Working with the Supreme Court Database

We will be using data from the Supreme Court Database available at: http://scdb.wustl.edu/index.php

UIC CS 111 Law, Fall 2016
Profs. Bob Sloan and Richard Warner

Primary author: Dr. Daniel M. Katz

Version: October 7, 2016

Downloading Data

The file we want is at:

http://scdb.wustl.edu/_brickFiles/2016_01/SCDB_2016_01_justiceCentered_Citation.csv.zip

There are ways to get it from inside Python, but there is no good reason to learn those. We will just download and unzip the file so we have a file named:
SCDB_2016_01_justiceCentered_Citation.csv
in a directory where we want to work.

Importing modules

It is good practice to put *all* modules you will be using at the top of your code.

We will be using (at least):

- the open-source Python Data Analysis Library pandas (http://pandas.pydata.org/)
- the standard Python plotting library, matplotlib.pyplot

So at the top of your Python file (if working in a file) or at the terminal before anything else, let's put

```python
import pandas
import matplotlib.pyplot as plt
```

Loading the data into Python from file

Open as usual, except file is encoded in ISO-8859-1, not ASCII (nor UTF-8 encoding of Unicode):

```python
fileref = open('SCDB_2016_01_justiceCentered_Citation.csv', encoding="ISO-8859-1")
```

Could use for loop over or read or readlines with fileref as usual, but we'd like to exploit the CSV format, and do some data analytics with pandas, so

```python
# Read from file with pandas preparing to exploit csv format
scdb = pandas.read_csv(fileref)
```

First Look at the Data

Data dimensions:

```python
>>> print(scdb.shape)

(78233, 61)
```

Subsetting by Rows:

- First 5 rows:

```python
scdb.head()
```

- Last 5 rows:

```python
scdb.tail()
```

- A specific row:

```python
scdb.loc[10]
```

First Look at the Data (cont.)

Rename All Columns:

First, let's create a subset called "scdb_subset" with the first 3 columns

```python
scdb_subset = scdb.iloc[:,0:3]
scdb_subset.head()
```

```
>>> scdb_subset.head()
     caseId     docketId    caseIssuesId
0  1946-001  1946-001-01  1946-001-01-01
1  1946-001  1946-001-01  1946-001-01-01
2  1946-001  1946-001-01  1946-001-01-01
3  1946-001  1946-001-01  1946-001-01-01
4  1946-001  1946-001-01  1946-001-01-01
```

Rename a specific column

```python
>>> scdb_subset.rename(columns = {'caseId': 'case_Identifier'}, inplace = True)
>>> scdb_subset.head()
```

```
  case_Identifier     docketId    caseIssuesId
0        1946-001  1946-001-01  1946-001-01-01
1        1946-001  1946-001-01  1946-001-01-01
2        1946-001  1946-001-01  1946-001-01-01
3        1946-001  1946-001-01  1946-001-01-01
4        1946-001  1946-001-01  1946-001-01-01
```

Rename all columns:

```python
>>> scdb_subset.columns = ['case_identifier', 'docket_identifier', 'caseIssues_identifier']
>>> scdb_subset.head()
```

```
  case_identifier docket_identifier caseIssues_identifier
0        1946-001       1946-001-01        1946-001-01-01
1        1946-001       1946-001-01        1946-001-01-01
2        1946-001       1946-001-01        1946-001-01-01
3        1946-001       1946-001-01        1946-001-01-01
4        1946-001       1946-001-01        1946-001-01-01
```

Visualizing Data: matplotlib

"matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. matplotlib can be used in python scripts, the python and ipython shell (ala MATLAB® or Mathematica®), web application servers, and six graphical user interface toolkits." http://matplotlib.org/

Learn More: http://matplotlib.org/

How Many Decisions Per Year Were There, Both In Total and by Justice?

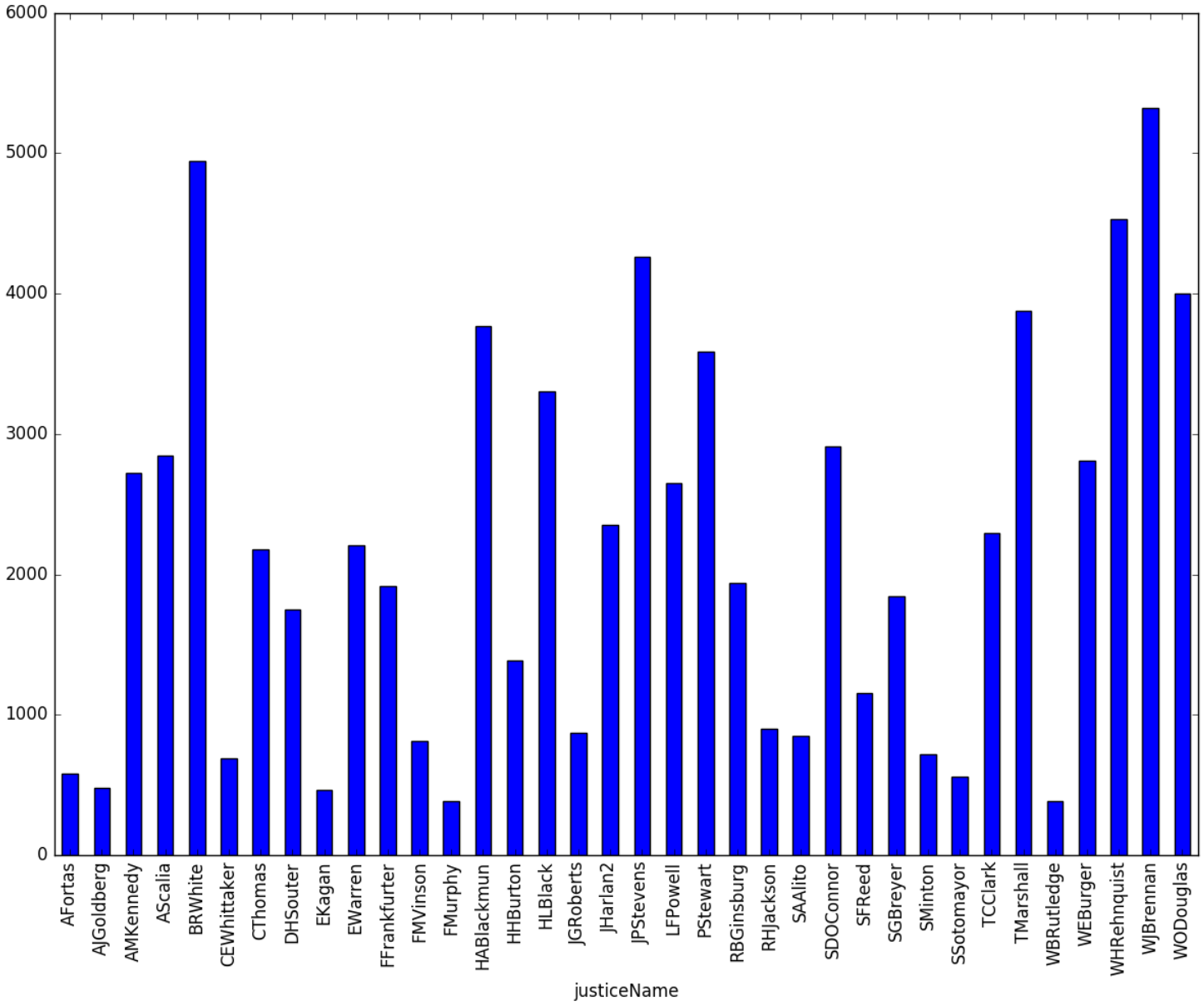
Let's plot the number of decisions by Justice in alphabetical order

```Python
>>> f = plt.figure(figsize=(12,10)) # make 12 x 10 (inches) figure
>>> scdb.groupby("justiceName")["docketId"].count().plot(kind="bar")
<matplotlib.axes._subplots.AxesSubplot object at 0x11773dc18>
>>> f.show()
```

There are a couple of minor problems:

1. If we want to keep working in our terminal window, we need to use matplotlib in its interactive mode, so in that case we should first issue the command `plt.ion()`
2. The Justice's names are long enough that the labels of the x-axis run a bit off the bottom of the page. For situations like this, we can issue the command `f.tight_layout()`

That will give

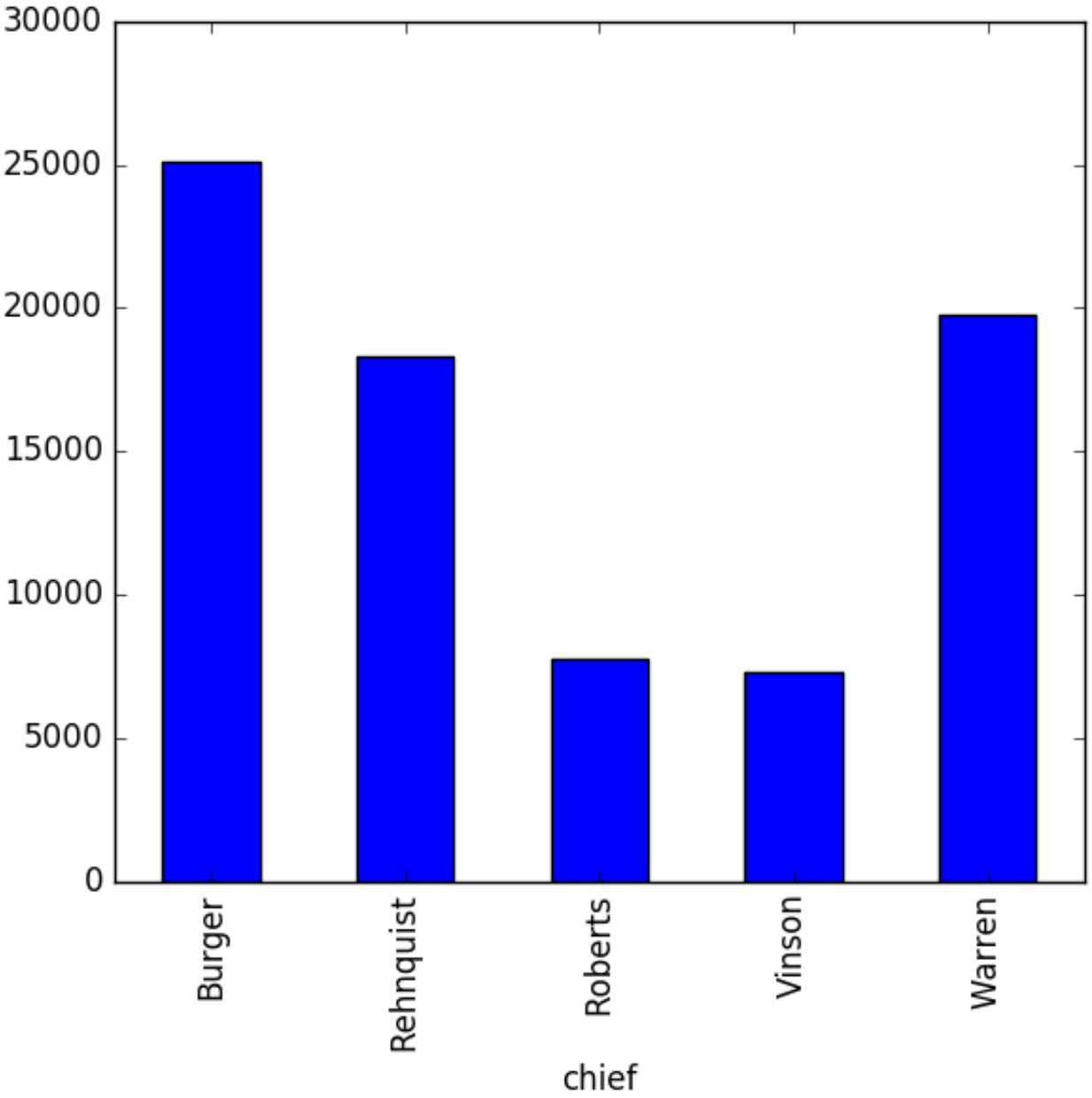


What About Cheif Justices Only?

We can try to change the groupby from "justiceName"" to "chief"

Python

```
f = plt.figure(figsize=(6,6))       # Make it a little smaller
scdb.groupby('chief')['docketId'].count().plot(kind='bar')
f.tight_layout()                          # Must come AFTER we put something in f
f.show()
```



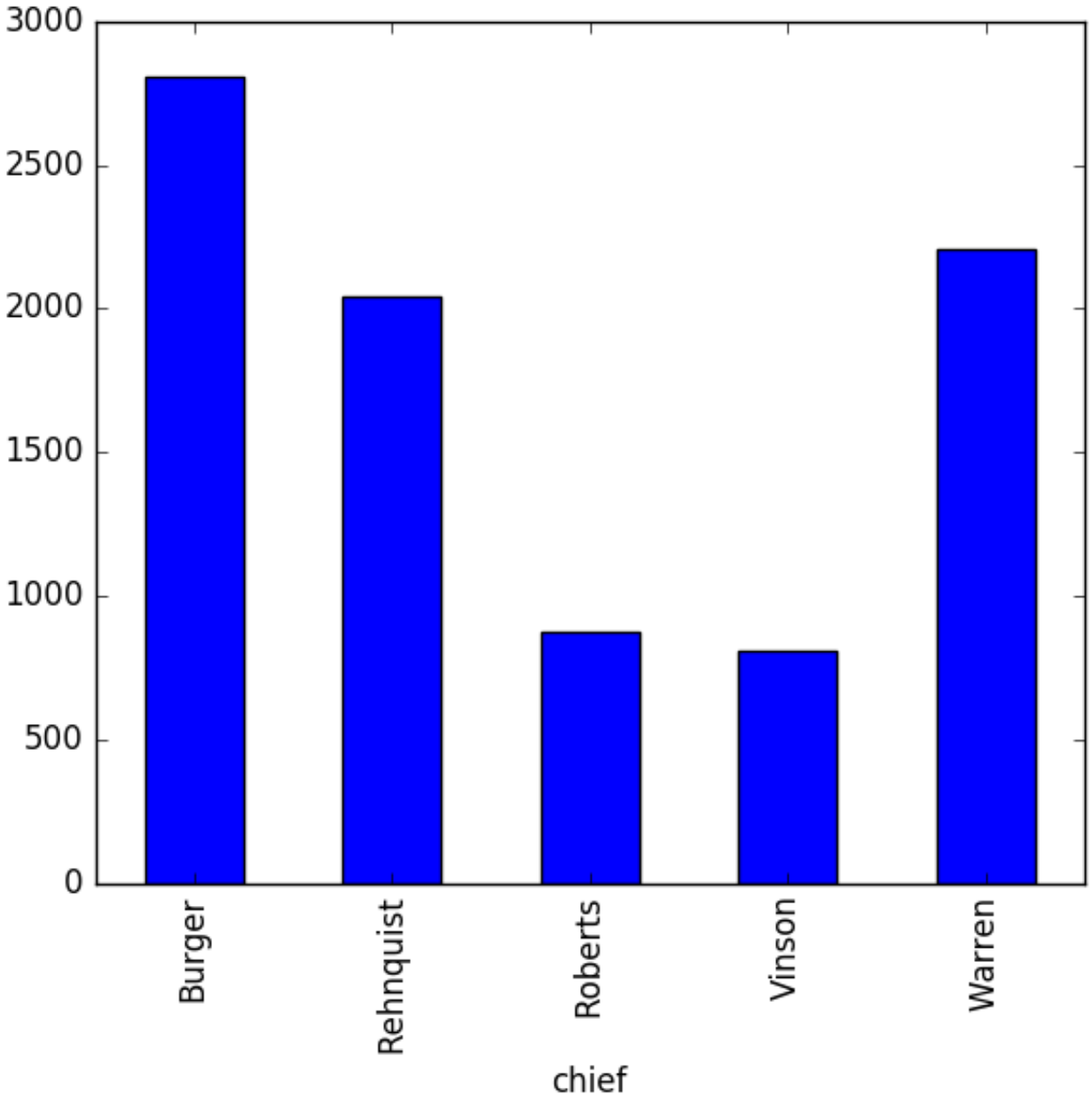
Look carefully at that graph and the one before. It's not really what we want, is it?

The numbers are a bit off.

To get the number of decisions for each chief justice, we need to use .nunique rather than .count.

Python

```
f = plt.figure(figsize=(6,6))
scdb.groupby('chief')['docketId'].nunique().plot(kind='bar')
f.tight_layout()                          # Must come AFTER we put something in f
f.show()
```



Why do we use .count for the number of decisions by each justice, but .nunique for the number of decisions by each chief justice?

Let's take a closer look at the data to see what is going on

```
>>> scdb_peek = scdb.loc[0:20, {'docketId', 'chief', 'justiceName'}]
>>> scdb_peek
     justiceName   chief     docketId
0       HHBurton  Vinson  1946-001-01
1      RHJackson  Vinson  1946-001-01
2      WODouglas  Vinson  1946-001-01
3   FFrankfurter  Vinson  1946-001-01
4         SFReed  Vinson  1946-001-01
5        HLBlack  Vinson  1946-001-01
6     WBRutledge  Vinson  1946-001-01
7        FMurphy  Vinson  1946-001-01
8       FMVinson  Vinson  1946-001-01
9       HHBurton  Vinson  1946-002-01
10     RHJackson  Vinson  1946-002-01
11     WODouglas  Vinson  1946-002-01
12  FFrankfurter  Vinson  1946-002-01
13        SFReed  Vinson  1946-002-01
14       HLBlack  Vinson  1946-002-01
15    WBRutledge  Vinson  1946-002-01
16       FMurphy  Vinson  1946-002-01
17      FMVinson  Vinson  1946-002-01
18      HHBurton  Vinson  1946-003-01
19     RHJackson  Vinson  1946-003-01
20     WODouglas  Vinson  1946-003-01
```

The first two dockets ("1946-001-01" and "1946-002-01) each have 9 rows (or records in data science speak). Each row or record represents a vote by the identified justiceName and each justice votes only once per docketId.

The "chief" column represents the Chief Justice for the given docketId and there is only one chief per docketId.

Thus, when we counted the number of rows for each Chief Justice we were counting the number of votes cast for all the docketId's that each Chief Justice presided over, including their own vote, giving us the number of decisions rendered by each chief justices * approx. 9

".count()" gives us the number of rows for each chief. This is the plot above that was incorrect:

```
>>> scdb.groupby('chief')['docketId'].count()
chief
Burger       25076
Rehnquist    18322
Roberts       7792
Vinson        7307
Warren       19736
Name: docketId, dtype: int64
```

"nunique()" gives us the number of unique docketId's for each chief, rather than the number of rows.

```
>>> scdb.groupby('chief')['docketId'].nunique()
chief
Burger       2807
Rehnquist    2040
Roberts       873
Vinson        812
Warren       2205
Name: docketId, dtype: int64
```

Let's run the count and nunique figures by justiceName to compare. Display only the last 5 rows which contain two justices that were chief justices (Burger and Rehnquist) so we can easily compare the results.

```
>>> justice_count = scdb.groupby('justiceName')['docketId'].count()
>>> justice_count.tail()
justiceName
WBRutledge     387
WEBurger      2807
WHRehnquist   4529
WJBrennan     5325
WODouglas     4001
Name: docketId, dtype: int64
```

Unlike chief, grouping by justiceName gives us the same result for .count and .nunique because each row represents a justice's vote and each justice only votes once per docketId.

Hopefully you now understand the scdb data structure well enough that you see why `scdb.groupby('chief')['docketId'].nunique().plot(kind='bar')` gave the proper plot above.

What if we were only interested in cases from certain terms?

```python
#The 2010 Term
scdb_subset = scdb[scdb.term == 2010]


#Terms 2010 to current: Let's use this one
scdb_subset = scdb[scdb.term >= 2010]
```

Data Exploration - Descriptive Statistics

Since the 2010 term, how many cases (caseId) has the court reviewed?

```python
# We use "nunique" rather than "count" because our data
# has 1 row for each voting Justice (usually 9 per case)
# but we want to know the number of distinct caseId's, not rows.
>>> scdb_subset.caseId.nunique()
463

# See the difference with count?
>>> scdb_subset.caseId.count()
4104
```

Data Exploration - Descriptive Statistics (cont.)

How many cases for each term from 2010 on?

```python
>>> scdb_subset.groupby('term').caseId.nunique()
term
2010    84
2011    77
2012    79
2013    75
2014    70
2015    78
Name: caseId, dtype: int64
```

Data Exploration - Descriptive Statistics (cont.)

-What is the average number of cases per term?

```
>>> scdb_subset.groupby('term').caseId.nunique().mean()
77.16666666666667
```

Data Exploration - Bar Plots again

- Visualize the number of cases for each term

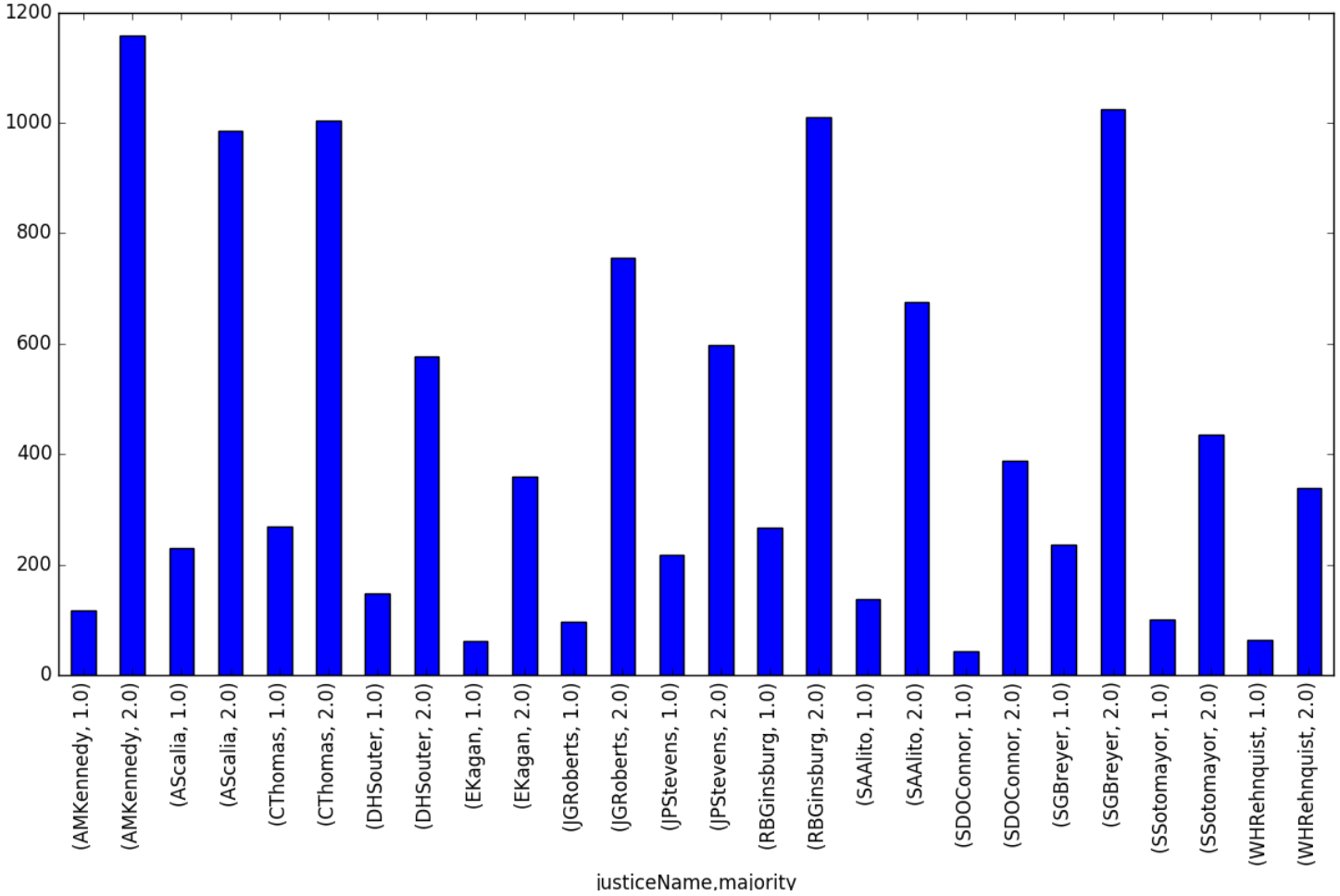
```
#Plot the number of cases for each term
f = plt.figure(figsize=(8,6))
scdb_subset.groupby('term')['caseId'].nunique().plot(kind="bar")
```

Subsetting Data again, this time inline

Since the 2000 term, see how many times each justice has voted for the dissent and majority.

And just to demo it, we'll do the subset of the terms inline

```
f = plt.figure(figsize=(12,8))
scdb[scdb.term >= 2000].groupby(['justiceName', 'majority'])['caseId'].nunique().plot(
                                                                        kind="bar")
f.tight_layout()
f.show()
```


1200
1000
800
600
400
200
0
(AMKennedy, 1.0)
(AMKennedy, 2.0)
(AScalia, 1.0)
(AScalia, 2.0)
(CThomas, 1.0)
(CThomas, 2.0)
(DHSouter, 1.0)
(DHSouter, 2.0)
(EKagan, 1.0)
(EKagan, 2.0)
(JGRoberts, 1.0)
(JGRoberts, 2.0)
(JPStevens, 1.0)
(JPStevens, 2.0)
(RBGinsburg, 1.0)
(RBGinsburg, 2.0)
(SAAlito, 1.0)
(SAAlito, 2.0)
(SDOConnor, 1.0)
(SDOConnor, 2.0)
(SGBreyer, 1.0)
(SGBreyer, 2.0)
(SSotomayor, 1.0)
(SSotomayor, 2.0)
(WHRehnquist, 1.0)
(WHRehnquist, 2.0)
justiceName,majority