starter

November 29, 2018

In [20]:
```
from urllib.request import urlopen
import re
from collections import defaultdict
import numpy as np
import chardet
import pandas as pd
import csv
import json
from bs4 import BeautifulSoup
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
def get_variable_names():
    #contains all variables corresponding to a string
    varNames_dict = {}

    # Grab the page from the web
    f = urlopen("http://scdb.wustl.edu/documentation.php?var=chief")
    html = str(f.read())


    reviews = html.split('<li class="toc">')[1:]
    for variable in reviews:
        str_1 = variable.split("</a>")[0].split('class="toc')
        if (re.split(".var=",str_1[0])[1].replace('" ','') == 'intro'):
            continue
        if (re.split(".var=",str_1[0])[1].replace('" ','') == 'cite'):
            continue
        varName_dict[str_1[1].replace('">','')] = re.split(".var=",str_1[0])[1].replace

    return varName_dict
```

In [3]:
```
def get_all_var_values(varName_dict):
    #contains all values possible for a var
    varValues_dict = {}

    for var in varName_dict.keys():
        #print("--------------------------")
```

```
#print(str(var)+":"+str(varName_dict[var]))
f = urlopen("http://scdb.wustl.edu/documentation.php?var="+str(varName_dict[va
html = str(f.read())

#for variables which are not encoded and stright values are used
if ('<strong>Values:</strong><br />' not in html):
    continue

#print(html)
# Extract the (30) review elements
reviews = html.split('<strong>Values:</strong><br />')[1]
reviews = reviews.split('<div class="toc" onClick="toggleBlock')[0]
#print(reviews)

soup = BeautifulSoup(reviews)
#print(soup.text)

final_str = soup.text.replace("\\t","#").replace("\\n","#")
#print(final_str)
list_val = re.split("#+",final_str)
list_val = [x for x in list_val if x not in ["","  "]]

assert(len(list_val)%2 == 0 or len(list_val)%3 == 0)

#choose which format it is
pattern = re.compile("\d+")

#print(len(list_val))
if (len(list_val)>4):
    if (pattern.match(list_val[0]) and pattern.match(list_val[2]) and pattern.
        div = 2
    elif (pattern.match(list_val[0]) and pattern.match(list_val[3]) and patter
        div = 3
    else:
        print("diff pattern found")
elif (len(list_val)<4):
    if (len(list_val)%2==0):
        div = 2
    elif (len(list_val)%2==0):
        div = 3
#print(div)

varValues_dict[str(varName_dict[var])] = {}

if (div == 2):
    for x in np.arange(0,len(list_val)//2):
        #print("varValues_dict["+str(varName_dict[var])+"]["+str(list_val[x*2]
        varValues_dict[str(varName_dict[var])][list_val[x*2]] = str(list_val[(
```

```
            elif (div == 3):
                for x in np.arange(0,len(list_val)//3):
                    #print("varValues_dict["+str(varName_dict[var])+"]["+str(list_val[x*3])
                    varValues_dict[str(varName_dict[var])][list_val[x*3]] = str(list_val[(x
        return varValues_dict
```

In [4]:
```
#fill dicts
varName_dict = {}
varName_dict = get_variable_names()

varValues_dict = {}
varValues_dict = get_all_var_values(varName_dict)
```

In [5]:
```
def fill_csv_with_var_values(varValues_dict,fileName,readLinesCount=None):
    #entireListDict has values instead of keys
    entireListDict = []

    #csv file to read
    csvfile = open(fileName, 'r')

    #fieldnames = ("caseId","sctCite")
    reader = csv.DictReader( csvfile)

    count = 0
    for row in reader:
        entireListDict.append(row)
        for column in row:
            if column in varValues_dict.keys():
                if (row[column] in varValues_dict[column].keys()):
                    entireListDict[count][column] = str(varValues_dict[column][row[colu
                else:
                    entireListDict[count][column] = str(row[column])
            else:
                entireListDict[count][column] = str(row[column])
        count += 1

        #how many lines to read
        if readLinesCount is not None:
            if count >= 1:
                break
    return entireListDict
```

In [6]:
```
def write_csv_with_values(entireListDict,outputFileName):
    #write the output replacing keys with values
    with open(outputFileName,'w') as f:
        wr = csv.writer(f)
        for rowNum in np.arange(len(entireListDict)):
            if rowNum == 0:
```

```
                    wr.writerow([key for key,val in entireListDict[rowNum].items()])
                wr.writerow([val for key,val in entireListDict[rowNum].items()])
            print("File "+str(outputFileName)+" is generated!")
```

In [7]:
```
##for reading csv as dict format
# with open('sample.csv') as csv_file:
#     csv_reader = csv.reader(csv_file, delimiter=',')
#     line_count = 0
#     for row in csv_reader:
#         if line_count == 0:
#             print(f'Column names are {", ".join(row)}')
#             line_count += 1
#             #break
#         else:
#             print(row)
#             line_count += 1
#     print(f'Processed {line_count} lines.')
```

In [8]:
```
#test dicts
for var in varName_dict.keys():
    print(str(var)+":"+str(varName_dict[var]))

x = 'chief'
for y in varValues_dict[x]:
    print(str(y)+"\t"+str(varValues_dict[x][y]))

#get list of dicts aka csv file
entireListDict = fill_csv_with_var_values(varValues_dict,"JusticeData.csv")
print(len(entireListDict))
#write_csv_with_values(entireListDict,"sample_output_2.csv")
```

```
SCDB Case ID:caseId
SCDB Docket ID:docketId
SCDB Issues ID:caseIssuesId
SCDB Vote ID:voteId
U.S. Reporter Citation:usCite
Supreme Court Citation:sctCite
Lawyers Edition Citation:ledCite
LEXIS Citation:lexisCite
Docket Number:docket
Case Name:caseName
Petitioner:petitioner
Petitioner State:petitionerState
Respondent:respondent
Respondent State:respondentState
Manner in which the Court takes Jurisdiction:jurisdiction
Administrative Action Preceeding Litigation:adminAction
Administrative Action Preceeding Litigation State:adminActionState
```

```
Three-Judge District Court:threeJudgeFdc
Origin of Case:caseOrigin
Origin of Case State:caseOriginState
Source of Case:caseSource
Source of Case State:caseSourceState
Lower Court Disagreement:lcDisagreement
Reason for Granting Cert:certReason
Lower Court Disposition:lcDisposition
Lower Court Disposition Direction:lcDispositionDirection
Date of Decision:dateDecision
Term of Court:term
Natural Court:naturalCourt
OnChief Justice:chief
Date of Oral Argument:dateArgument
Date of Reargument:dateRearg
Issue:issue
Issue Area:issueArea
Decision Direction:decisionDirection
Decision Direction Dissent:decisionDirectionDissent
Authority for Decision 1:authorityDecision1
Authority for Decision 2:authorityDecision2
Legal Provisions Considered by the Court:lawType
Legal Provision Supplement:lawSupp
Legal Provision Minor Supplement:lawMinor
Decision Type:decisionType
Declaration of Unconstitutionality:declarationUncon
Disposition of Case:caseDisposition
Unusual Disposition:caseDispositionUnusual
Winning Party:partyWinning
Formal Alteration of Precedent:precedentAlteration
Vote Not Clearly Specified:voteUnclear
Majority Opinion Writer:majOpinWriter
Majority Opinion Assigner:majOpinAssigner
Split Vote:splitVote
Majority Votes:majVotes
Minority Votes:minVotes
Justice ID:justice
Justice Name:justiceName
The Vote in the Case:vote
Opinion:opinion
Direction of the Individual Justice\'s Votes:direction
Majority and Minority Voting by Justice:majority
First Agreement:firstAgreement
Second Agreement:secondAgreement
1        Jay
2        Rutledge
3        Ellsworth
4        Marshall
```

```
5         Taney
6         Chase
7         Waite
8         Fuller
9         White
10         Taft
11         Hughes
12         Stone
13         Vinson
14         Warren
15         Burger
16         Rehnquist
17         Roberts
119838
```

```
In [9]: df = pd.DataFrame(entireListDict)

In [10]: df['firstAgreement'].value_counts(normalize=True)

Out[10]:                0.872495
         WJBrennan      0.011040
         JPStevens      0.010706
         HLBlack        0.010297
         0              0.008203
         BRWhite        0.007093
         WHRehnquist    0.005975
         AScalia        0.005741
         HABlackmun     0.005449
         TMarshall      0.005441
         JHarlan2       0.004673
         AMKennedy      0.004548
         LFPowell       0.004389
         PStewart       0.004281
         SGBreyer       0.004131
         SDOConnor      0.004039
         FFrankfurter   0.003655
         RBGinsburg     0.003355
         CThomas        0.003329
         DHSouter       0.003121
         TCClark        0.002153
         WODouglas      0.001769
         JGRoberts      0.001369
         SAAlito        0.001285
         WEBurger       0.001210
         SFReed         0.001127
         RHJackson      0.001101
         SSotomayor     0.001026
```

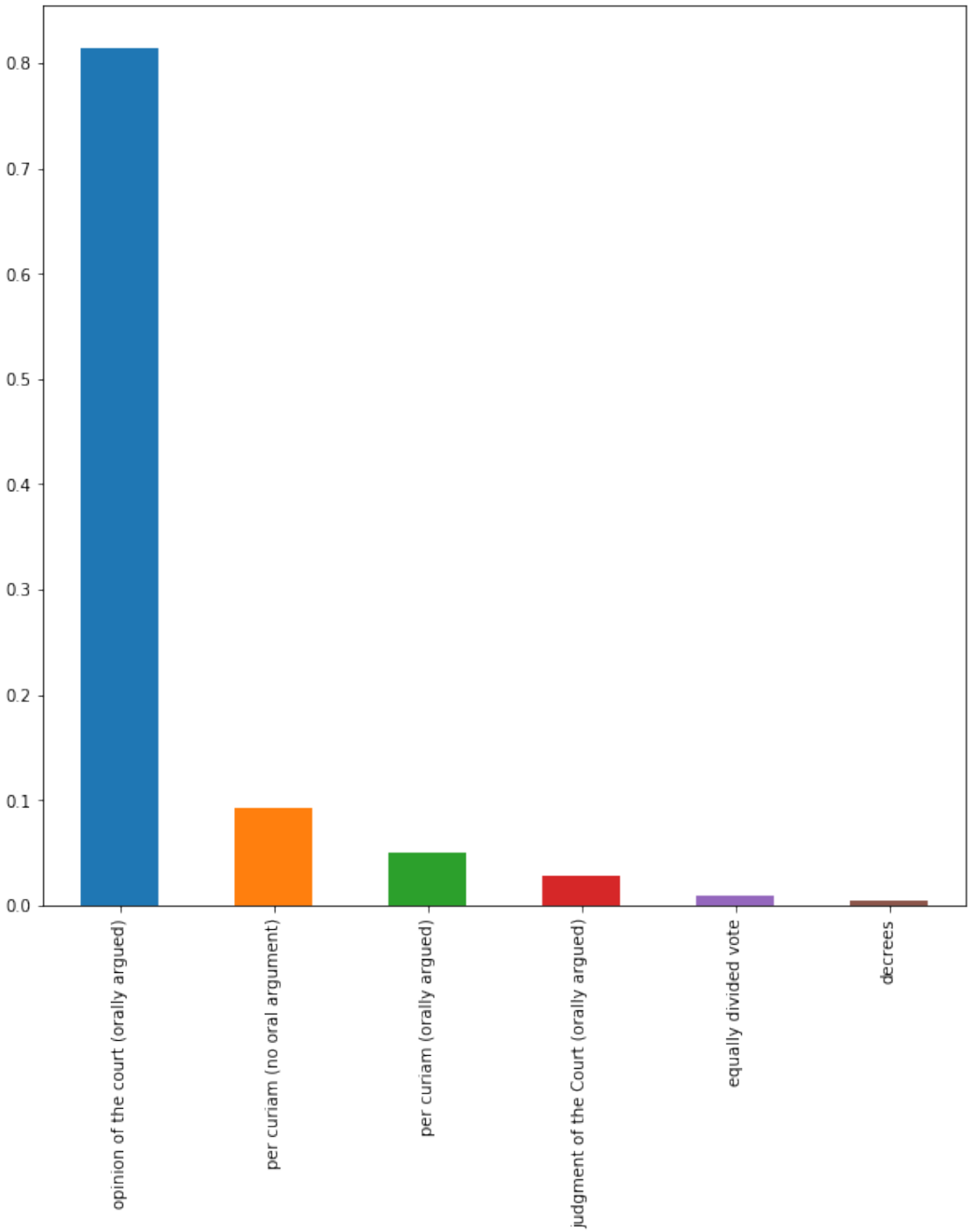
```
HHBurton        0.001018
EWarren         0.000851
WBRutledge      0.000826
AJGoldberg      0.000726
CEWhittaker     0.000676
AFortas         0.000651
SMinton         0.000651
FMurphy         0.000618
EKagan          0.000551
FMVinson        0.000317
NMGorsuch       0.000117
Name: firstAgreement, dtype: float64
```

In [16]:
```
# for var in entireListDict[0].keys():
#     print("make_graphs_var.append("+str(var)+")")
make_graphs_var = []

make_graphs_var.append('decisionType')
make_graphs_var.append('term')
make_graphs_var.append('naturalCourt')
make_graphs_var.append('chief')
make_graphs_var.append('docket')

make_graphs_var.append('petitioner')
make_graphs_var.append('petitionerState')
make_graphs_var.append('respondent')
make_graphs_var.append('respondentState')
make_graphs_var.append('jurisdiction')
make_graphs_var.append('adminAction')
make_graphs_var.append('adminActionState')
make_graphs_var.append('threeJudgeFdc')
make_graphs_var.append('caseOrigin')
make_graphs_var.append('caseOriginState')
make_graphs_var.append('caseSource')
make_graphs_var.append('caseSourceState')
make_graphs_var.append('lcDisagreement')
make_graphs_var.append('certReason')
make_graphs_var.append('lcDisposition')
make_graphs_var.append('lcDispositionDirection')
make_graphs_var.append('declarationUncon')
make_graphs_var.append('caseDisposition')
make_graphs_var.append('caseDispositionUnusual')
make_graphs_var.append('partyWinning')
make_graphs_var.append('precedentAlteration')
make_graphs_var.append('voteUnclear')
make_graphs_var.append('issue')
make_graphs_var.append('issueArea')
make_graphs_var.append('decisionDirection')
```

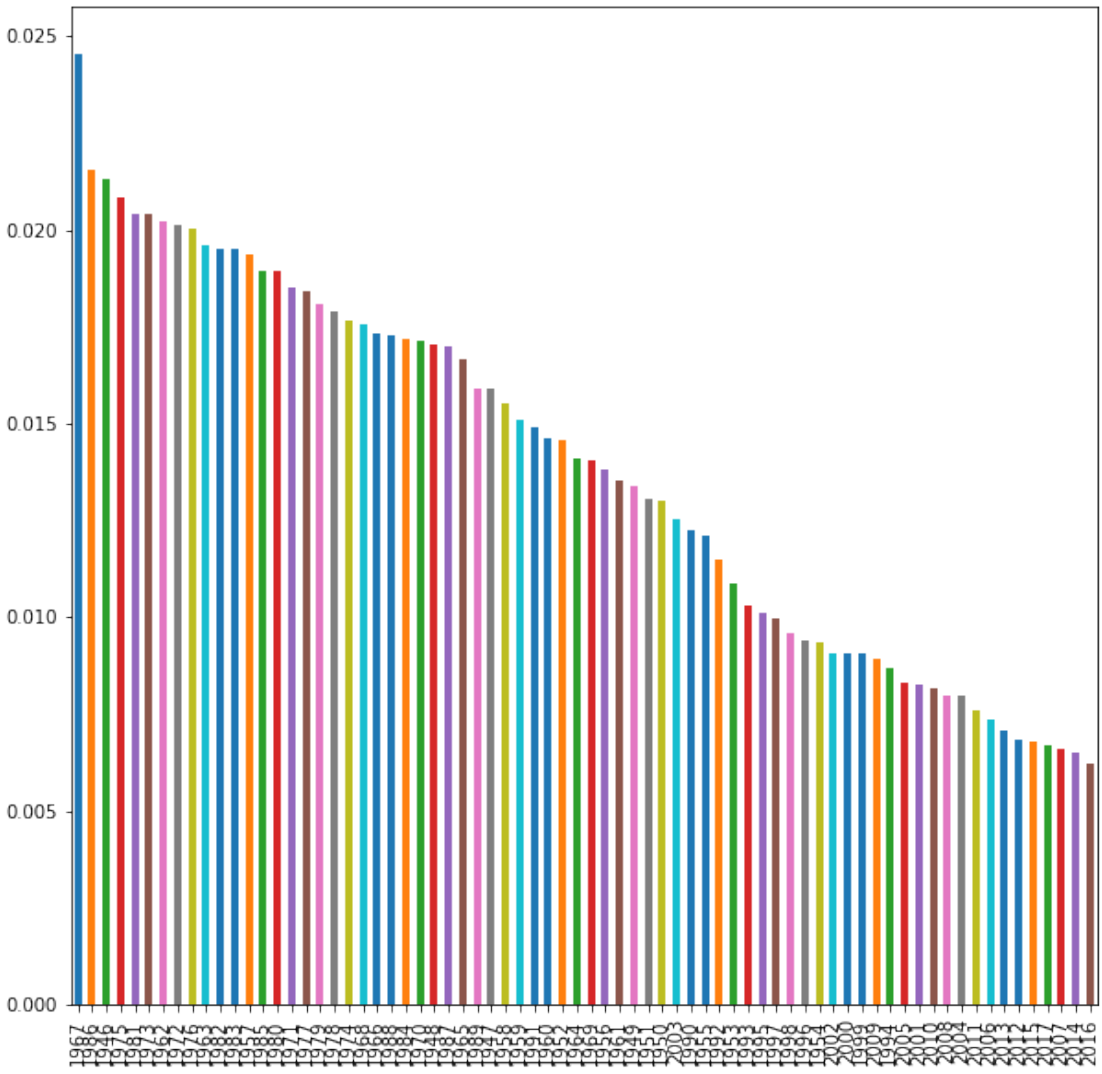
```
make_graphs_var.append('decisionDirectionDissent')
make_graphs_var.append('authorityDecision1')
make_graphs_var.append('authorityDecision2')
make_graphs_var.append('lawType')
make_graphs_var.append('lawSupp')

make_graphs_var.append('majOpinWriter')
make_graphs_var.append('majOpinAssigner')
make_graphs_var.append('splitVote')
make_graphs_var.append('majVotes')
make_graphs_var.append('minVotes')
make_graphs_var.append('justice')
make_graphs_var.append('justiceName')
make_graphs_var.append('vote')
make_graphs_var.append('opinion')
make_graphs_var.append('direction')
make_graphs_var.append('majority')
make_graphs_var.append('firstAgreement')
make_graphs_var.append('secondAgreement')
```

In [21]:
```
for var in make_graphs_var:
    print(str(var))
    plot = df[var].value_counts(normalize=True).plot(kind="bar", figsize=(10, 10))
    plt.show()
```

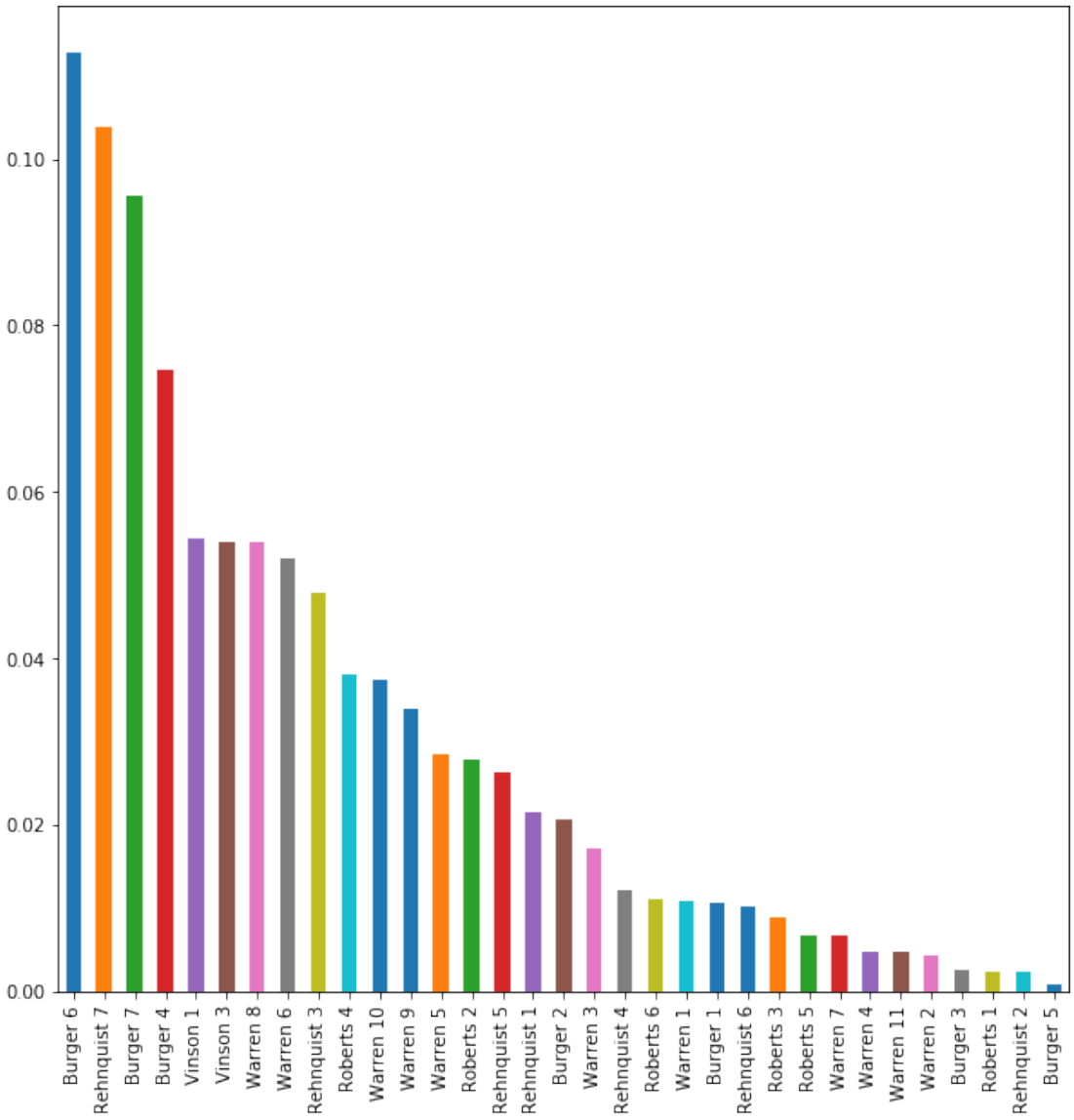
```
decisionType
```


0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
opinion of the court (orally argued)
per curiam (no oral argument)
per curiam (orally argued)
judgment of the Court (orally argued)
equally divided vote
decrees

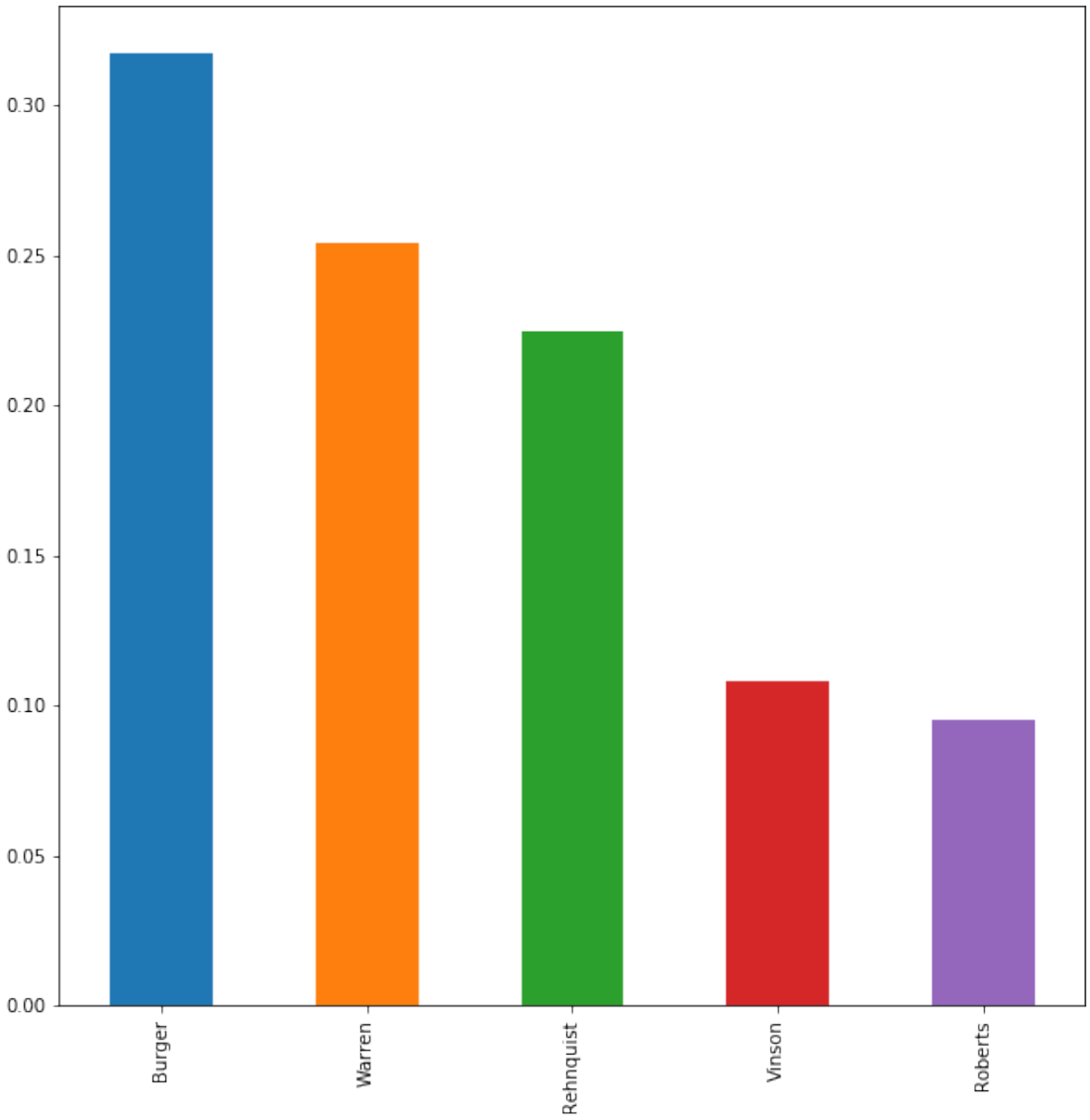
term



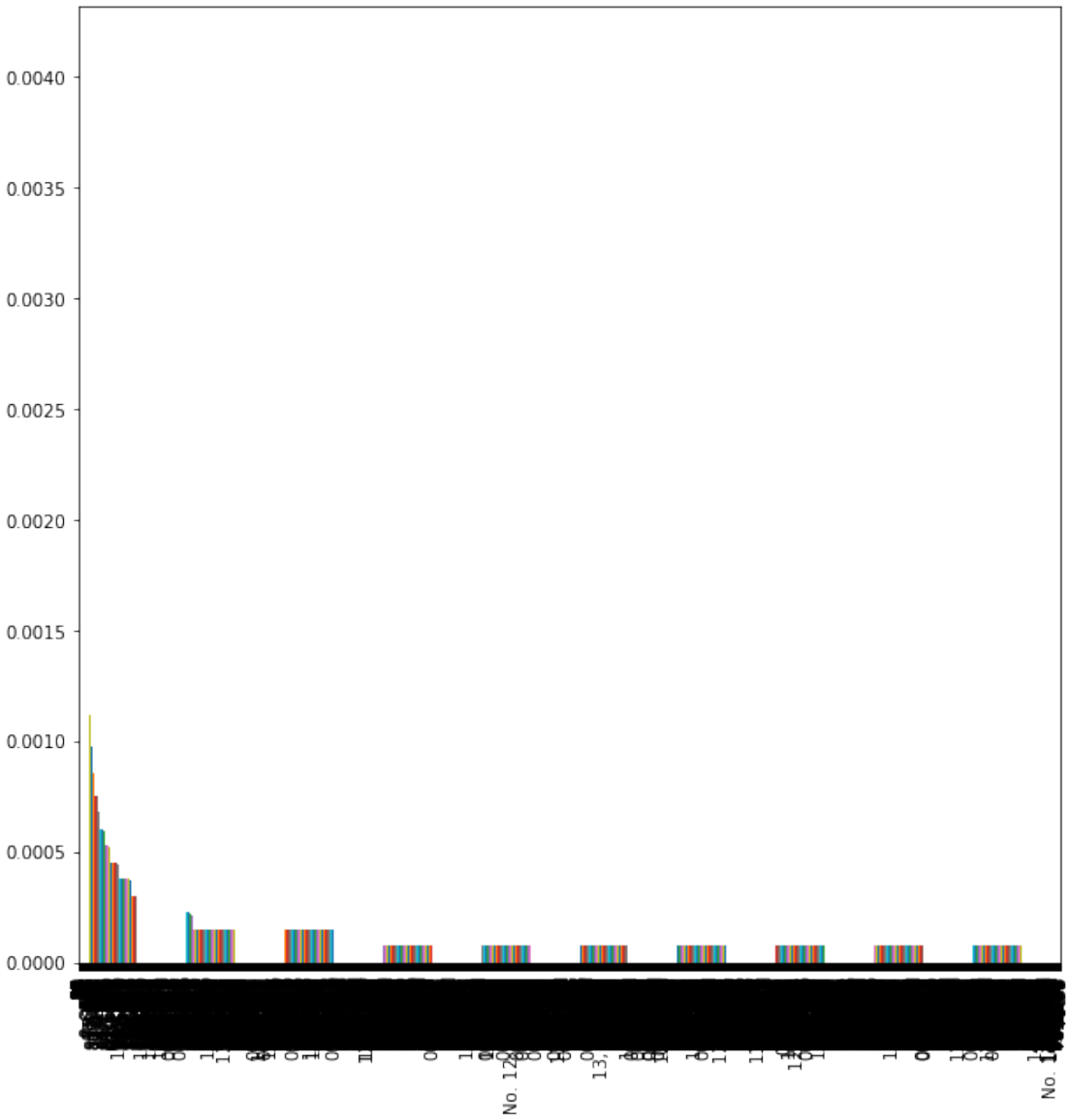
naturalCourt


0.10
0.08
0.06
0.04
0.02
0.00
Burger 6
Rehnquist 7
Burger 7
Burger 4
Vinson 1
Vinson 3
Warren 8
Warren 6
Rehnquist 3
Roberts 4
Warren 10
Warren 9
Warren 5
Roberts 2
Rehnquist 5
Rehnquist 1
Burger 2
Warren 3
Rehnquist 4
Roberts 6
Warren 1
Burger 1
Rehnquist 6
Roberts 3
Roberts 5
Warren 7
Warren 4
Warren 11
Warren 2
Burger 3
Roberts 1
Rehnquist 2
Burger 5

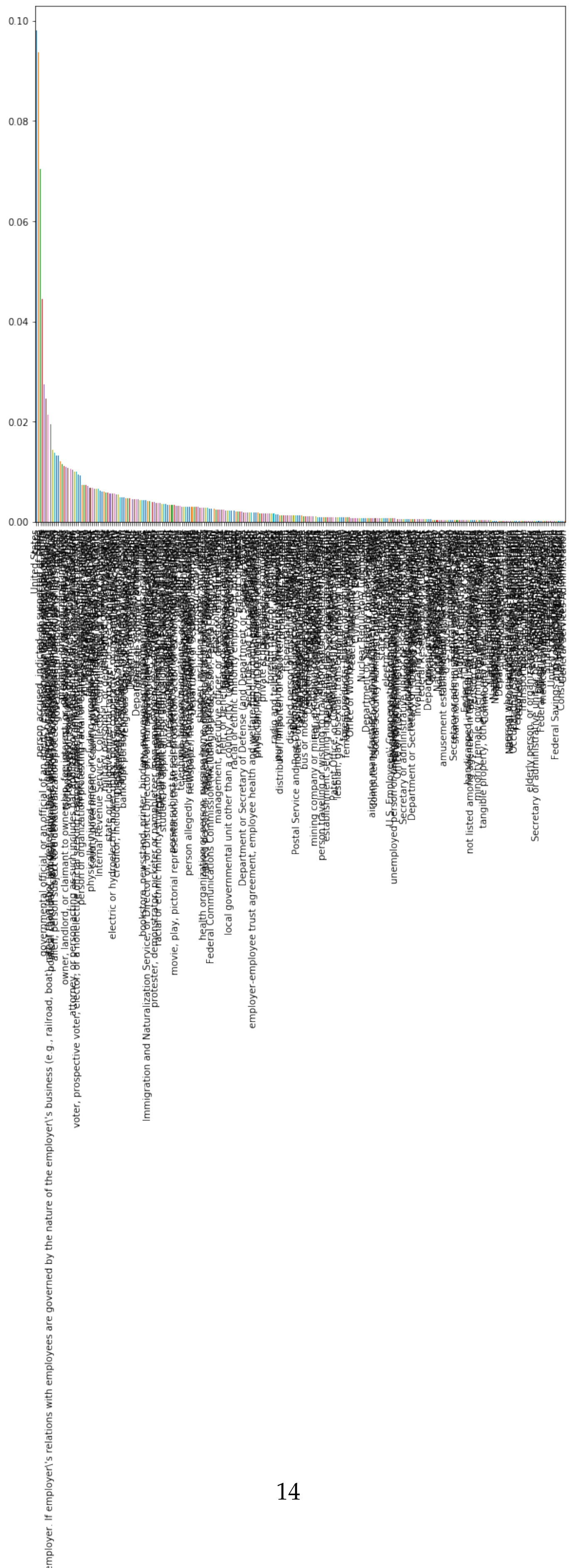
```
chief
```


0.30
0.25
0.20
0.15
0.10
0.05
0.00
Burger
Warren
Rehnquist
Vinson
Roberts

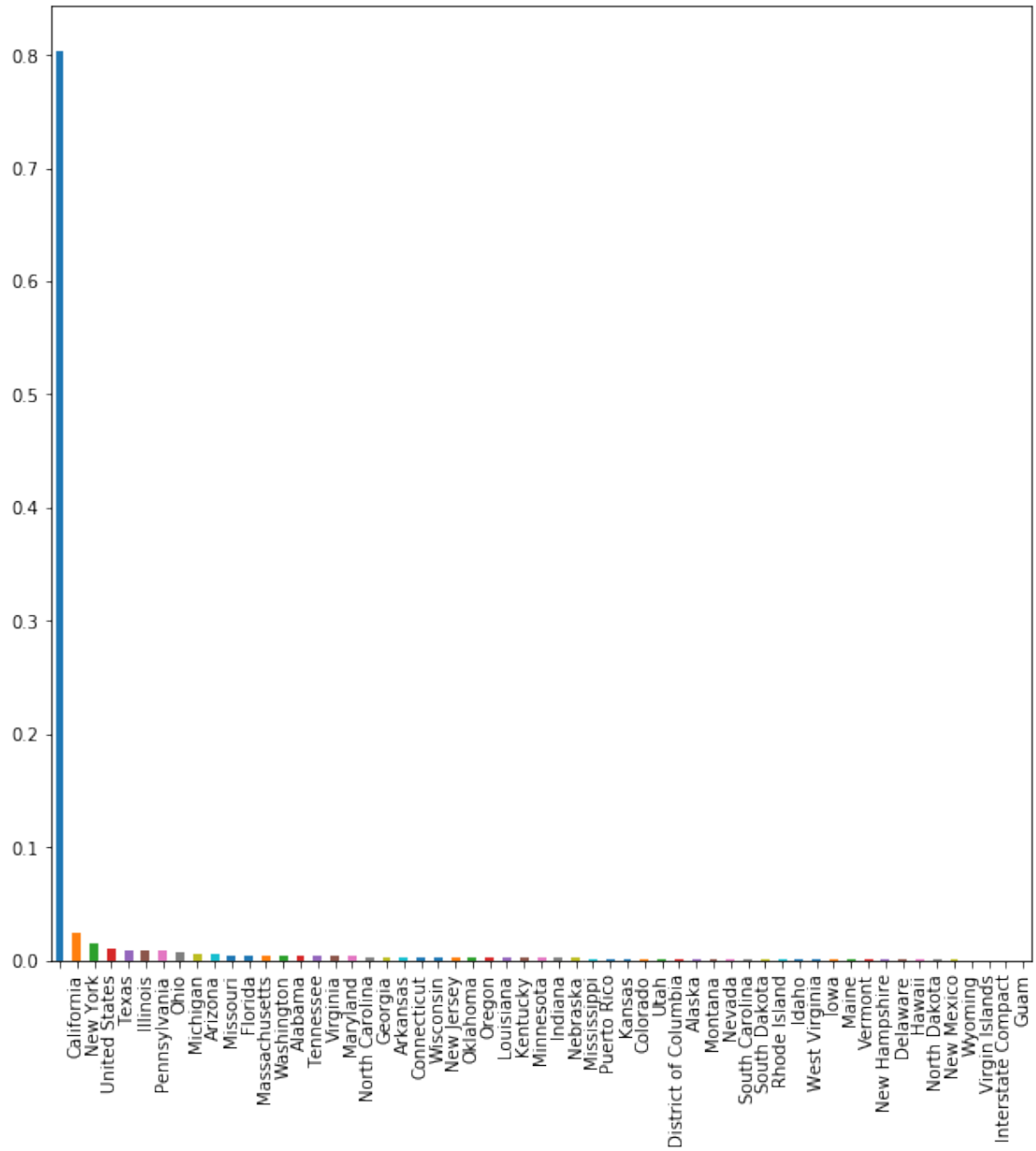
docket



petitioner


0.10
0.08
0.06
0.04
0.02
0.00
United States
Internal Revenue Service
Immigration and Naturalization Service
Federal Communications Commission
local governmental unit other than a county, city, town, township, village, or municipality
employer-employee trust agreement, employee health and welfare fund, or multi-employer pension plan
Postal Service and Post Office
Nuclear Regulatory Commission
Federal Savings

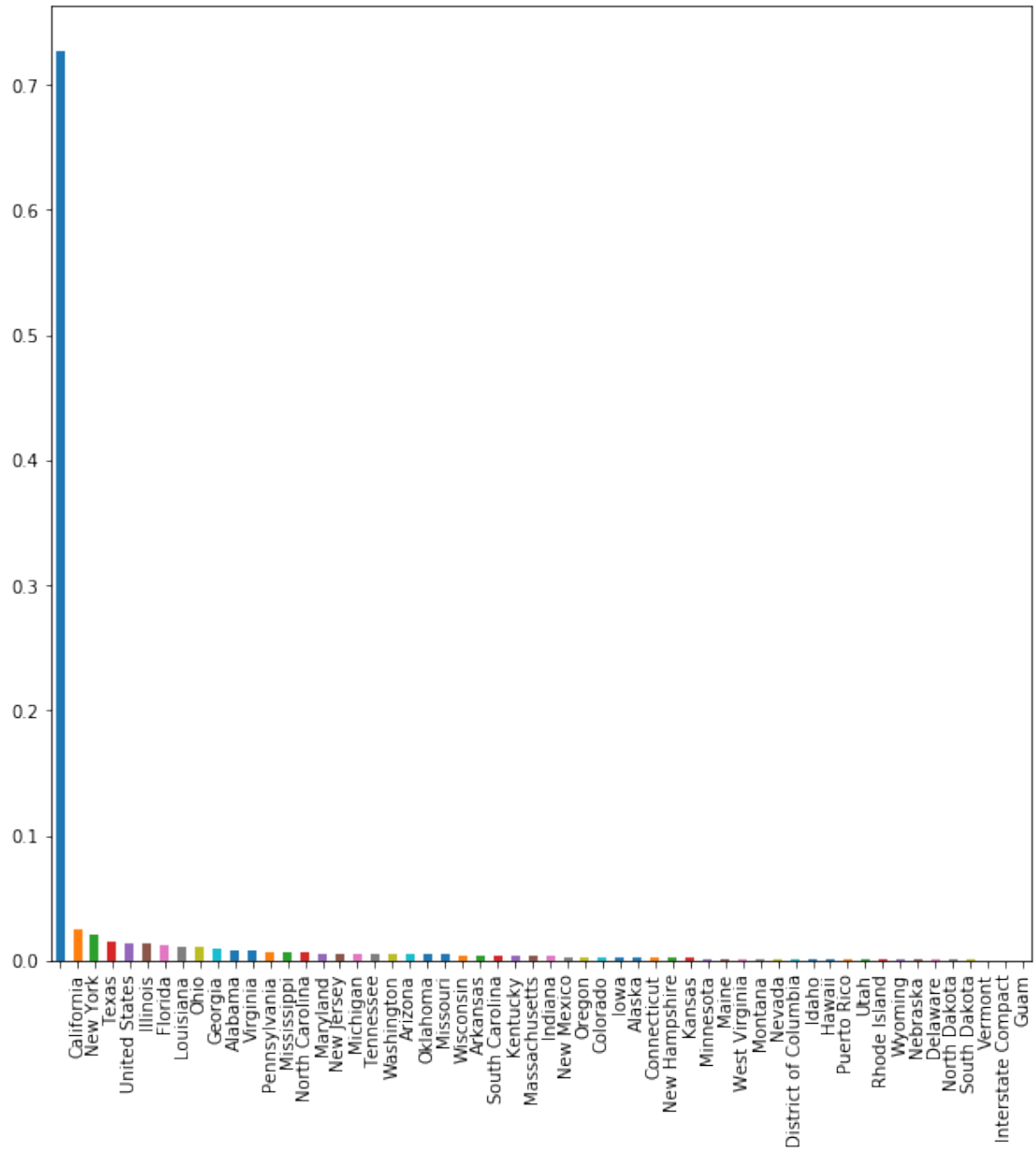
petitionerState



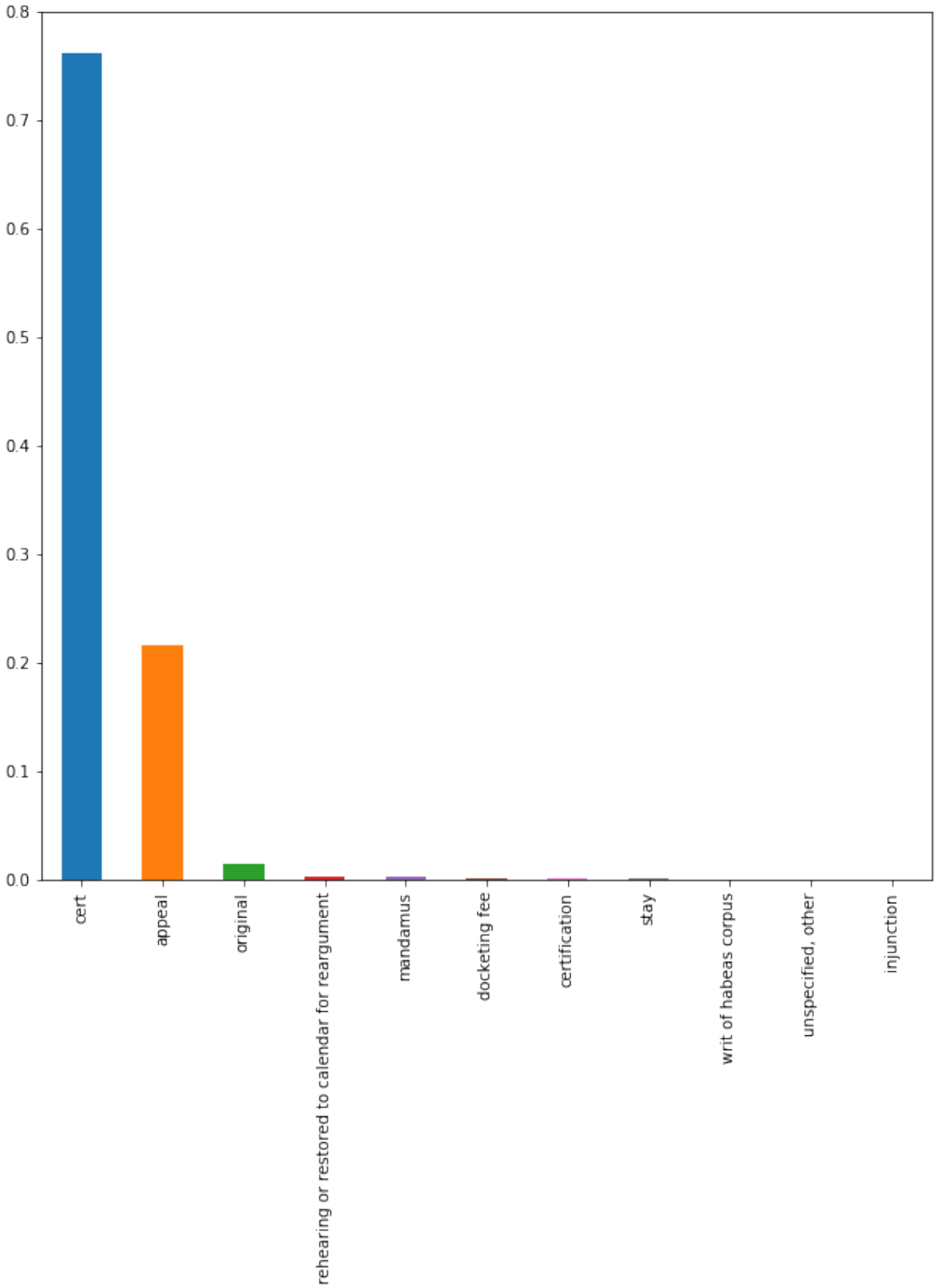
respondent

0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
employer. If employer\'s relations with employees are governed by the nature of the employer\'s business (e.g., railroad, boat),
voter, prospective voter, elector
Immigration and Naturalization Service, or Director of, or District Director of,
attorney, or
Federal Communications Commission
movie, play, pictorial
employer-employee trust agreement
racial or ethnic minority student or applicant
person traveling or
not listed among agencies in the
Customs Service or Commissioner of Customs

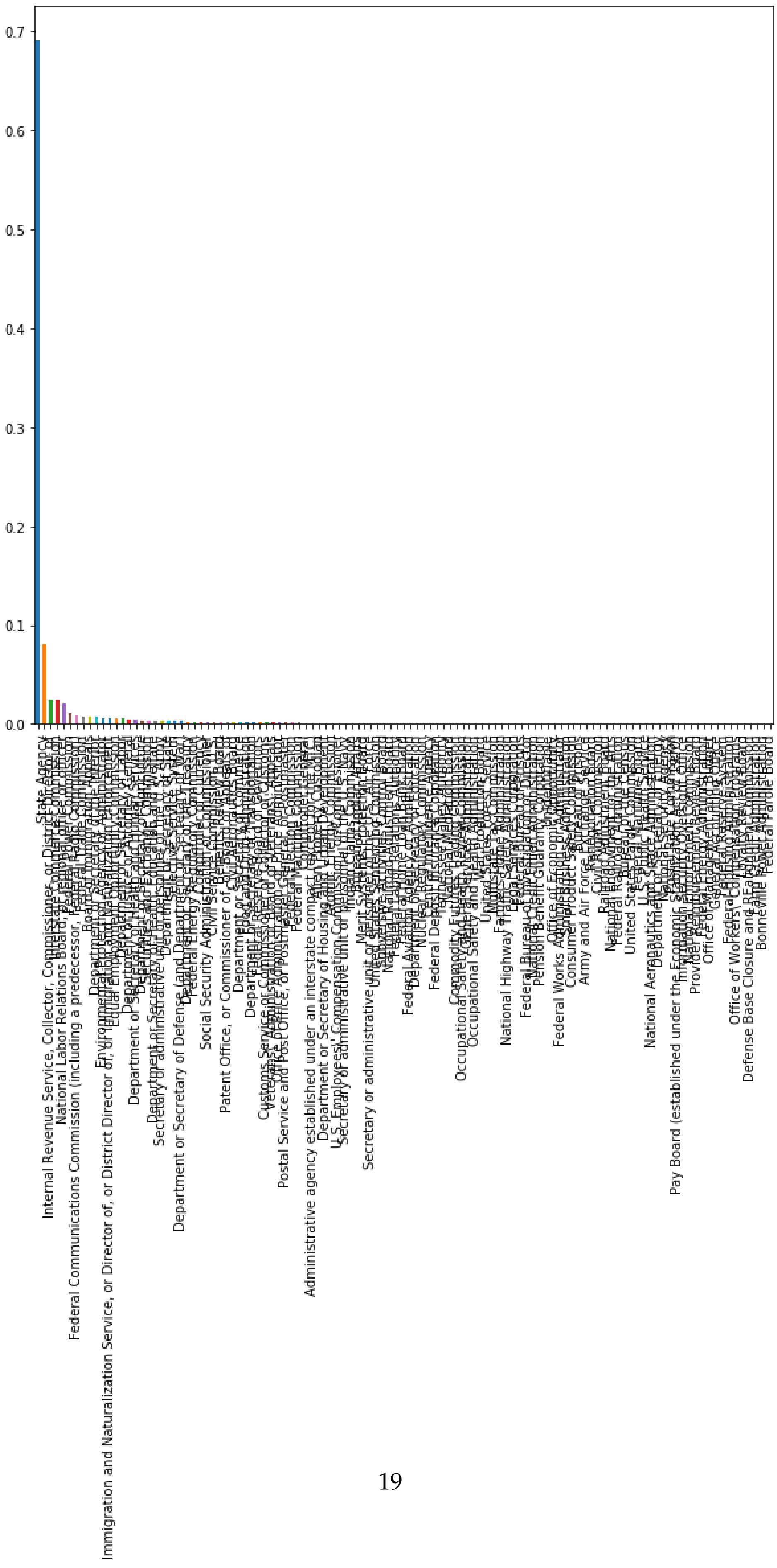
respondentState



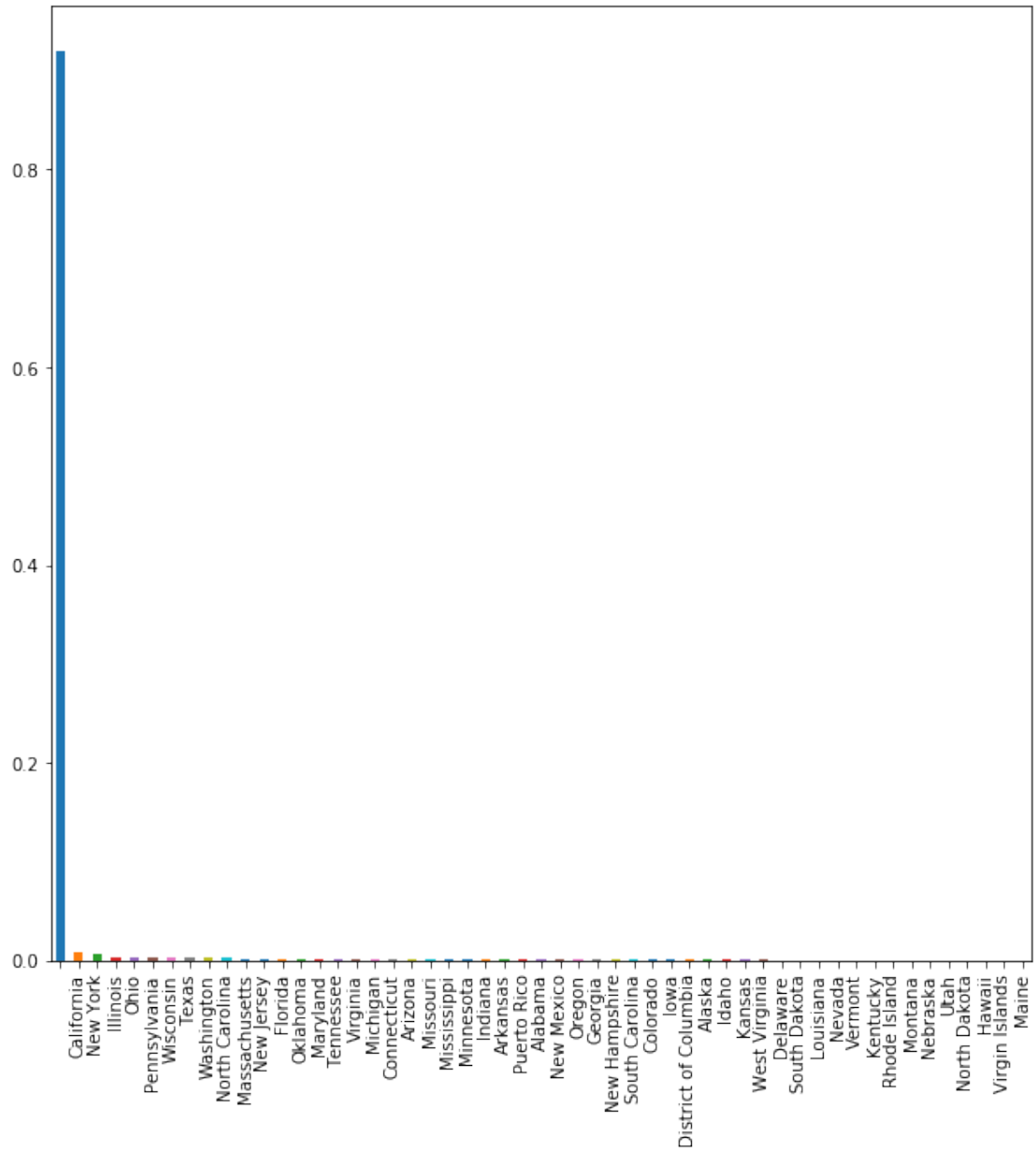
jurisdiction



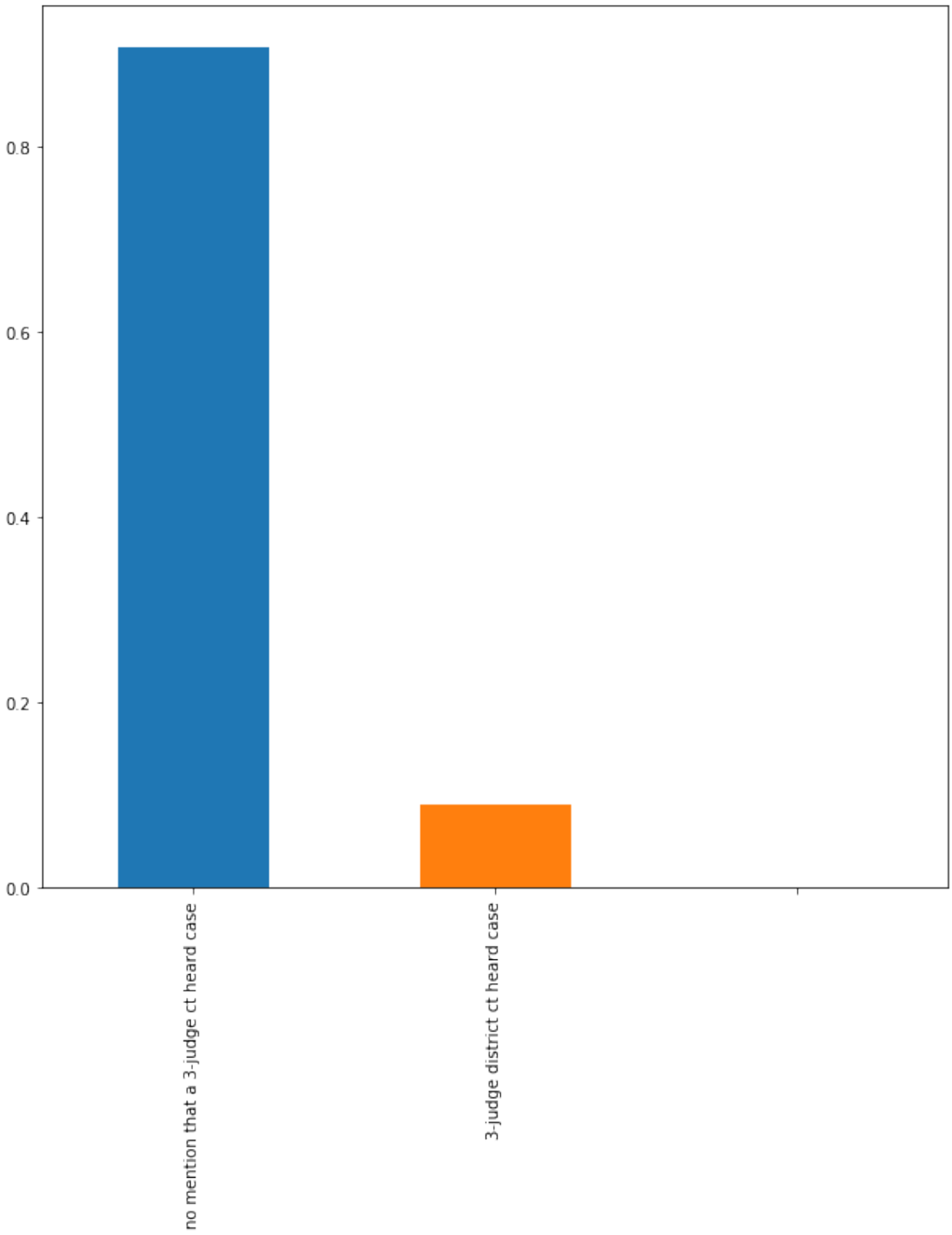
`adminAction`


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
State Agency
National Labor Relations Board
Social Security Administration
Postal Service and Post Office, or Postmaster
Administrative agency established under an interstate compact
National Highway
Army and Air Force
Federal Parole Board

adminActionState



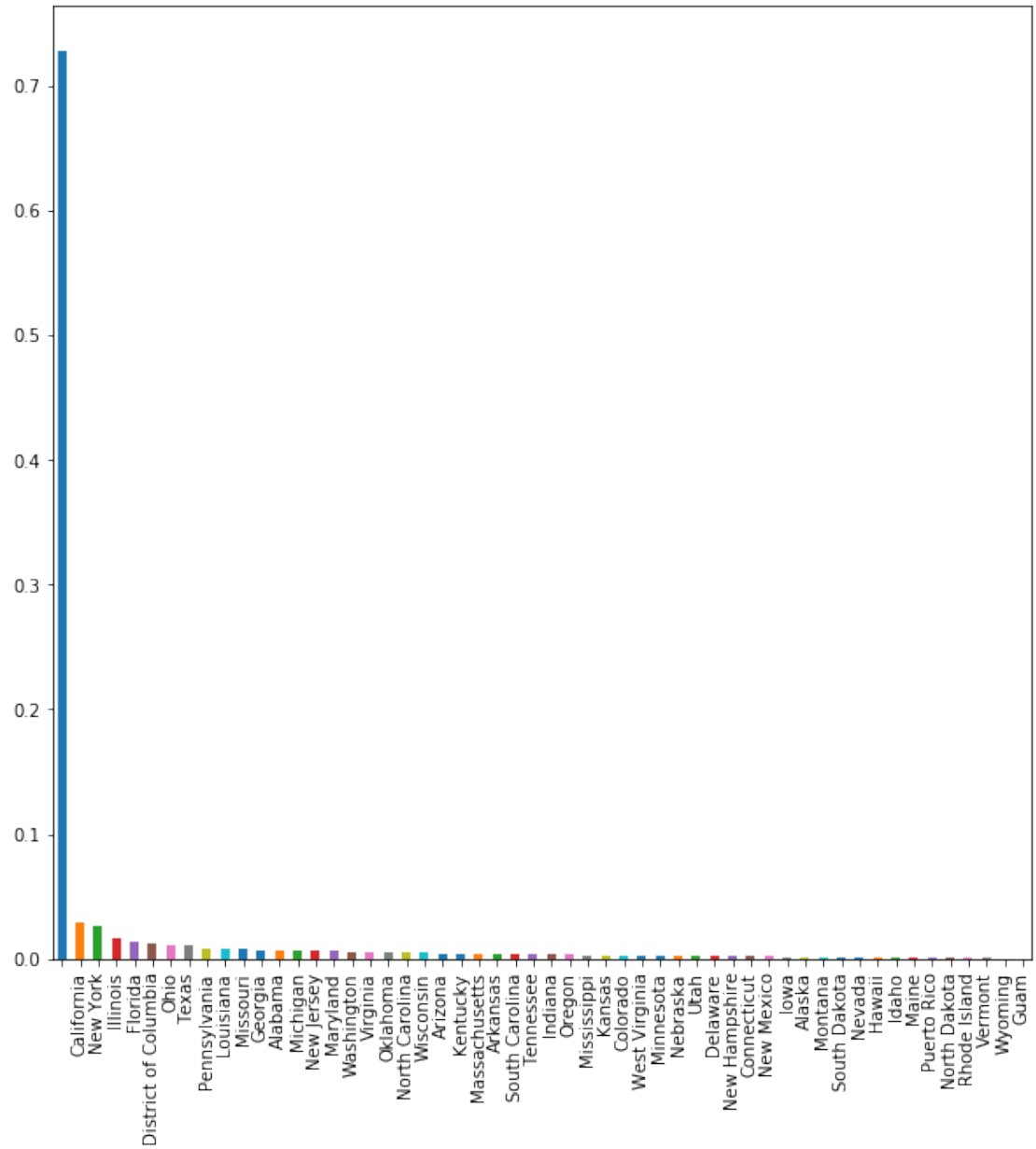
threeJudgeFdc


0.8
0.6
0.4
0.2
0.0
no mention that a 3-judge ct heard case
3-judge district ct heard case

caseOrigin

0.200
0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
U.S. Court of Appeals, District of Columbia Circuit (includes the Court of Appeals for the District of Columbia but not the District of Columbia Court of Appeals)
U.S. Court of Military Appeals, renamed as Court of Appeals for the Armed Forces
Temporary Emergency Court of Appeals
Northern Mariana Islands U.S. District Court
Territorial Appellate Court

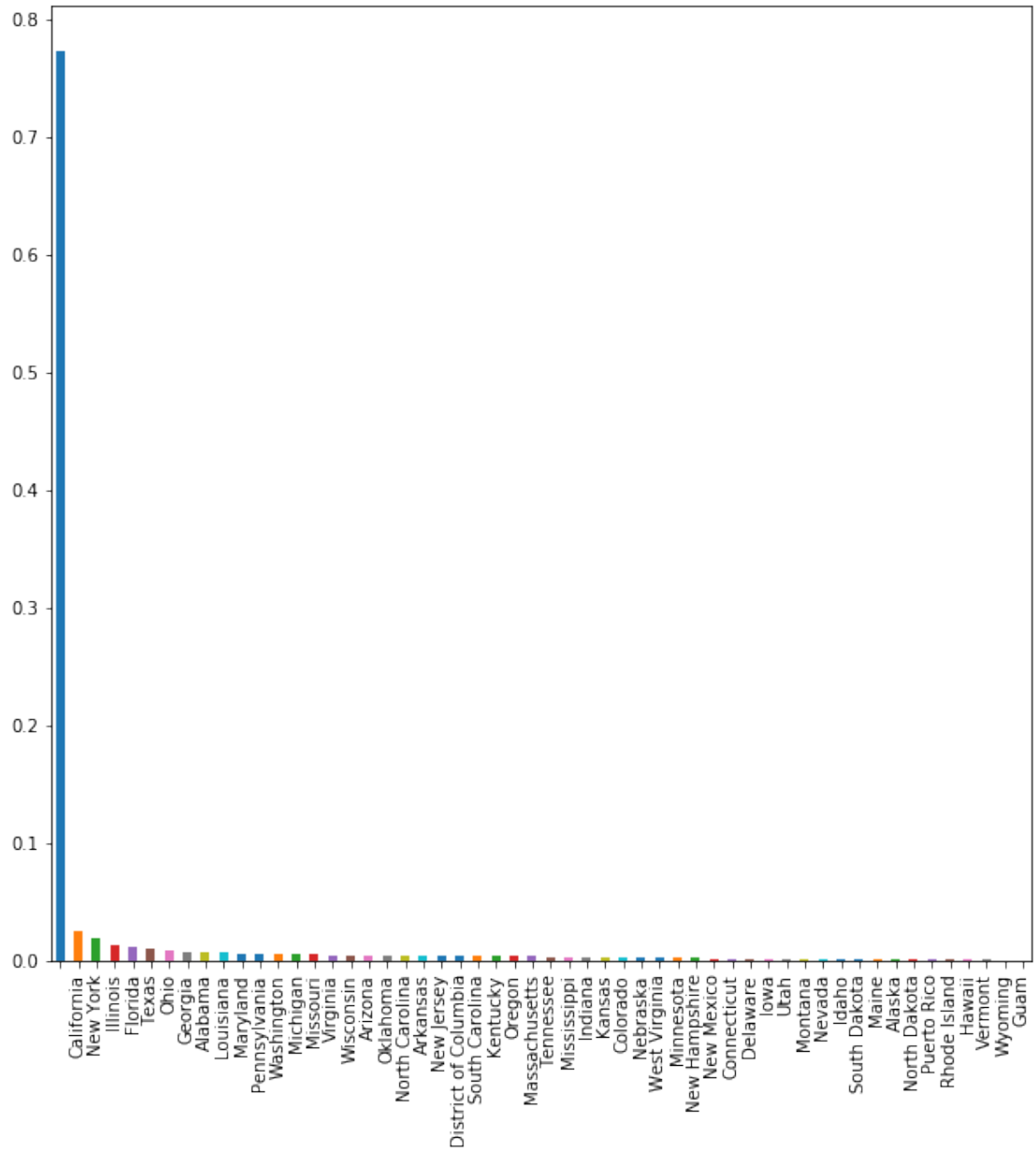
caseOriginState



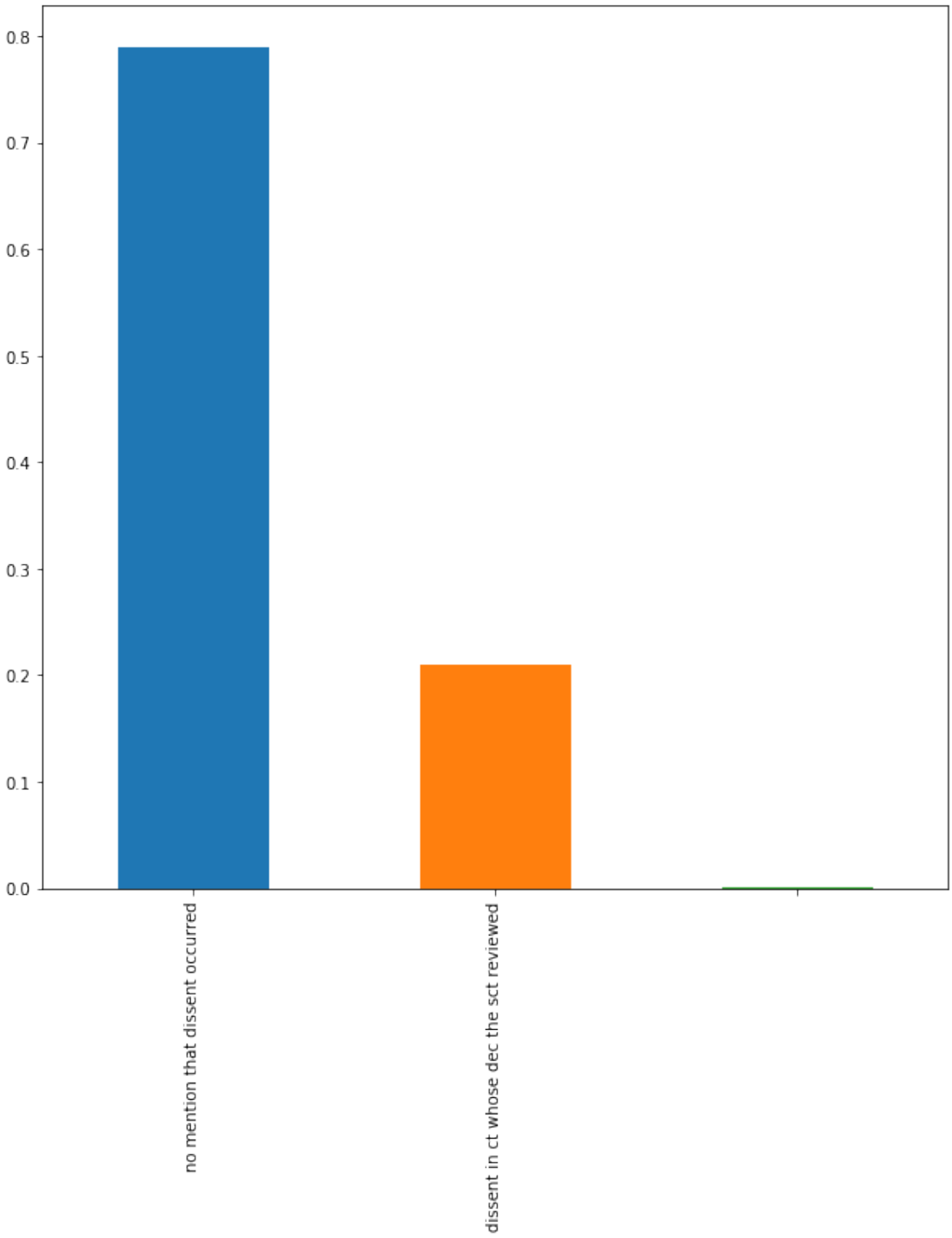
caseSource

0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
U.S. Court of Appeals, District of Columbia Circuit (includes the Court of Appeals for the District of Columbia but not the District of Columbia Court of Appeals)
U.S. Court of Military Appeals, renamed as U.S. Court of Appeals for the Armed Forces

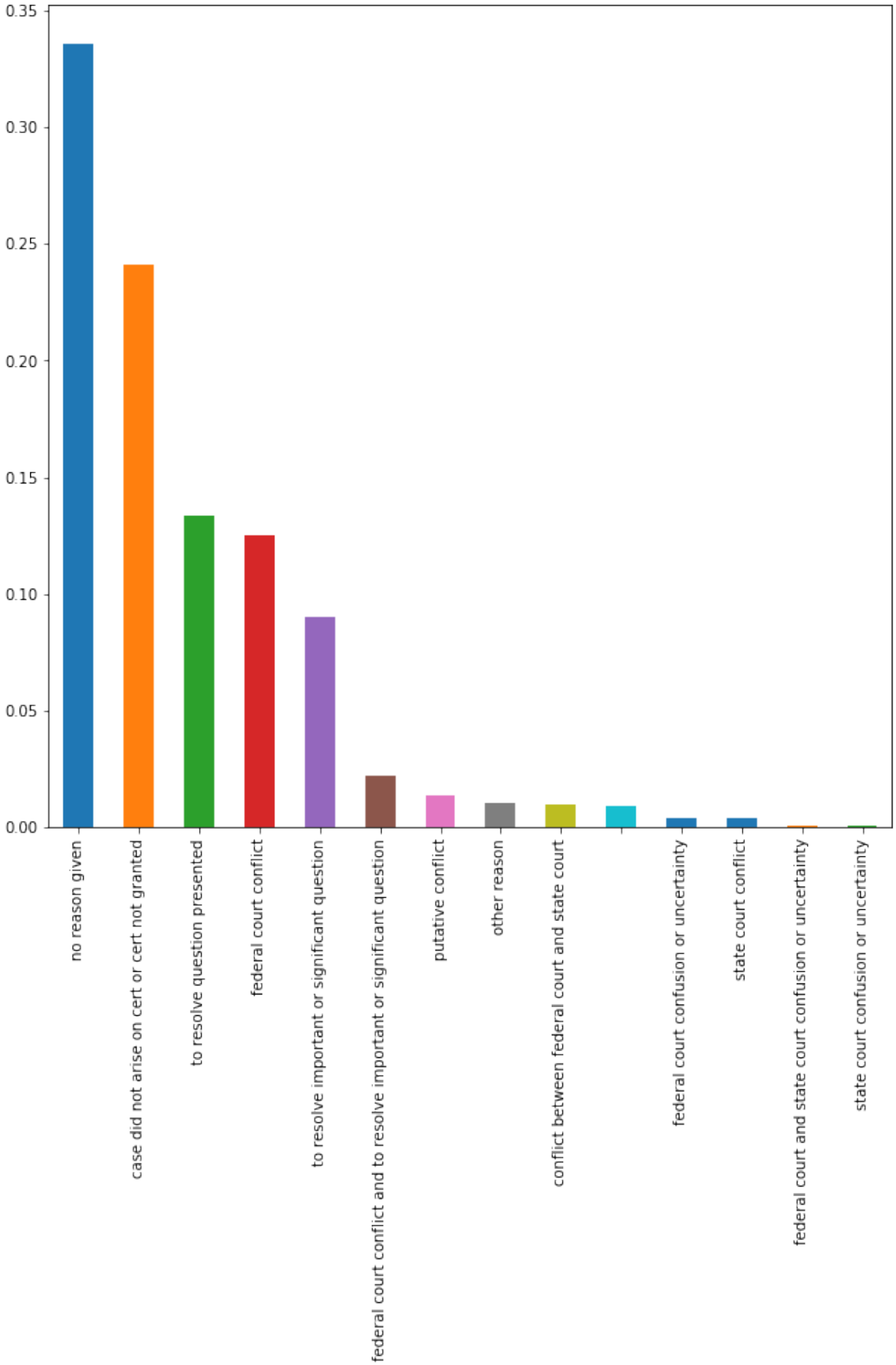
caseSourceState



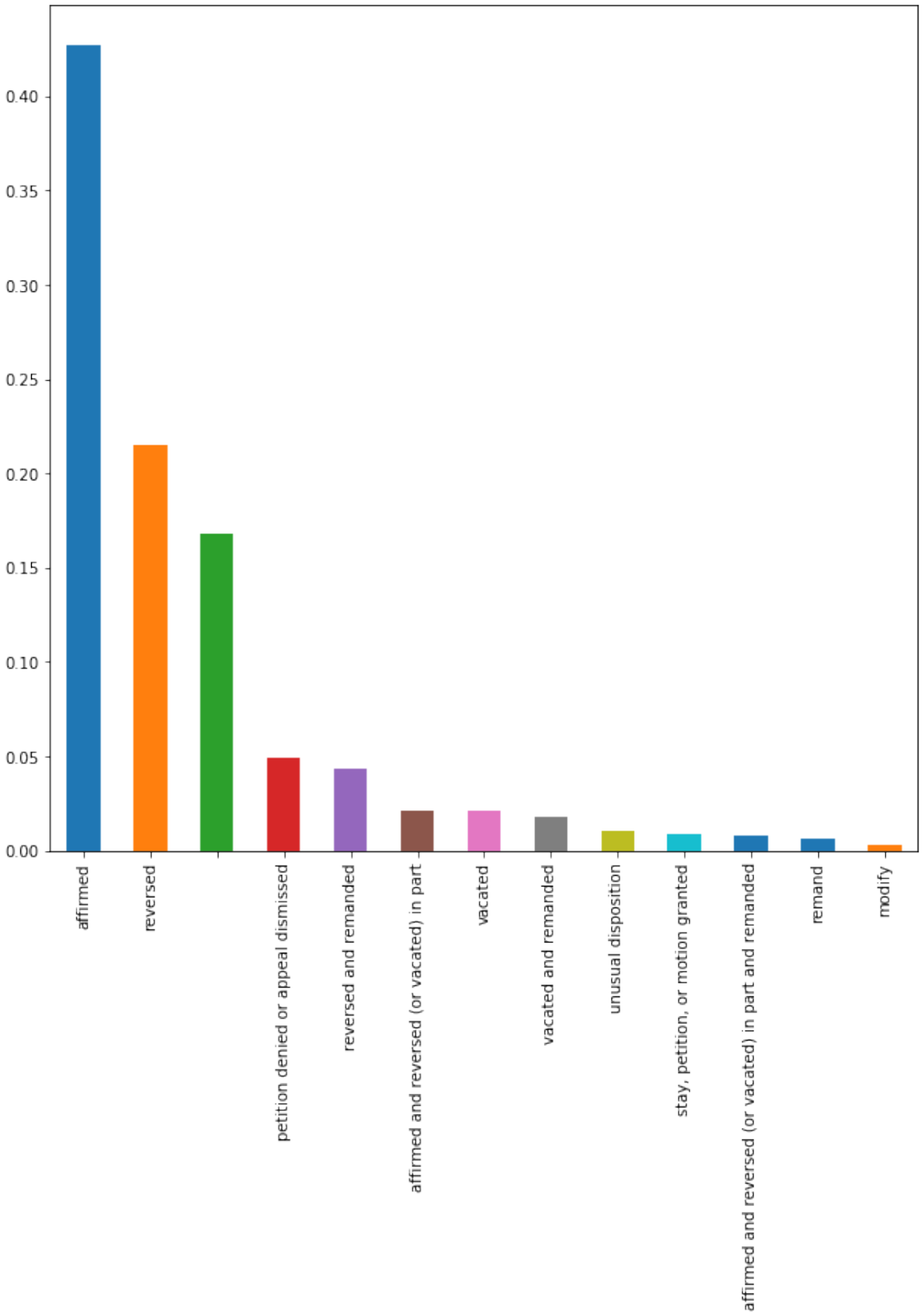
lcDisagreement


0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
no mention that dissent occurred
dissent in ct whose dec the sct reviewed

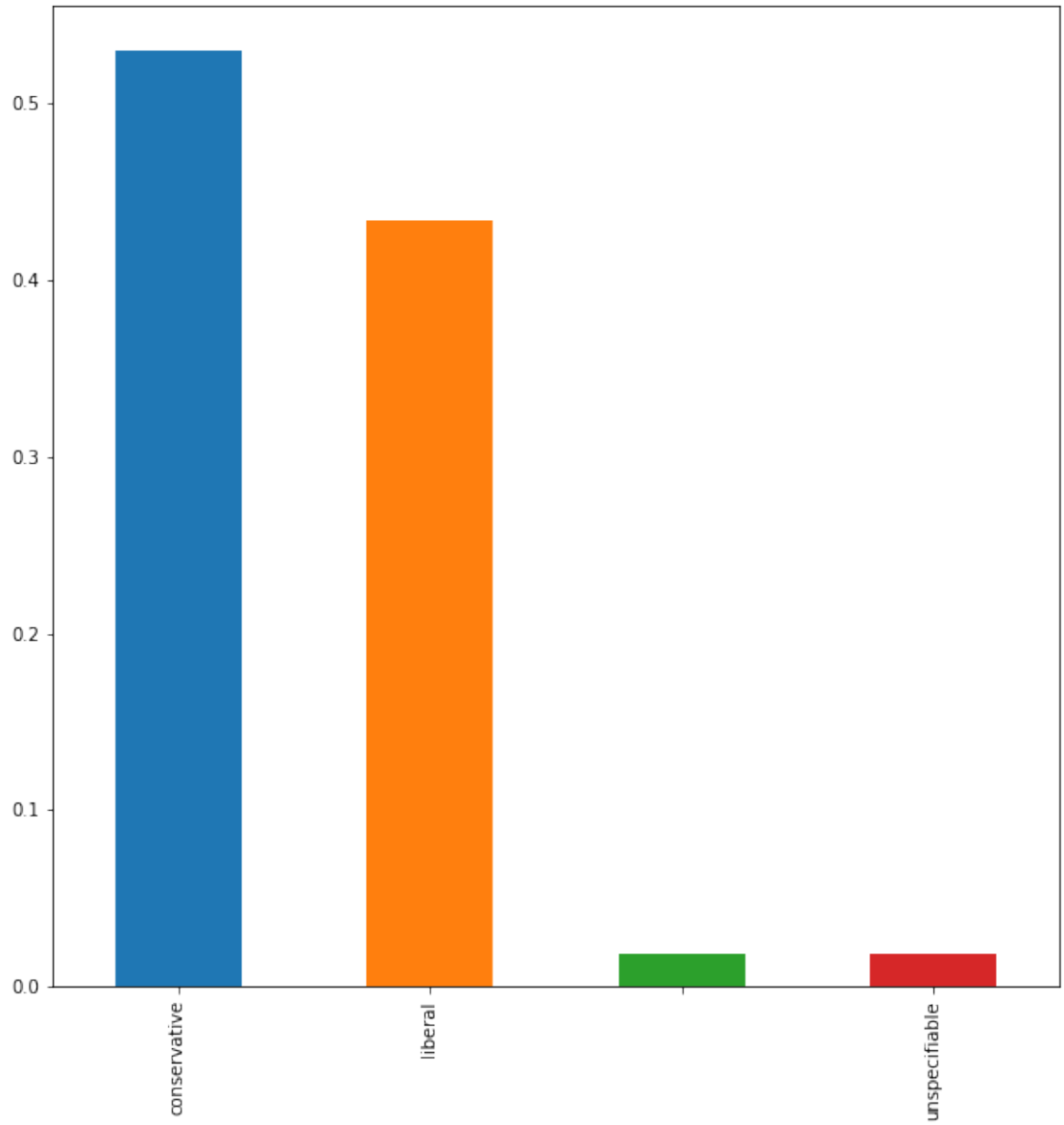
certReason


0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
no reason given
case did not arise on cert or cert not granted
to resolve question presented
federal court conflict
to resolve important or significant question
federal court conflict and to resolve important or significant question
putative conflict
other reason
conflict between federal court and state court
federal court confusion or uncertainty
state court conflict
federal court and state court confusion or uncertainty
state court confusion or uncertainty

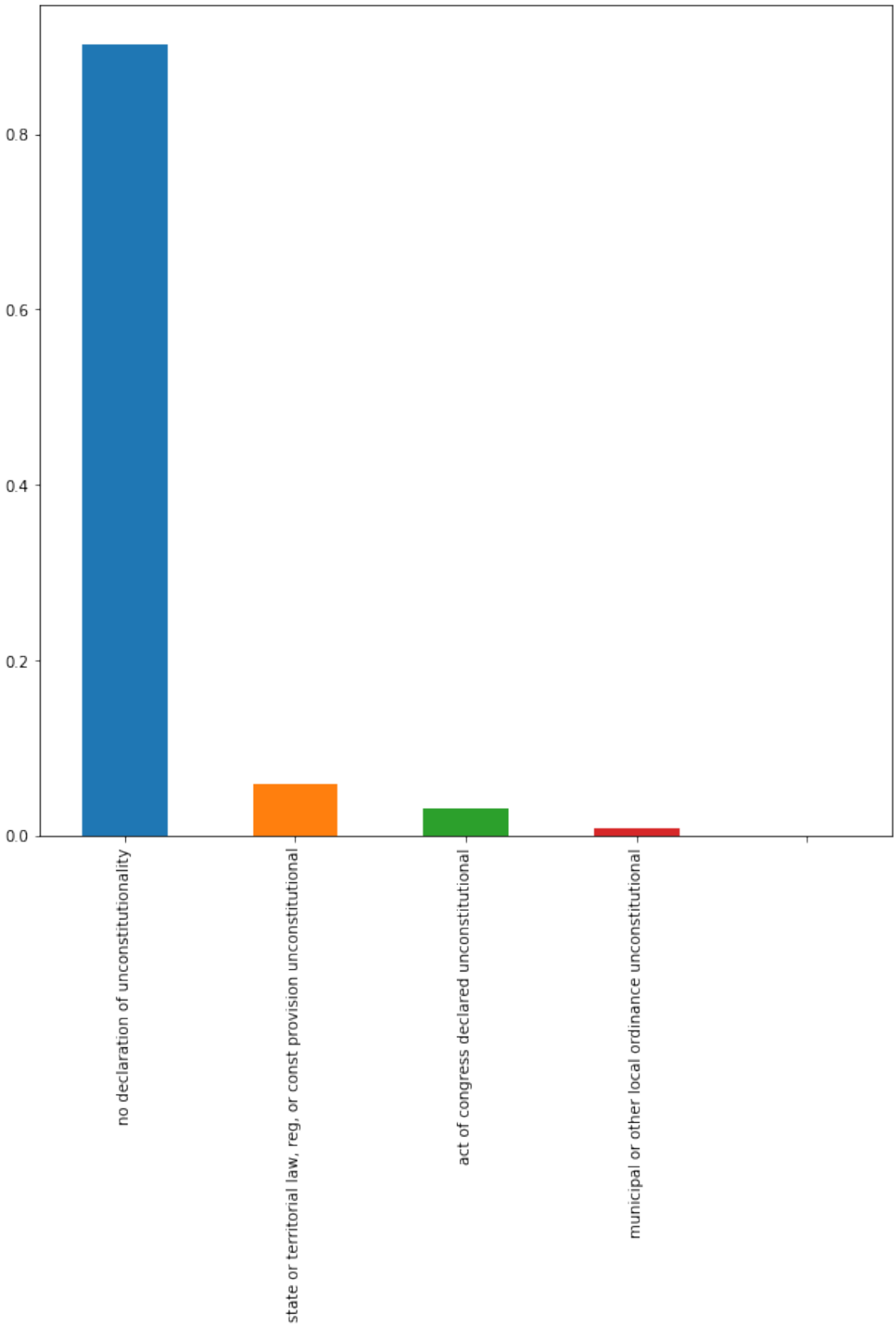
lcDisposition


0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
affirmed
reversed
petition denied or appeal dismissed
reversed and remanded
affirmed and reversed (or vacated) in part
vacated
vacated and remanded
unusual disposition
stay, petition, or motion granted
affirmed and reversed (or vacated) in part and remanded
remand
modify

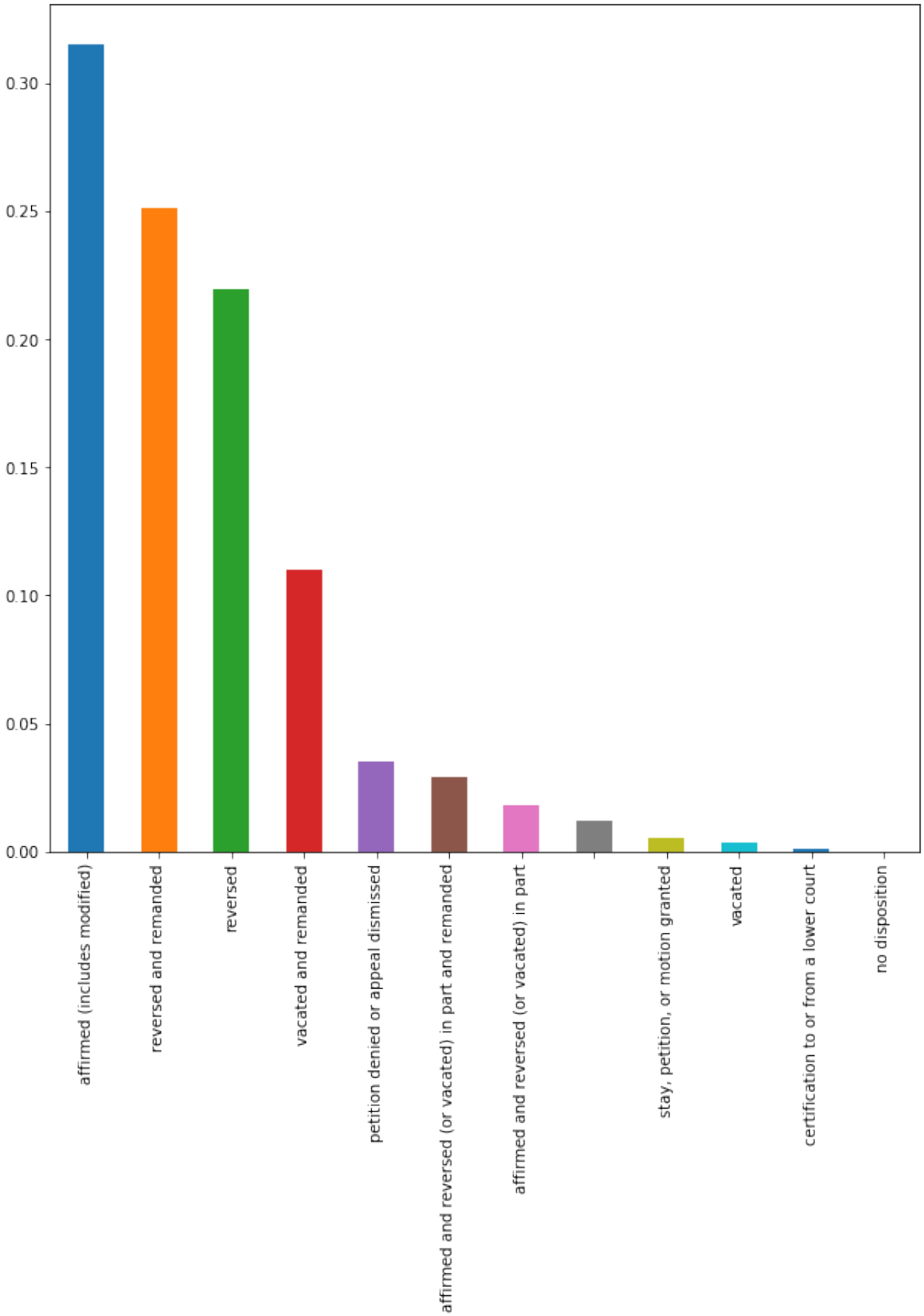
lcDispositionDirection



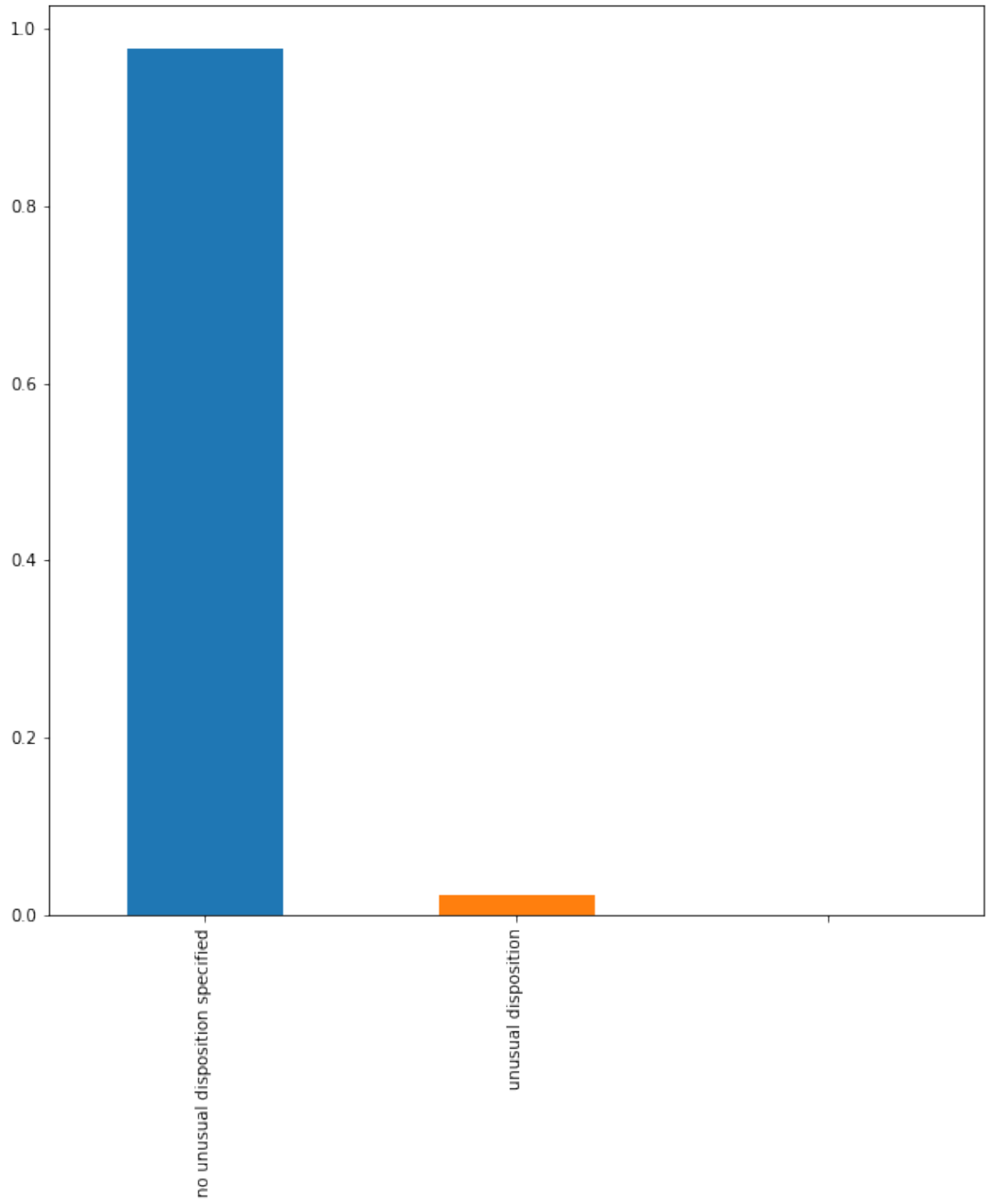
declarationUncon


0.8
0.6
0.4
0.2
0.0
no declaration of unconstitutionality
state or territorial law, reg, or const provision unconstitutional
act of congress declared unconstitutional
municipal or other local ordinance unconstitutional

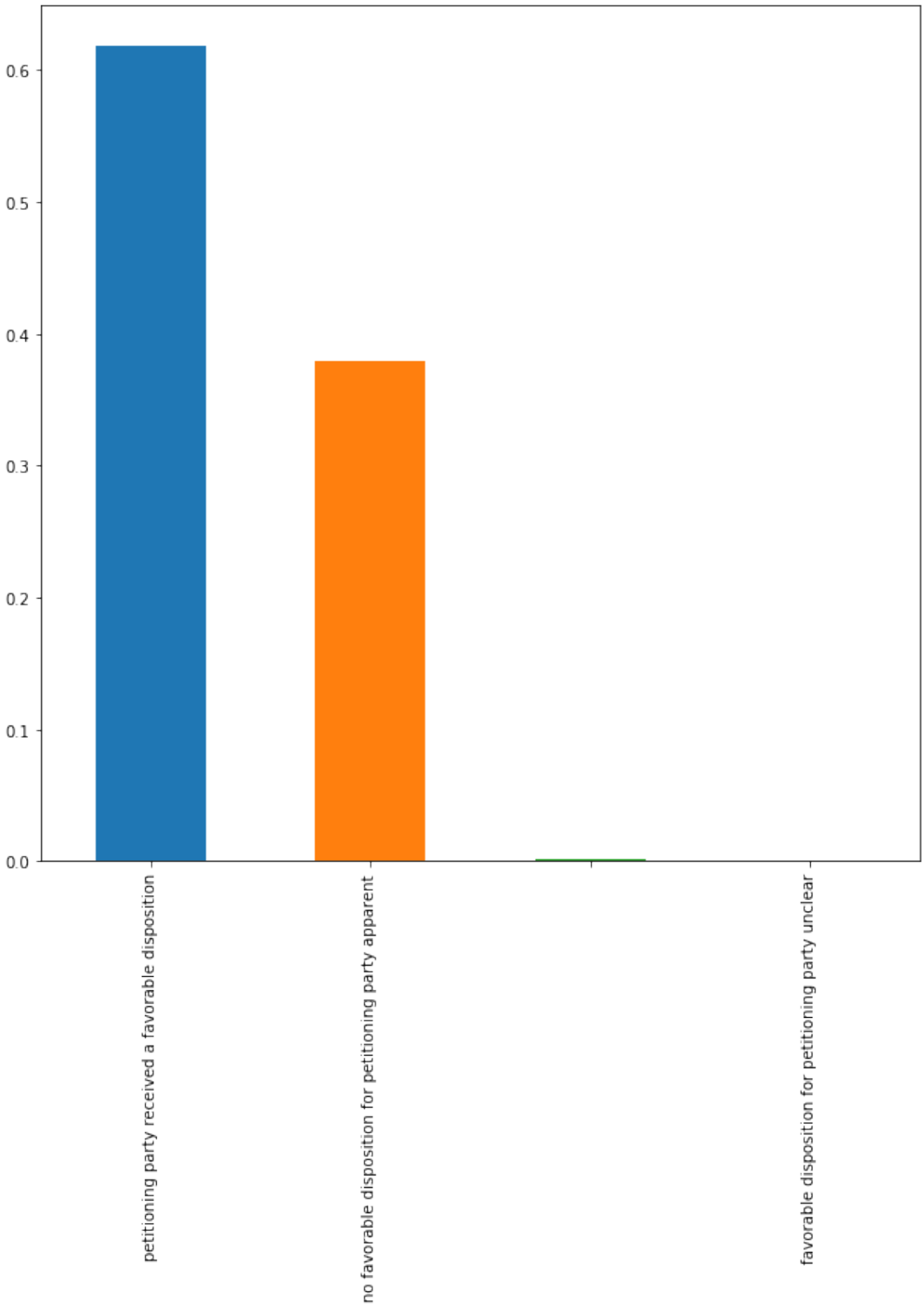
caseDisposition



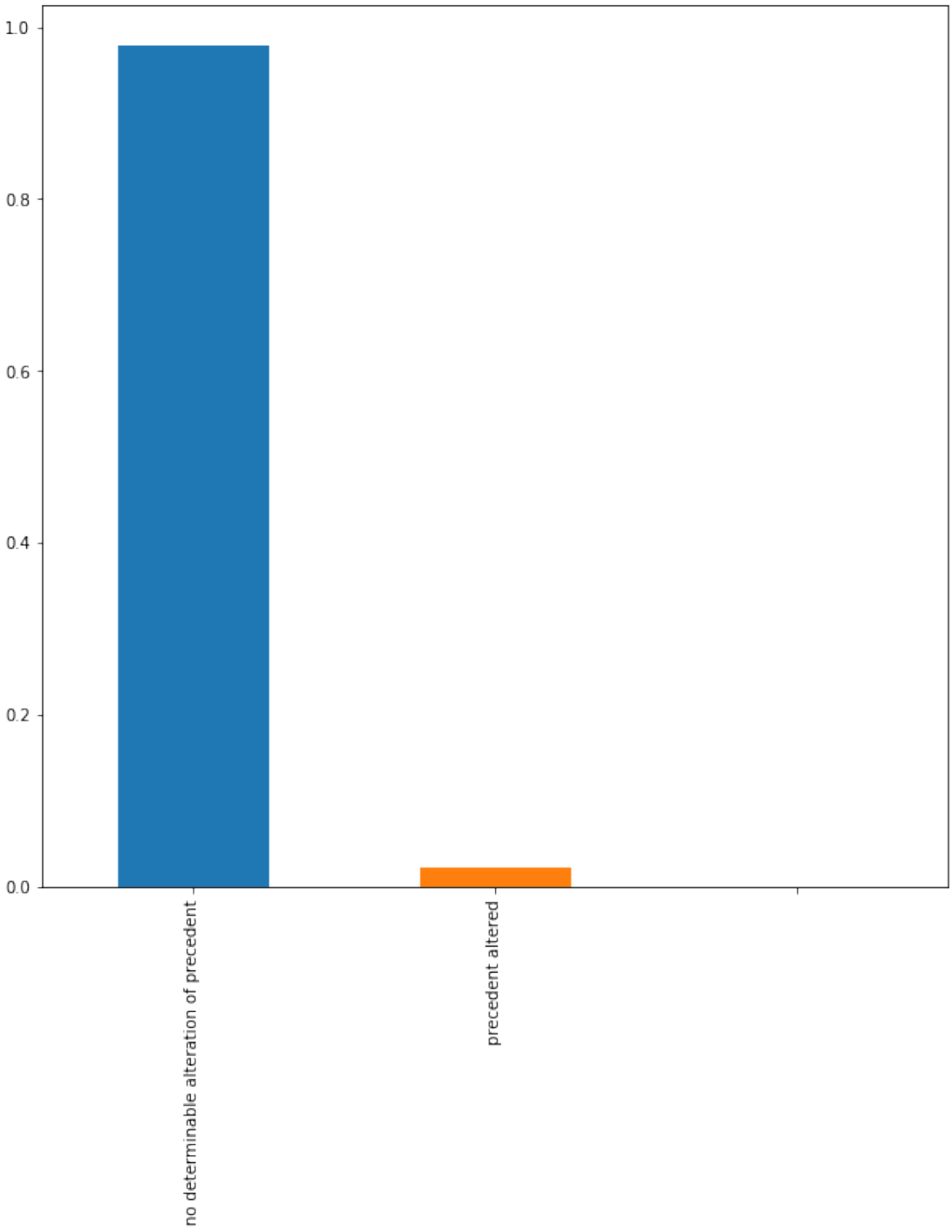
caseDispositionUnusual



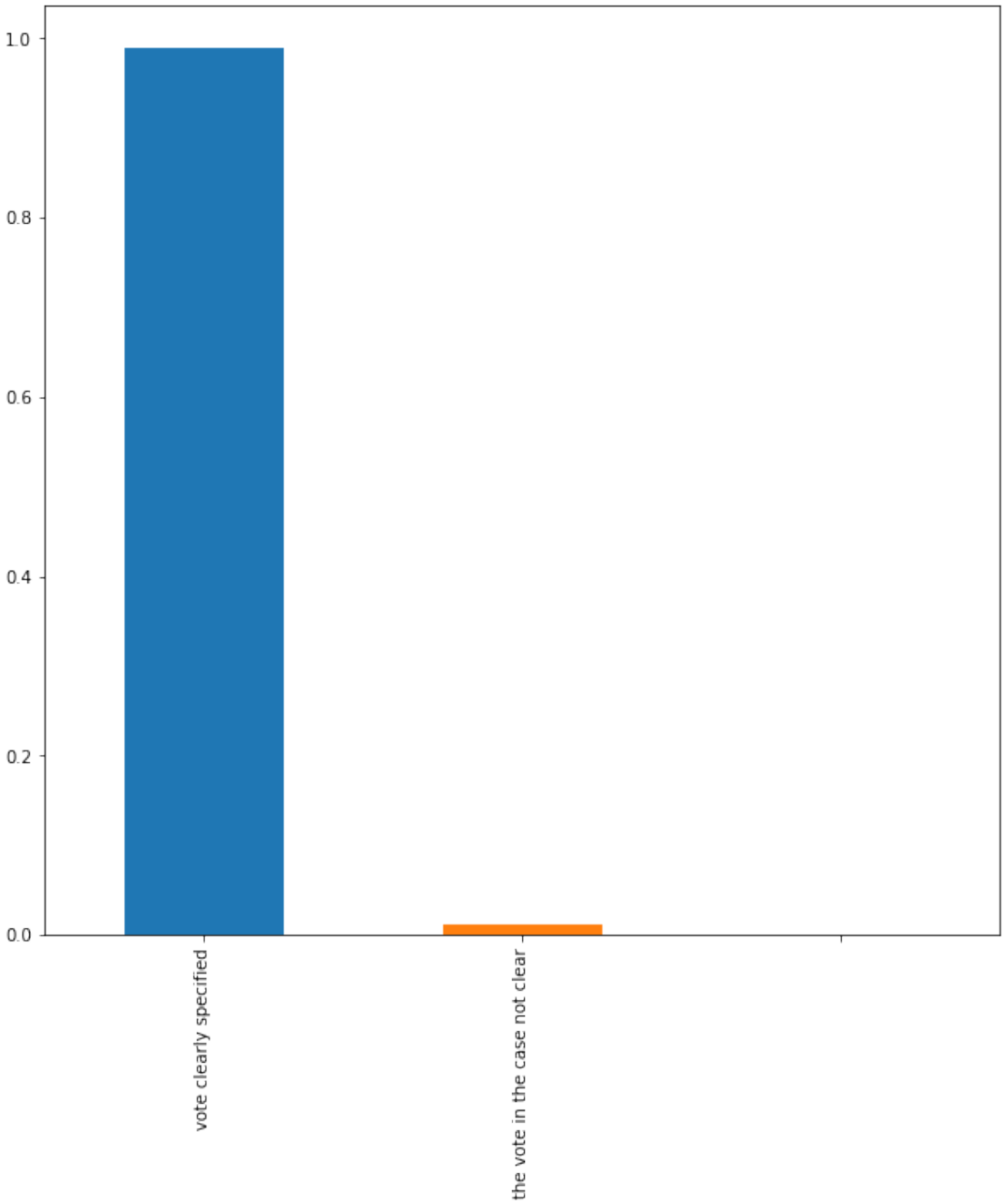
partyWinning



precedentAlteration



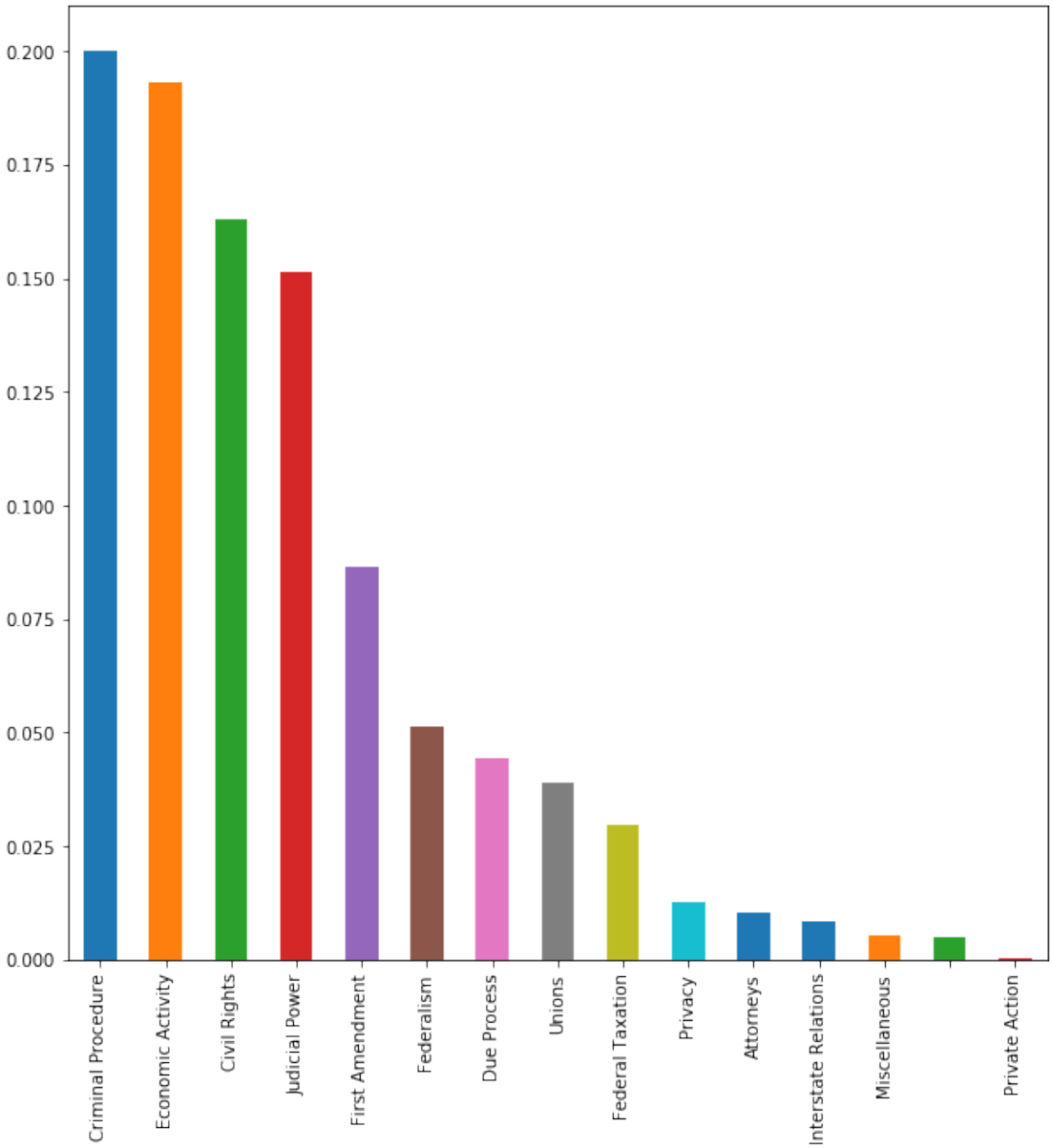
voteUnclear


1.0
0.8
0.6
0.4
0.2
0.0
vote clearly specified
the vote in the case not clear

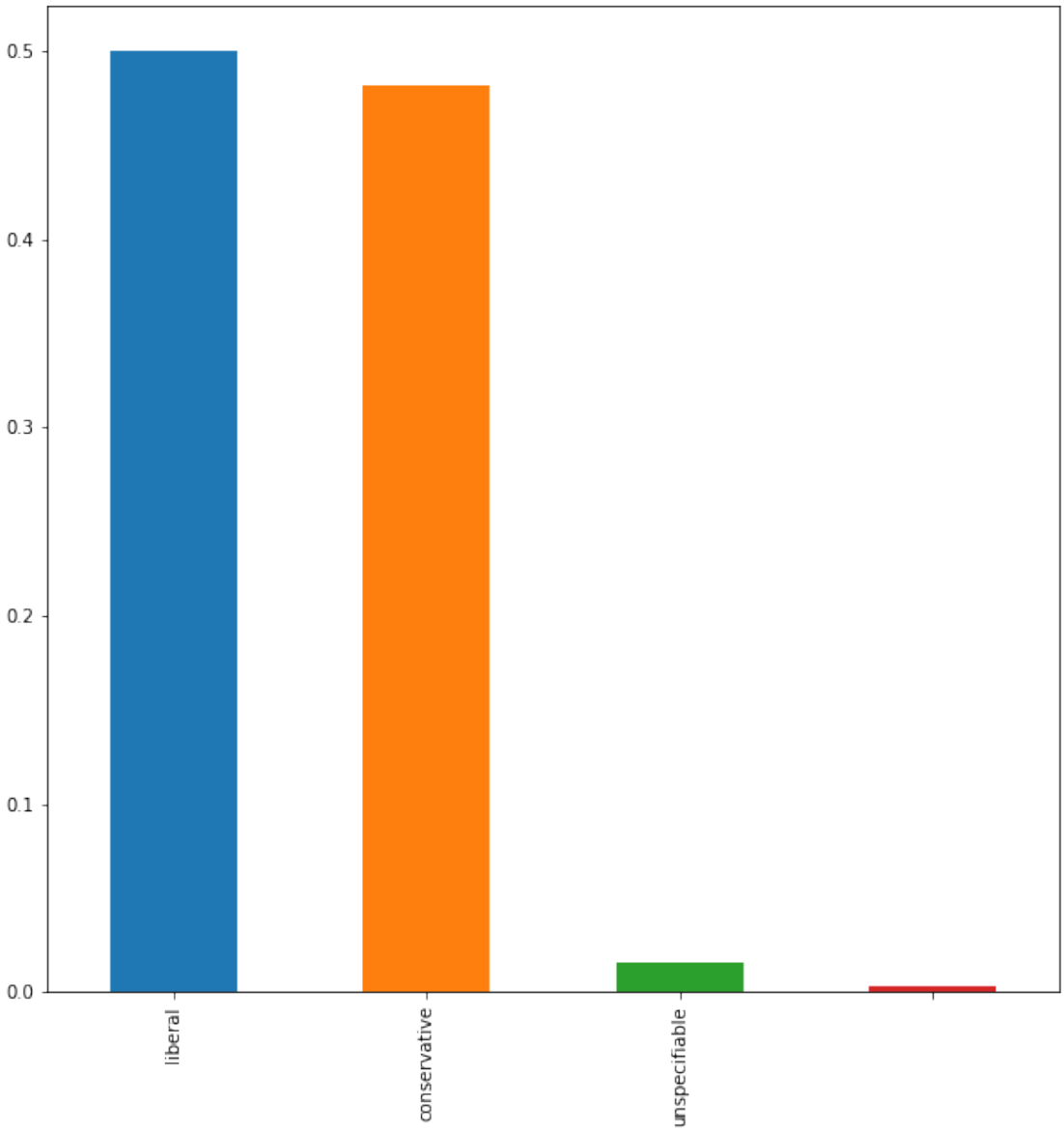
issue

0.030
0.025
0.020
0.015
0.010
0.005
0.000

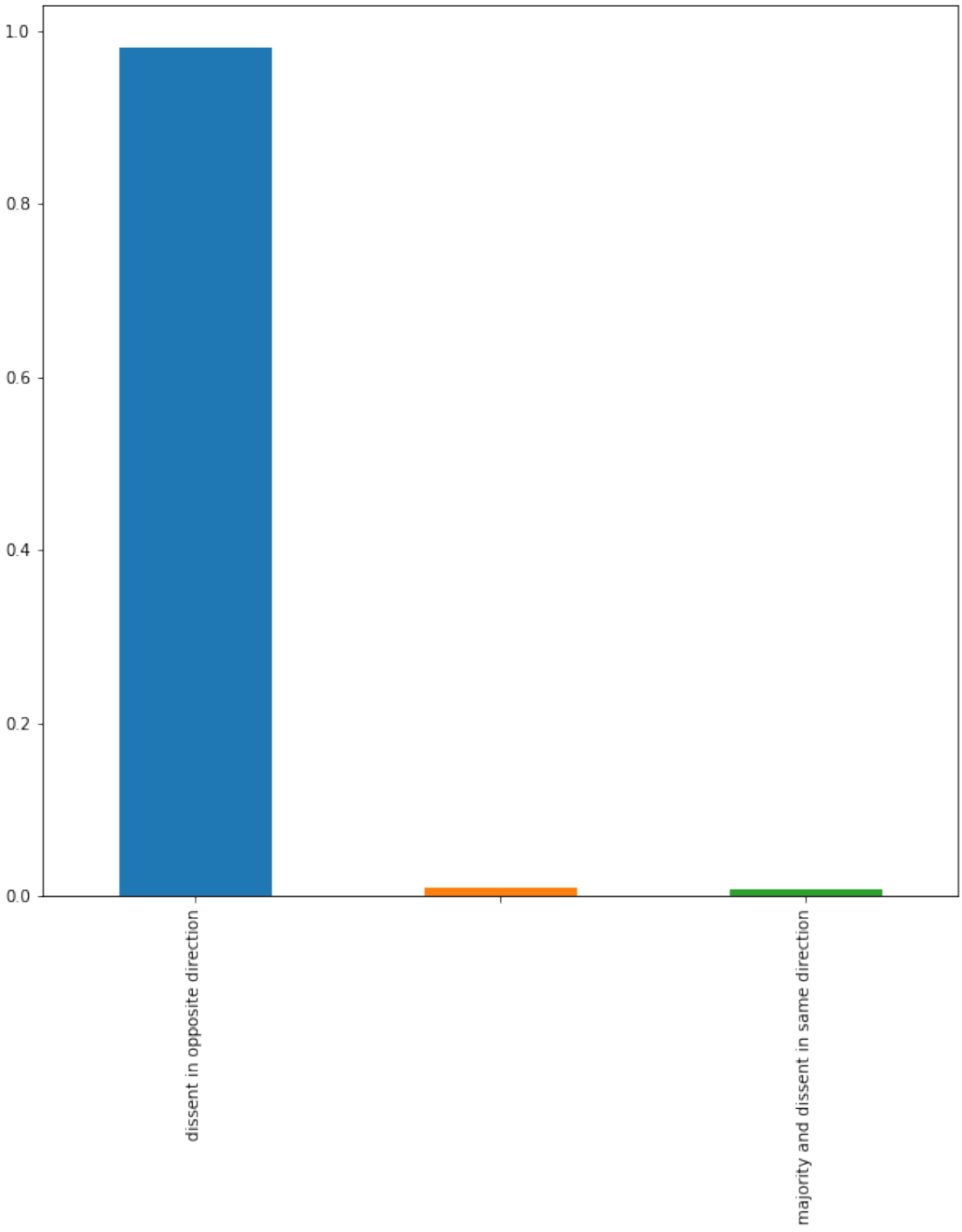
issueArea



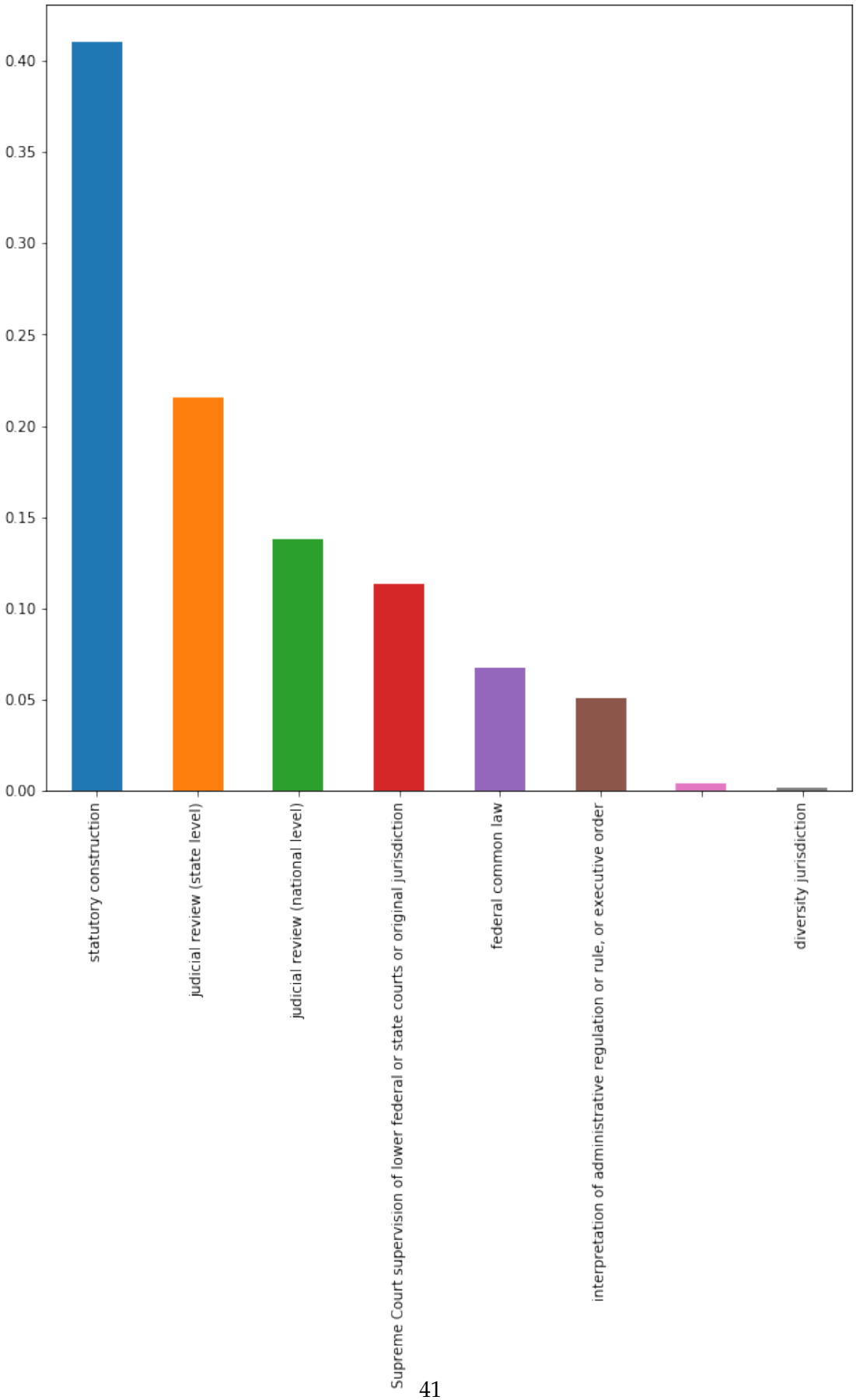
decisionDirection



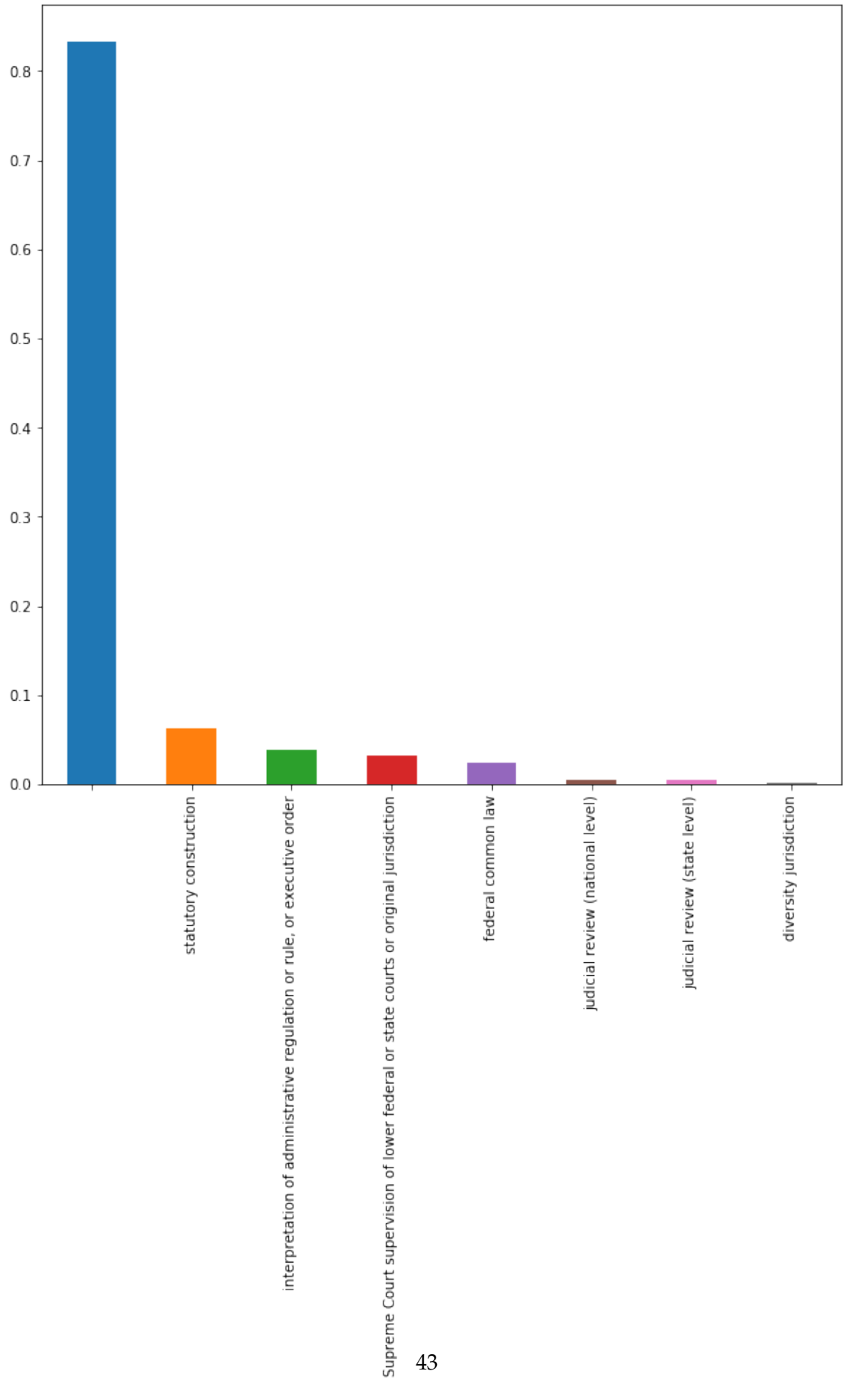

```
decisionDirectionDissent
```



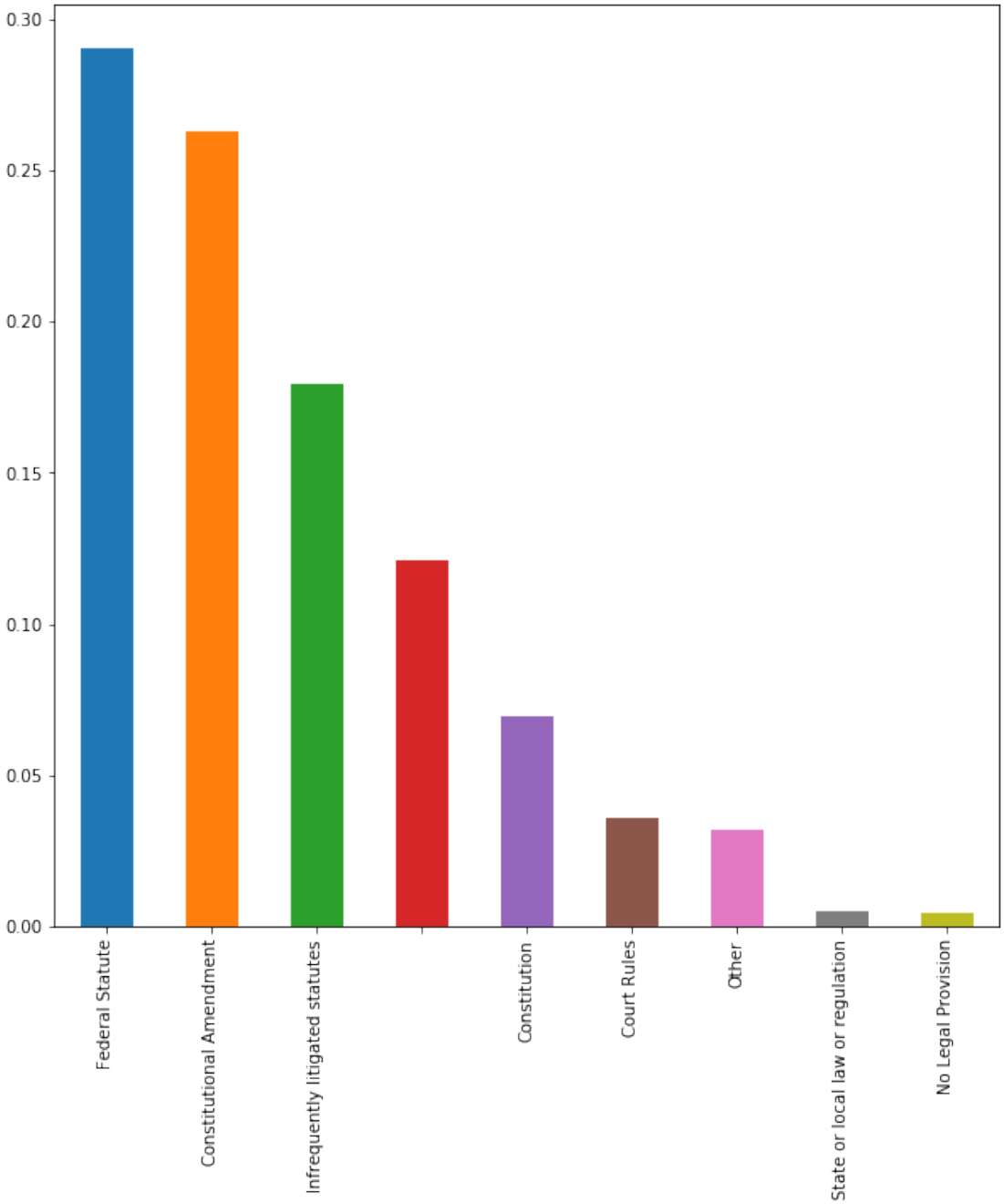
authorityDecision1


0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
statutory construction
judicial review (state level)
judicial review (national level)
Supreme Court supervision of lower federal or state courts or original jurisdiction
federal common law
interpretation of administrative regulation or rule, or executive order
diversity jurisdiction

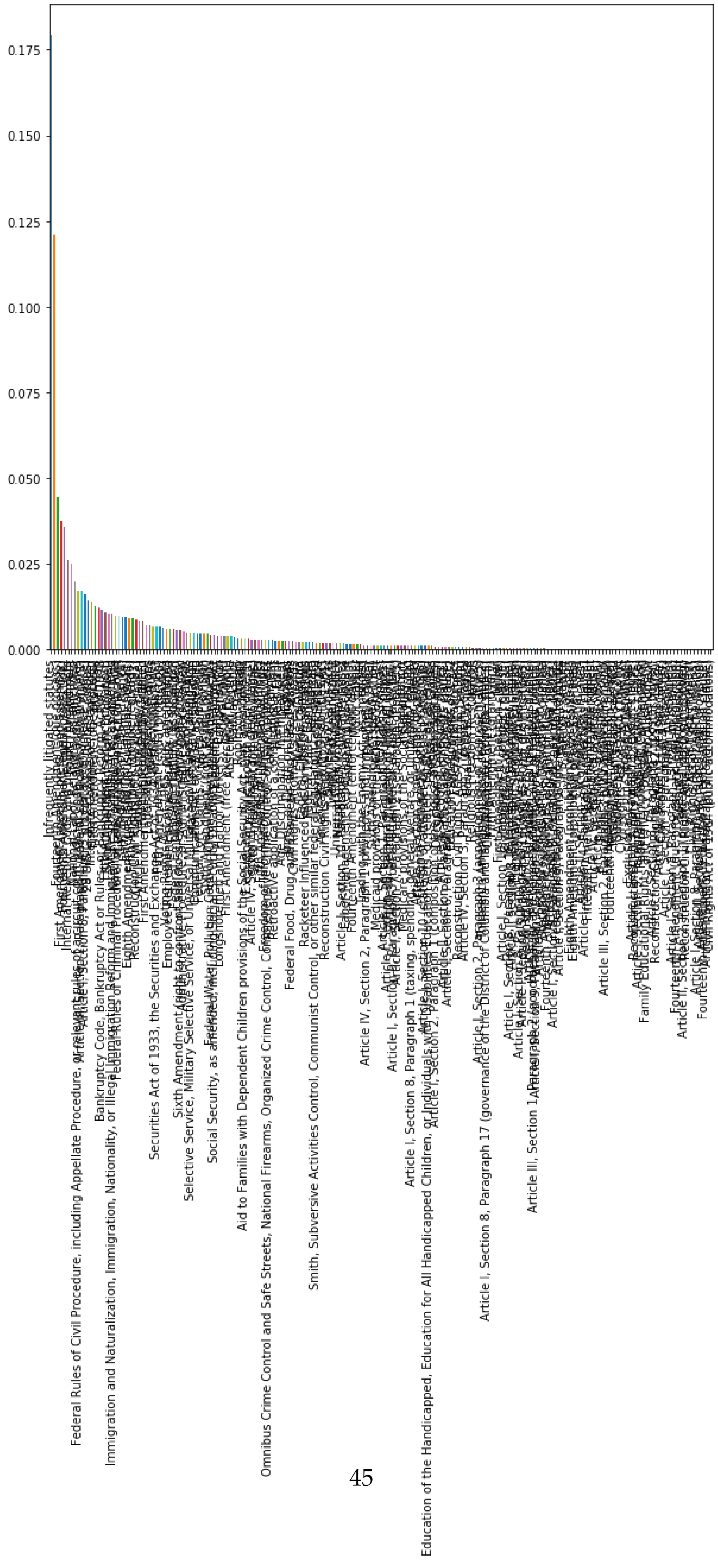
authorityDecision2


0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
statutory construction
interpretation of administrative regulation or rule, or executive order
Supreme Court supervision of lower federal or state courts or original jurisdiction
federal common law
judicial review (national level)
judicial review (state level)
diversity jurisdiction

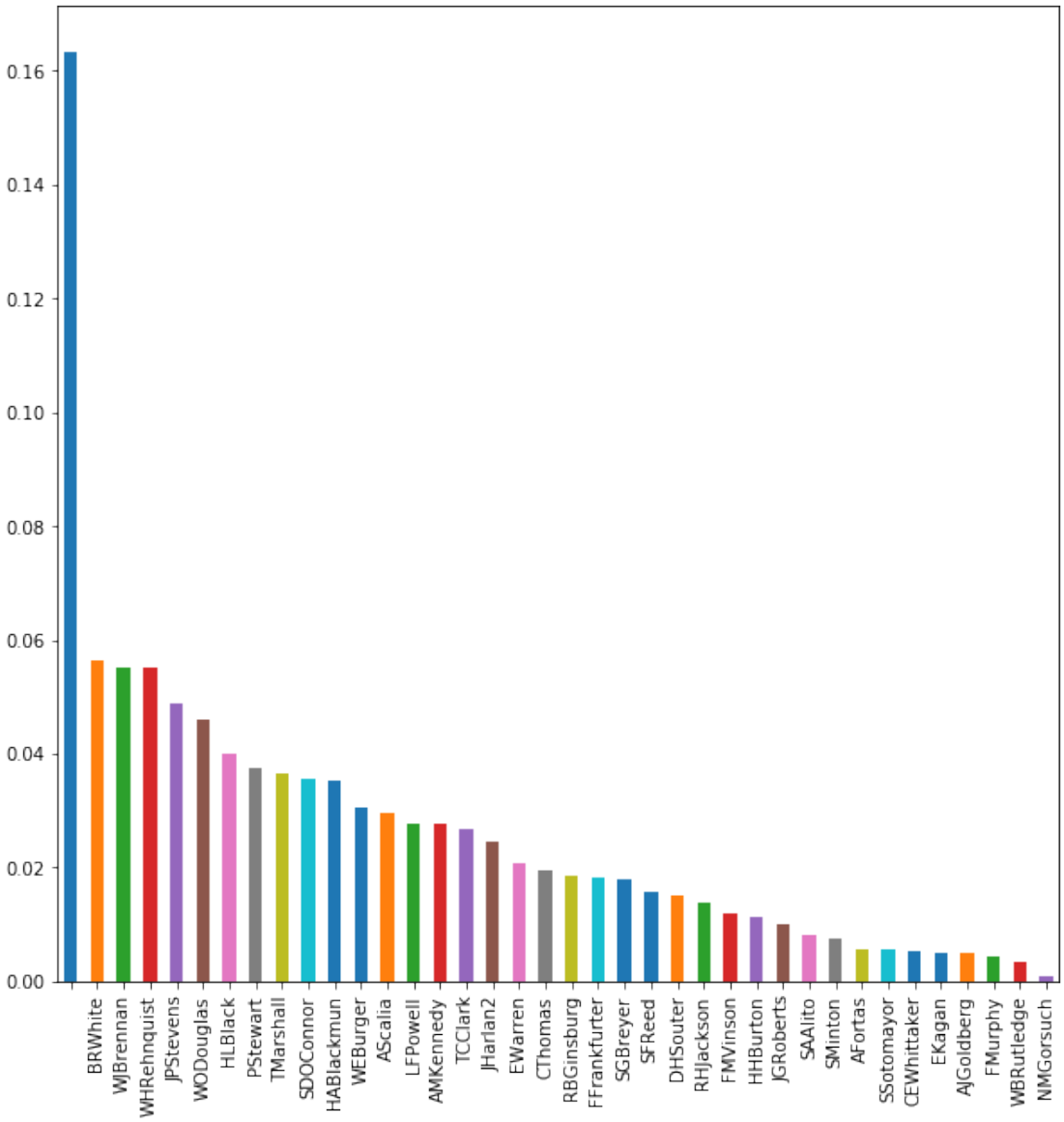
lawType



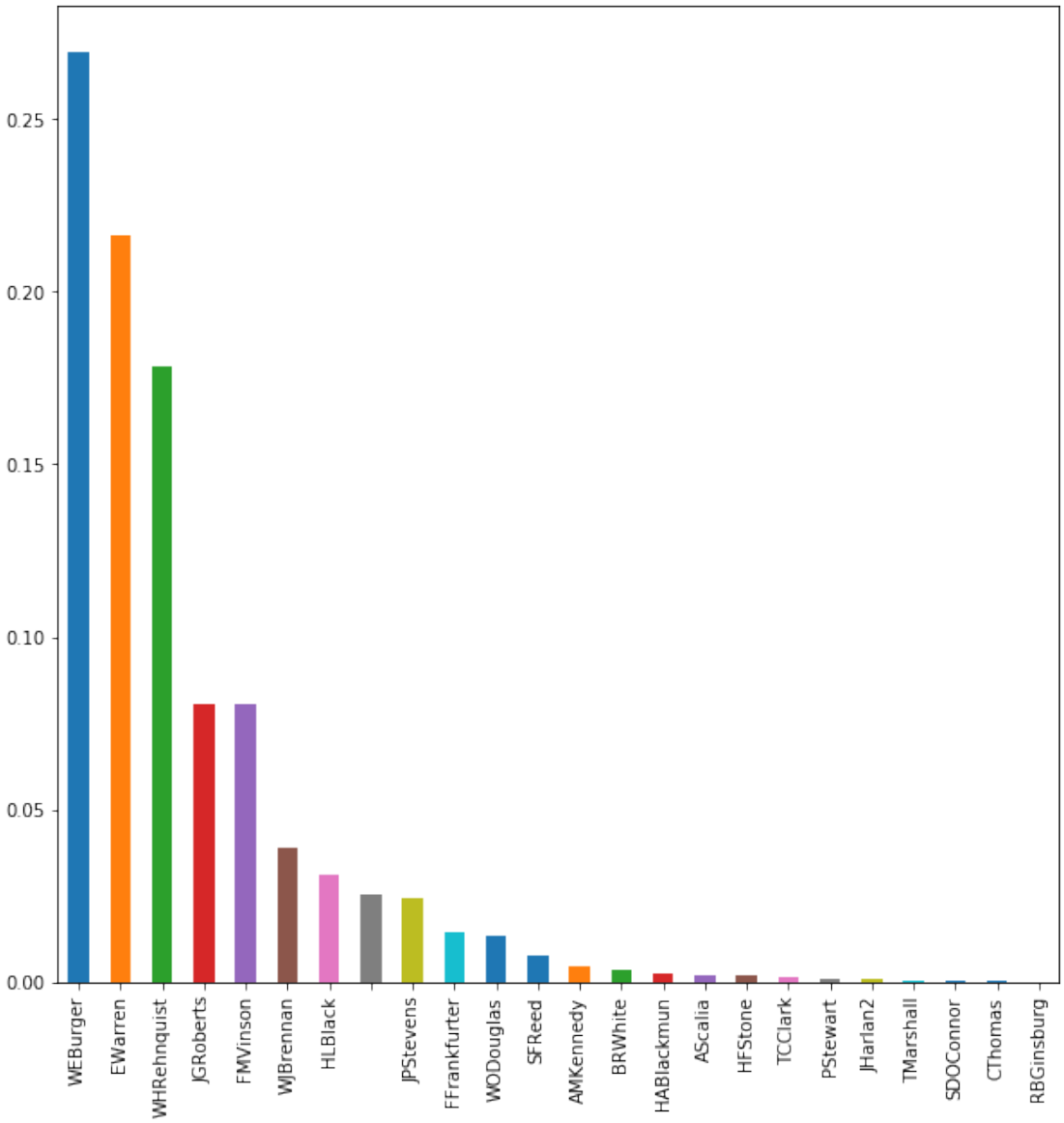
lawSupp


0.175
0.150
0.125
0.100
0.075
0.050
0.025
0.000
Infrequently litigated statutes
Aid to Families with Dependent Children provisions of the Social Security Act
Article I, Section 8, Paragraph 17 (governance of the District of Columbia and

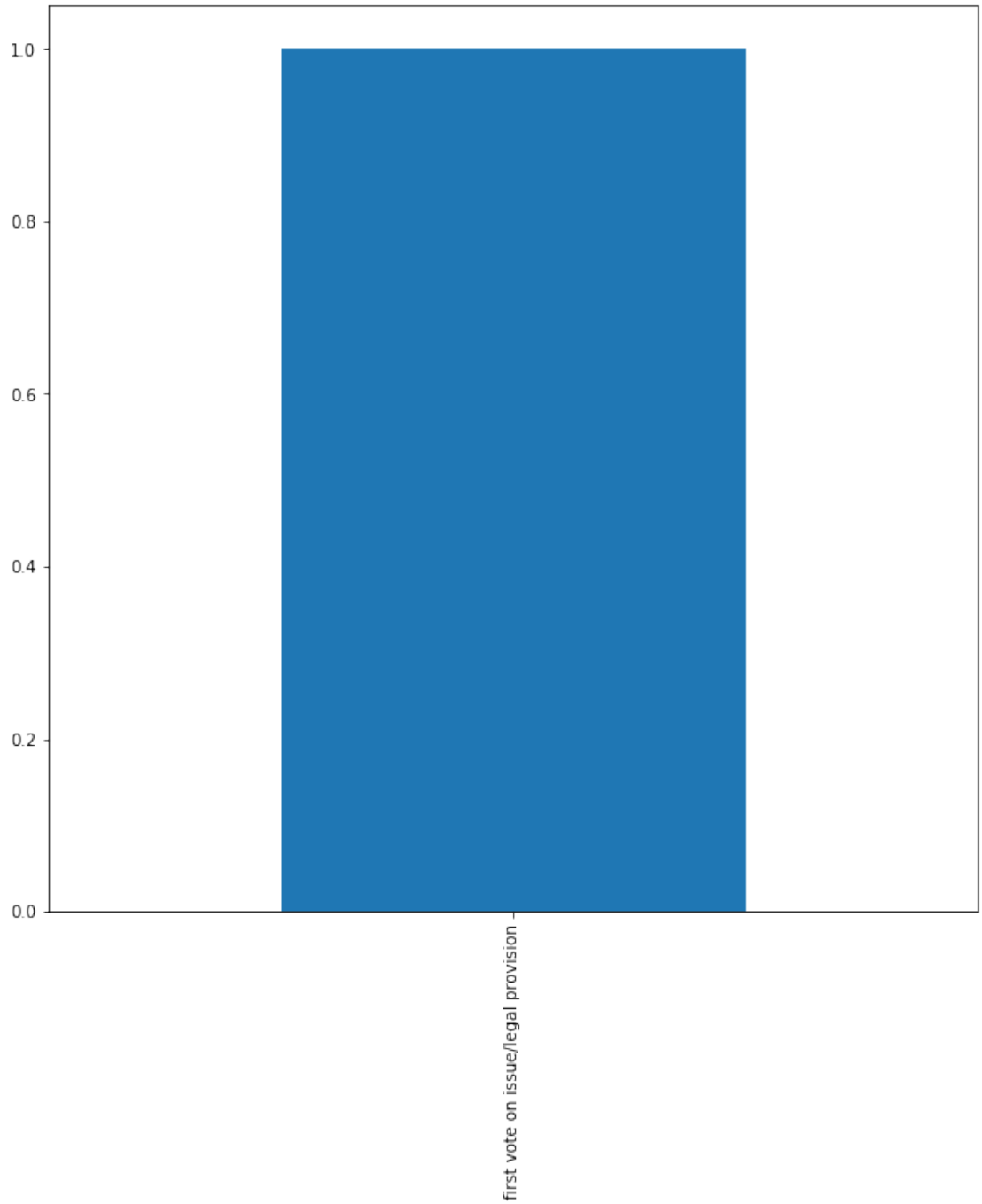
majOpinWriter



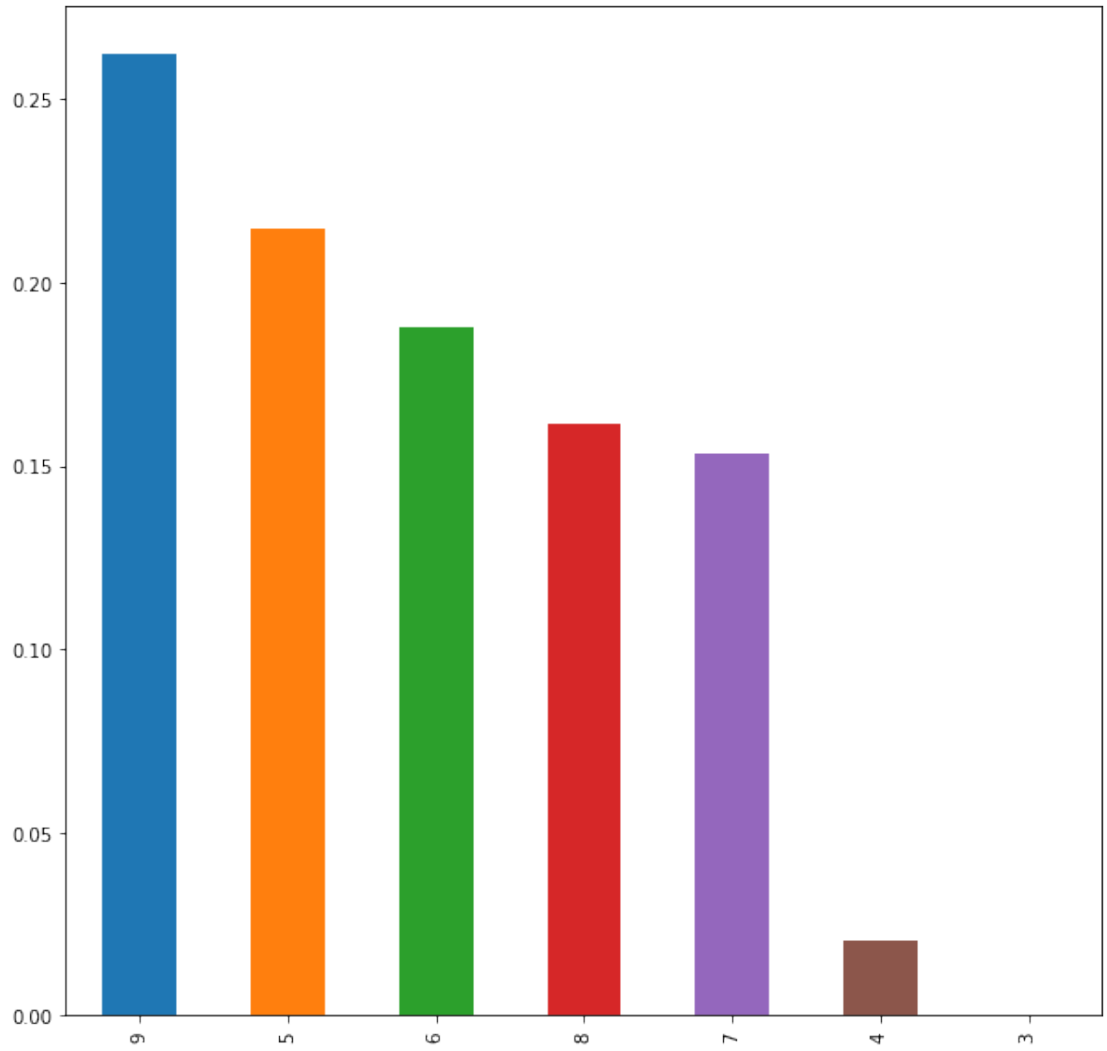
majOpinAssigner


0.25
0.20
0.15
0.10
0.05
0.00
WEBurger
EWarren
WHRehnquist
JGRoberts
FMVinson
WJBrennan
HLBlack
JPStevens
FFrankfurter
WODouglas
SFReed
AMKennedy
BRWhite
HABlackmun
AScalia
HFStone
TCClark
PStewart
JHarlan2
TMarshall
SDOConnor
CThomas
RBGinsburg

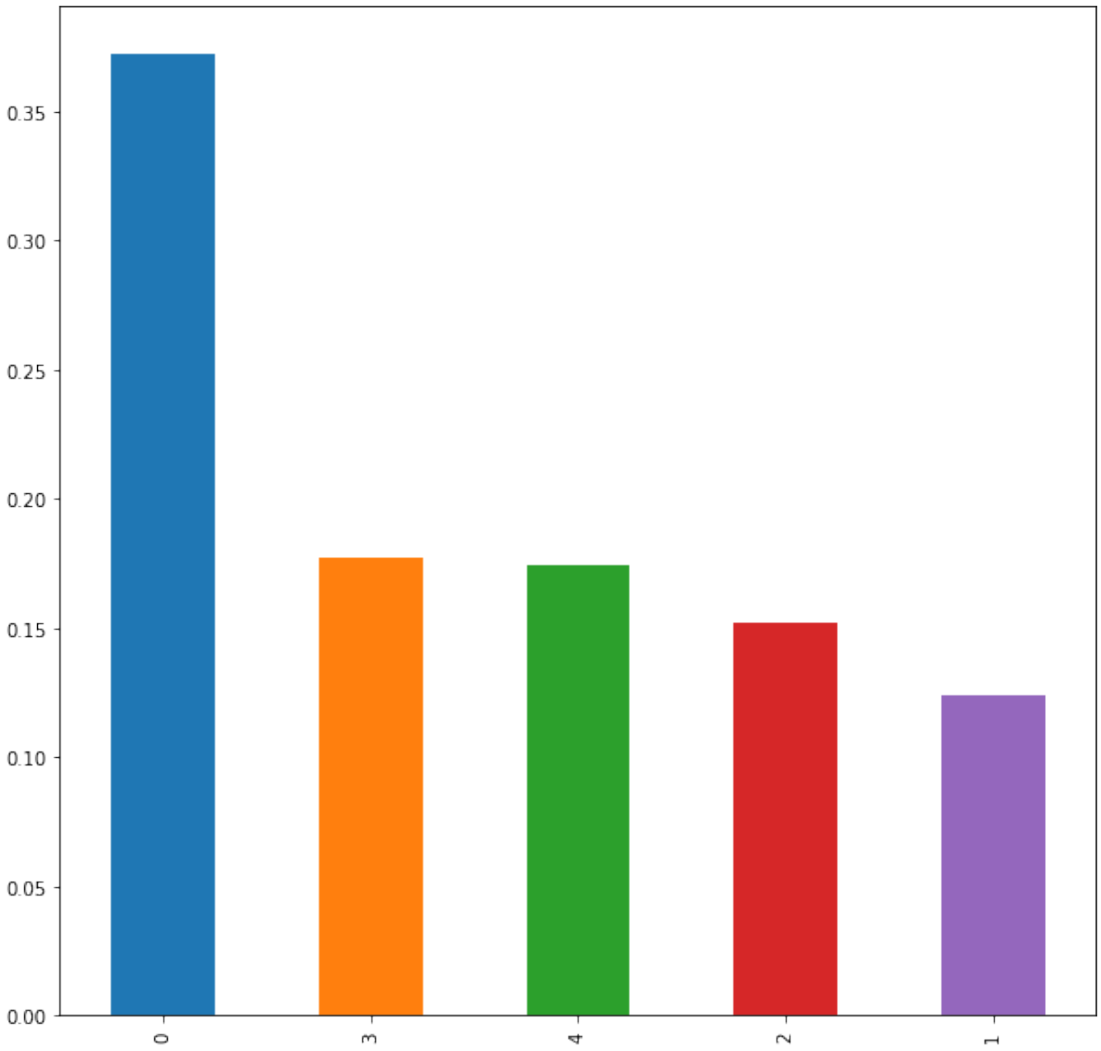
splitVote



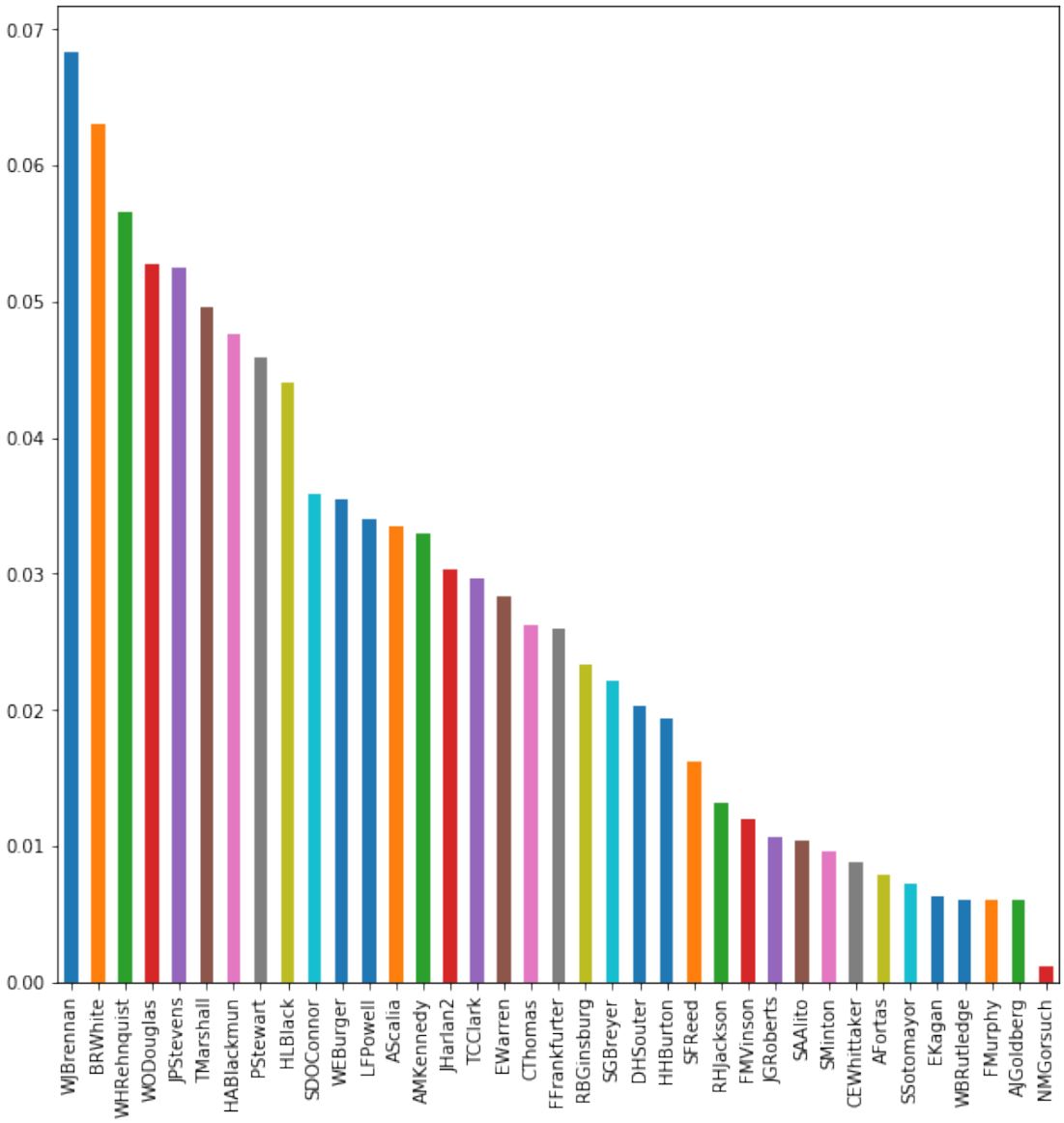
majVotes



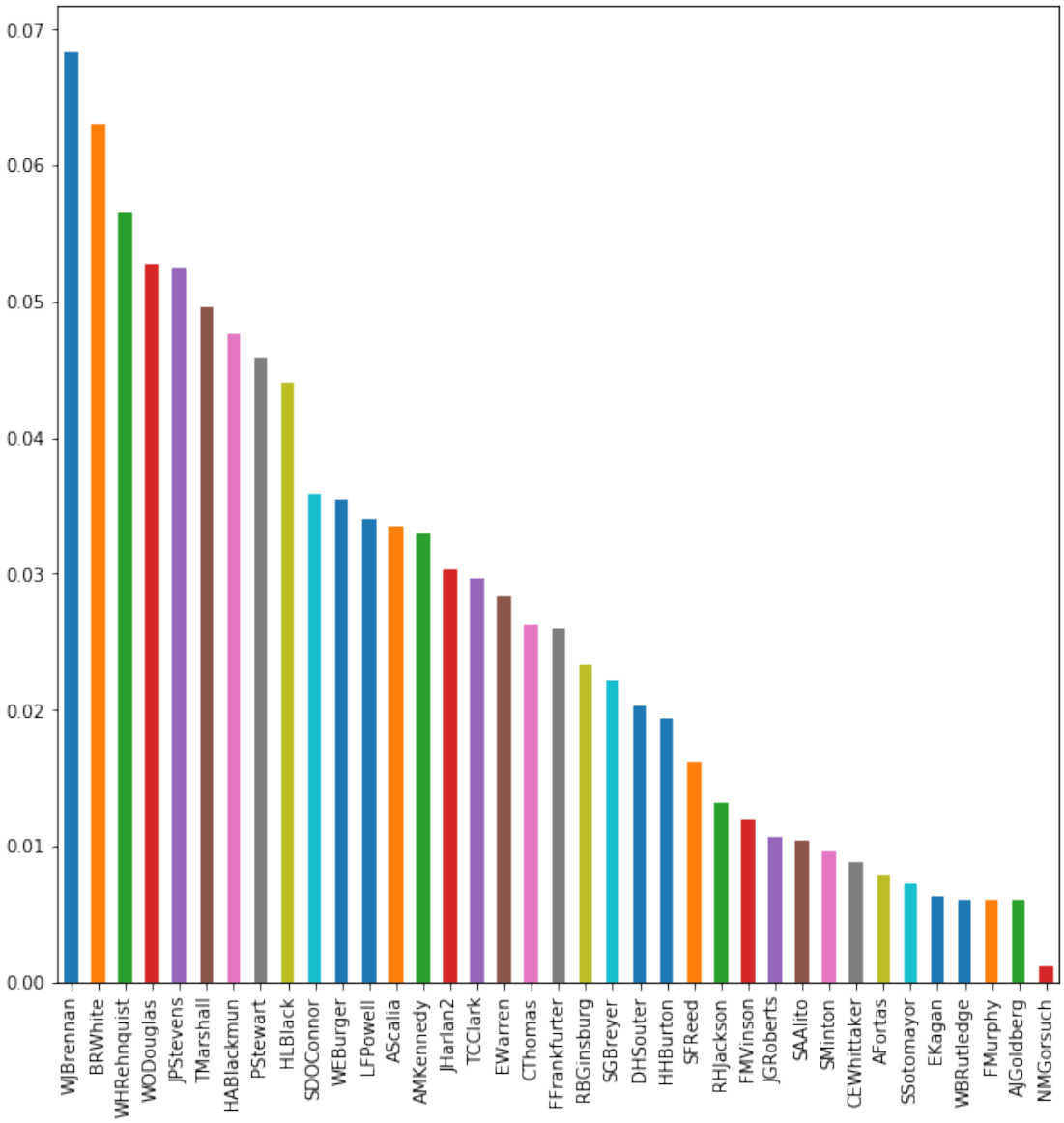

```
minVotes
```



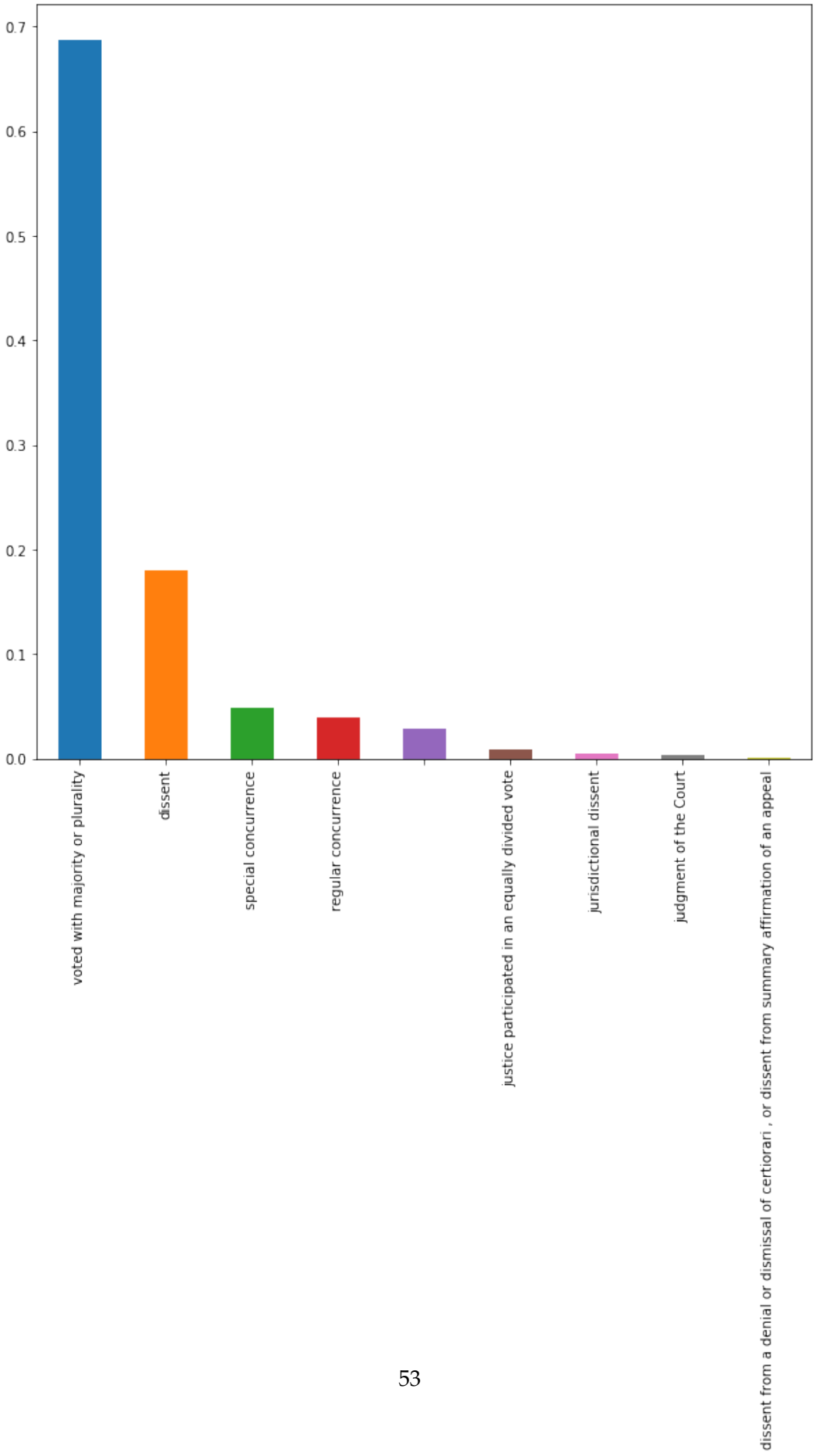

```
justice
```



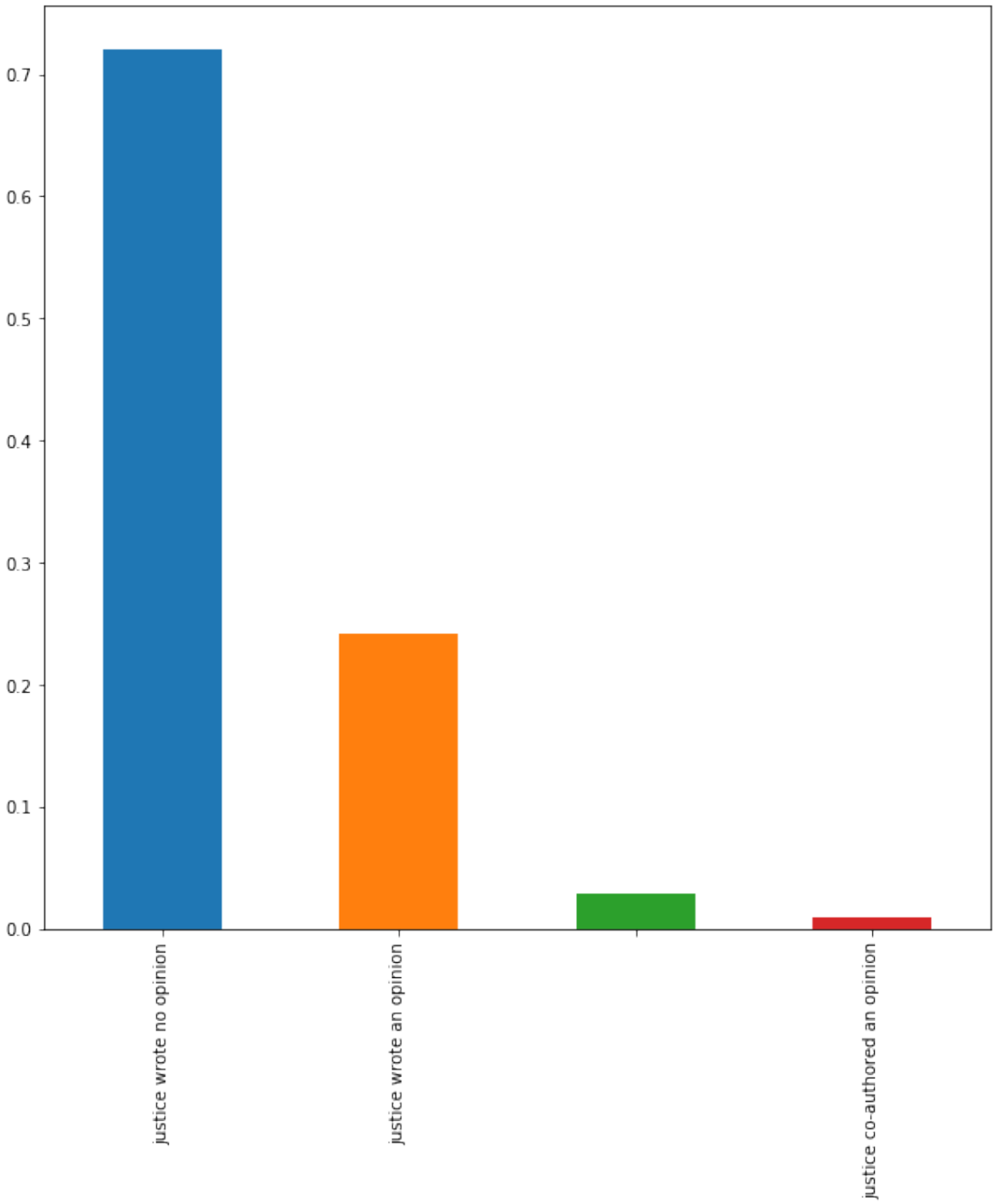
justiceName


0.07
0.06
0.05
0.04
0.03
0.02
0.01
0.00
WJBrennan
BRWhite
WHRehnquist
WODouglas
JPStevens
TMarshall
HABlackmun
PStewart
HLBlack
SDOConnor
WEBurger
LFPowell
AScalia
AMKennedy
JHarlan2
TCClark
EWarren
CThomas
FFrankfurter
RBGinsburg
SGBreyer
DHSouter
HHBurton
SFReed
RHJackson
FMVinson
JGRoberts
SAAlito
SMinton
CEWhittaker
AFortas
SSotomayor
EKagan
WBRutledge
FMurphy
AJGoldberg
NMGorsuch

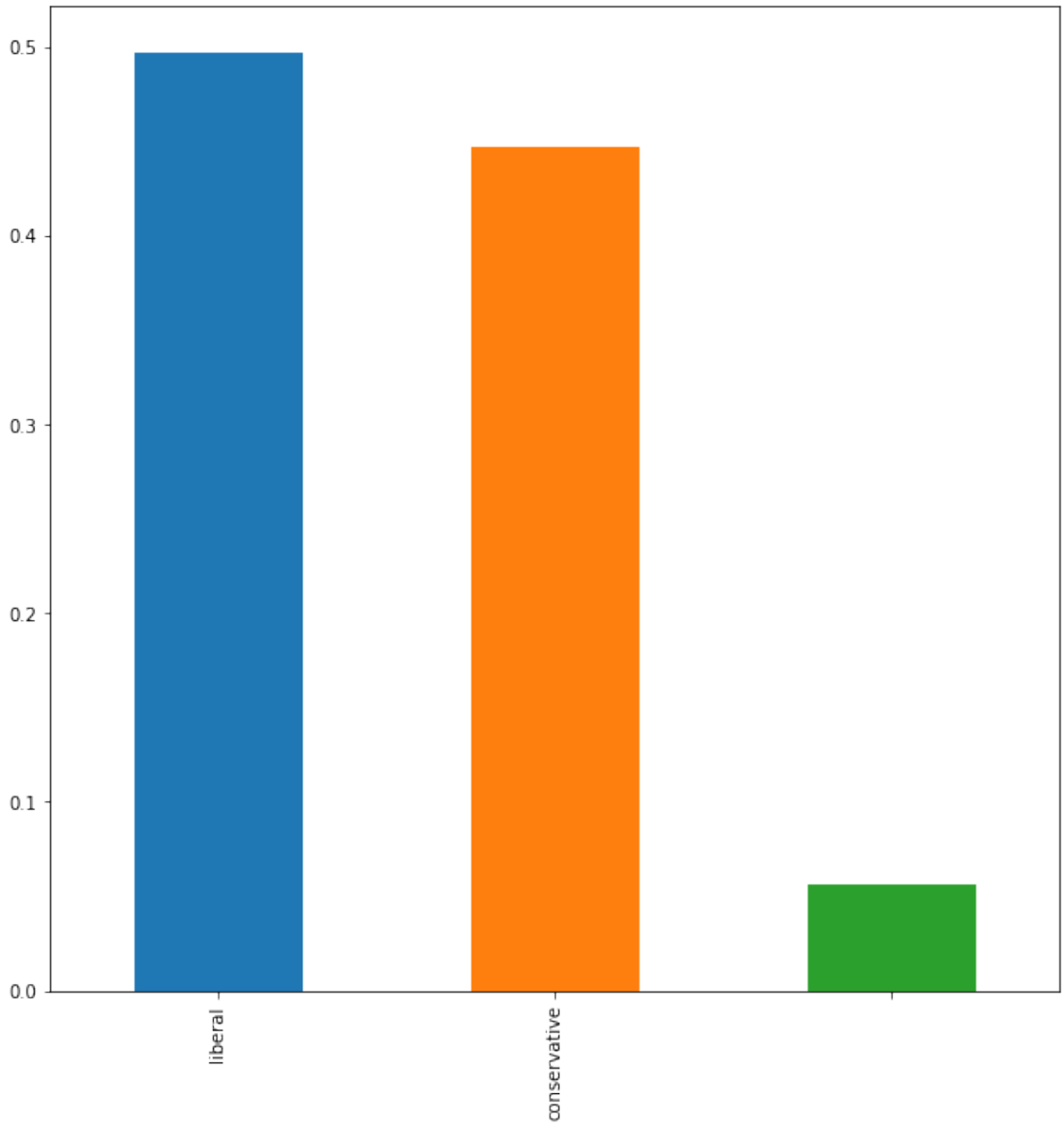
vote


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
voted with majority or plurality
dissent
special concurrence
regular concurrence
justice participated in an equally divided vote
jurisdictional dissent
judgment of the Court
dissent from a denial or dismissal of certiorari , or dissent from summary affirmation of an appeal

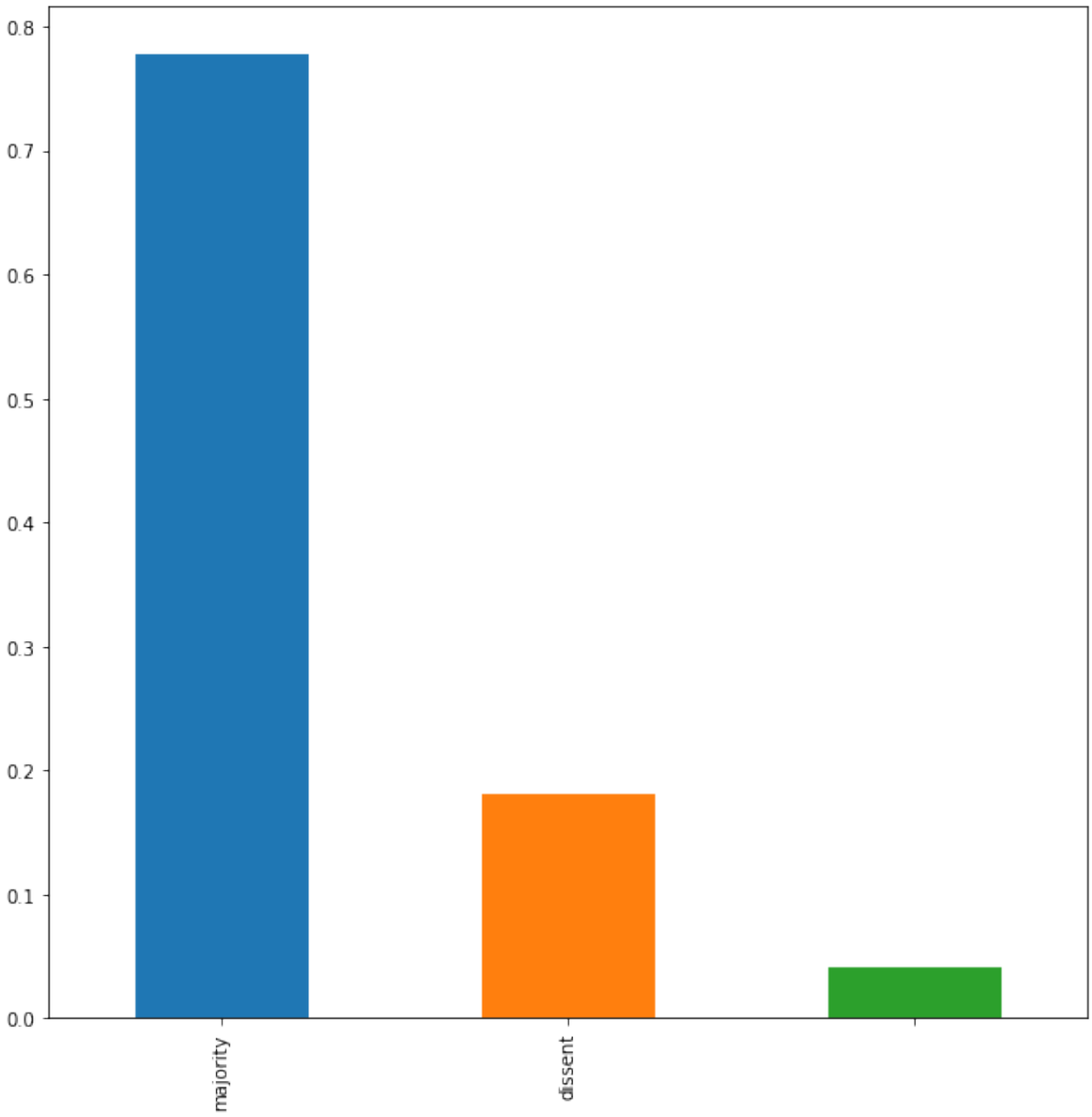
opinion



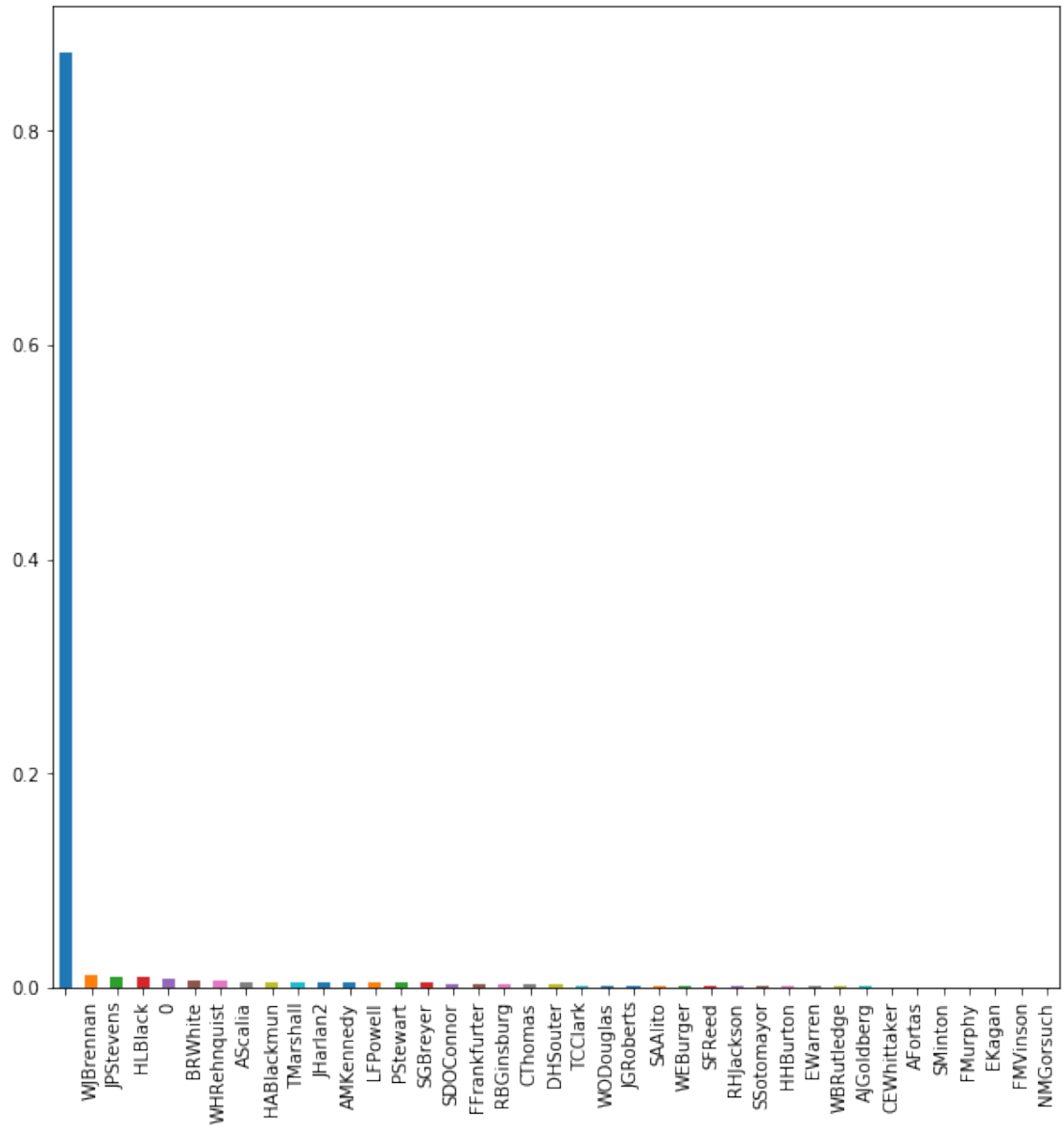
direction



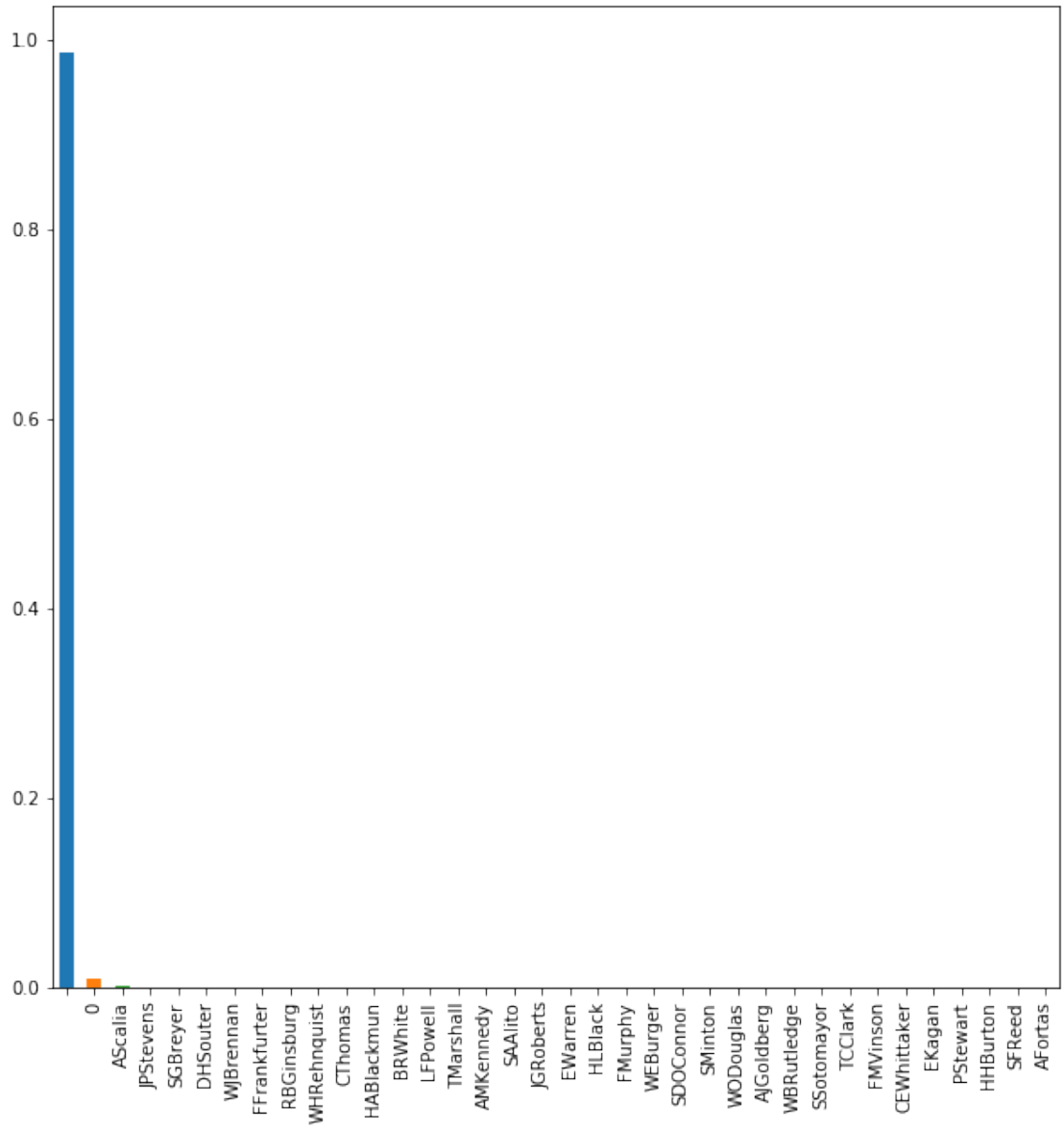

```
majority
```



`firstAgreement`



`secondAgreement`



In [ ]:

df['firstAgreement'].value_counts(normalize=True).plot(kind="bar", figsize=(10, 10))

In [ ]: