
2012-03-01                0.638958        0.158189
2012-04-01                0.696379        0.146936
2012-05-01                0.695146        0.144337
…                              …               …
2020-11-01                0.606925        0.154786
2020-12-01                0.646182        0.158287
2021-01-01                0.621924        0.111857
2021-02-01                0.651731        0.134420
2021-03-01                0.680451        0.148496

[111 rows x 5 columns]
```

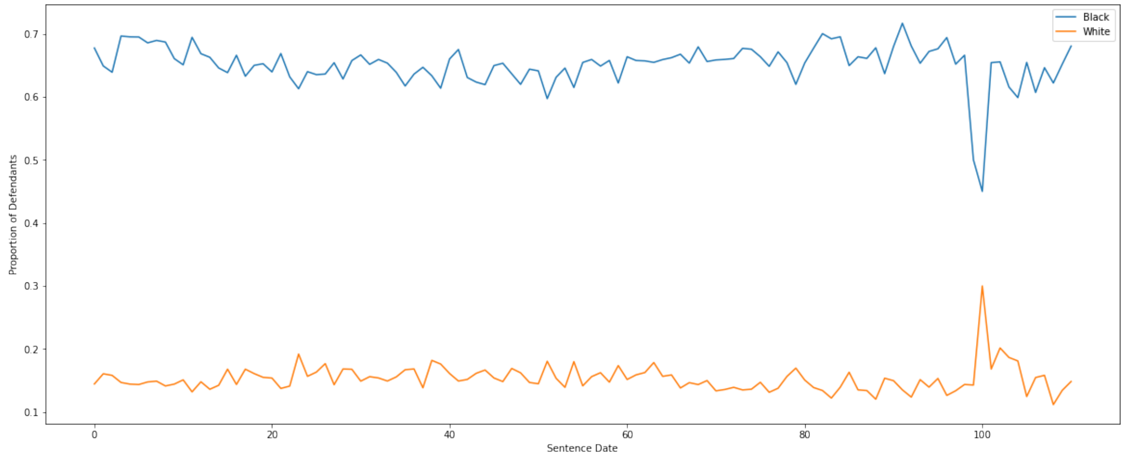
[88]:

```
plot = table.rename(columns = {"black_fraction": "Black", "white_fraction" :␣
↪"White"}).reset_index()

plot = plot[["Black", "White"]].plot(kind="line", figsize=(20, 8))



plot.set_xlabel("Sentence Date")
```

```
plot.set_ylabel("Proportion of Defendants")
```

[88]: Text(0.5, 0, 'Sentence Date')

[88]: Text(0, 0.5, 'Proportion of Defendants')



$\frac{\sum BlackDefendants}{\sum Defendants}$ per month is signficinatly higher (about 6 times higher) than $\frac{\sum WhiteDefendants}{\sum Defendants}$ per month throughout most of the time between 2012 and 2021. However, during April and May of 2020, $\frac{\sum BlackDefendants}{\sum Defendants}$ per month drops signficiantly (but still higher than $\frac{\sum WhiteDefendants}{\sum Defendants}$); meanwhile $\frac{\sum WhiteDefendants}{\sum Defendants}$ of that two months increases obviously.

3.3 2.2: Investigating the first type of disparity: probation versus incaceration (10 points)

One type of disparity beyond who arrives at the sentencing stage is whether the defendant receives probation or incaceration.

According to the codebook, incarceration is indicated by `COMMITMENT_TYPE` == "Illinois Department of Corrections"

Recreate the previous plot but where the y axis represents the difference between the following proportions (can be either Black - White or White - Black but make sure to label), adding a smoothed line:

- Percent of black defendants who are incarcerated out of all black defendants that month/year
- Percent of white defendants who are incarcerated out of all white defendants that month/year

In a markdown cell after, write 1-2 sentences on your observations of trends over time. Do gaps seem to be widening or increasing?

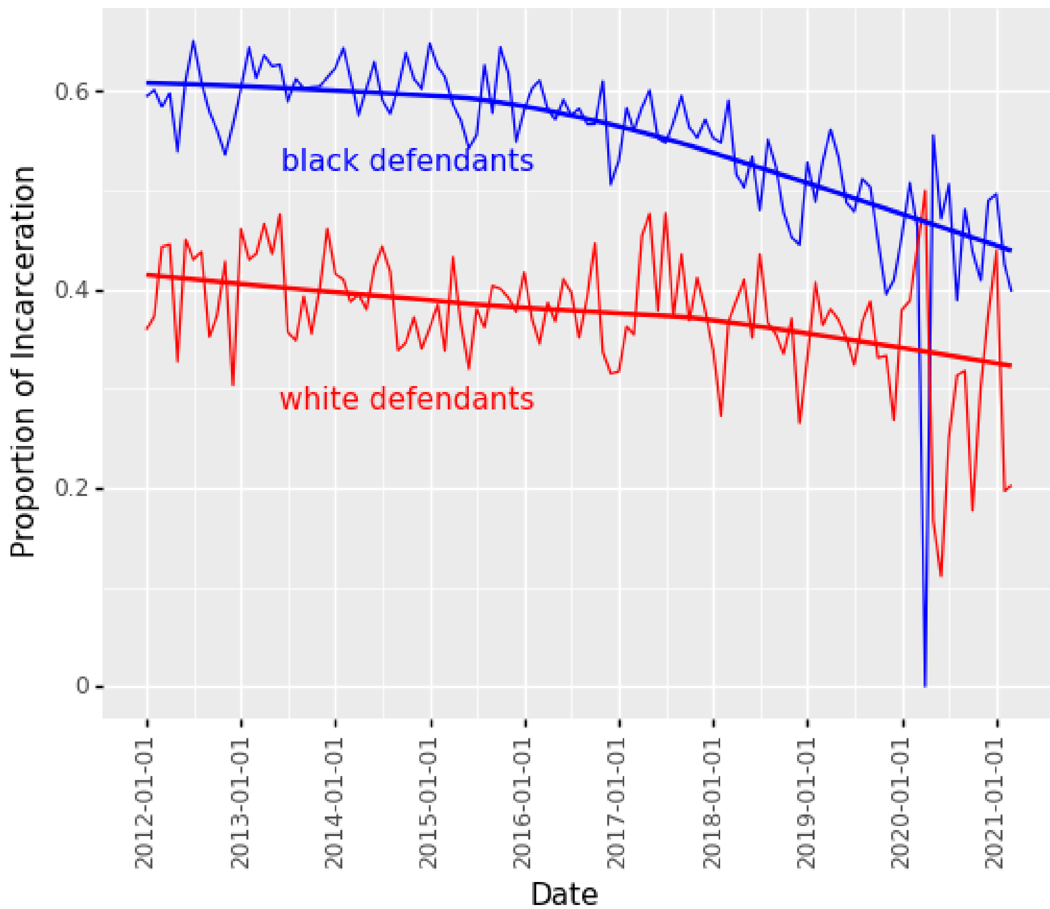
[89]:
```
sent_cleaned["black_incarc"] = np.where((sent_cleaned.is_black_derived == True)␣
↪&
```

```
                                        (sent_cleaned.COMMITMENT_TYPE ==␣
↪"Illinois Department of Corrections"),
                                        True, False)

sent_cleaned["white_incarc"] = np.where((sent_cleaned.is_white_derived == True)␣
↪&
                                        (sent_cleaned.COMMITMENT_TYPE ==␣
↪"Illinois Department of Corrections"),
                                        True, False)

table2 = sent_cleaned.groupby("sentenceym_derived").agg(black_total =␣
↪("is_black_derived","sum"),
                                                        white_total =␣
↪("is_white_derived", "sum"),
                                                        all_total = ("CASE_ID",␣
↪"nunique"),
                                                        black_incarc =␣
↪("black_incarc", "sum"),
                                                        white_incarc =␣
↪("white_incarc", "sum")).reset_index()

table2["black_incarc_proportion"] = table2.black_incarc / table2.black_total
table2["white_incarc_proportion"] = table2.white_incarc / table2.white_total
table2["difference(black-white)"] = table2.black_incarc_proportion - table2.
↪white_incarc_proportion
```

[90]:

```
plot_table2 = ggplot(table2, aes(x = "sentenceym_derived")) + \
    geom_line(aes(y = 'black_incarc_proportion'), color = "blue") + \
    geom_line(aes(y = 'white_incarc_proportion'), color = "red") + \
    stat_smooth(aes(y = 'black_incarc_proportion'), color = "blue") + \
    stat_smooth(aes(y = 'white_incarc_proportion'), color = "red") + \
    annotate("text", x = pd.to_datetime("2014-10-05"), y = .53, label = "black␣
↪defendants", color = "blue") + \
    annotate("text", x = pd.to_datetime("2014-10-05"), y = .29, label = "white␣
↪defendants", color = "red") + \
    theme(axis_text_x = element_text(angle = 90)) + \
    labs(x="Date", y= "Proportion of Incarceration")


plot_table2
```

```
/opt/conda/lib/python3.8/site-packages/plotnine/stats/smoothers.py:310:
PlotnineWarning: Confidence intervals are not yet implementedfor lowess
smoothings.
/opt/conda/lib/python3.8/site-packages/plotnine/stats/smoothers.py:310:
PlotnineWarning: Confidence intervals are not yet implementedfor lowess
smoothings.
```



[90]: <ggplot: (8748630725599)>

[91]:
```
# Proportions of incarceration seem to be decreasing for white and black␣
↪defendants,
# and the gap between races is also shrinking slightly. Notably, in the first␣
↪few months of 2020,
# the incarceration proportions for black defendants decrease signficantly for␣
↪black defendant to
# and extent that is lower than White defendent; however, the decrease is␣
↪follows by an immediate increase.
```

3.4 2.3: Investigating mechanisms: incaceration rates by charge

Your colleague sees the previous graph and is worried that the gap could be different—either wider or smaller—if you adjust for the fact that prosecutors have discretion in what crimes to charge defendants with. If white defendants are charged with crimes that tend to receive probation rather than incarceration, that could explain some of the gaps.

In the next questions, you'll begin to investigate this.

3.4.1 2.3.1: Find the most common offenses (3 points)

First, create a set of 'frequent offenses' that represent (over the entire period) the union of the 10 offenses Black defendant are most likely to be charged with and the 10 offenses white defendants are most likely to be charged with (might be far less than 20 total if there's a lot of overlap in common charges)

Use the `simplified_offense_derived` for this

[92]:
```
white_frequent = sent_cleaned["simplified_offense_derived"][sent_cleaned.
↪is_white_derived==True].value_counts().head(10).to_frame().reset_index()
white_frequent = set(white_frequent["index"])

black_frequent = sent_cleaned["simplified_offense_derived"][sent_cleaned.
↪is_black_derived==True].value_counts().head(10).to_frame().reset_index()
black_frequent = set(black_frequent["index"])

not_in_black = set(white_frequent - black_frequent)
frequent = list(black_frequent) + list(not_in_black)
frequent
```

[92]:
```
['Battery',
 'Theft',
 'Vehicle-related',
 'Burglary',
 'Driving With Suspended Or Revoked License',
 'DUI',
 'Robbery',
 'Narcotics',
 'Retail Theft',
 'UUW - Unlawful Use of Weapon',
 'Other Offense',
 'Residential Burglary']
```

3.4.2 2.3.2: Look at incarceration rates (again just whether incarcerated) by race and offense type for these top offenses (3 points)

Print a wide-format version of the resulting table (so each row is an offense type, one col is black incarceration rate for that offense type; another is the white incarceration rate) and interpret. Which offenses show the largest disparities in judges being less likely to sentence White defendants to incarceration/more likely to offer those defendants probation?

[93]:
```
table3 = sent_cleaned[sent_cleaned.simplified_offense_derived.isin(frequent)]

table3 = table3.groupby("simplified_offense_derived").agg(black_total =␣
↪("is_black_derived","sum"),
                                                          white_total =␣
↪("is_white_derived", "sum"),
```

```
                                                    all_total = ("CASE_ID",␣
↪"nunique"),
                                                    black_incarc =␣
↪("black_incarc", "sum"),
                                                    white_incarc =␣
↪("white_incarc", "sum"))


table3["black_incarceration_rate"] = table3.black_incarc / table3.black_total
table3["white_incarceration_rate"] = table3.white_incarc / table3.white_total
table3["Difference_Btw_Rate_Black_Minus_White"] =␣
↪table3["black_incarceration_rate"] - table3["white_incarceration_rate"]

table3[["black_incarceration_rate", "white_incarceration_rate",␣
↪"Difference_Btw_Rate_Black_Minus_White"]].
↪sort_values("Difference_Btw_Rate_Black_Minus_White", ascending=False)
```

[93]:

```
                                            black_incarceration_rate  \
simplified_offense_derived
Narcotics                                                   0.537876
Battery                                                     0.509032
UUW - Unlawful Use of Weapon                                0.696191
Retail Theft                                                0.595713
Burglary                                                    0.661927
Other Offense                                               0.340580
Driving With Suspended Or Revoked License                   0.582193
Theft                                                       0.386961
Residential Burglary                                        0.717791
DUI                                                         0.404997
Robbery                                                     0.650341
Vehicle-related                                             0.553750

                                            white_incarceration_rate  \
simplified_offense_derived
Narcotics                                                   0.242535
Battery                                                     0.275552
UUW - Unlawful Use of Weapon                                0.481973
Retail Theft                                                0.449582
Burglary                                                    0.535763
Other Offense                                               0.230228
Driving With Suspended Or Revoked License                   0.496552
Theft                                                       0.318721
Residential Burglary                                        0.654114
DUI                                                         0.341855
Robbery                                                     0.589286
Vehicle-related                                             0.585635
```

```
                                        Difference_Btw_Rate_Black_Minus_White
simplified_offense_derived
Narcotics                                                            0.295341
Battery                                                              0.233480
UUW - Unlawful Use of Weapon                                         0.214217
Retail Theft                                                         0.146132
Burglary                                                             0.126164
Other Offense                                                        0.110351
Driving With Suspended Or Revoked License                            0.085642
Theft                                                                0.068240
Residential Burglary                                                 0.063677
DUI                                                                  0.063143
Robbery                                                              0.061055
Vehicle-related                                                     -0.031885
```

[94]:
```
# Racial disparities in incarceration rates are greatest for Narcotics,␣
↪Battery, and UUW charges. For all charges
# other than vehicle-realted offenses, judges are more likely to incarcerate␣
↪black defendents than white. This may be
# a result of judges being more likely to offer white defendants probation.
```

3.4.3 2.3.3: Examine whether this changes pre and post change to charging threshold for retail theft (13 points)

One important question is not only whether there are disparities by offense type but also whether these disparities are changing over time.

The SAO, for instance, announced in December of 2016 that they would no longer default to charging retail thefts of under $1,000 as felonies. This change might have (1) decreased disparities or (2) increased disparities, depending on the correlation between race/ethnicity and magnitude of goods stolen: news coverage.

Focusing on `simplified_offense_derived` == "Retail theft." Using a function and/or loop (Dec. 2016 is always excluded as a transition month):

- Compare Black-White disparities before and after the change using a two-month bandwidth (so pre is October and November 2016; post is January and February 2017)
- Compare Black-White disparities before and after the change using a four-month bandwidth (so pre is August- November 2016; post is January - April 2017)
- Compare Black-White disparities using an eight-month bandwidth
- Compare Black-White disparities using a twelve-month bandwidth

---

- Print a table with the results (any organization is fine as long as it's clear)
- Create a bar chart where the x axis represents different bandwidths (2, 4, etc); the y axis the size of the Black-White gap in whether the defendant receives incarceration, and for each of the x axis points, you have one shaded bar representing "before" the change, another representing

"after" the change (make sure that before is ordered before after and the bandwidths are from smallest to largest)

Note: for each of the bandwidths include dates the entire month (e.g., for the first, include not only 02-01-2017 but everything up through 02-28-2017; easiest way is for the subsetting to use the rounded `sentenceym_derived`). Also make sure to only include white or black defendants.

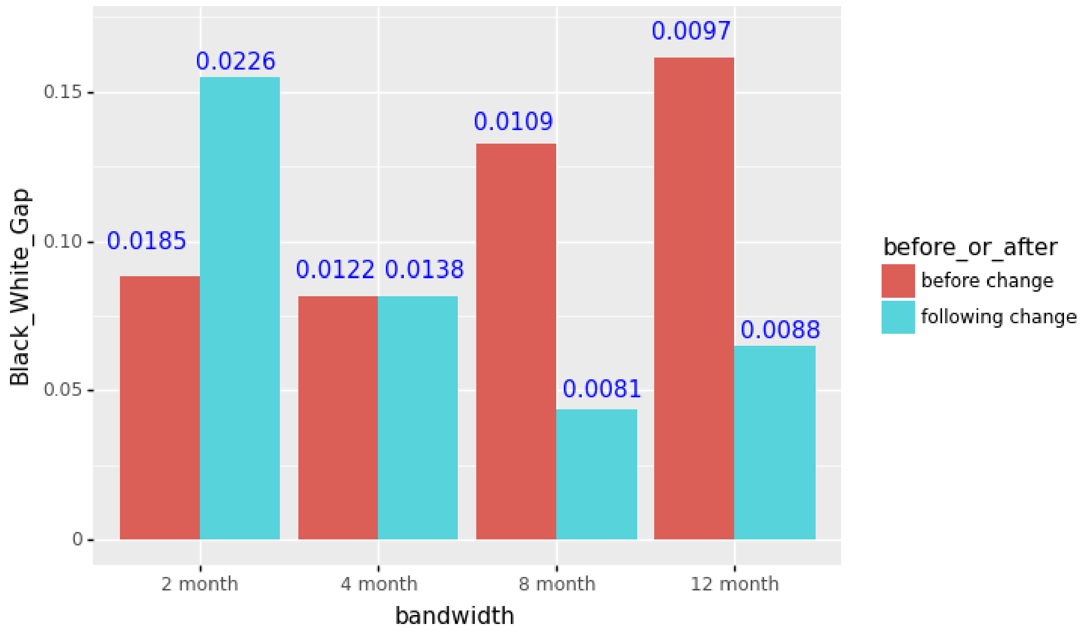
Extra credit: because the bandwidths have different sample sizes, a better viz incorporates measures of uncertainty. Add standard errors to the estimates using the formula: $(\frac{p(1-p)}{n})^{0.5}$ where p is the gap and N is the number of cases in each bandwidth period

[95]:
```
retail_theft = sent_cleaned[sent_cleaned.simplified_offense_derived == "Retail␣
↪Theft"]
retail_theft = retail_theft[(retail_theft.is_black_derived == True) |␣
↪(retail_theft.is_white_derived == True)]

def bandwidth(search_in, date_min, date_max):

    new_frame = search_in[(search_in.sentenceymd_derived >= date_min) &
                          (search_in.sentenceymd_derived <= date_max)]

    new_frame = new_frame[(new_frame.sentenceymd_derived < "2016-12-01") |
                          (new_frame.sentenceymd_derived > "2016-12-31")]

    new_frame["before_or_after"] = np.where(new_frame.sentenceymd_derived <␣
↪"2016-12-01", "before change", "following change")

    table = new_frame.groupby("before_or_after").agg(black_total =␣
↪("is_black_derived","sum"),
                                                     white_total =␣
↪("is_white_derived", "sum"),
                                                     black_incarc =␣
↪("black_incarc", "sum"),
                                                     white_incarc =␣
↪("white_incarc", "sum"))

    table["black_incarc_proportion"] = table.black_incarc / table.black_total
    table["white_incarc_proportion"] = table.white_incarc / table.white_total
    table["Black_White_Gap"] = table["black_incarc_proportion"] -␣
↪table["white_incarc_proportion"]

    return(table)

two_month = bandwidth(retail_theft, "2016-10-01", "2017-02-28")
two_month["bandwidth"] = "2 month"
four_month = bandwidth(retail_theft, "2016-08-01", "2017-04-30")
four_month["bandwidth"] = "4 month"
```

```
eight_month = bandwidth(retail_theft, "2016-04-01", "2017-08-31")
eight_month["bandwidth"] = "8 month"
twelve_month = bandwidth(retail_theft, "2015-12-01", "2017-12-31")
twelve_month["bandwidth"] = "12 month"

combined = two_month.append([four_month, eight_month, twelve_month]).
 ↪reset_index()
combined

ggplot(combined, aes(x = "bandwidth", y = "Black_White_Gap", fill =␣
 ↪"before_or_after")) + \
    geom_bar(position = "dodge", stat="identity") + \
    scale_x_discrete(limits=["2 month", "4 month", "8 month", "12 month"])+ \
    annotate("text", x = 0.7, y = .1, label = "0.0185", color = "blue") + \
    annotate("text", x = 1.2, y = .16, label = "0.0226", color = "blue") + \
    annotate("text", x = 1.76, y = .09, label = "0.0122", color = "blue") + \
    annotate("text", x = 2.26, y = .09, label = "0.0138", color = "blue")+ \
    annotate("text", x = 2.76, y = .14, label = "0.0109", color = "blue") + \
    annotate("text", x = 3.26, y = .05, label = "0.0081", color = "blue") + \
    annotate("text", x = 3.76, y = .17, label = "0.0097", color = "blue") + \
    annotate("text", x = 4.26, y = .07, label = "0.0088", color = "blue")

#     annotate("text", x = pd.to_datetime("2014-10-05"), y = .29, label =␣
 ↪"white defendants", color = "red") + \
```

[95]:

```
     before_or_after  black_total  white_total  black_incarc  white_incarc  \
0     before change          182           52           114            28
1  following change          188           69            70            15
2     before change          376          130           210            62
3  following change          284          112           117            37
4     before change          703          266           400           116
5  following change          458          180           226            81
6     before change         1057          384           614           161
7  following change          560          213           286            95

   black_incarc_proportion  white_incarc_proportion  Black_White_Gap bandwidth
0                 0.626374                 0.538462         0.087912   2 month
1                 0.372340                 0.217391         0.154949   2 month
2                 0.558511                 0.476923         0.081588   4 month
3                 0.411972                 0.330357         0.081615   4 month
4                 0.568990                 0.436090         0.132900   8 month
5                 0.493450                 0.450000         0.043450   8 month
6                 0.580889                 0.419271         0.161618  12 month
7                 0.510714                 0.446009         0.064705  12 month
```



[95]: <ggplot: (8748566109344)>

3.4.4 2.3.3.1: Interpret the results (2 points)

Write a two-sentence interpretation of the results. What might this show about how people on both sides of the issue—those who argued that the retail theft policy change would narrow disparities; those who argued that the change may widen disparities–could support their claims?

[96]:
```
## Indeed, in the two month bandwidth, the policy change widens the disparities
↪significantly as shown in the graph.
## However, when we increase the bandwidth to 8 or 12 month, those who argued
↪that the policy change narrow
## dispairities are valid. Overall, the results evidence that bandwidth matters
↪and would show different results in terms of disparities.
## Lastly, it is important to note that disparities still exist even when we
↪expand the bandwidth.
```