

Live Session Module 9

H. Diana McSpadden (hdm5s)

All court filings in district and circuit courts in Virginia are public, so any time an individual appears in court, the case is filed under that individual's publicly accessible criminal record. That happens regardless of the outcome of the case: even dismissed charges appear on a person's record. These records, especially convictions, are compiled by private companies that perform criminal background checks for rental, job, and loan applications. The effect is the creation of a permanent underclass of citizens in Virginia: people who have already served all legal punishments who continue to be punished by the lack of access to housing, employment, and financing.

Record sealing is the process by which a person's criminal record is removed from the public record. It doesn't destroy the records (that would be "expungement") as the records persist in a private database for use by state legal and law enforcement agencies, but it does prevent these records from appearing on non-governmental criminal record background checks. Until 2021, the only records that could be sealed for individuals in Virginia's legal system were charges that were outright dismissed or applied to the wrong individual due to clerical errors or identity theft. Even then, to seal a record, an individual was required to use a lengthy and difficult petition process to do so. As such, very few records in Virginia have been sealed under the previous law.

In 2021 the Virginia state legislature passed a new law [Links to an external site.](#) allowing for automatic sealing of certain criminal records. The Legal Aid Justice Center [Links to an external site.](#) has taken on the work of lobbying state government to remove some of the restrictions included in the law for the purpose of simplifying the law, expanding the number of people who can benefit from criminal record sealing, and reduce the racial disparity among individuals who qualify for record sealing. Code for Charlottesville [Links to an external site.](#) is assisting this effort by providing data analysis to (1) encode whether any given case can be automatically sealed, sealed by petition, or is ineligible for record sealing, (2) report frequencies of these outcomes statewide and within particular localities, and (3) consider how these frequencies would change given one or more hypothetical changes that can be made to the record sealing law in the future.

The data originate from Virginia's Online Case Information System which allows anyone to look up any record by name, location, date, and other fields. The public data were compiled and made available for bulk download [Links to an external site.](#) for the first time by Code for Hampton Roads volunteer Ben Schoenfeld. The files contain records of all cases that have appeared in the Virginia district and circuit courts since 2009 (with some courts providing data back to 2000). The data contain about 9 million cases from about 3 million different

people. The following data file is a random sample of 100,000 people (anonymized) from this dataset with their whole records (about 350,000 rows total):

```
In [ ]: import pandas as pd
import numpy as np
```

```
In [ ]: # try to load the large dataset
df_100k = pd.read_csv('data100k.csv')
```

```
In [ ]: df_100k.head()
```

```
Out[ ]:
```

	person_id	HearingDate	CodeSection	codesection	ChargeType	chargetype	Class	D
0	102090000000110	2019-02-28	A.46.2-862	covered elsewhere	Misdemeanor	Misdemeanor	1	
1	343221000000125	2009-12-07	B.46.2-301	covered elsewhere	Misdemeanor	Misdemeanor	1	
2	343221000000125	2011-01-20	A.46.2-707	covered elsewhere	Misdemeanor	Misdemeanor	3	
3	343221000000125	2011-07-01	B.46.2-301	covered elsewhere	Misdemeanor	Misdemeanor	1	
4	343221000000125	2012-10-15	B.46.2-301	covered elsewhere	Misdemeanor	Misdemeanor	1	

5 rows × 28 columns

Goals for this live coding session:

1. We will decide in class on some interesting questions we can answer with this dataset, such as
 - What code sections are most frequent?
 - Which ones most often lead to convictions?
 - Which ones have the most severe racial disparities?
 - In what localities (fips) are these disparities most severe?
2. But we will use pandas as needed to clean the data in specific, necessary ways prior to these analyses, including:

- Loading the data
- Recoding and collapsing categories
- Dropping extraneous columns
- Filtering rows to specific code sections or localities
- Aggregating to race, locality, and/or code section level data

```
In [ ]: df_100k.head(1).T
```

Out[]:

0

person_id	102090000000110
HearingDate	2019-02-28
CodeSection	A.46.2-862
codesection	covered elsewhere
ChargeType	Misdemeanor
chargetype	Misdemeanor
Class	1
DispositionCode	Guilty
disposition	Conviction
Plea	NaN
Race	Black(Non-Hispanic)
Sex	Male
fips	25
convictions	True
arrests	False
felony10	False
sevenyear	False
tenyear	False
within7	True
within10	True
class1_2	False
class3_4	False
expungable	Automatic (pending)
old_expungable	False
expungable_no_lifetimelimit	Automatic (pending)
reason	Conviction of misdemeanor charges listed in 19...
sameday	False
lifetime	False

What violations of the law have the greatest racial disparity?

- Black(Non-Hispanic)

In []: `df_100k['Race'].value_counts()`

```
Out[ ]: White Caucasian(Non-Hispanic)      114421
        Black(Non-Hispanic)                80173
        White Caucasian (Non-Hispanic)     41679
        Black (Non-Hispanic)              33254
        Hispanic                          9319
        White                             3527
        Other(Includes Not Applicable.. Unknown) 3452
        Asian Or Pacific Islander         2787
        Black                             2200
        MISSING                           1022
        Unknown (Includes Not Applicable.. Unknown) 785
        Other (Includes Not Applicable.. Unknown) 615
        American Indian                   302
        Unknown                           54
        Asian or Pacific Islander          7
        American Indian Or Alaskan Native  1
        Name: Race, dtype: int64
```

```
In [ ]: df_100k['Race'].unique()
```

```
Out[ ]: array(['Black(Non-Hispanic)', 'Hispanic', 'White Caucasian(Non-Hispanic)',
              'MISSING', 'Asian Or Pacific Islander', 'Black (Non-Hispanic)',
              'White Caucasian (Non-Hispanic)',
              'Other(Includes Not Applicable.. Unknown)',
              'Other (Includes Not Applicable.. Unknown)', 'Black', 'White',
              'Unknown (Includes Not Applicable.. Unknown)', 'American Indian',
              'Unknown', 'Asian or Pacific Islander',
              'American Indian Or Alaskan Native'], dtype=object)
```

```
In [ ]: # Black (Non-Hispanic) & Black(Non-Hispanic) should be the same class
        # Let's fix that
        # make courts a copy of the df_100k
        courts = df_100k.copy()

        race_map = {'Black(Non-Hispanic)': 'Black',
                    'Hispanic': 'Hispanic',
                    'White Caucasian(Non-Hispanic)': 'White',
                    'MISSING': 'Other',
                    'Asian Or Pacific Islander': 'Asian or Pacific Islander',
                    'Black (Non-Hispanic)': 'Black',
                    'White Caucasian (Non-Hispanic)': 'White',
                    'Other(Includes Not Applicable.. Unknown)': 'Other',
                    'Other (Includes Not Applicable.. Unknown)': 'Other',
                    'Black': 'Black',
                    'White': 'White',
                    'Unknown (Includes Not Applicable.. Unknown)': 'Other',
                    'American Indian': 'American Indian or Alaskan Native',
                    'Unknown': 'Other',
                    'Asian or Pacific Islander': 'Asian or Pacific Islander',
                    'American Indian Or Alaskan Native': 'American Indian or Alaskan Native'
                    }

        courts['Race'] = courts['Race'].replace(race_map)

In [ ]: courts['Race'].value_counts()
```

```
Out[ ]: White      159627
        Black      115627
        Hispanic    9319
        Other       5928
        Asian or Pacific Islander 2794
        American Indian or Alaskan Native 303
        Name: Race, dtype: int64
```

What we need is to get a count of CodeSection and Race

```
In [ ]: race_codesection = courts.groupby(['CodeSection','Race']).size().reset_index()
        race_codesection = race_codesection.rename(columns={0:'Count'})
```

```
In [ ]: race_codesection
```

```
Out[ ]:
```

	CodeSection	Race	Count
0	(74-4) 26-123	Black	1
1	01-2007	White	1
2	1	Black	5
3	1	White	3
4	1-12	Black	62
...
6635	Z.18.2-91	White	166
6636	Z.18.2-91; 26	Black	1
6637	Z.18.2-92	Black	1
6638	Z.18.2-95	Black	2
6639	Z18.2-47	Black	1

6640 rows × 3 columns

```
In [ ]: # do a reshape - one column for each of the race groups - we want it wider
        race_codesection_pivot = race_codesection.pivot_table(index=['CodeSection'],columns
```

```
In [ ]: race_codesection_pivot
```

Out[]:

CodeSection	Race	Count					
		American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White
(74-4) 26-123		0	0	1	0	0	0
01-2007		0	0	0	0	0	1
1		0	0	5	0	0	3
1-12		0	0	62	0	0	13
1-200		0	0	26	0	1	17
...	
Z.18.2-91		0	3	131	2	0	166
Z.18.2-91; 26		0	0	1	0	0	0
Z.18.2-92		0	0	1	0	0	0
Z.18.2-95		0	0	2	0	0	0
Z18.2-47		0	0	1	0	0	0

4207 rows × 6 columns

```
In [ ]: # want to get rid of the multi-index on the columns
        race_codesection_pivot = race_codesection_pivot.droplevel(0,axis=1)
```

```
In [ ]: race_codesection_pivot
```

Out[]:

Race	American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White
CodeSection						
(74-4) 26-123	0	0	1	0	0	0
01-2007	0	0	0	0	0	1
1	0	0	5	0	0	3
1-12	0	0	62	0	0	13
1-200	0	0	26	0	1	17
...
Z.18.2-91	0	3	131	2	0	166
Z.18.2-91; 26	0	0	1	0	0	0
Z.18.2-92	0	0	1	0	0	0
Z.18.2-95	0	0	2	0	0	0
Z18.2-47	0	0	1	0	0	0

4207 rows × 6 columns

```
In [ ]: race_codesection_pivot = race_codesection_pivot.assign(total = race_codesection_piv
+ race_codesection_pivot['As
+ race_codesection_pivot['Bl
+ race_codesection_pivot['Hi
+ race_codesection_pivot['Ot
+ race_codesection_pivot['Wh
```

```
In [ ]: race_codesection_pivot
```


Out[]:

Race	American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White	total
CodeSection							
(74-4) 26-123	0	0	1	0	0	0	1
01-2007	0	0	0	0	0	1	1
1	0	0	5	0	0	3	8
1-12	0	0	62	0	0	13	75
1-200	0	0	26	0	1	17	44
...
Z.18.2-91	0	3	131	2	0	166	302
Z.18.2-91; 26	0	0	1	0	0	0	1
Z.18.2-92	0	0	1	0	0	0	1
Z.18.2-95	0	0	2	0	0	0	2
Z18.2-47	0	0	1	0	0	0	1

4207 rows × 7 columns

```
In [ ]: race_codesection_pivot = race_codesection_pivot.assign(perc_black = race_codesectio
race_codesection_pivot = race_codesection_pivot.assign(perc_white = race_codesectio
race_codesection_pivot = race_codesection_pivot.assign(disparity = (race_codesectio
```

```
In [ ]: race_codesection_pivot
```

Out[]:

Race	American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White	total	perc_black	perc_white	di
CodeSection										
(74-4) 26-123	0	0	1	0	0	0	1	1.000000	0.000000	1.
01-2007	0	0	0	0	0	1	1	0.000000	1.000000	-1.
1	0	0	5	0	0	3	8	0.625000	0.375000	0.
1-12	0	0	62	0	0	13	75	0.826667	0.173333	0.
1-200	0	0	26	0	1	17	44	0.590909	0.386364	0.
...
Z.18.2-91	0	3	131	2	0	166	302	0.433775	0.549669	-0.
Z.18.2-91; 26	0	0	1	0	0	0	1	1.000000	0.000000	1.
Z.18.2-92	0	0	1	0	0	0	1	1.000000	0.000000	1.
Z.18.2-95	0	0	2	0	0	0	2	1.000000	0.000000	1.
Z18.2-47	0	0	1	0	0	0	1	1.000000	0.000000	1.

4207 rows × 10 columns

we don't care if the total is low, we need statistical significance

```
In [ ]: # filter the rows if the values are low
race_codesection_pivot_filtered = race_codesection_pivot.query("total >= 200")
race_codesection_pivot_filtered
```

Out[]:

Race	American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White	total	perc_black	perc_white	di
CodeSection										
16.1-253.2	0	0	56	1	0	152	209	0.267943	0.727273	-0.
17-7	0	10	43	0	0	151	204	0.210784	0.740196	-0.
18.2-102	0	1	129	2	1	127	260	0.496154	0.488462	0.
18.2-103	1	26	1295	31	17	2164	3534	0.366440	0.612337	-0.
18.2-104	0	2	266	1	7	349	625	0.425600	0.558400	-0.
...
C.46.2-894	2	4	91	2	6	216	321	0.283489	0.672897	-0.
C.46.2-896	0	5	99	3	4	273	384	0.257812	0.710938	-0.
MISSING	0	0	129	5	8	104	246	0.524390	0.422764	0.
NO DMV	0	4	175	16	6	202	403	0.434243	0.501241	-0.
Z.18.2-91	0	3	131	2	0	166	302	0.433775	0.549669	-0.

145 rows × 10 columns

when is % higher for blacks than whites and when it is reversed.

```
In [ ]: race_codesection_pivot_filtered.sort_values(by='disparity',ascending=False)
```

Out[]:

Race	American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White	total	perc_black	perc_white	di
CodeSection										
29-48	0	1	228	0	1	30	260	0.876923	0.115385	0.
18.2-53.1	0	5	1411	16	5	309	1746	0.808133	0.176976	0.
46.2-938	0	1	319	7	2	69	398	0.801508	0.173367	0.
18.2-58	0	6	1142	18	10	400	1576	0.724619	0.253807	0.
18.2-32	0	1	197	3	1	74	276	0.713768	0.268116	0.
...
4.1-305	3	26	444	41	29	2245	2788	0.159254	0.805237	-0.
54.1-3466	0	4	100	2	2	709	817	0.122399	0.867809	-0.
18.2-374.1:1	0	0	60	1	25	583	669	0.089686	0.871450	-0.
18.2-258.1	0	1	83	2	13	767	866	0.095843	0.885681	-0.
29.1-735	0	2	13	4	1	194	214	0.060748	0.906542	-0.

145 rows × 10 columns

Picking up from Last Week

There wasn't geography information in this dataset

We want to break this down by geographic location and compare to census data and compare local racial statistics

```
In [ ]: #UVA\ds6001\mod9\Live\Census_2019_RaceEstimates_forVA_0.xls
df_Race = pd.read_excel('Census_2019_RaceEstimates_forVA_0.xls', skiprows=[0,1,2,3,5
```

```
In [ ]: # rename columns
df_Race = df_Race.rename(columns={'Unnamed: 4': 'White_Prop', 'Unnamed: 6': 'AA_Pro
df_Race = df_Race[['FIPS', 'Jurisdiction', 'Total Population', 'White_Prop', 'AA_Pro
df_Race.head()
```

Out[]:

	FIPS	Jurisdiction	Total Population	White_Prop	AA_Prop	Asian_Prop	Other_Prop	Two_More_Prop
0	1	Accomack County	32316	0.677652	0.287907	0.007953	0.009067	0.017422
1	3	Albemarle County	109330	0.817598	0.096954	0.055346	0.004418	0.025684
2	5	Alleghany County	14860	0.927524	0.046972	0.003096	0.003769	0.018641
3	7	Amelia County	13145	0.764549	0.204488	0.006086	0.006466	0.018410
4	9	Amherst County	31605	0.768834	0.191141	0.005695	0.009650	0.024680

In []:

```
# UVa\ds6001\mod9\Live\Census_2019_HispanicEstimates_forVA_0.xls
df_Hispanic = pd.read_excel('Census_2019_HispanicEstimates_forVA_0.xls', skiprows=9)
df_Hispanic = df_Hispanic[['Unnamed: 0', 'Unnamed: 7']]
df_Hispanic = df_Hispanic.rename(columns={'Unnamed: 0': 'FIPS', 'Unnamed: 7': 'Hispanic_Prop'})
df_Hispanic.head()
```

Out[]:

	FIPS	Hispanic_Prop
0	1	0.091441
1	3	0.057743
2	5	0.016016
3	7	0.031799
4	9	0.024268

Now, we need to merge on the FIPS.

BUT do the FIPS match?

```
In [ ]: df_Race.FIPS.nunique()
```

Out[]: 133

```
In [ ]: df_Hispanic.FIPS.nunique()
```

Out[]: 133

At least they have the same number of unique values.

First a full join with diagnostics to see if the merge is OK

```
In [ ]: census = pd.merge(df_Race, df_Hispanic,
                          on='FIPS',
```

```
how = 'outer',  
indicator='matched',  
validate='one_to_one')
```

start with an outer join for diagnostics

indicator is the column that shows where the data came from

```
In [ ]: census['matched'].value_counts()
```

```
Out[ ]: both          133  
left_only         0  
right_only        0  
Name: matched, dtype: int64
```

Second, the merge we actually are doing

```
In [ ]: census = pd.merge(df_Race, df_Hispanic,  
                          on='FIPS',  
                          how = 'inner')
```

```
In [ ]: census
```

Out[]:

	FIPS	Jurisdiction	Total Population	White_Prop	AA_Prop	Asian_Prop	Other_Prop	Two_More_Prc
0	1	Accomack County	32316	0.677652	0.287907	0.007953	0.009067	0.0174
1	3	Albemarle County	109330	0.817598	0.096954	0.055346	0.004418	0.0256
2	5	Alleghany County	14860	0.927524	0.046972	0.003096	0.003769	0.0186
3	7	Amelia County	13145	0.764549	0.204488	0.006086	0.006466	0.0184
4	9	Amherst County	31605	0.768834	0.191141	0.005695	0.009650	0.0246
...
128	800	Suffolk city	92108	0.521051	0.426087	0.019043	0.005157	0.0286
129	810	Virginia Beach city	449974	0.673777	0.202592	0.073873	0.006460	0.0432
130	820	Waynesboro city	22630	0.807114	0.134335	0.016880	0.006584	0.0350
131	830	Williamsburg city	14954	0.736458	0.159890	0.060452	0.006955	0.0362
132	840	Winchester city	28078	0.814232	0.113042	0.027210	0.009937	0.0355

133 rows × 9 columns

Coming back to the court data, we need data that counts the number of cases in each code section/race/FIPS combination

In []: `courts['Race'].value_counts()`

```
Out[ ]: White          159627
        Black          115627
        Hispanic         9319
        Other           5928
        Asian or Pacific Islander  2794
        American Indian or Alaskan Native  303
        Name: Race, dtype: int64
```

In []: `courts.columns`

```
Out[ ]: Index(['person_id', 'HearingDate', 'CodeSection', 'codesection', 'ChargeType',
              'chargetype', 'Class', 'DispositionCode', 'disposition', 'Plea', 'Race',
              'Sex', 'fips', 'convictions', 'arrests', 'felony10', 'sevenyear',
              'tenyear', 'within7', 'within10', 'class1_2', 'class3_4', 'expungable',
              'old_expungable', 'expungable_no_lifetimelimit', 'reason', 'sameday',
              'lifetime'],
              dtype='object')
```

```
In [ ]: courts_fips = courts.groupby(by=['CodeSection', 'Race', 'fips']).size().reset_index()
```

```
In [ ]: courts_fips
```

```
Out[ ]:
```

	CodeSection	Race	fips	0
0	(74-4) 26-123	Black	550	1
1	01-2007	White	51	1
2		1 Black	550	5
3		1 White	550	2
4		1 White	810	1
...
37580	Z.18.2-91; 26	Black	700	1
37581	Z.18.2-92	Black	760	1
37582	Z.18.2-95	Black	67	1
37583	Z.18.2-95	Black	83	1
37584	Z18.2-47	Black	730	1

37585 rows × 4 columns

```
In [ ]: courts_fips = courts_fips.rename(columns={0: 'count'})
```

```
In [ ]: courts_fips
```


Out[]:

	CodeSection	Race	fips	count
0	(74-4) 26-123	Black	550	1
1	01-2007	White	51	1
2	1	Black	550	5
3	1	White	550	2
4	1	White	810	1
...
37580	Z.18.2-91; 26	Black	700	1
37581	Z.18.2-92	Black	760	1
37582	Z.18.2-95	Black	67	1
37583	Z.18.2-95	Black	83	1
37584	Z18.2-47	Black	730	1

37585 rows × 4 columns

```
In [ ]: # do a reshape
courts_fips_pivot = courts_fips.pivot_table(index=['CodeSection', 'fips'],
                                             columns='Race',
                                             values='count',
                                             fill_value=0)
```

```
In [ ]: courts_fips_pivot
```

Out[]:

	Race	American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White
CodeSection	fips						
(74-4) 26-123	550	0	0	1	0	0	0
01-2007	51	0	0	0	0	0	1
1	550	0	0	5	0	0	2
	810	0	0	0	0	0	1
1-12	650	0	0	62	0	0	13
...
Z.18.2-91; 26	700	0	0	1	0	0	0
Z.18.2-92	760	0	0	1	0	0	0
Z.18.2-95	67	0	0	1	0	0	0
	83	0	0	1	0	0	0
Z18.2-47	730	0	0	1	0	0	0

25077 rows × 6 columns

```
In [ ]: # now we need to merge this with the census data
# we need to convert numbers to percentages

courts_fips_pivot = courts_fips_pivot.assign(total = courts_fips_pivot['American In
courts_fips_pivot['Asian or Pacific Is
courts_fips_pivot['Black'] +
courts_fips_pivot['Hispanic'] +
courts_fips_pivot['Other'] +
courts_fips_pivot['White'])
```

```
In [ ]: courts_fips_pivot['amerind_perc'] = courts_fips_pivot['American Indian or Alaskan
courts_fips_pivot['asian_perc'] = courts_fips_pivot['Asian or Pacific Islander'] /
courts_fips_pivot['black_perc'] = courts_fips_pivot['Black'] / courts_fips_pivot['
courts_fips_pivot['hispanic_perc'] = courts_fips_pivot['Hispanic'] / courts_fips_p
courts_fips_pivot['other_perc'] = courts_fips_pivot['Other'] / courts_fips_pivot['
courts_fips_pivot['white_perc'] = courts_fips_pivot['White'] / courts_fips_pivot['
```

```
In [ ]: courts_fips_pivot
```

Out[]:

	Race	American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White	total	amerind_perc	asian
CodeSection	fips									
(74-4) 26-123	550	0	0	1	0	0	0	1	0.0	
01-2007	51	0	0	0	0	0	1	1	0.0	
1	550	0	0	5	0	0	2	7	0.0	
	810	0	0	0	0	0	1	1	0.0	
1-12	650	0	0	62	0	0	13	75	0.0	
...
Z.18.2-91; 26	700	0	0	1	0	0	0	1	0.0	
Z.18.2-92	760	0	0	1	0	0	0	1	0.0	
Z.18.2-95	67	0	0	1	0	0	0	1	0.0	
	83	0	0	1	0	0	0	1	0.0	
Z18.2-47	730	0	0	1	0	0	0	1	0.0	

25077 rows × 13 columns

```
In [ ]: courts_fips_pivot.reset_index(inplace=True)
```

Join this with census stuff

We NEED TO DO DIAGNOSTICS!!!

```
In [ ]: court_census_validate = pd.merge(courts_fips_pivot, census,
      left_on='fips', right_on='FIPS',
      how = 'outer',
      indicator='matched',
      validate='many_to_one')
```

```
In [ ]: court_census_validate['matched'].value_counts()
```

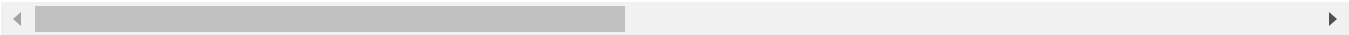
```
Out[ ]: both          23513
left_only         1564
right_only           8
Name: matched, dtype: int64
```

```
In [ ]: court_census_validate.query("matched == 'left_only'")
```

Out[]:

	CodeSection	fips	American Indian or Alaskan Native	Asian or Pacific Islander	Black	Hispanic	Other	White	total	amerind_per
4247	1-200	701.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.
4248	10-18	701.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.
4249	10-24	701.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	0.
4250	10-31	701.0	0.0	0.0	1.0	0.0	0.0	1.0	2.0	0.
4251	11-14	701.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.
...
22736	C.46.2-896	702.0	0.0	0.0	2.0	0.0	0.0	5.0	7.0	0.
22737	D.46.2-894	702.0	0.0	0.0	2.0	0.0	0.0	0.0	2.0	0.
22738	NO DMV	702.0	0.0	0.0	8.0	0.0	0.0	7.0	15.0	0.
25075	18.2-172	560.0	0.0	0.0	0.0	0.0	0.0	4.0	4.0	0.
25076	18.2-248	560.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	0.

1564 rows × 25 columns



```
In [ ]: court_census_validate.query("matched == 'right_only']").T
```

Out[]:

	25077	25078	25079	25080	25081	25082	25083	
CodeSection	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
fips	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
American Indian or Alaskan Native	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Asian or Pacific Islander	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Black	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Hispanic	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
Other	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
White	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
total	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
amerind_perc	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
asian_perc	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
black_perc	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
hispanic_perc	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
other_perc	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
white_perc	NaN	NaN	NaN	NaN	NaN	NaN	NaN	
FIPS	95.0	580.0	660.0	678.0	683.0	685.0	720.0	
Jurisdiction	James City County	Covington city	Harrisonburg city	Lexington city	Manassas city	Manassas Park city	Norton city	Pocahontas city
Total Population	76523.0	5538.0	53016.0	7446.0	41085.0	17478.0	3981.0	1000
White_Prop	0.803327	0.823221	0.825826	0.851867	0.730559	0.674105	0.897011	0.900000
AA_Prop	0.136338	0.137414	0.089011	0.085952	0.153998	0.156196	0.05903	0.000000
Asian_Prop	0.026436	0.00957	0.0441	0.02968	0.063381	0.115173	0.018086	0.000000
Other_Prop	0.005293	0.004695	0.010393	0.003358	0.015967	0.019396	0.004019	0.000000
Two_More_Prop	0.028606	0.025099	0.03067	0.029143	0.036096	0.03513	0.021854	0.000000
Hispanic_Prop	0.061315	0.02221	0.207447	0.035858	0.381064	0.410459	0.038433	0.000000
matched	right_only	right_only	right_only	right_only	right_only	right_only	right_only	right_only

Let's pretend these diagnostics were okay

```
In [ ]: court_census_validate['black_disparity'] = court_census_validate['black_perc']/cour
```

```
In [ ]: court_census_validate.sort_values(by='black_disparity',ascending=False).head(20).T
```

Out[]:

	23129	23163	23149	23152	23153	23158	2315
CodeSection	18.2-286	4VAC15-90-231	29.1-520	29.1-530.1	29.1-548	46.2-328	29.1-52
fips	45.0	45.0	45.0	45.0	45.0	45.0	45.
American Indian or Alaskan Native	0.0	0.0	0.0	0.0	0.0	0.0	0.
Asian or Pacific Islander	0.0	0.0	0.0	0.0	0.0	0.0	0.
Black	1.0	1.0	1.0	1.0	1.0	1.0	2.
Hispanic	0.0	0.0	0.0	0.0	0.0	0.0	0.
Other	0.0	0.0	0.0	0.0	0.0	0.0	0.
White	0.0	0.0	0.0	0.0	0.0	0.0	1.
total	1.0	1.0	1.0	1.0	1.0	1.0	3.
amerind_perc	0.0	0.0	0.0	0.0	0.0	0.0	0.
asian_perc	0.0	0.0	0.0	0.0	0.0	0.0	0.
black_perc	1.0	1.0	1.0	1.0	1.0	1.0	0.66666
hispanic_perc	0.0	0.0	0.0	0.0	0.0	0.0	0.
other_perc	0.0	0.0	0.0	0.0	0.0	0.0	0.
white_perc	0.0	0.0	0.0	0.0	0.0	0.0	0.33333
FIPS	45.0	45.0	45.0	45.0	45.0	45.0	45.
Jurisdiction	Craig County	Craig County	Craig County	Craig County	Craig County	Craig County	Crai Count
Total Population	5131.0	5131.0	5131.0	5131.0	5131.0	5131.0	5131.
White_Prop	0.980706	0.980706	0.980706	0.980706	0.980706	0.980706	0.98070
AA_Prop	0.004483	0.004483	0.004483	0.004483	0.004483	0.004483	0.00448
Asian_Prop	0.001559	0.001559	0.001559	0.001559	0.001559	0.001559	0.00155
Other_Prop	0.004093	0.004093	0.004093	0.004093	0.004093	0.004093	0.00409
Two_More_Prop	0.00916	0.00916	0.00916	0.00916	0.00916	0.00916	0.0091
Hispanic_Prop	0.015786	0.015786	0.015786	0.015786	0.015786	0.015786	0.01578
matched	both	both	both	both	both	both	bot
black_disparity	223.086957	223.086957	223.086957	223.086957	223.086957	223.086957	148.72463

In []: