CS146 7.2 PCW (Updated)

October 24, 2019

1 Modeling 2016 US Presidential Elections

```
In [1]: from scipy import stats
        import numpy as np
        import matplotlib.pyplot as plt
        import pystan
```

1.1 Data

The `electoral_votes` variable is a dictionary containing the number of Electoral College votes for each state. For example

```
>>> electoral_votes['Indiana']
11
```

Data from Wikipedia: United_States_Electoral_College

The `survey_results` variable is a dictionary mapping from states to an array of survey results for each candidate. Each row in a survey results array represents one survey and each column represents one candidate. There are 4 columns, representing Clinton, Trump, Johnson, and Stein in that order. In the example below, Clinton got 340 votes in the first survey, Trump got 258, Johnson got 27, and Stein got 13.

```
>>> survey_results['Indiana']
array([[340, 258,  27,  13],
       [240, 155,   5,   5],
       [235, 155,  50,  20],
       [308, 266,  49,  35],
       [222, 161,  80,  30]])
```

Data from Wikipedia: Statewide opinion polling for the United States presidential election, 2016

```
In [2]: electoral_votes = {
            'California': 55, 'Texas': 38, 'Florida': 29, 'New York': 29, 'Illinois': 20,
            'Pennsylvania': 20, 'Ohio': 18, 'Georgia': 16, 'Michigan': 16,
            'North Carolina': 15, 'New Jersey': 14, 'Virginia': 13, 'Washington': 12,
            'Arizona': 11, 'Indiana': 11, 'Massachusetts': 11, 'Tennessee': 11,
            'Maryland': 10, 'Minnesota': 10, 'Missouri': 10, 'Wisconsin': 10, 'Alabama': 9,
```

```
    'Colorado': 9, 'South Carolina': 9, 'Kentucky': 8, 'Louisiana': 8,
    'Connecticut': 7, 'Oklahoma': 7, 'Oregon': 7, 'Arkansas': 6, 'Iowa': 6,
    'Kansas': 6, 'Mississippi': 6, 'Nevada': 6, 'Utah': 6, 'Nebraska': 5,
    'New Mexico': 5, 'West Virginia': 5, 'Hawaii': 4, 'Idaho': 4, 'Maine': 4,
    'New Hampshire': 4, 'Rhode Island': 4, 'Alaska': 3, 'Delaware': 3,
    'District of Columbia': 3, 'Montana': 3, 'North Dakota': 3, 'South Dakota': 3,
    'Vermont': 3, 'Wyoming': 3}

survey_results = {
    'Alabama': np.array([], dtype=int).reshape(0, 4),
    'Alaska': np.array([400 * np.array([.47, .43, .07, .03]), 500 * np.array([.36, .37
    'Arizona': np.array([392 * np.array([.45, .47, .05, .02]), 550 * np.array([.39, .47
    'Arkansas': np.array([463 * np.array([.33, .56, .04, .02]), 831 * np.array([.34, .5
    'California': np.array([401 * np.array([.58, .35, .03, .02]), 747 * np.array([.56,
    'Colorado': np.array([1150 * np.array([.45, .44, .05, .04]), 500 * np.array([.44,
    'Connecticut': np.array([1000 * np.array([.50 , .35 , .09 , .04])], dtype=int),
    'Delaware': np.array([762 * np.array([.51 , .30 , .07 , .02])], dtype=int),
    'District of Columbia': np.array([], dtype=int).reshape(0, 4),
    'Florida': np.array([1100 * np.array([.46, .50, .02, .01]), 884 * np.array([.46, .4
    'Georgia': np.array([1250 * np.array([.45, .52, .02, .00]), 650 * np.array([.42, .5
    'Hawaii': np.array([], dtype=int).reshape(0, 4),
    'Idaho': np.array([608 * np.array([.30 , .40 , .10 , .03])], dtype=int),
    'Illinois': np.array([500 * np.array([.53, .41, .02, .00]), 600 * np.array([.45, .3
    'Indiana': np.array([1313 * np.array([.36, .44, .10, .03])], dtype=int),
    'Iowa': np.array([800 * np.array([.39 , .46 , .06 , .01]), 700 * np.array([.41 , .4
    'Kansas': np.array([624 * np.array([.38 , .49 , .07 , .01]), 581 * np.array([.36 ,
    'Kentucky': np.array([602 * np.array([.37 , .54 , .01 , .01]), 602 * np.array([.37
    'Louisiana': np.array([603 * np.array([.35 , .49 , .07 , .02]), 625 * np.array([.34
    'Maine': np.array([855 * np.array([.45 , .39 , .07 , .04]), 750 * np.array([.46 ,
    'Maryland': np.array([706 * np.array([.63, .27, .04, .02]), 514 * np.array([.58, .2
    'Massachusetts': np.array([417 * np.array([.56, .26, .08, .03]), 500 * np.array([.5
    'Michigan': np.array([1200 * np.array([.47, .49, .03, .01]), 957 * np.array([.46,
    'Minnesota': np.array([656 * np.array([.49, .39, .05, .02]), 625 * np.array([.46,
    'Mississippi': np.array([], dtype=int).reshape(0, 4),
    'Missouri': np.array([750 * np.array([.41, .47, .07, .02]), 871 * np.array([.37, .5
    'Montana': np.array([590 * np.array([.27 , .43 , .07 , .02])], dtype=int),
    'Nebraska': np.array([700 * np.array([.29 , .56 , .07 , .01])], dtype=int),
    'Nevada': np.array([387 * np.array([.46 , .46 , .05 , .01]), 1158 * np.array([.45
    'New Hampshire': np.array([701 * np.array([.49 , .38 , .06 , .01]), 1000 * np.array
    'New Jersey': np.array([678 * np.array([.51, .40, .03, .01]), 293 * np.array([.49,
    'New Mexico': np.array([8439 * np.array([.46 , .44 , .06 , .01]), 504 * np.array([
    'New York': np.array([617 * np.array([.51, .34, .05, .02]), 611 * np.array([.54, .3
    'North Carolina': np.array([1154 * np.array([.44, .49, .04, .00]), 992 * np.array(
    'North Dakota': np.array([400 * np.array([.32 , .43 , .08 , .01])], dtype=int),
    'Ohio': np.array([900 * np.array([.39, .46, .07, .03]), 1189 * np.array([.45, .46,
    'Oklahoma': np.array([], dtype=int).reshape(0, 4),
    'Oregon': np.array([504 * np.array([.41 , .34 , .04 , .02]), 608 * np.array([.46 ,
    'Pennsylvania': np.array([1220 * np.array([.46, .40, .07, .02]), 1300 * np.array([
```

```
    'Rhode Island': np.array([600 * np.array([.52 , .32 , .05 , .05]), 800 * np.array(
    'South Carolina': np.array([475 * np.array([.38, .42, .06, .03]), 1247 * np.array(
    'South Dakota': np.array([], dtype=int).reshape(0, 4),
    'Tennessee': np.array([508 * np.array([.34, .44, .07, .02]), 472 * np.array([.38,
    'Texas': np.array([700 * np.array([.35, .49, .05, .04]), 679 * np.array([.40, .49,
    'Utah': np.array([500 * np.array([.24 , .33 , .05 , .03]), 1000 * np.array([.20 ,
    'Vermont': np.array([1052 * np.array([.52 , .26 , .05 , .02]), 603 * np.array([.50
    'Virginia': np.array([800 * np.array([.49, .45, .03, .01]), 1024 * np.array([.48,
    'Washington': np.array([681 * np.array([.50, .38, .04, .02]), 502 * np.array([.48,
    'West Virginia': np.array([], dtype=int).reshape(0, 4),
    'Wisconsin': np.array([500 * np.array([.44, .38, .07, .02]), 1190 * np.array([.46,
    'Wyoming': np.array([722 * np.array([.20 , .58 , .09 , .02]), 402 * np.array([.19
}

states = sorted(survey_results.keys())
print('Modeling', len(states), 'states with', sum(electoral_votes[s] for s in states),
```

```
Modeling 51 states with 538 electoral college votes
```

In [3]:
```
stan_code = '''
data {
    int S;  // Number of surveys
    int C;  // Number of candidates
    int survey_results[S, C];  // Number of votes for each candidate in each survey
    real cauchy_location;  // Prior hyperparameters for half-Cauchy over alpha
    real cauchy_scale;
}

parameters {
    vector<lower=0>[C] alpha;
    simplex[C] p[S];  // One probability vector for each survey
}

model {
    alpha ~ cauchy(cauchy_location, cauchy_scale);
    for (i in 1:S) {
        p[i] ~ dirichlet(alpha);
        survey_results[i] ~ multinomial(p[i]);
    }
}
'''
stan_model = pystan.StanModel(model_code=stan_code)
```

```
INFO:pystan:COMPILING THE C++ CODE FOR MODEL anon_model_a89b9d288c9c4fecf50d2e6275e90c2e NOW.
```

In [4]:
```
# Sample results for all states
```

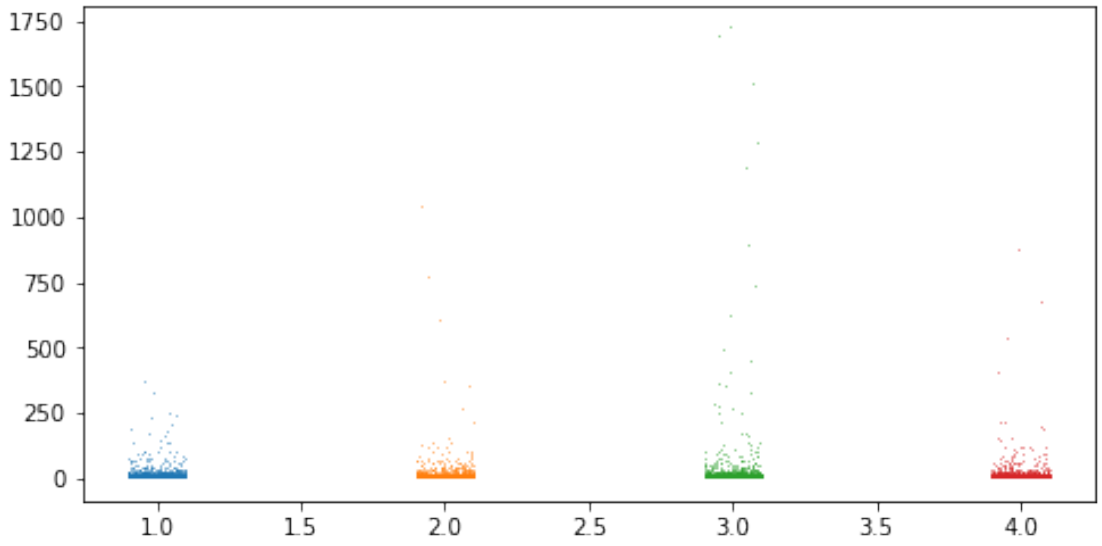
```
print('Posterior  samples and 95% confidence intervals for each state\n')

results = {}
for state in states:
    data = {
        'S': survey_results[state].shape[0],
        'C': 4,
        'survey_results': survey_results[state],
        'cauchy_location': 0,
        'cauchy_scale': 1}
    results[state] = stan_model.sampling(data=data)
    samples = results[state].extract()

    print(state)
    print(np.percentile(samples['alpha'], [2.5, 97.5], axis=0))
    plt.figure(figsize=(8,4))
    for i in range(4):
        plt.plot(stats.uniform.rvs(loc=i+1-0.1, scale=0.2, size=4000), samples['alpha']
    plt.show()
```

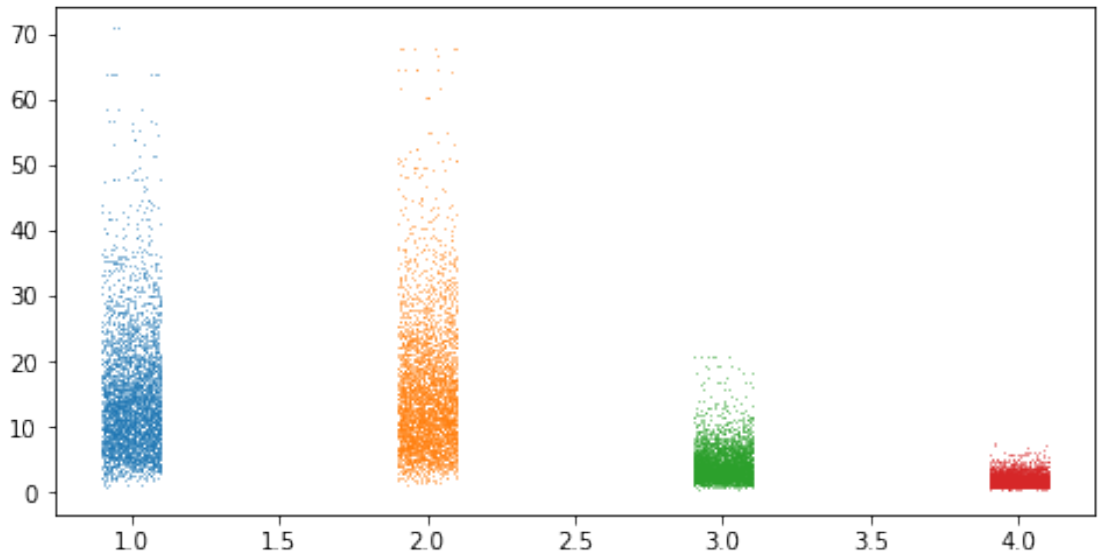
```
Posterior  samples and 95% confidence intervals for each state

Alabama
[[ 0.04552129  0.04159969  0.03581599  0.04365394]
 [22.02612964 24.53545759 29.07338238 20.99180703]]
```

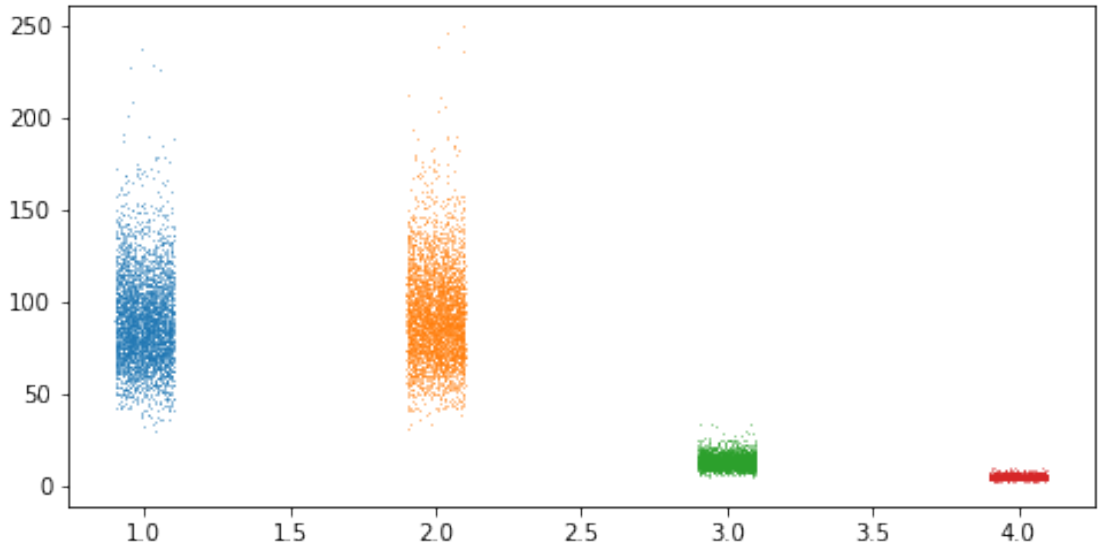



```
WARNING:pystan:4 of 4000 iterations ended with a divergence (0.1 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Alaska
[[ 3.10445592  3.3195319   0.96105629  0.44528596]
 [36.31680227 38.30405667 10.3917121   3.84869898]]
```

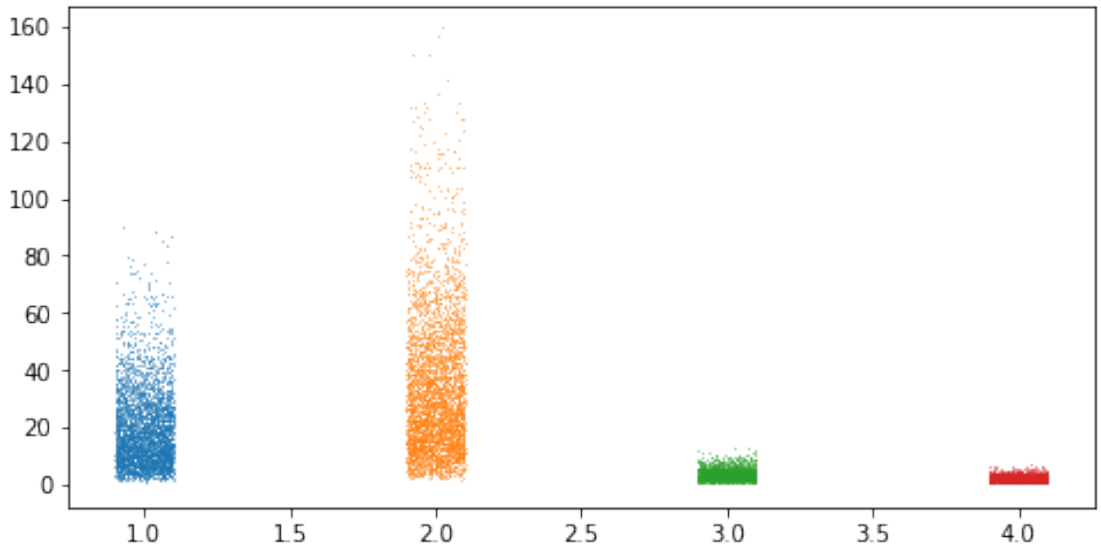



```
Arizona
[[ 49.06207848  51.44879017   6.55407642   1.9175871 ]
 [141.32400241 146.32183992  19.25651578   5.71397806]]
```

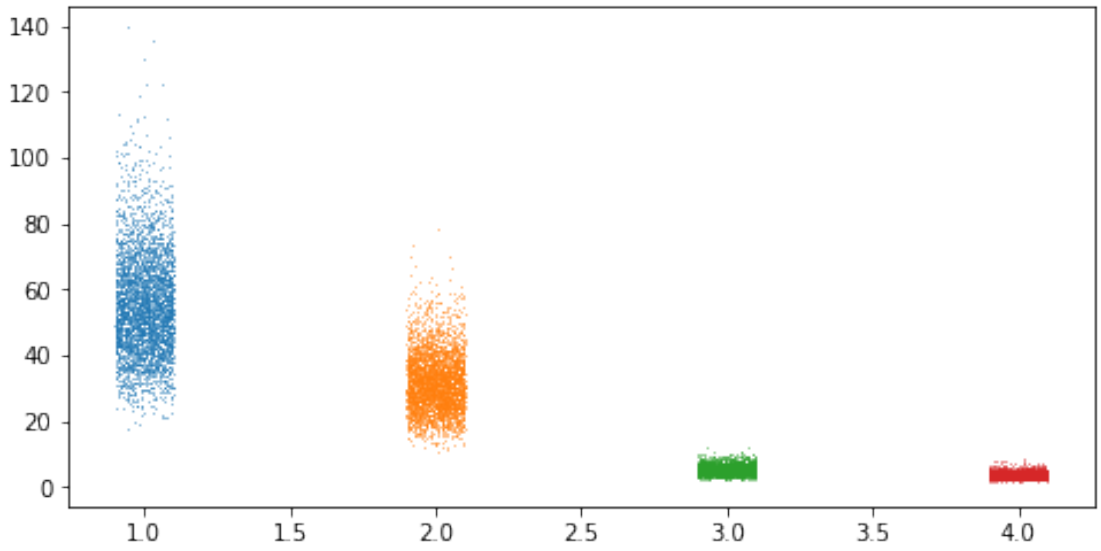


```
WARNING:pystan:24 of 4000 iterations ended with a divergence (0.6 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

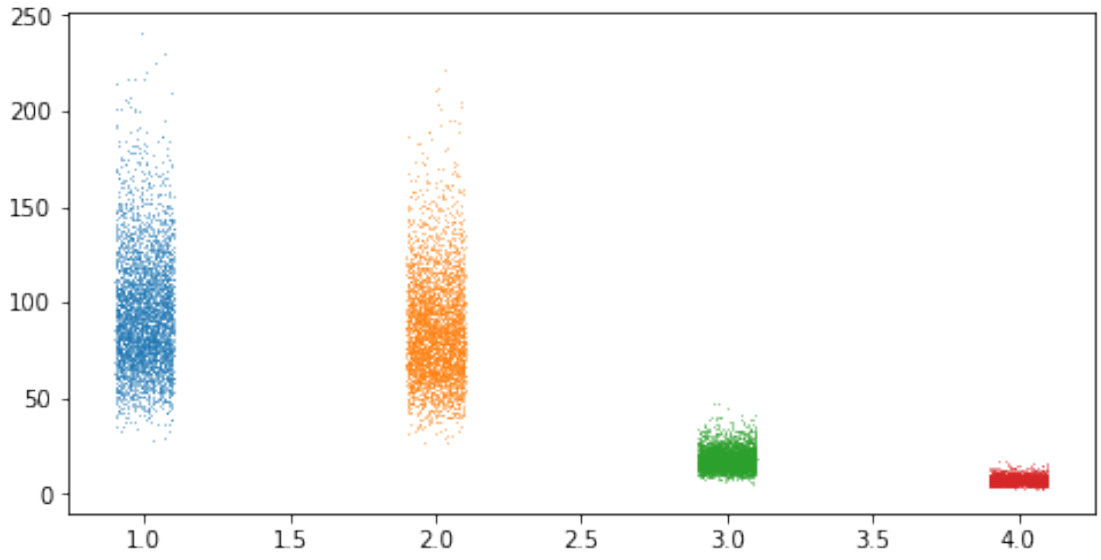
```
Arkansas
[[ 2.38054824  4.11534323  0.44838565  0.31734287]
 [52.26380095 91.17927265  6.92614309  3.49335928]]
```




```
California
[[29.59305969 17.09342303  2.48676108  1.61356474]
 [87.99094145 50.46919586  7.64177378  4.92356348]]
```

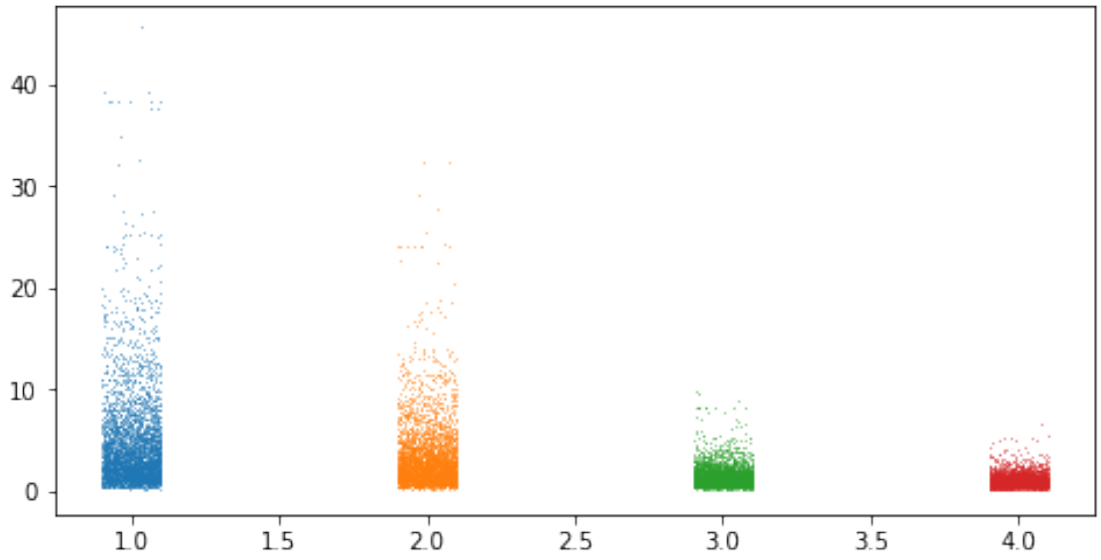


```
Colorado
[[ 48.08543187  43.87947667   8.69726229   3.08441528]
 [161.46554112 147.18362857  29.79416077  10.13537687]]
```



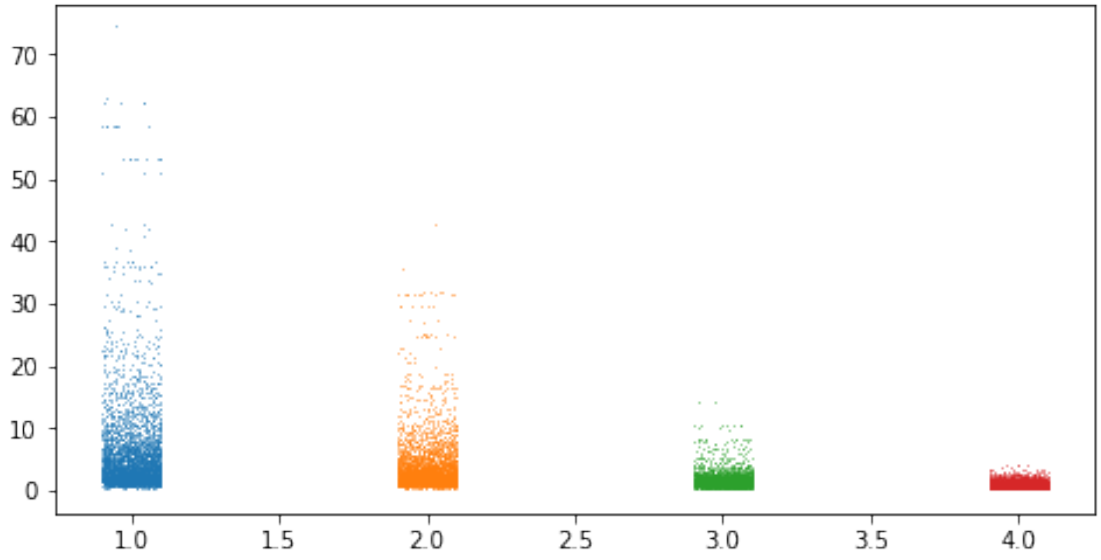

```
WARNING:pystan:2 of 4000 iterations ended with a divergence (0.05 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Connecticut
[[ 0.37239881  0.2891285   0.14364978  0.10290417]
 [16.60413262 10.88146385  3.67780758  2.3237785 ]]
```

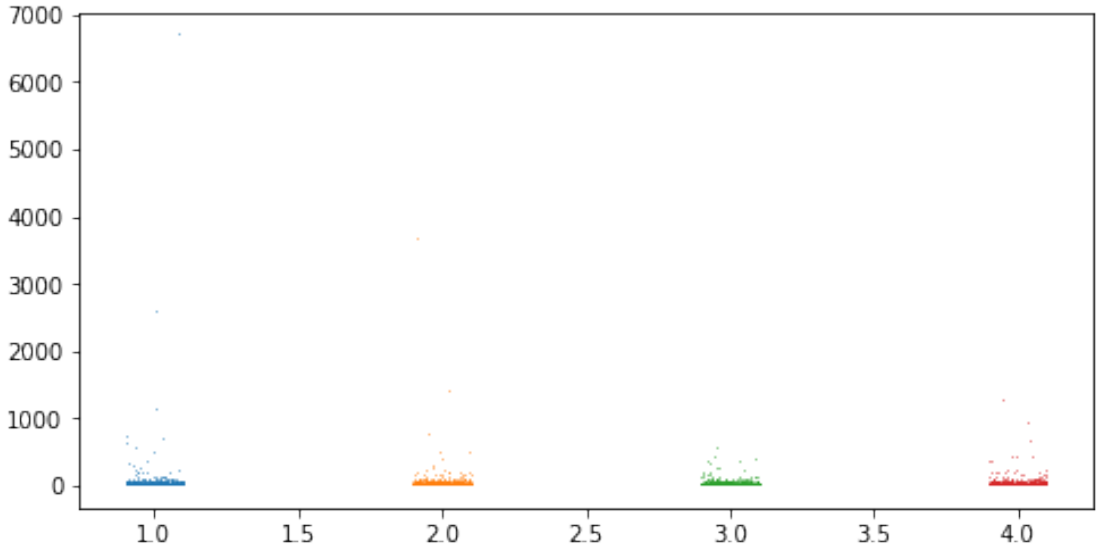



```
WARNING:pystan:33 of 4000 iterations ended with a divergence (0.825 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

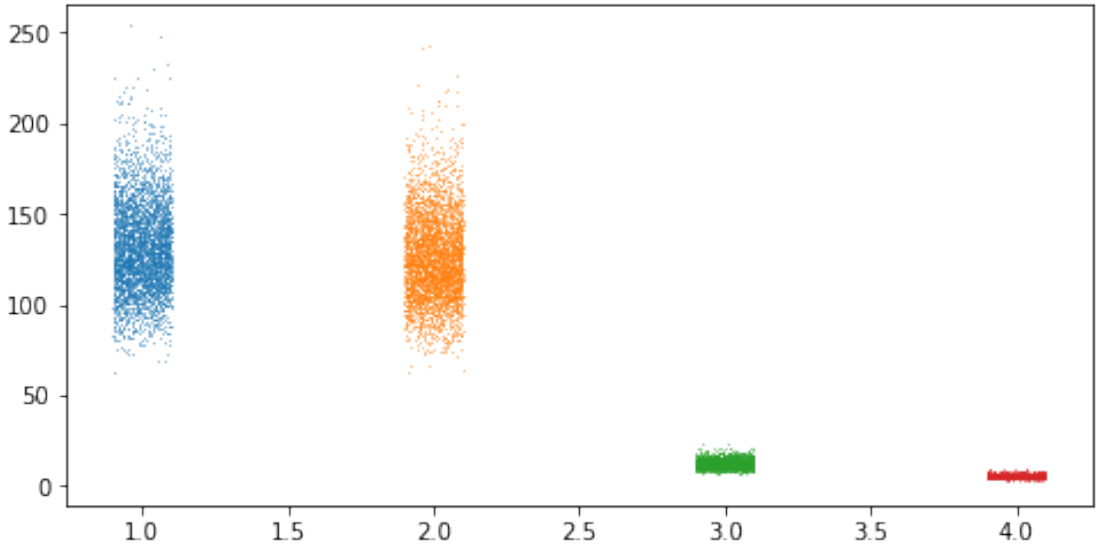
```
Delaware
[[ 0.38399608  0.26249106  0.13217618  0.08986722]
 [23.72769868 14.42121576  4.04848175  1.86975432]]
```



```
District of Columbia
[[ 0.03590147  0.04333459  0.04184546  0.03824849]
 [25.11034243 26.45206963 20.83811867 27.9236798 ]]
```

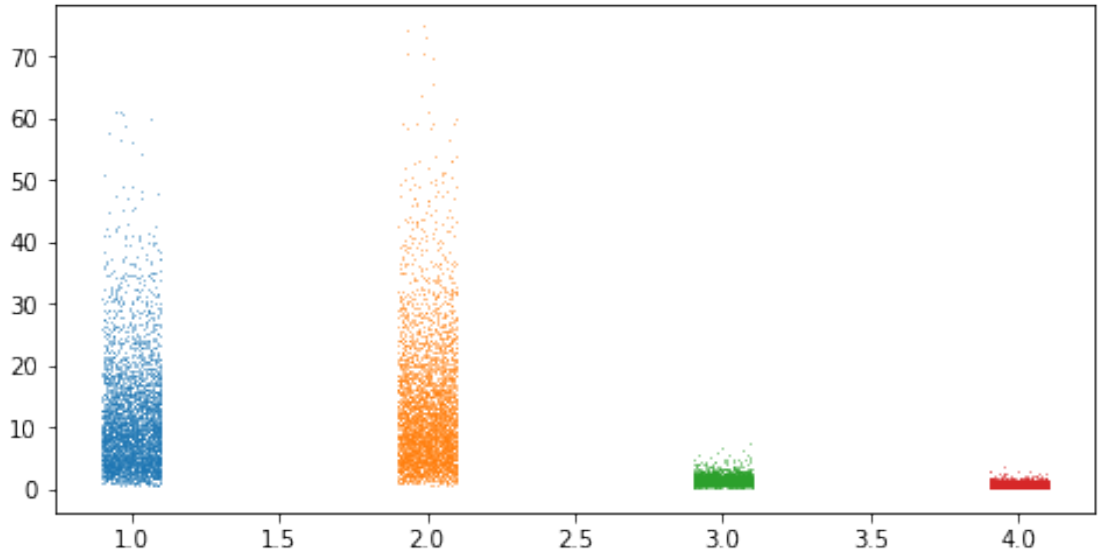



```
Florida
[[ 87.79643229  84.95080084   7.61913547   2.82914217]
 [182.9782196  175.29481991  15.85255596   6.02654406]]
```

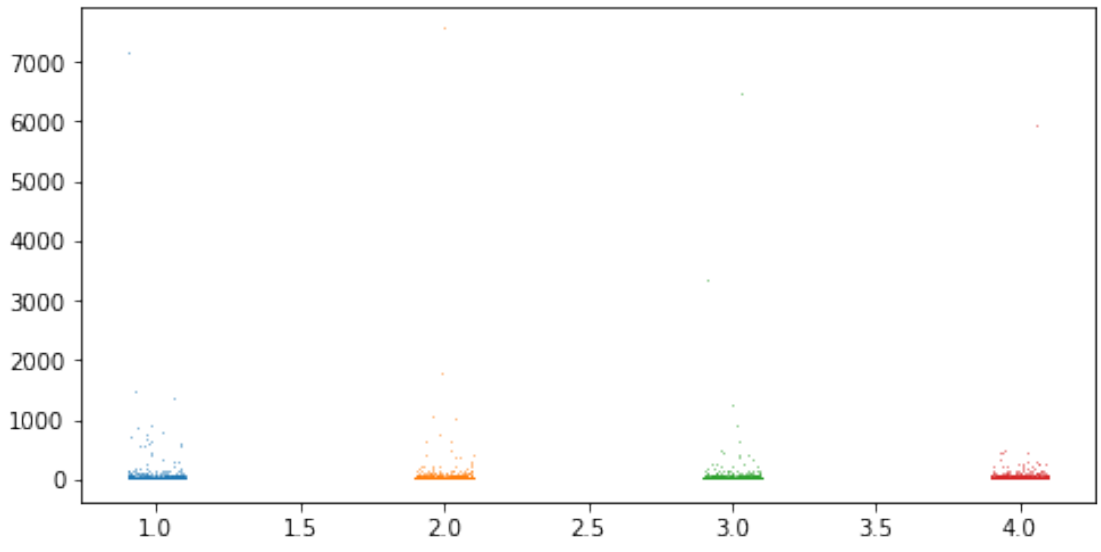


```
WARNING:pystan:12 of 4000 iterations ended with a divergence (0.3 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Georgia
[[ 1.39105631  1.59548047  0.24288649  0.10126778]
 [32.63192461 38.72673206  2.78369006  1.27481451]]
```

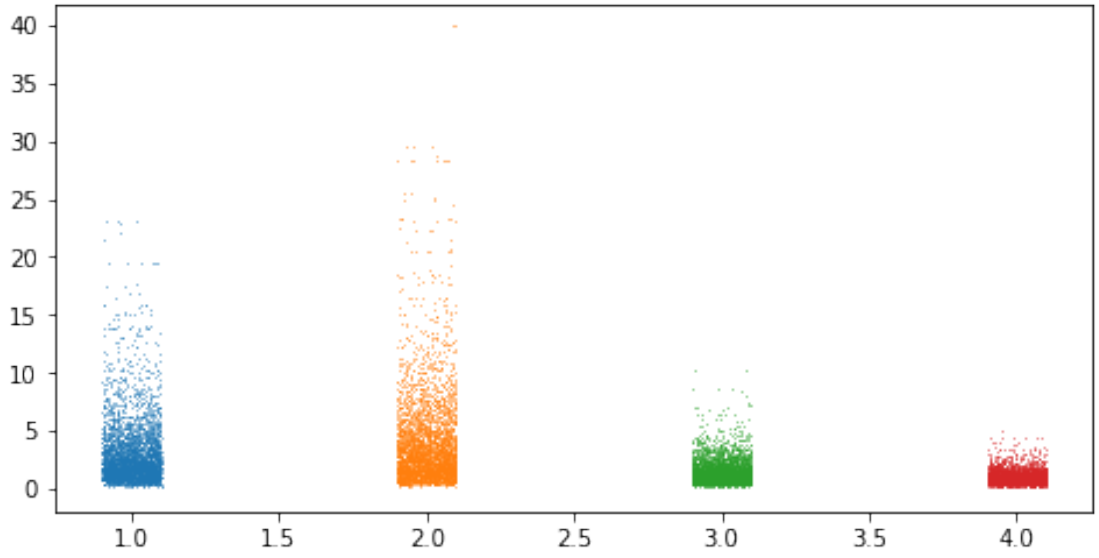



```
Hawaii
[[3.95841663e-02 3.68346394e-02 4.00837799e-02 2.75750191e-02]
 [2.63960997e+01 2.57822773e+01 2.24535635e+01 3.45070635e+01]]
```

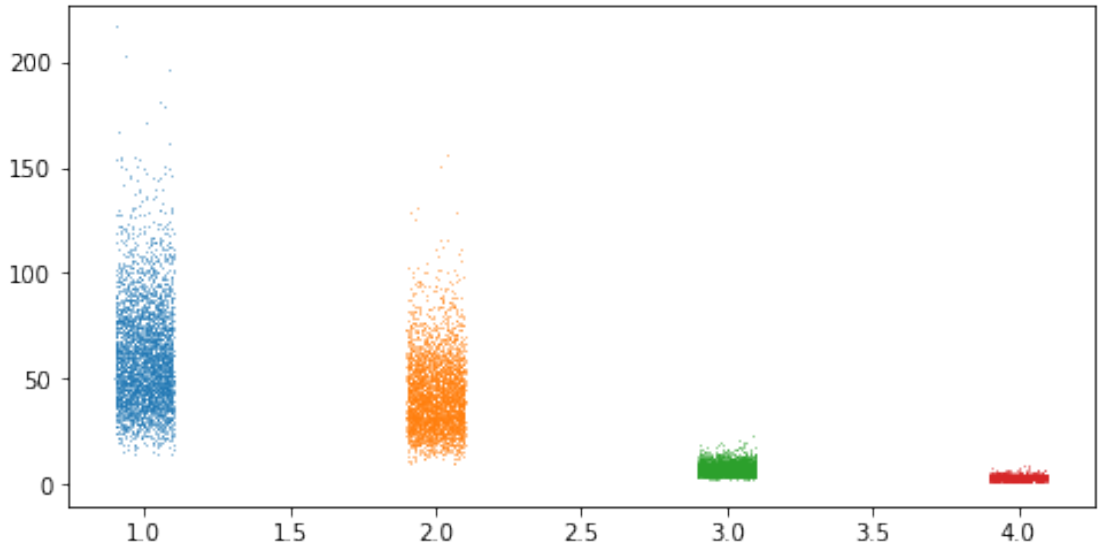


```
WARNING:pystan:7 of 4000 iterations ended with a divergence (0.175 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Idaho
[[ 0.27541501  0.32876759  0.13700278  0.09941157]
 [10.26863571 13.76572167  3.84671593  2.2246579 ]]
```

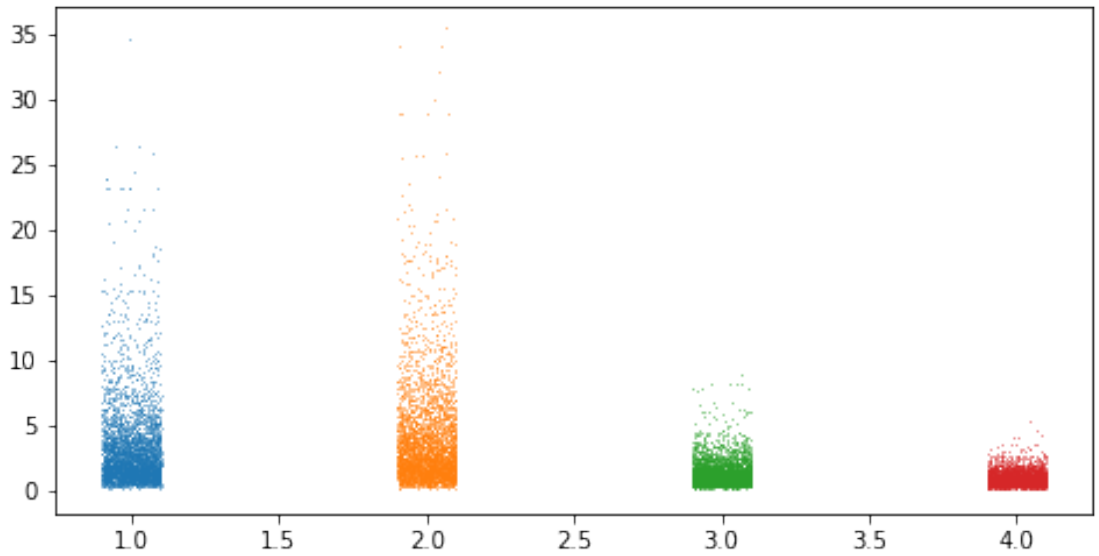



```
Illinois
[[ 24.23858002  16.98500041   2.56521626   0.78554429]
 [115.69482715  81.84921086  12.40963752   4.01624294]]
```

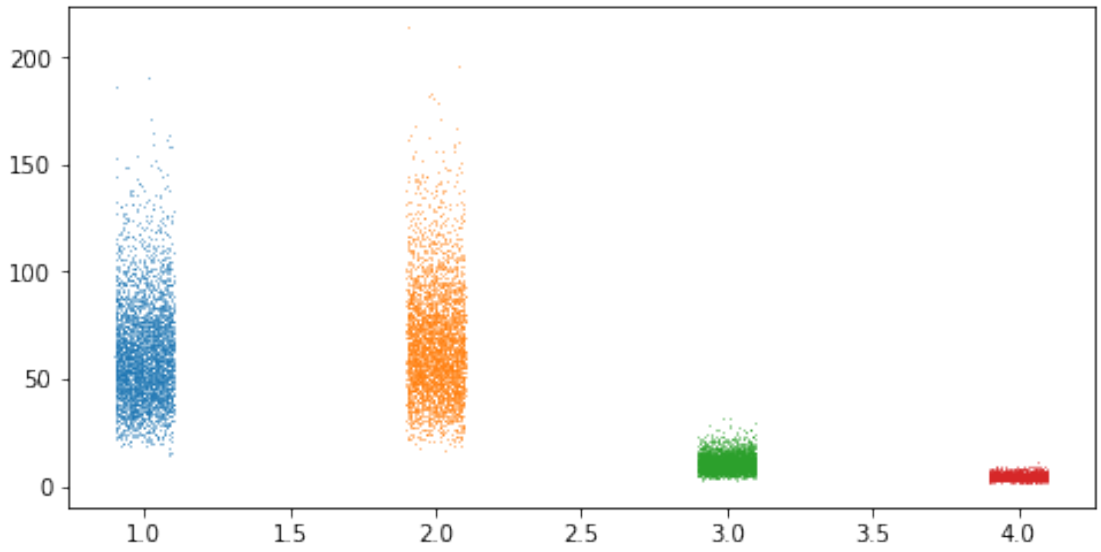



```
WARNING:pystan:4 of 4000 iterations ended with a divergence (0.1 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Indiana
[[ 0.32203206  0.33341565  0.14017555  0.09459214]
 [11.55025069 13.64499177  3.48498192  2.06450516]]
```

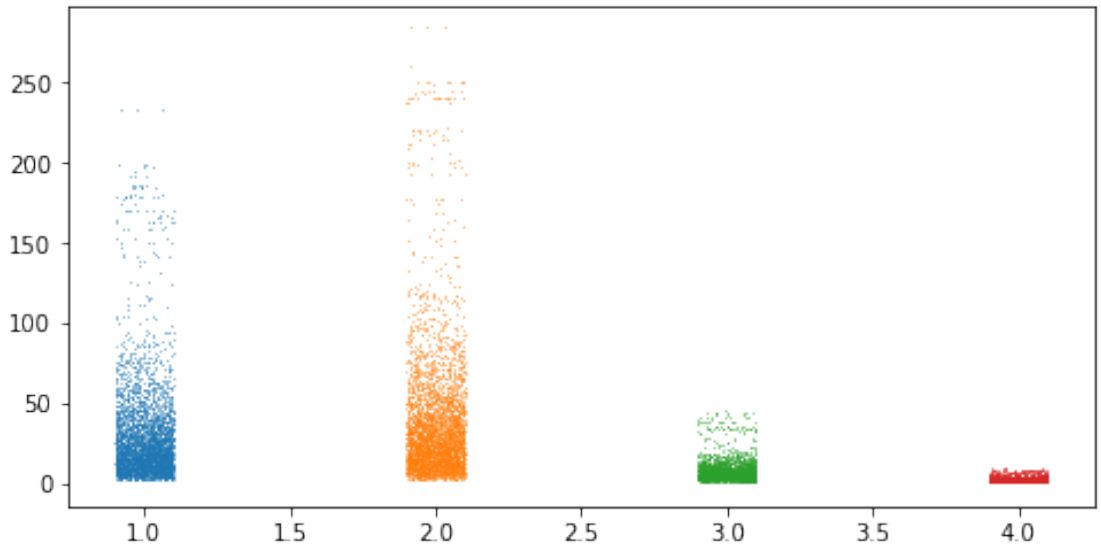


```
Iowa
[[ 26.09902658  28.29086395   4.14776897   1.39593775]
 [116.99031042 125.93615672  19.01007808   6.22546918]]
```



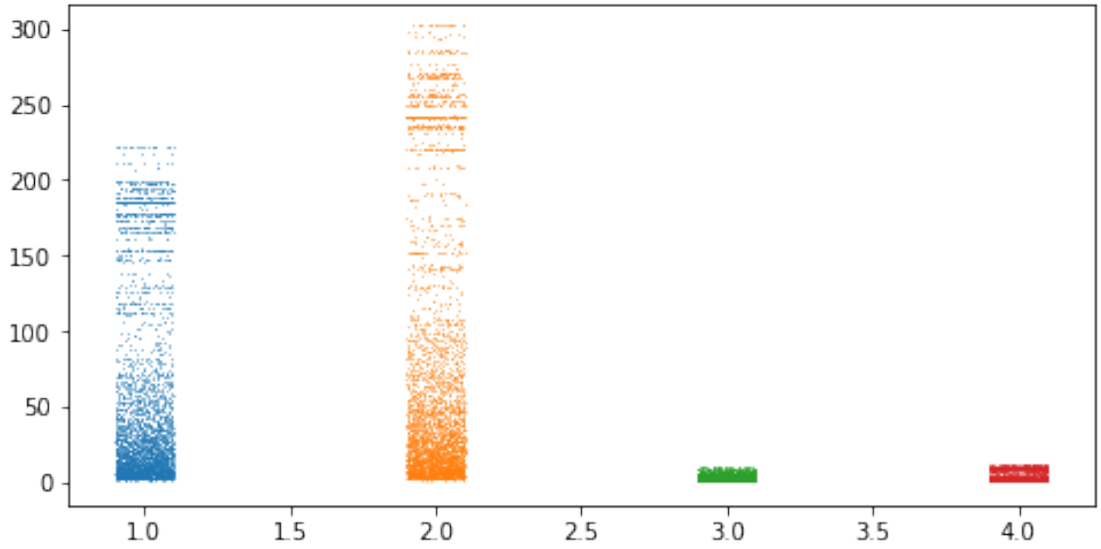

```
WARNING:pystan:82 of 4000 iterations ended with a divergence (2.05 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Kansas
[[  2.6697565    3.57087371   0.66924233   0.28771764]
 [103.34446766 139.7189639   21.73241017   5.02336989]]
```



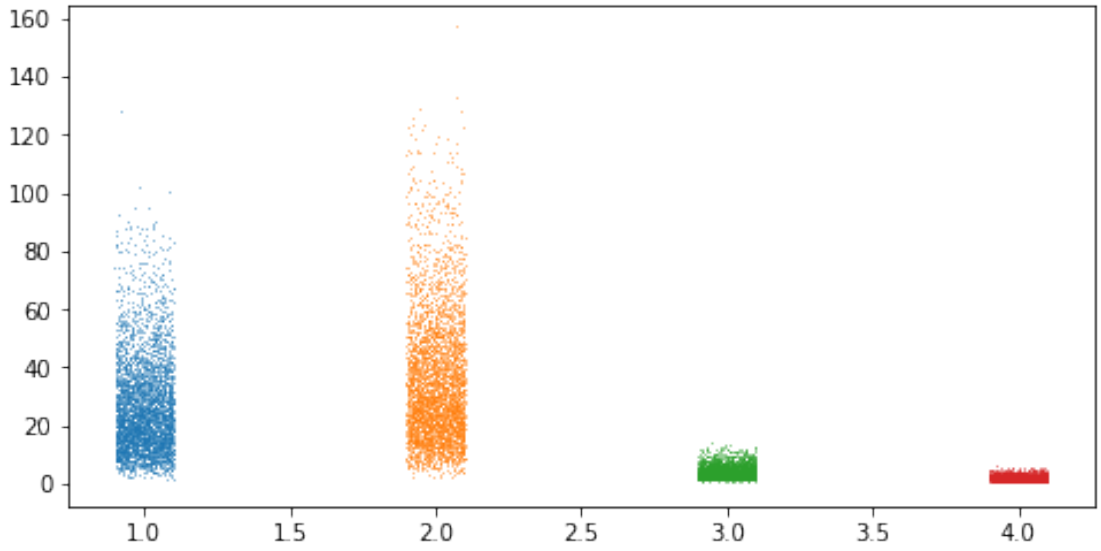
```
WARNING:pystan:n_eff / iter below 0.001 indicates that the effective sample size has likely bee
WARNING:pystan:Rhat above 1.1 or below 0.9 indicates that the chains very likely have not mixed
WARNING:pystan:795 of 4000 iterations ended with a divergence (19.9 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Kentucky
[[1.23647513e+00 1.63376687e+00 1.73507652e-01 1.76545954e-01]
 [1.93426457e+02 2.68674289e+02 7.13324477e+00 9.17744816e+00]]
```



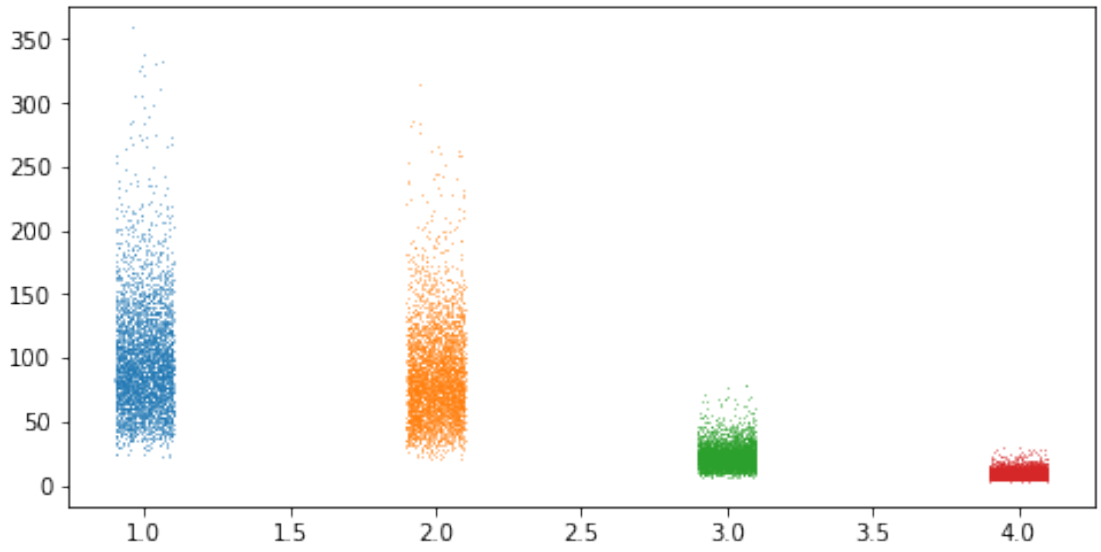

```
WARNING:pystan:21 of 4000 iterations ended with a divergence (0.525 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Louisiana
[[ 5.11319063  7.04786514  0.76669579  0.3680796 ]
 [66.84109109 91.54774659  8.72874409  3.29485536]]
```



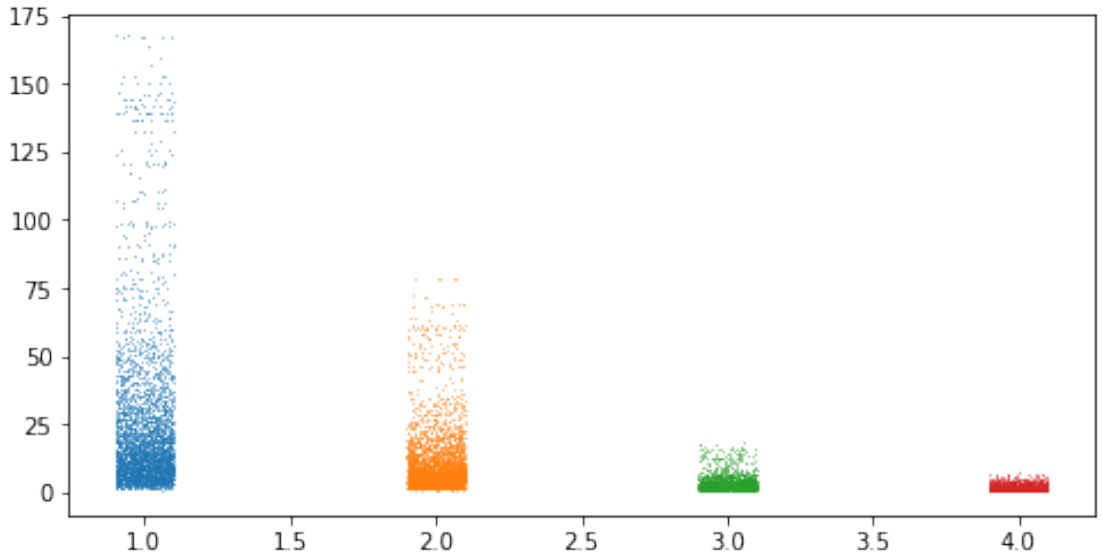

```
WARNING:pystan:3 of 4000 iterations ended with a divergence (0.075 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Maine
[[ 38.22660319  33.32662796   8.20196856   3.10574987]
 [196.24497888 169.13371153  43.04947     15.87877814]]
```



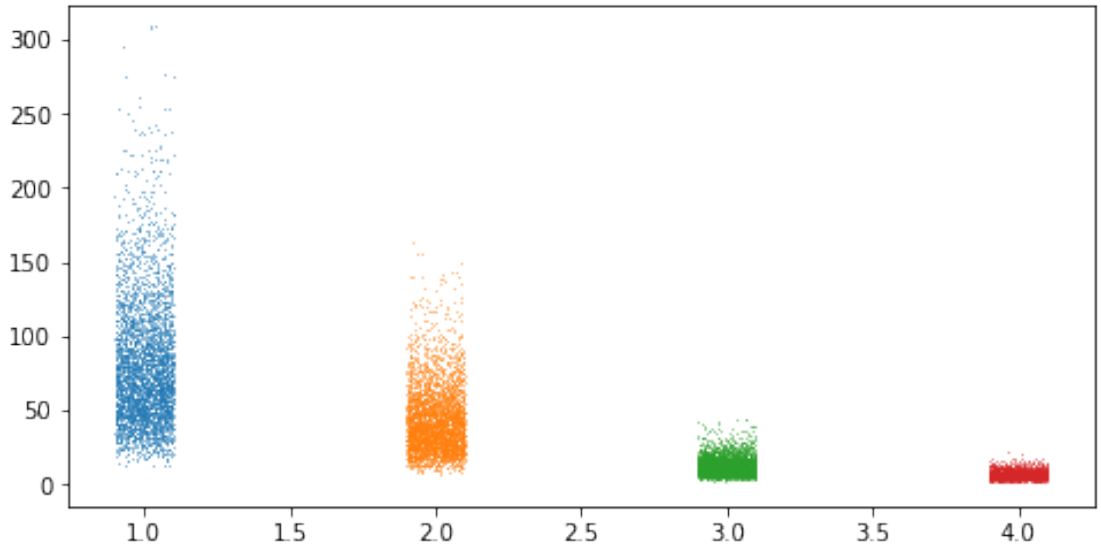
```
WARNING:pystan:Rhat above 1.1 or below 0.9 indicates that the chains very likely have not mixed
WARNING:pystan:124 of 4000 iterations ended with a divergence (3.1 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Maryland
[[  1.7169786    0.79122096   0.27322312   0.18074073]
 [110.06218037  48.68552494   9.24002305   3.4325879 ]]
```

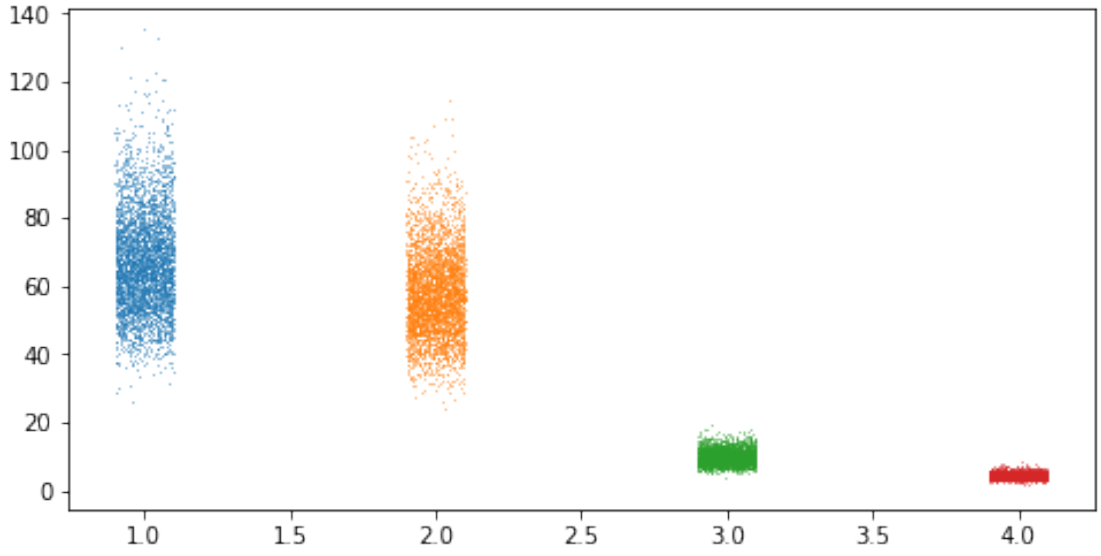



```
WARNING:pystan:2 of 4000 iterations ended with a divergence (0.05 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Massachusetts
[[ 24.12931936  12.83880137   3.4161795    1.50408293]
 [179.00378564  95.61802925  25.24567106  10.83859294]]
```

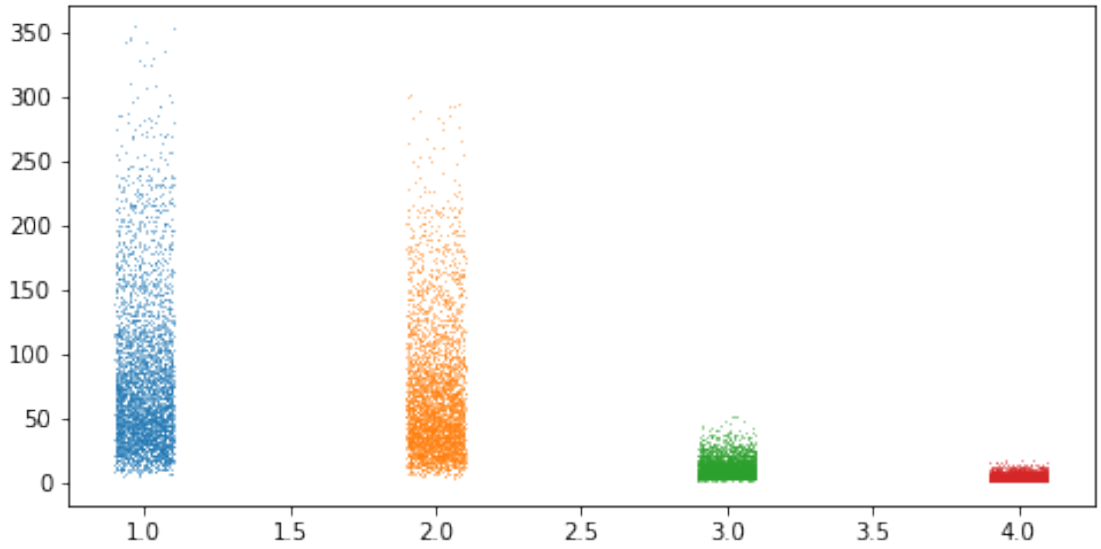



```
Michigan
[[42.5172594  36.30309696  5.75078987  2.32581496]
 [97.08753655 84.21683415 13.61333722  5.37687619]]
```

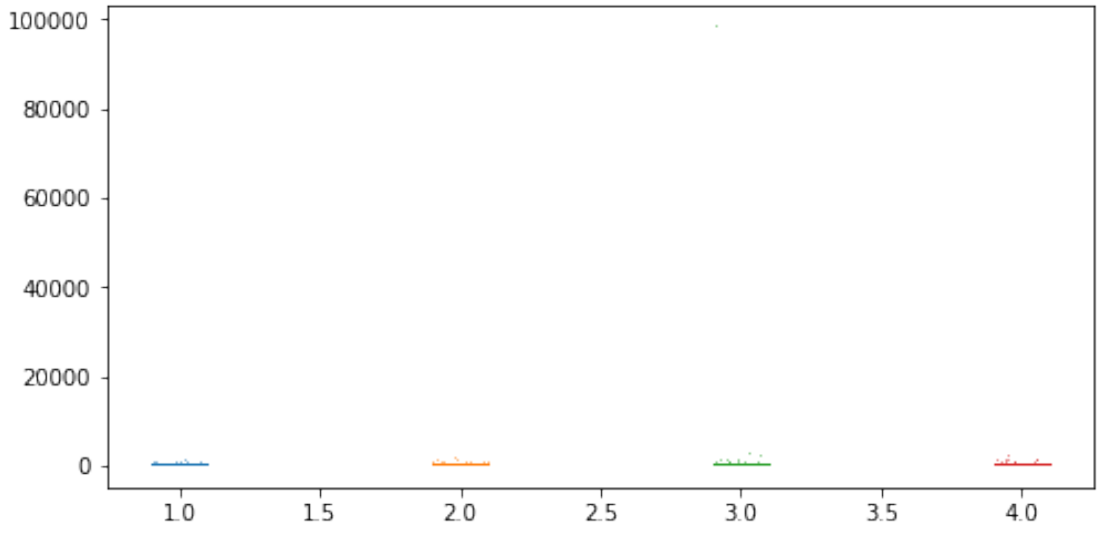



```
WARNING:pystan:51 of 4000 iterations ended with a divergence (1.27 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

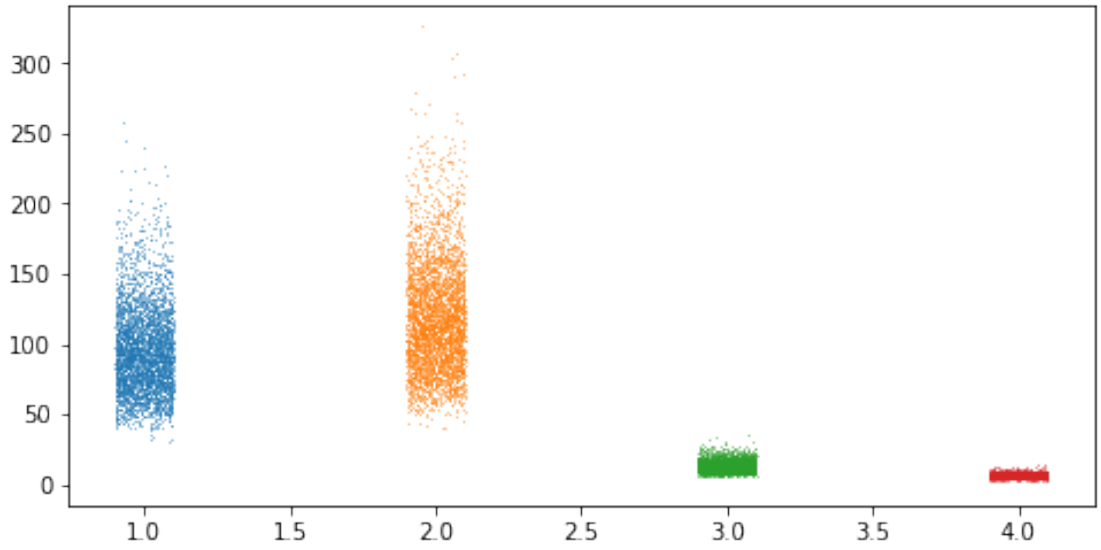
```
Minnesota
[[ 12.77295409  10.74353027   1.59184736   0.62526503]
 [230.20036291 191.80265102  28.50571998   9.18673605]]
```




```
Mississippi
[[ 0.03569364  0.04248256  0.03447347  0.04568398]
 [21.9339737  30.1450285  23.93079315 21.3823139 ]]
```

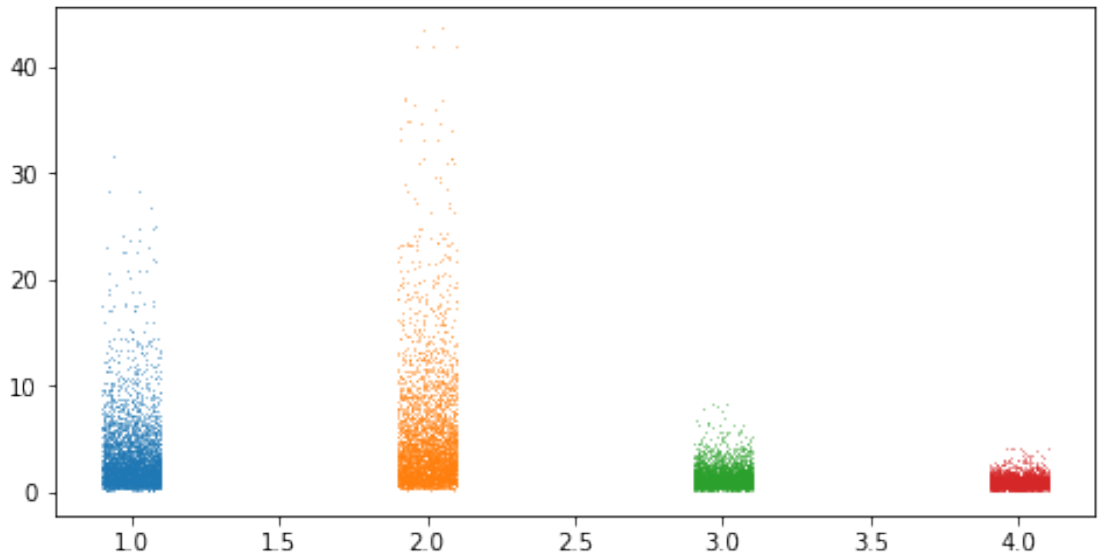


```
Missouri
[[ 48.10153028  59.69871183   5.8675095    2.28276374]
 [165.4365027  205.99045073  20.48550095   7.95315767]]
```



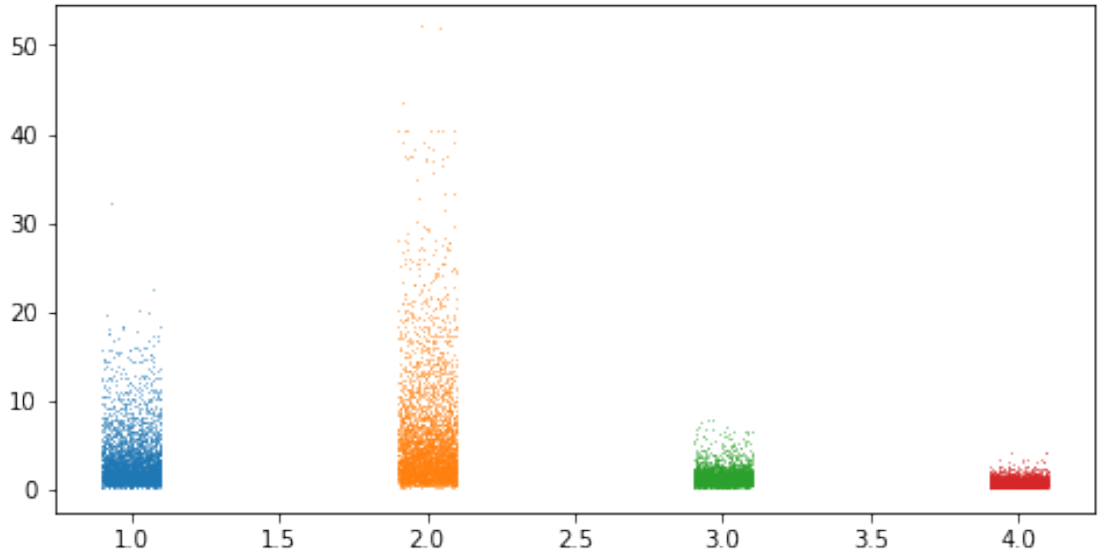

```
WARNING:pystan:14 of 4000 iterations ended with a divergence (0.35 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Montana
[[ 0.26134707  0.37317136  0.13531398  0.08975695]
 [11.70383323 19.19657278  3.63198862  1.95629493]]
```

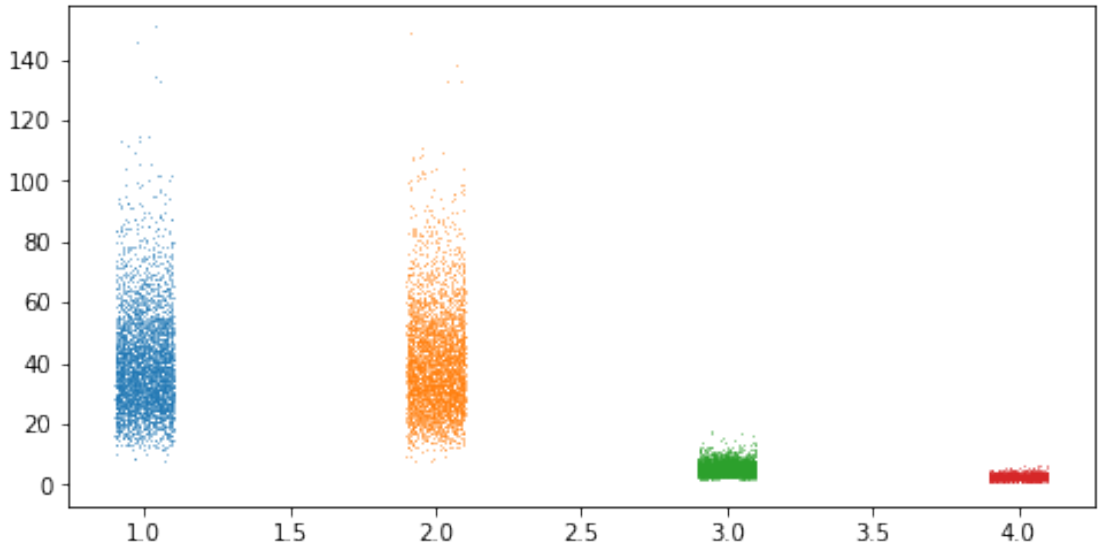


```
WARNING:pystan:9 of 4000 iterations ended with a divergence (0.225 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

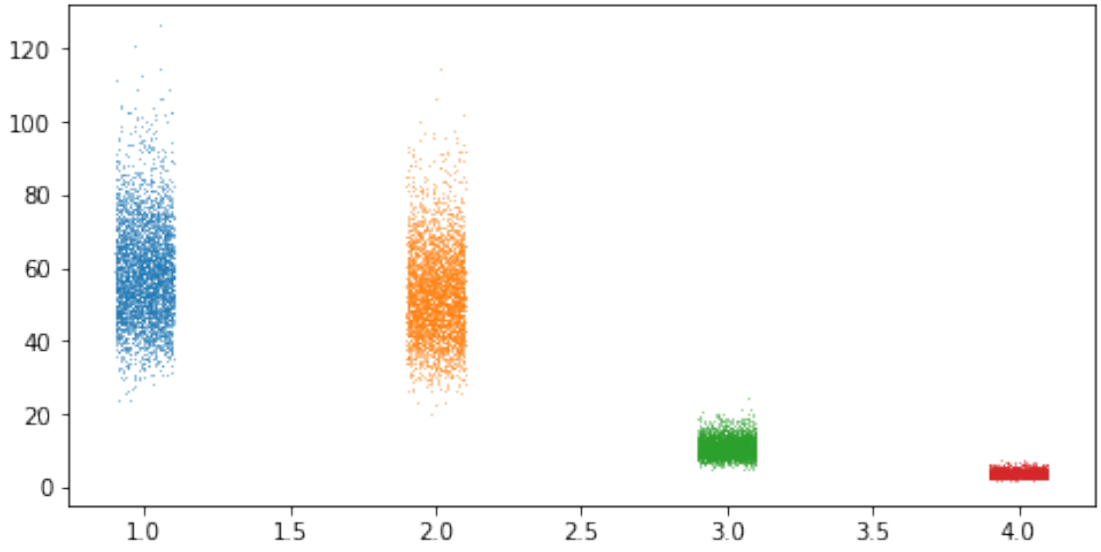
```
Nebraska
[[ 0.24638858  0.4185181   0.12740634  0.07434623]
 [11.52707993 21.88731452  3.54086501  1.6467466 ]]
```




```
Nevada
[[15.65765525 15.80077864  1.83461265  0.6290719 ]
 [78.07286299 79.85182687  9.45383704  3.26357351]]
```

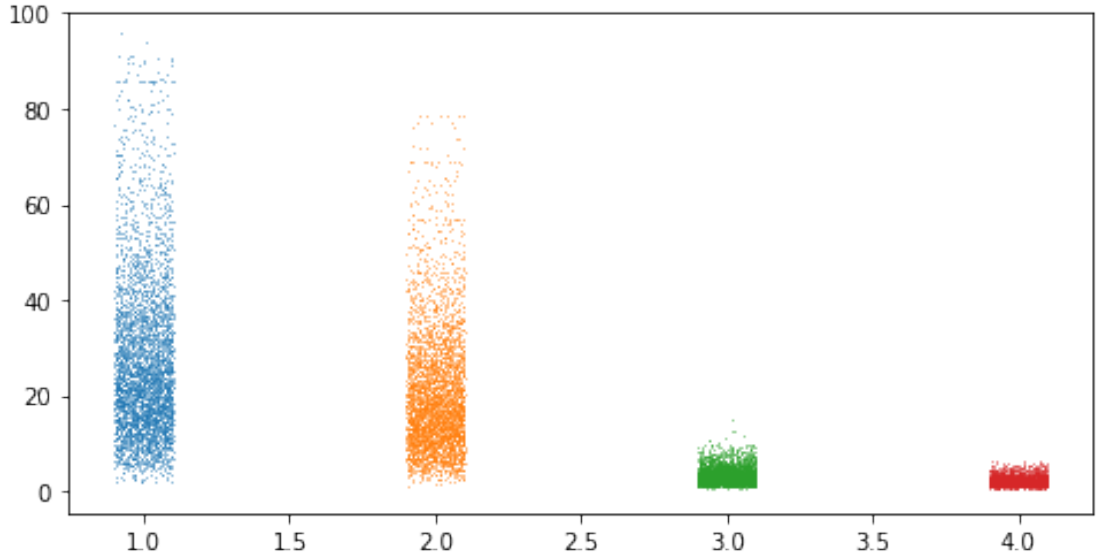



```
New Hampshire
[[36.36661269 32.2170279   6.49296118  1.98416954]
 [86.25167172 78.45306927 15.75428341  4.94535739]]
```

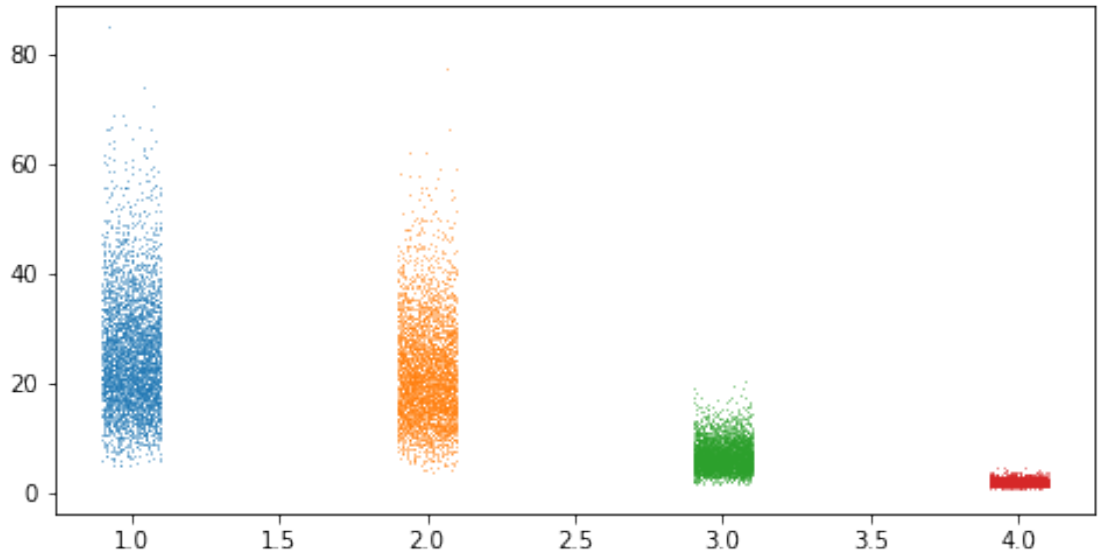



```
WARNING:pystan:28 of 4000 iterations ended with a divergence (0.7 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
New Jersey
[[ 5.52134612  4.09827708  0.6977142   0.44267282]
 [71.11411928 54.00499447  7.16949158  4.05878302]]
```

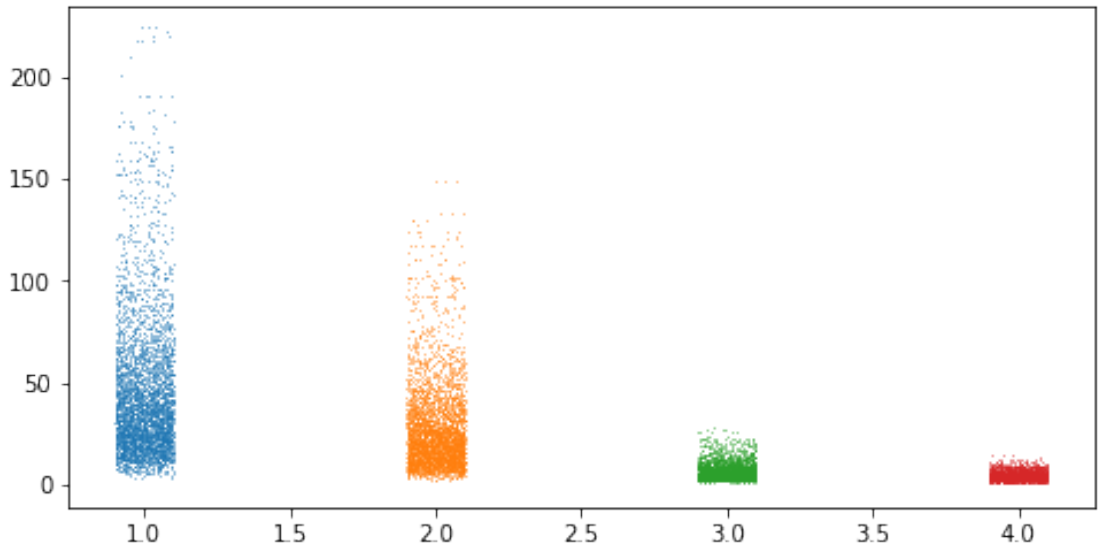



```
New Mexico
[[ 9.01522528  7.84371257  2.32630095  0.54930785]
 [49.12978965 42.50642788 12.48806326  2.54174607]]
```



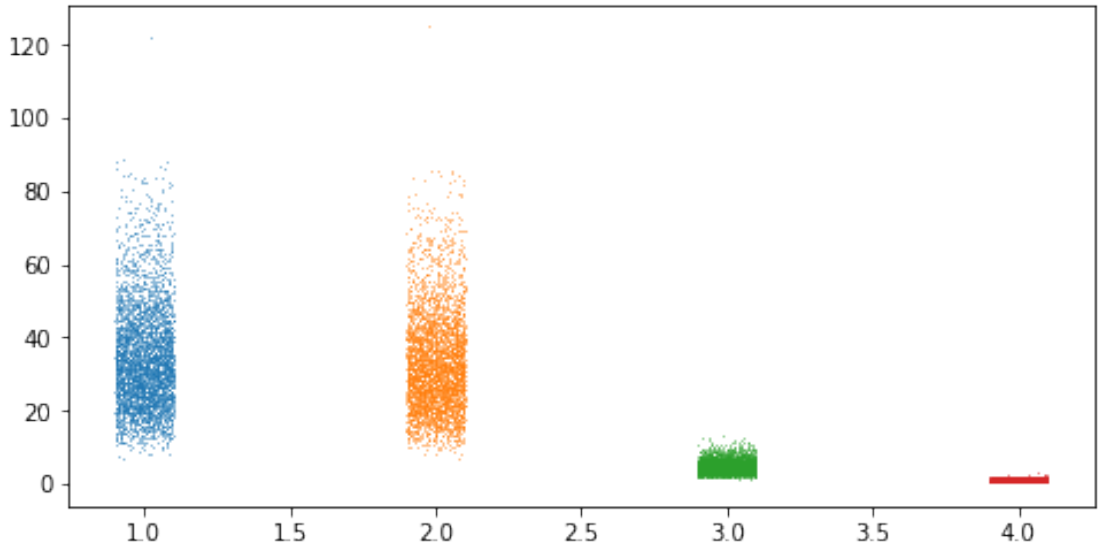
```
WARNING:pystan:77 of 4000 iterations ended with a divergence (1.93 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
New York
[[  8.23602907   4.83526263   0.99755124   0.63840628]
 [141.37045736  88.05191241  16.21210281   8.24230205]]
```



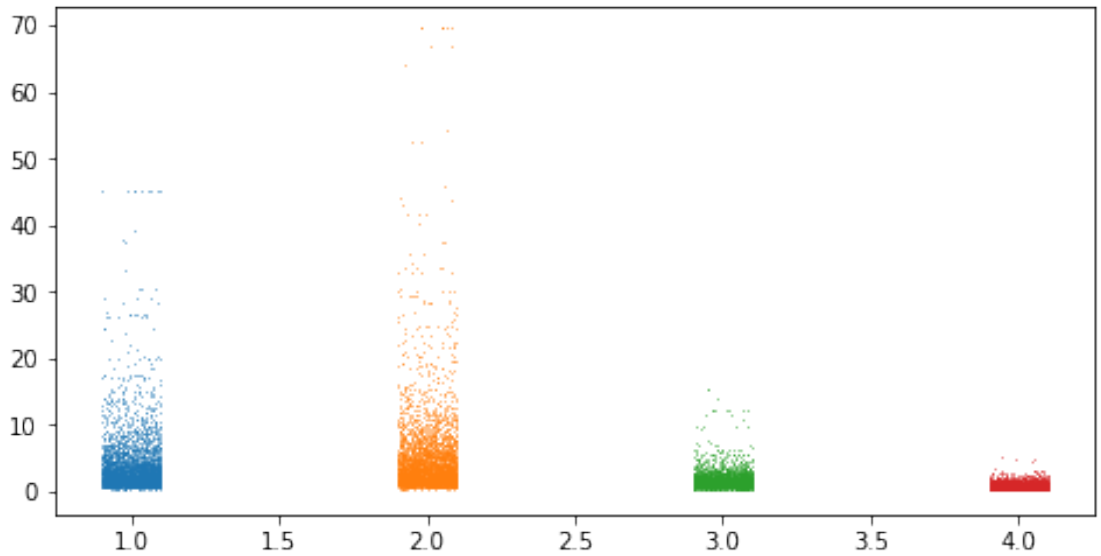

```
WARNING:pystan:5 of 4000 iterations ended with a divergence (0.125 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
North Carolina
[[13.3328549  12.92583159  1.6300671   0.19015715]
 [67.56996193 64.53812747  8.29026871  1.14952887]]
```

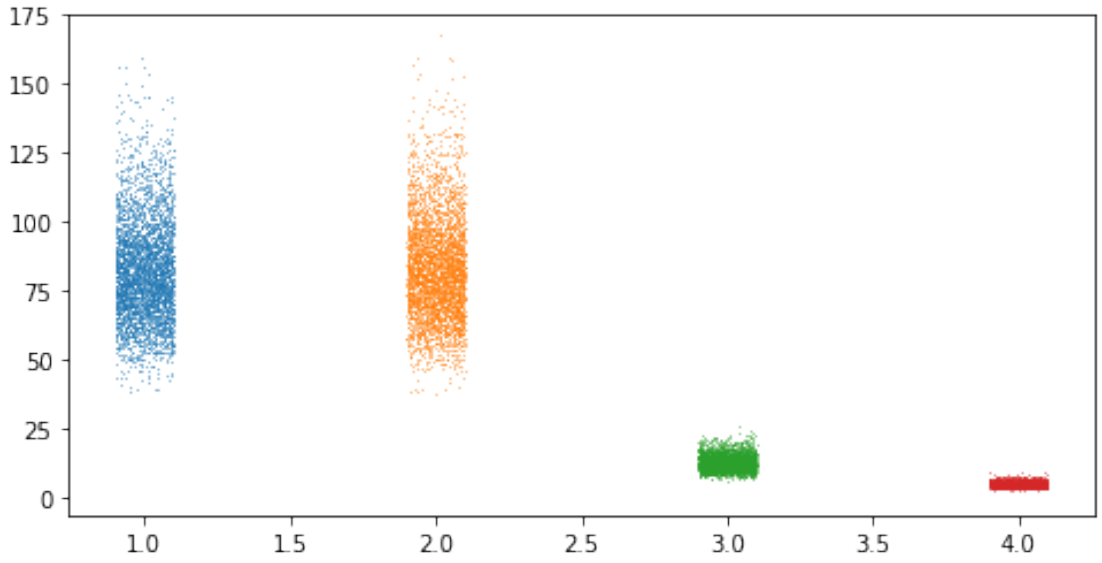



```
WARNING:pystan:17 of 4000 iterations ended with a divergence (0.425 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

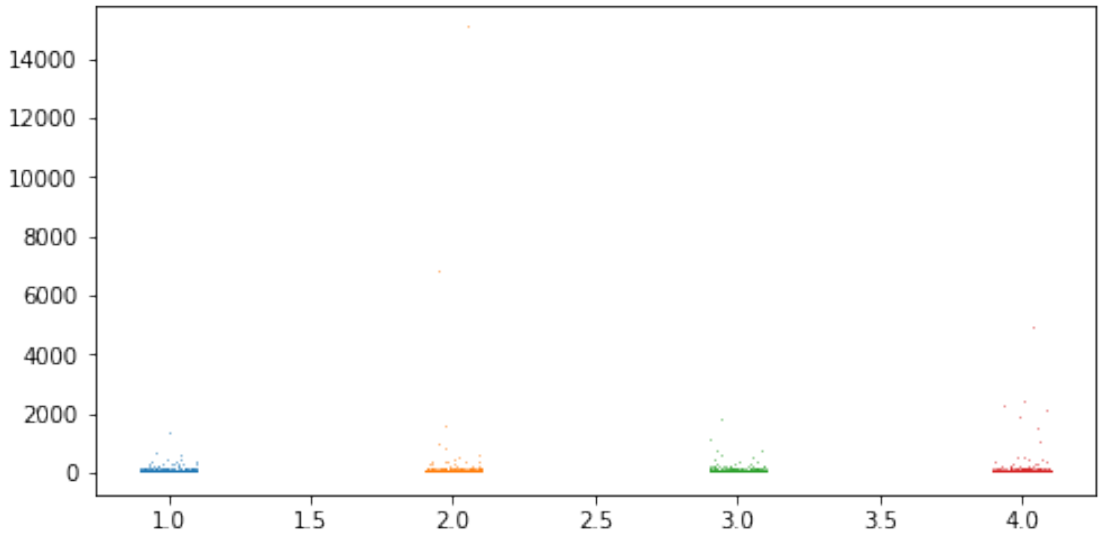
```
North Dakota
[[ 0.33335492  0.37206292  0.15131156  0.07968454]
 [14.10829555 20.02389445  4.09935927  1.6479846 ]]
```



```
Ohio
[[ 51.72637126  52.51180447   7.29001512   2.3605492 ]
 [122.00952787 123.688051    17.92266118   5.72246143]]
```

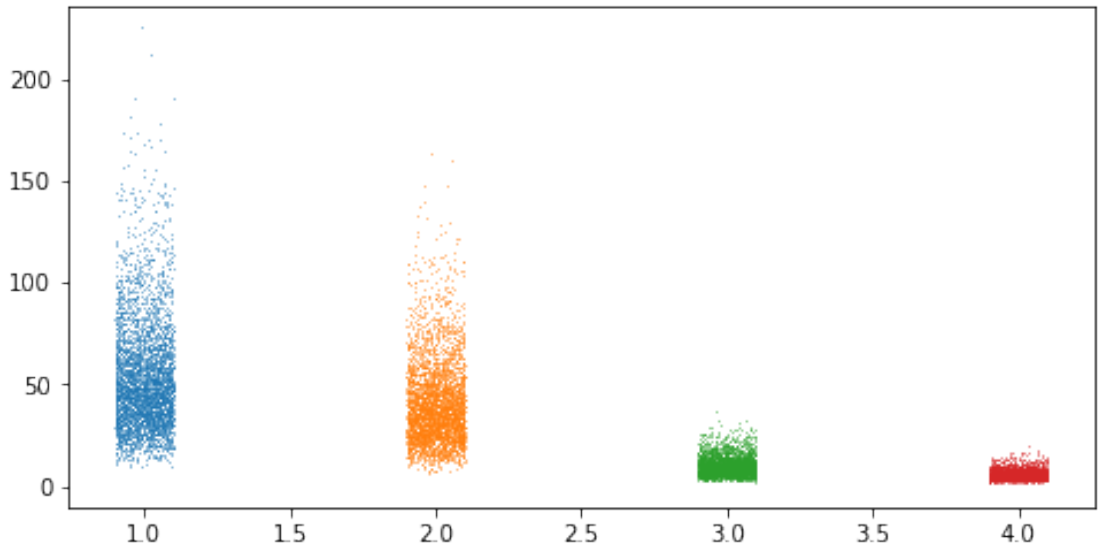



```
Oklahoma
[[ 0.0467213   0.0476332   0.03909465  0.04544567]
 [27.26317229 24.86120252 25.72784919 26.11515385]]
```

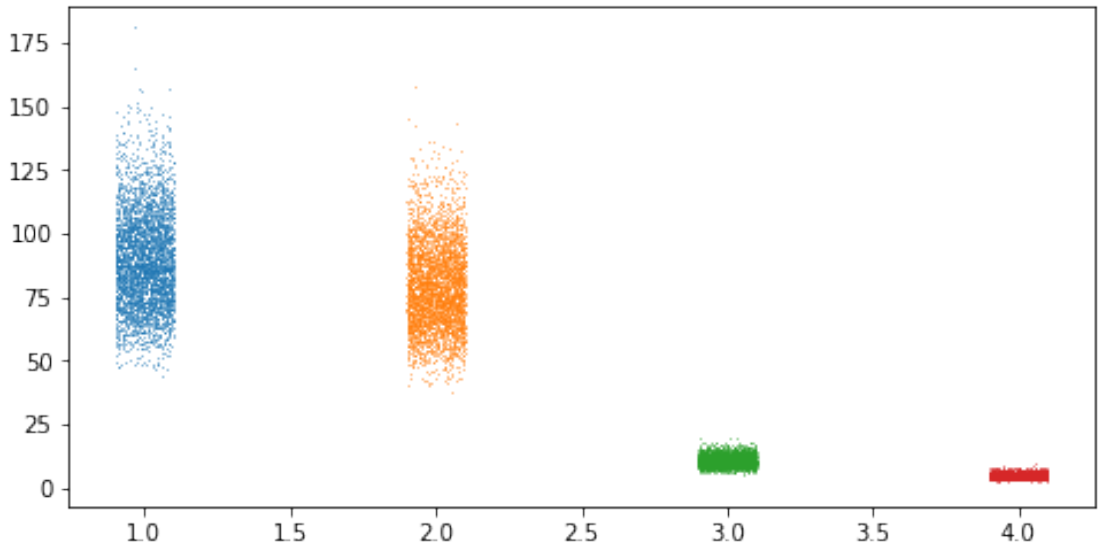


```
WARNING:pystan:1 of 4000 iterations ended with a divergence (0.025 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Oregon
[[ 17.37185473  13.02080772   2.82458303   1.50060529]
 [118.77160154  90.21772176  19.8667095   10.1587027 ]]
```

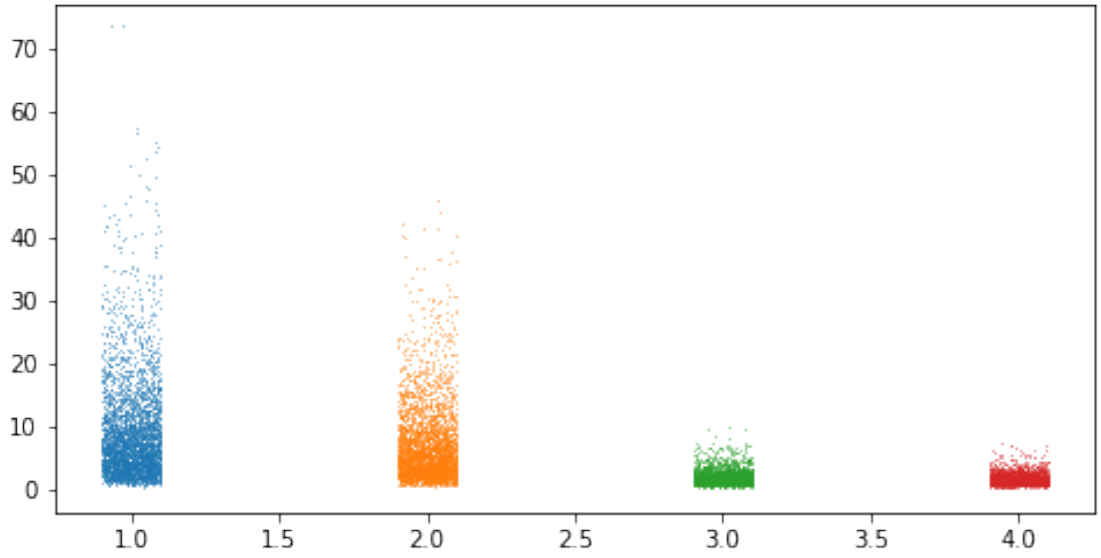



```
Pennsylvania
[[ 57.29667264  50.83856776   6.48272051   2.63363872]
 [128.84645059 114.19479989  14.52624645   6.10660464]]
```



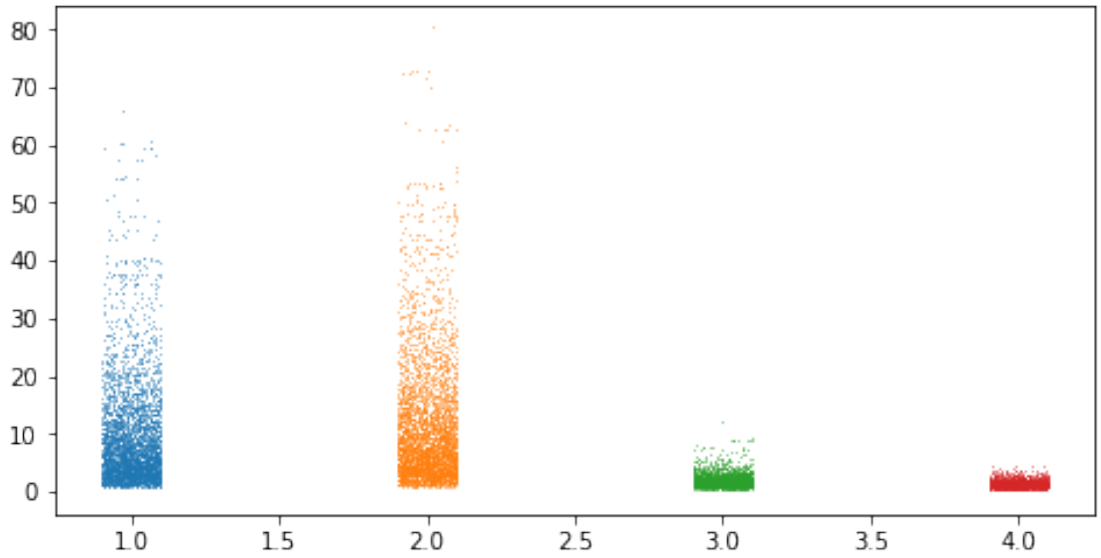
```
WARNING:pystan:13 of 4000 iterations ended with a divergence (0.325 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
Rhode Island
[[ 0.98936838  0.74756634  0.27270091  0.23539074]
 [28.96642027 22.55895744  4.3053249   3.31029545]]
```

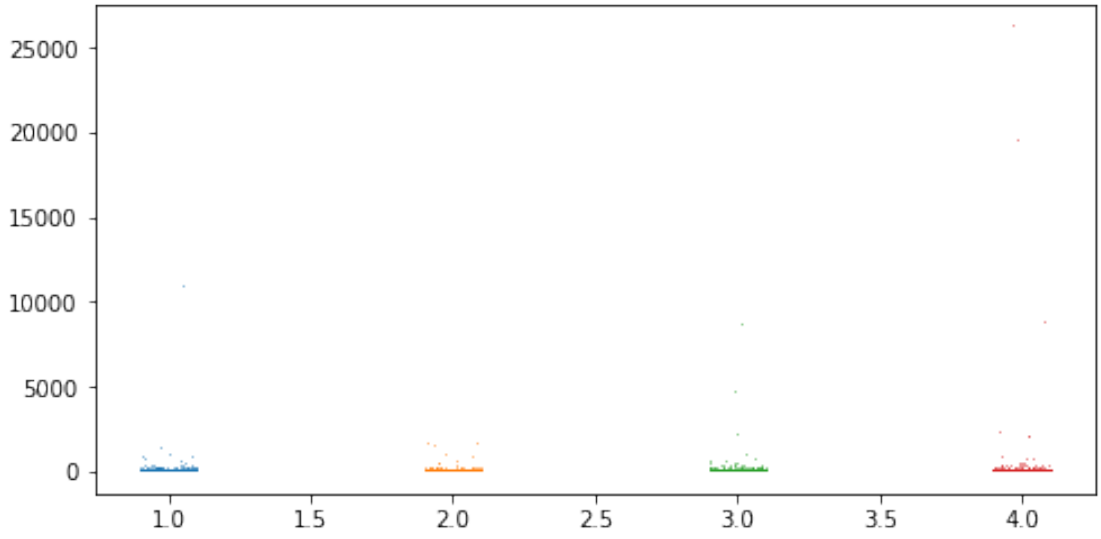



```
WARNING:pystan:23 of 4000 iterations ended with a divergence (0.575 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

```
South Carolina
[[ 0.84320193  1.00166753  0.21665969  0.15887607]
 [34.62928045 41.54857695  3.8793247   2.19856573]]
```

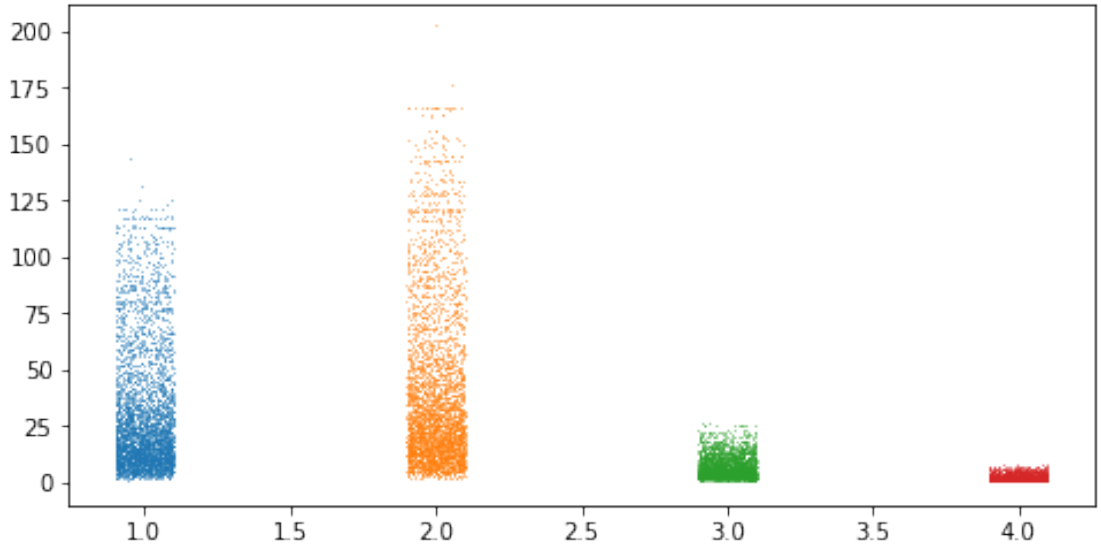



```
South Dakota
[[ 0.03430097  0.04555082  0.04040387  0.03904226]
 [26.46290922 21.37945027 25.88501266 26.1544208 ]]
```

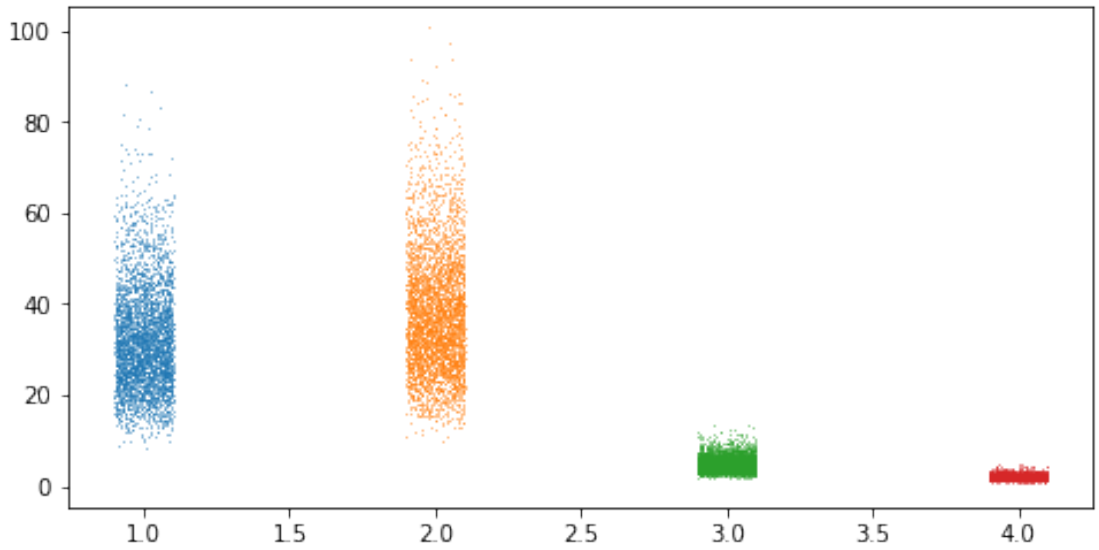



```
WARNING:pystan:Rhat above 1.1 or below 0.9 indicates that the chains very likely have not mixed
WARNING:pystan:294 of 4000 iterations ended with a divergence (7.35 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

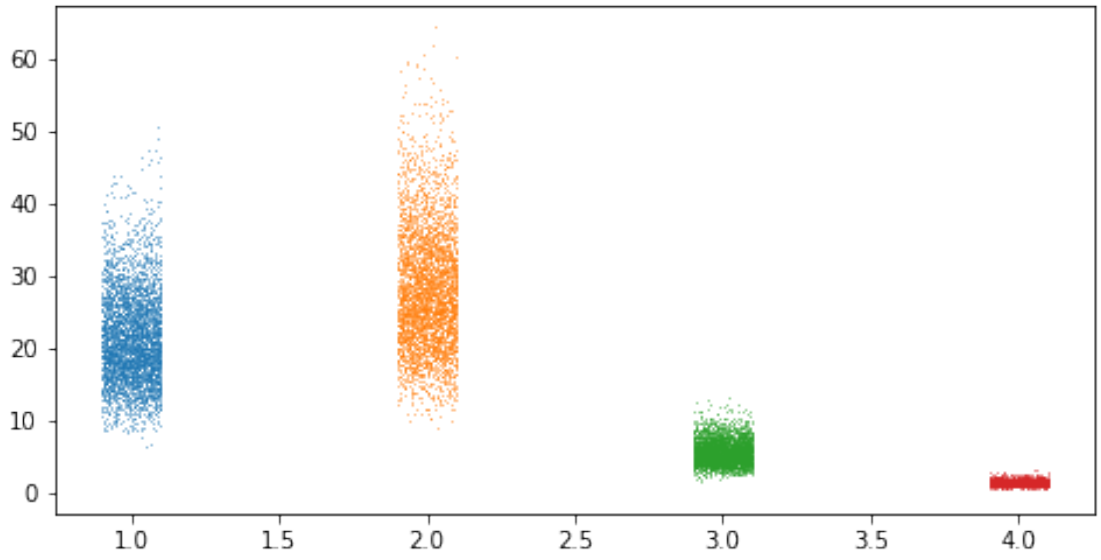
```
Tennessee
[[  2.71354664   3.75424159   0.62751088   0.27533352]
 [102.69535644 132.74603199  17.06174087   4.65105583]]
```




```
Texas
[[14.64490754 17.08123114  2.09522547  0.68101001]
 [57.93252506 67.56470379  8.33282569  2.76828876]]
```

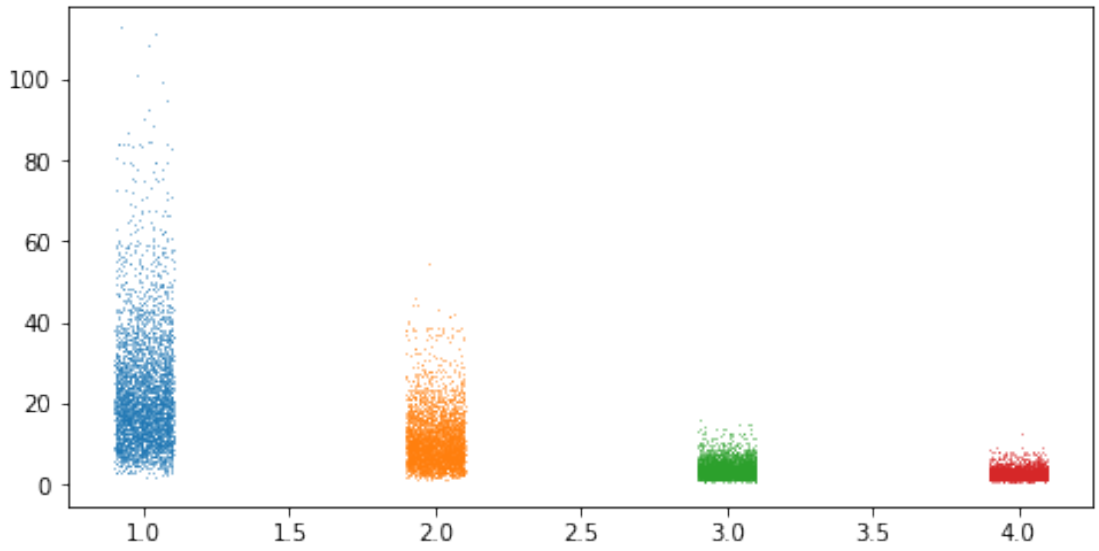


```
Utah
[[11.04998019 14.43828844  2.76373502  0.53618584]
 [35.06337495 46.36940099  9.01886133  1.82330914]]
```

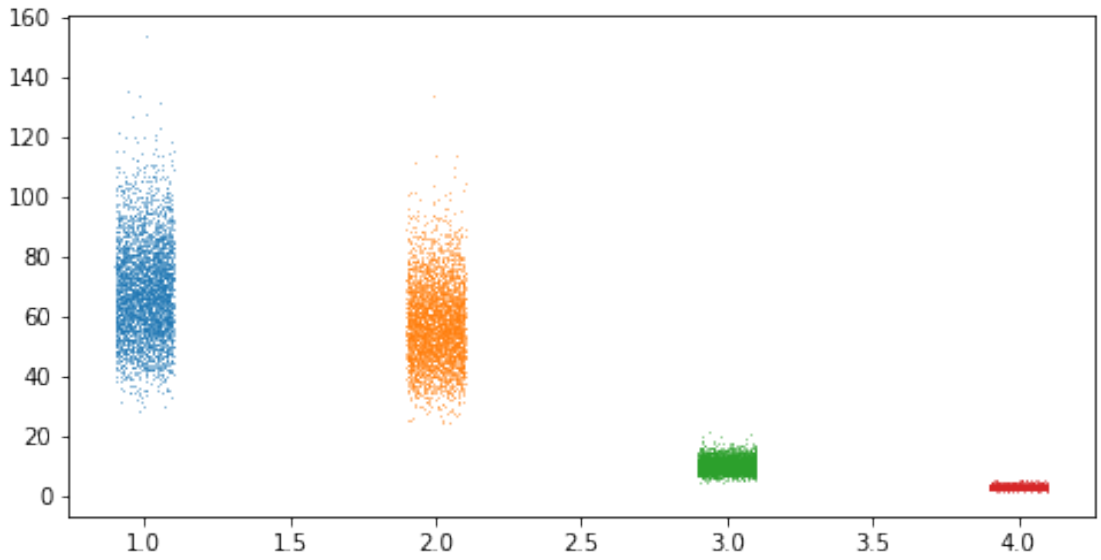



```
WARNING:pystan:5 of 4000 iterations ended with a divergence (0.125 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.
```

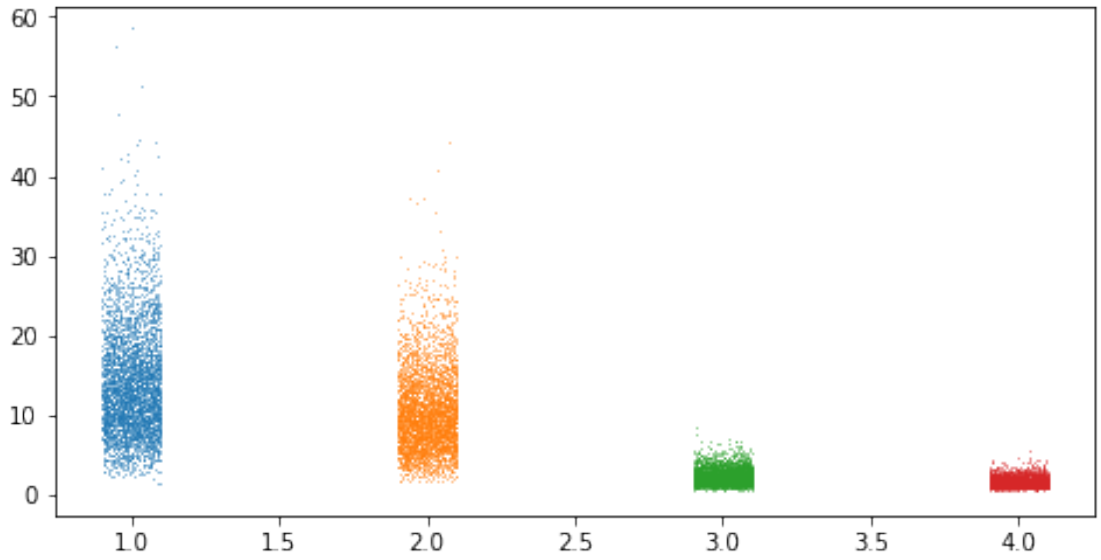
```
Vermont
[[ 4.9977826   2.17244648  0.79849016  0.54152922]
 [57.68388384 26.48247697  8.22764428  5.13902217]]
```



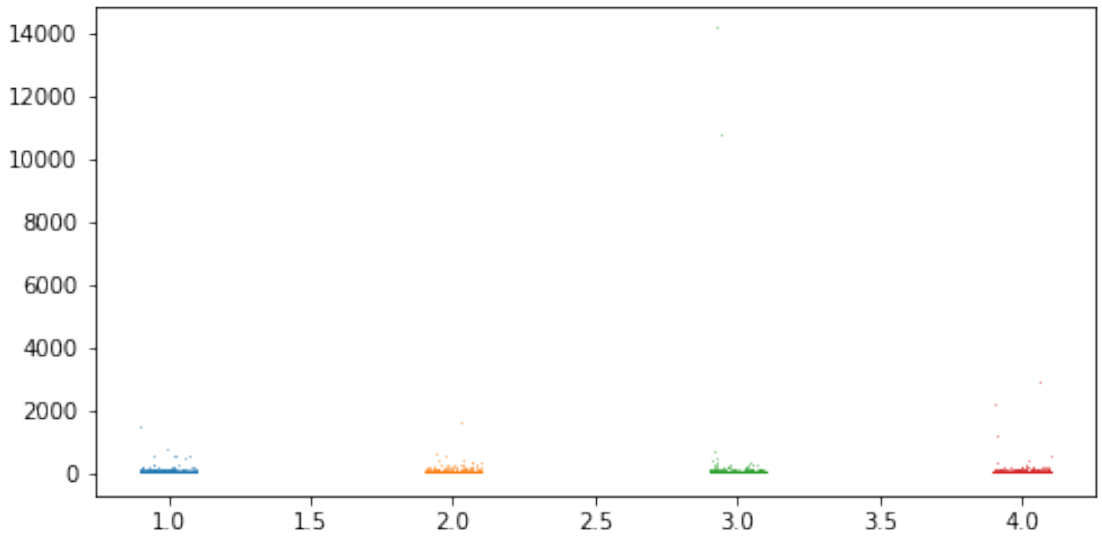
```
Virginia
[[ 41.57479352  34.62807713   5.91027094   1.34827707]
 [101.20860302  85.14056746  14.38226744   3.37086879]]
```



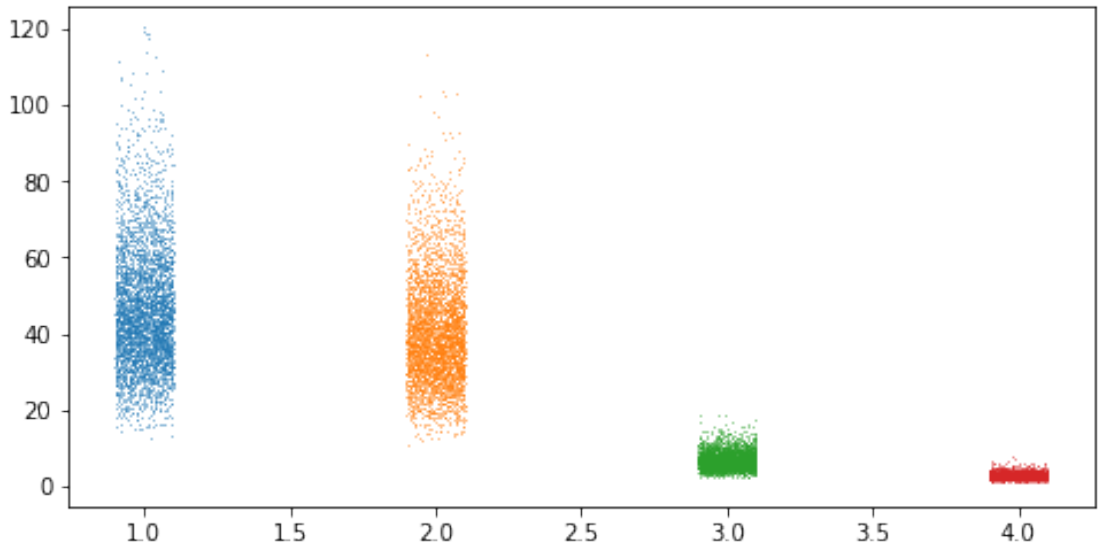

```
Washington
[[ 3.96481953  2.92754765  0.62231196  0.43776381]
 [29.8760953  22.09623192  4.28190889  2.6410507 ]]
```



```
West Virginia
[[ 0.04288542  0.04749731  0.04263964  0.03231327]
 [27.97376896 25.1519764  22.81768605 26.14446064]]
```

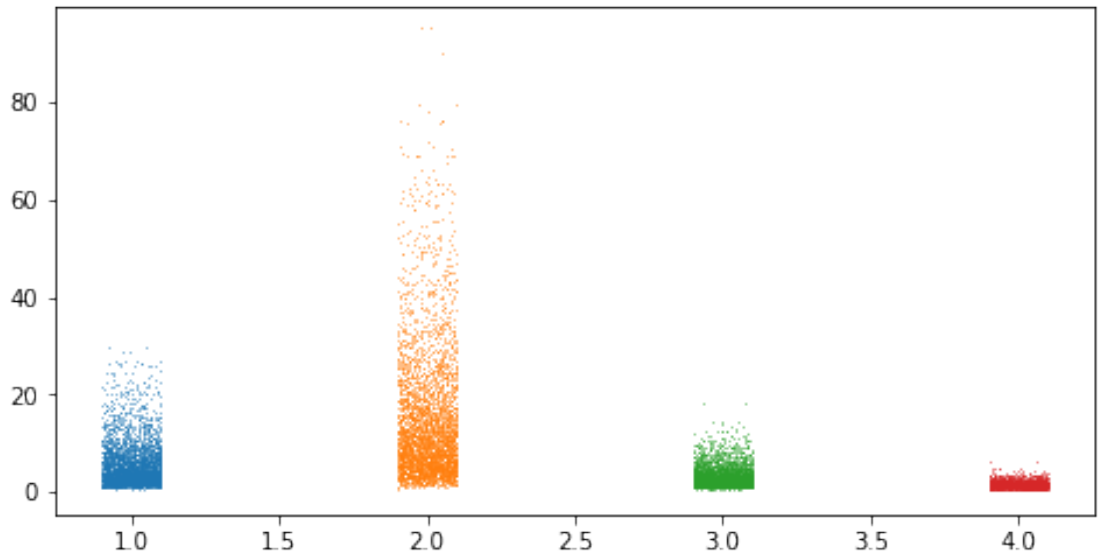



```
Wisconsin
[[21.610267   18.7907745   2.92305144  1.03360356]
 [83.74071924 71.95585972 11.43847686  3.85704316]]
```



```
WARNING:pystan:15 of 4000 iterations ended with a divergence (0.375 %).
WARNING:pystan:Try running with adapt_delta larger than 0.8 to remove the divergences.


Wyoming
[[ 0.62564836  1.47826462  0.38891635  0.1879286 ]
 [17.16964312 50.06014187  8.35248141  2.51490804]]
```



1.2 Simulation time

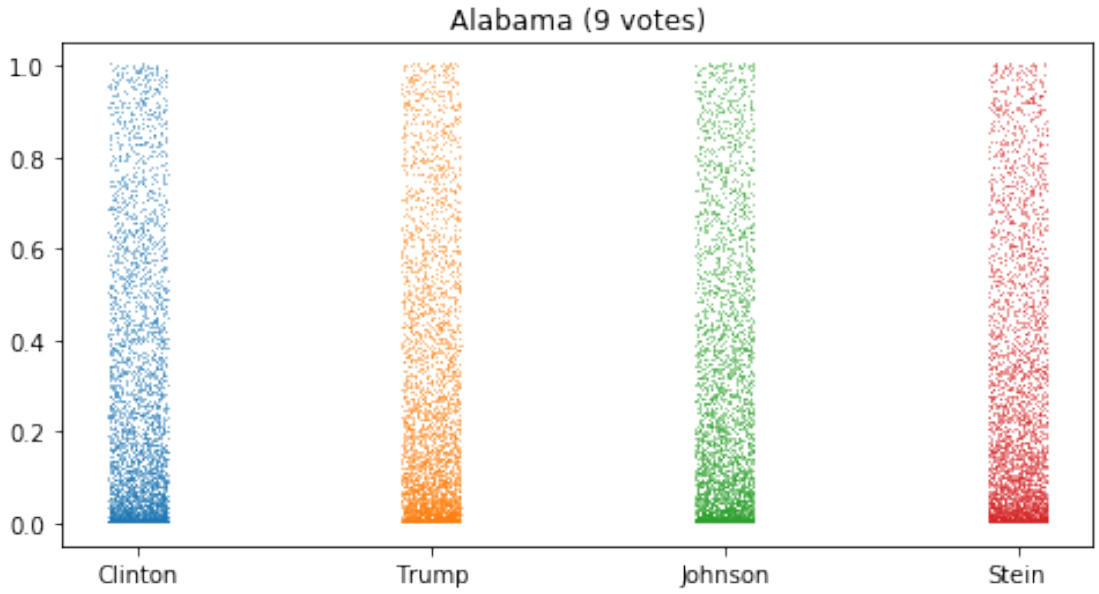
Use the posterior samples to predict the outcome of the presidential elections.

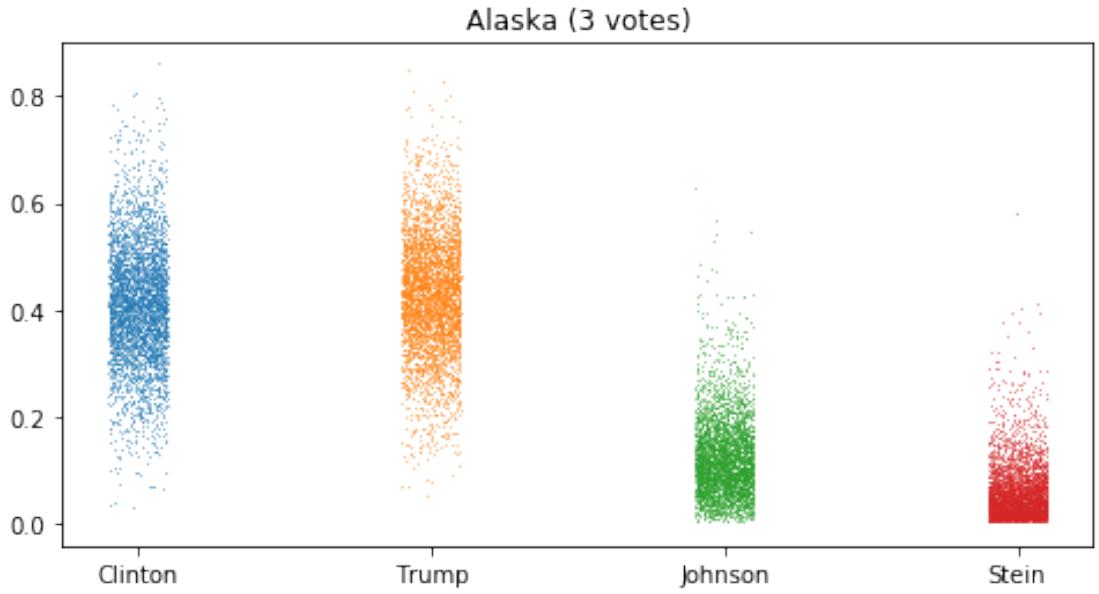
- Predict the probability that each candidate will win each state.
- Use the posterior α samples to generate posterior predictive samples for p — the proportion of votes each candidate would get in each state in an election.
- Use these p samples to estimate the probability that each candidate will win each state.
- Predict the probability that each candidate will win the presidential election.
- Use the posterior predictive probability that each candidate will win each state to generate samples over the total number Electoral College votes each candidate would get in an election.
- Use the total number of votes to generate samples over who would win the election.

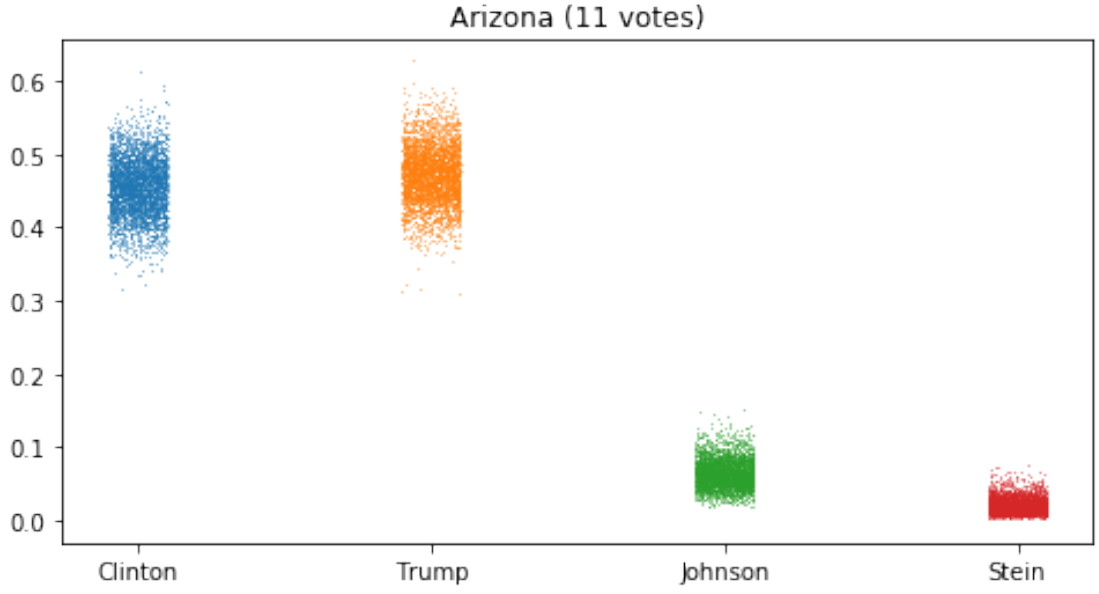
```
In [5]: # Generating predictive samples using SciPy
        for state in states:
            samples = results[state].extract()
```

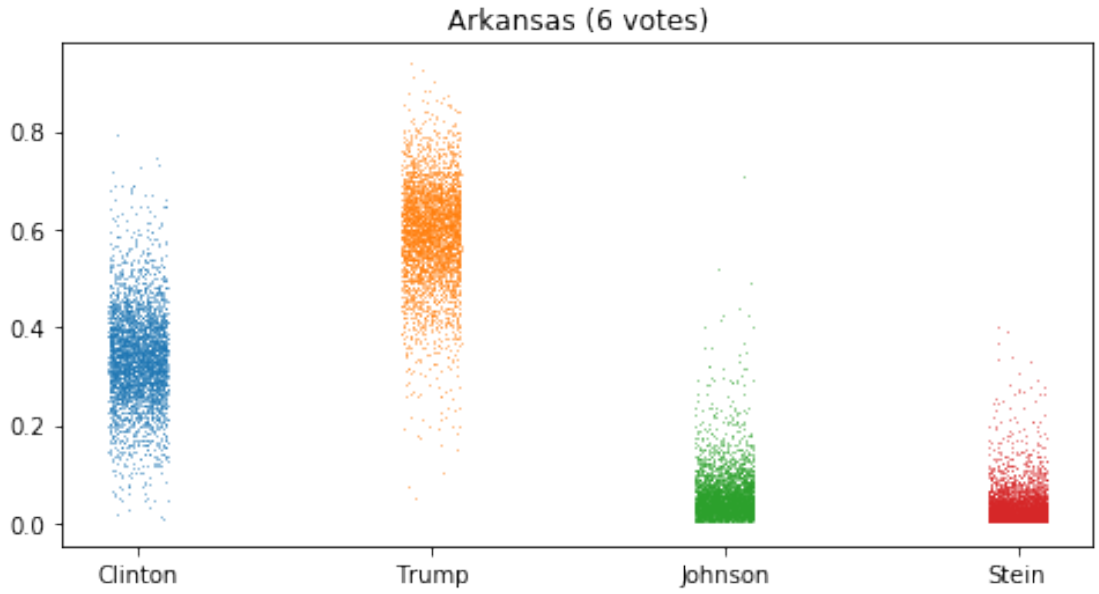
```
            alpha = samples['alpha']
            p_predicted = np.empty(alpha.shape)
            for i in range(alpha.shape[0]):
                p_predicted[i] = stats.dirichlet.rvs(alpha[i])
            plt.figure(figsize=(8, 4))
            for i in range(4):
                plt.plot(stats.uniform.rvs(loc=i-0.1, scale=0.2, size=4000), p_predicted[:,i],
            plt.title(state + ' (' + str(electoral_votes[state]) + ' votes)')
            plt.xticks([0, 1, 2, 3], ['Clinton', 'Trump', 'Johnson', 'Stein'])
        plt.show()
```

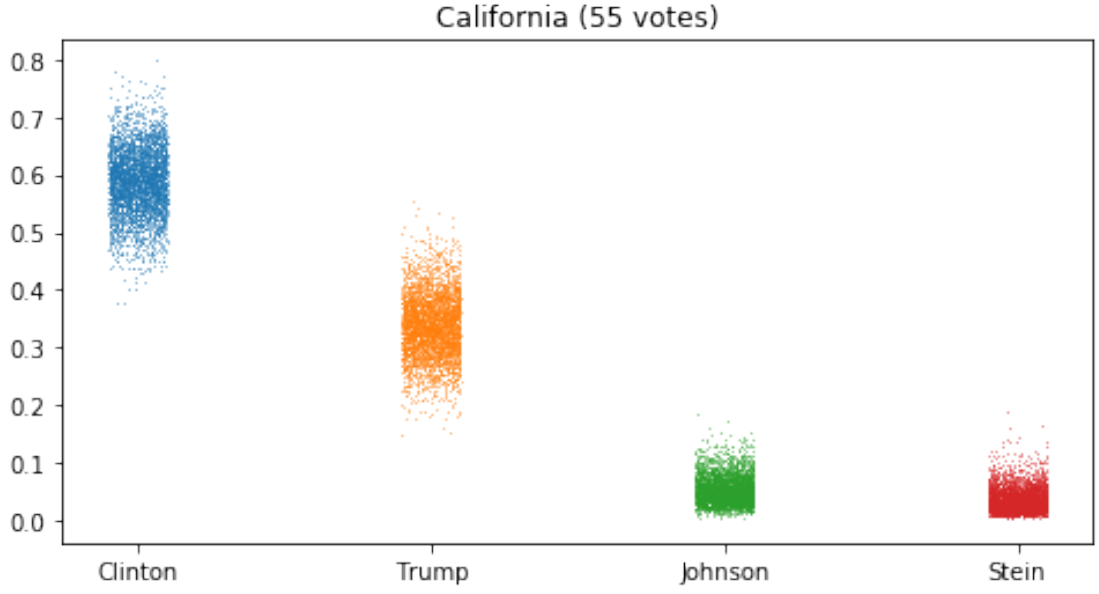
```
C:\Users\Taha\Anaconda3\Anaconda4\lib\site-packages\matplotlib\pyplot.py:514: RuntimeWarning: M
  max_open_warning, RuntimeWarning)
```

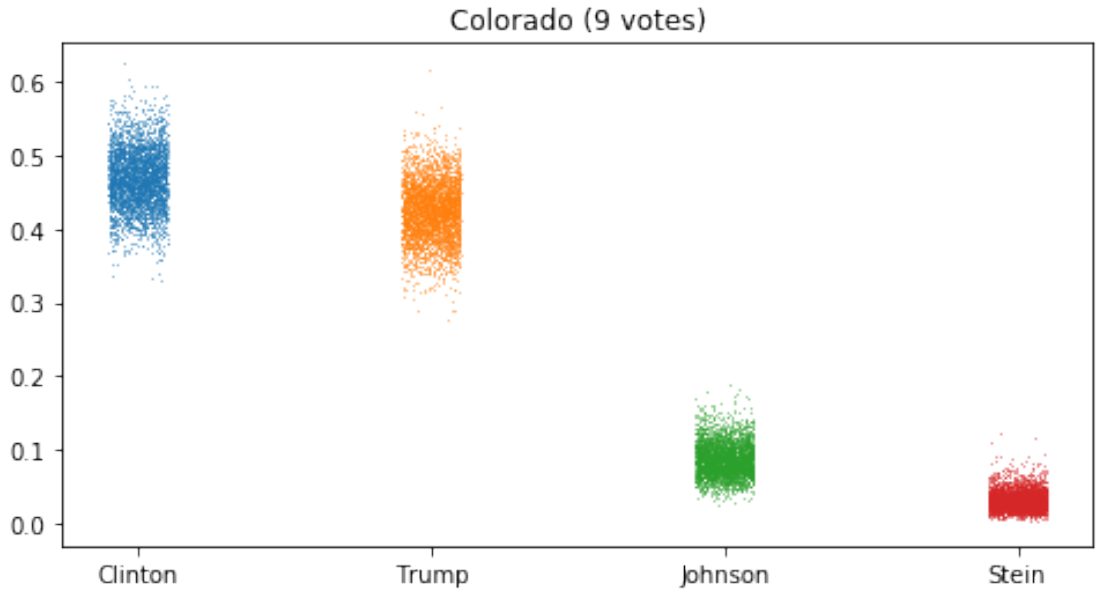


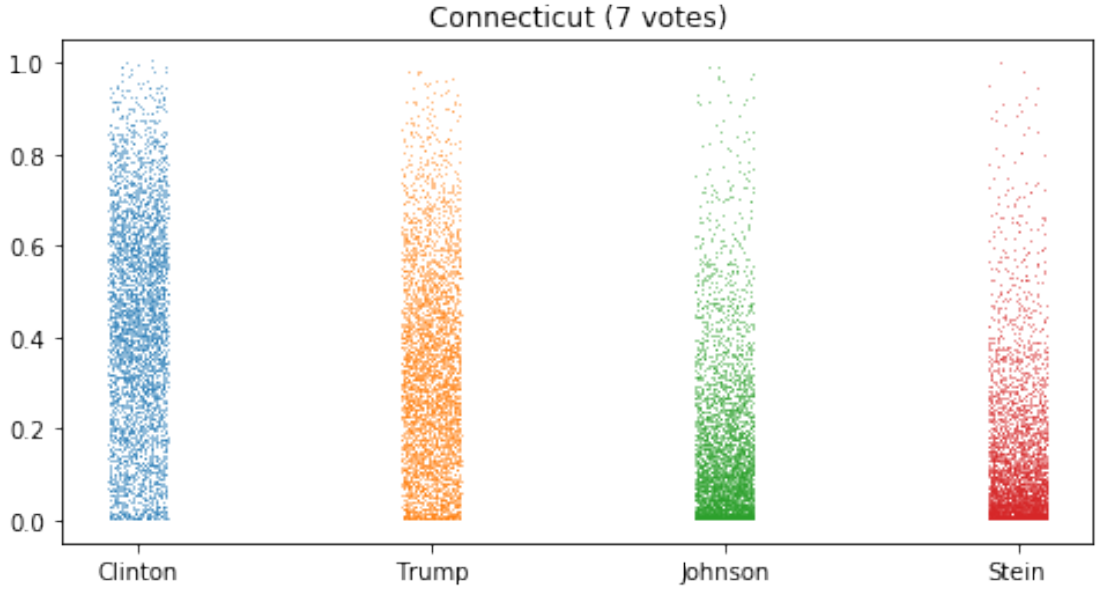

Alaska (3 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

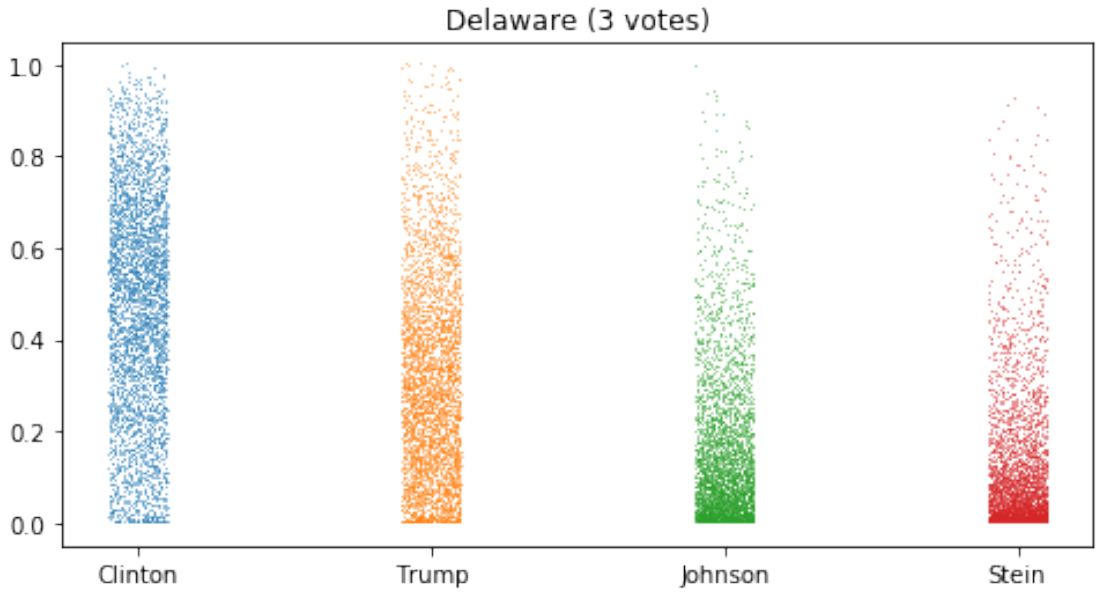

Arizona (11 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

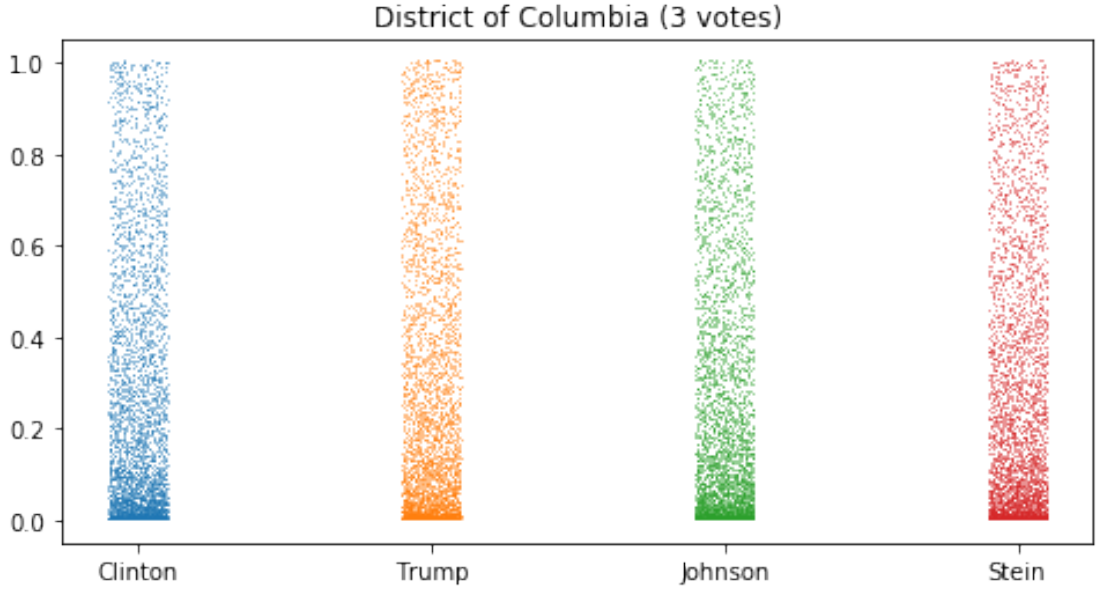

Arkansas (6 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

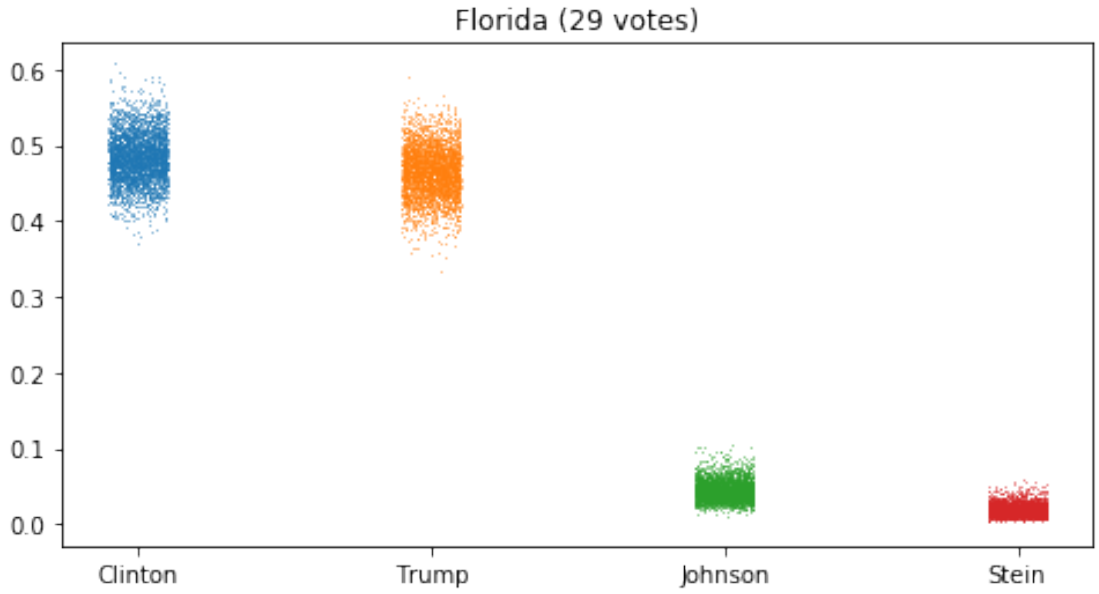

California (55 votes)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

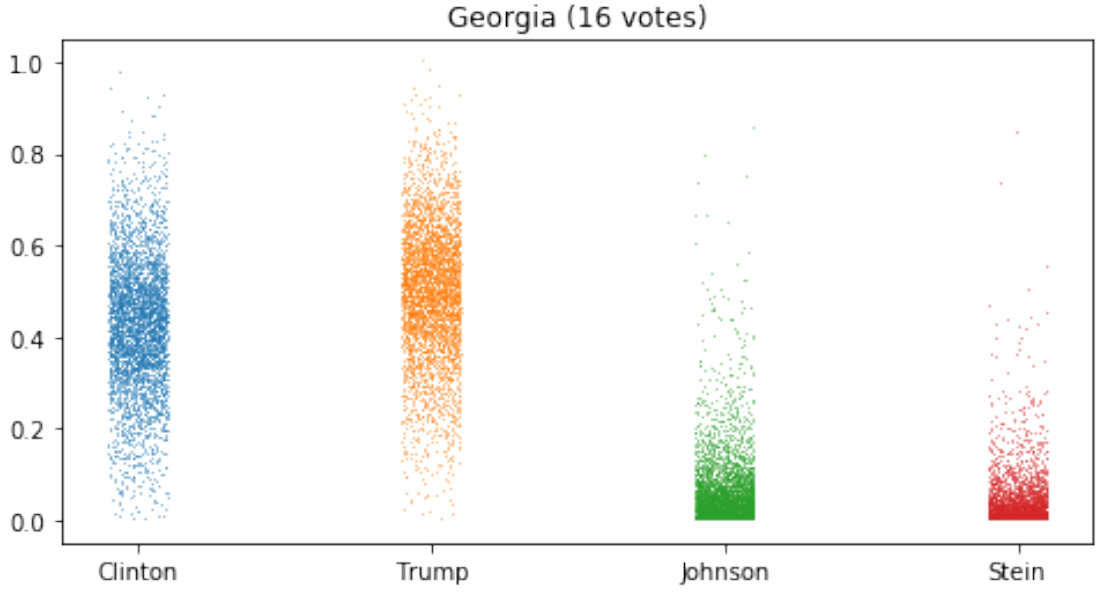

Colorado (9 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

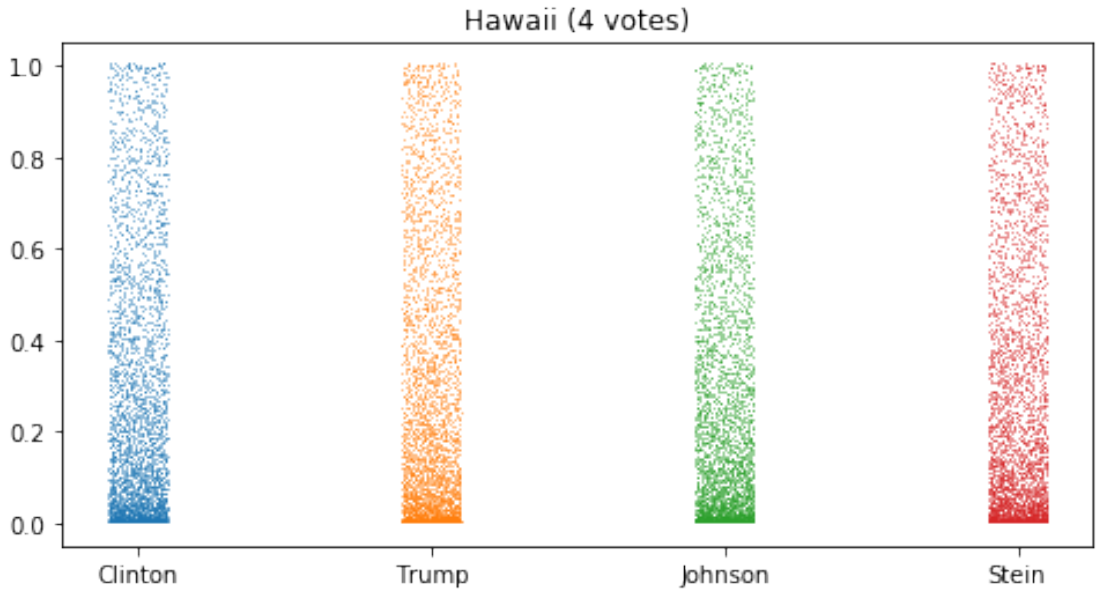

Connecticut (7 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

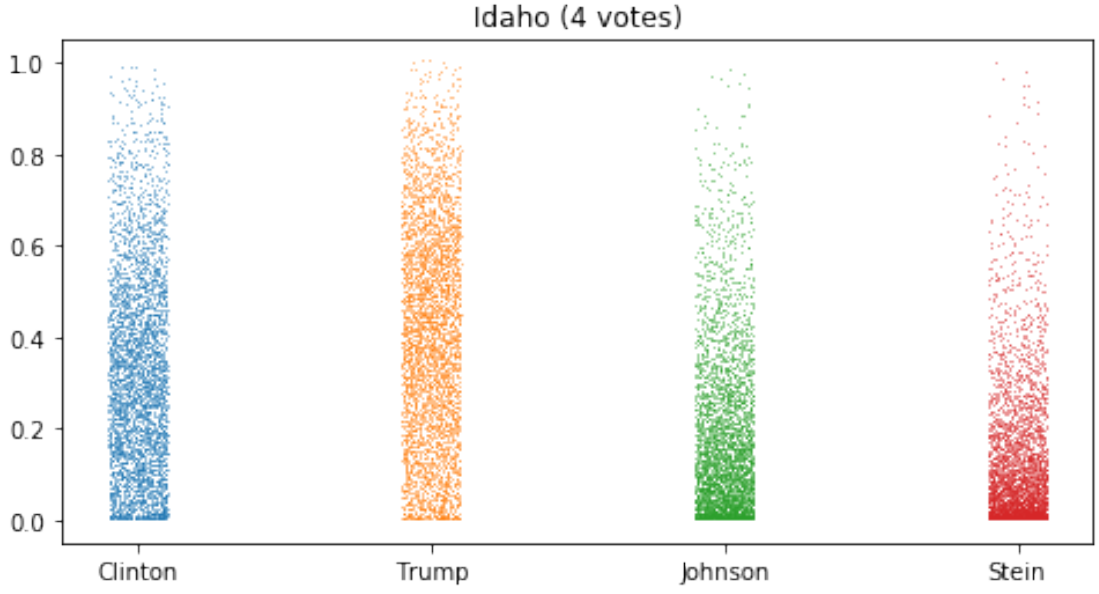

Delaware (3 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

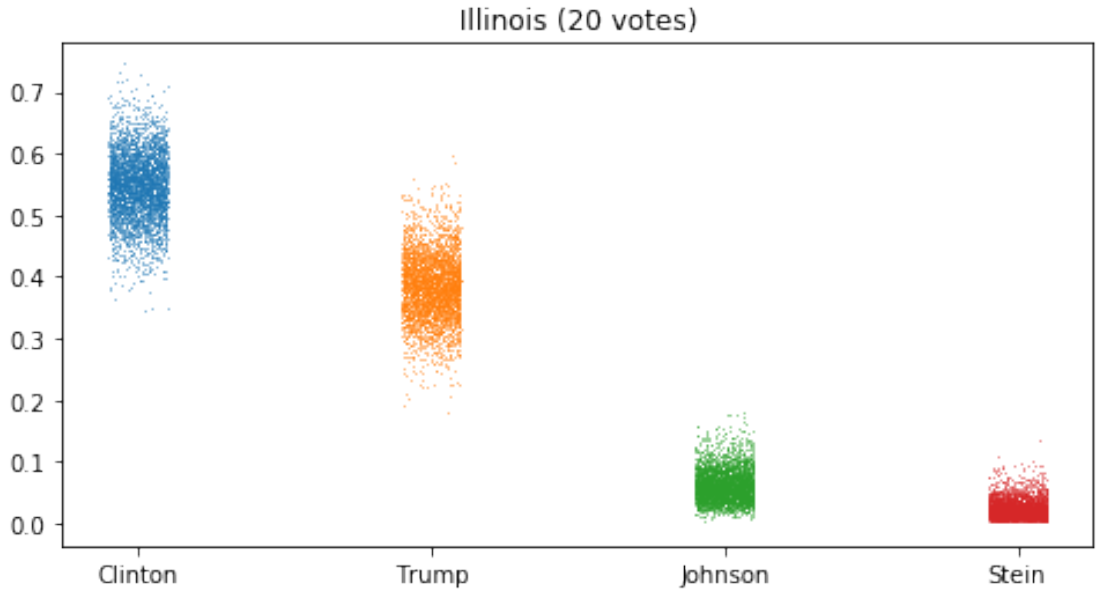

District of Columbia (3 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

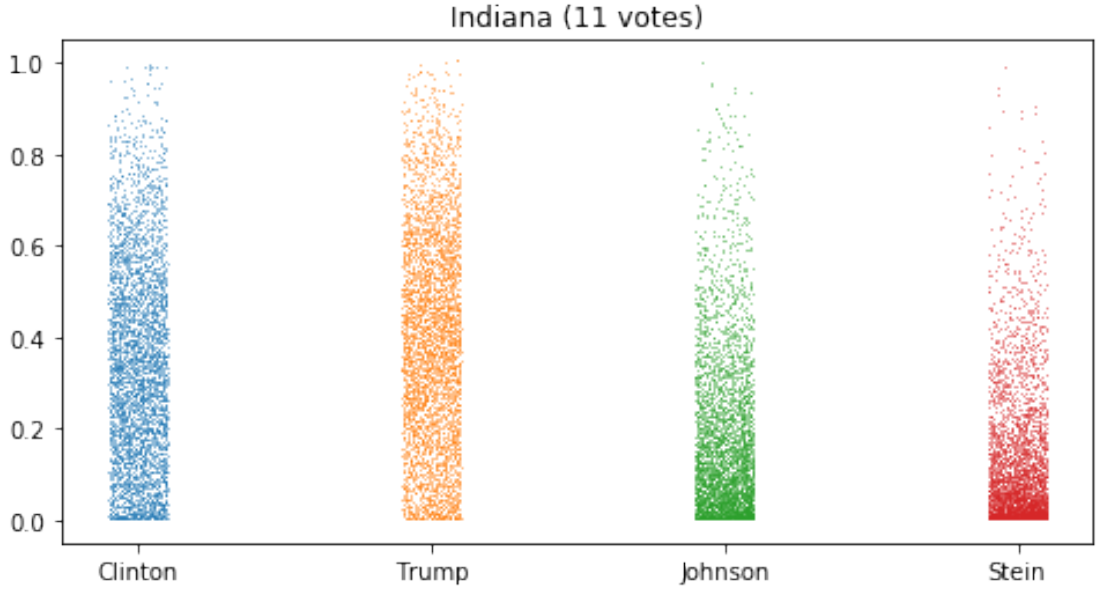

Florida (29 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

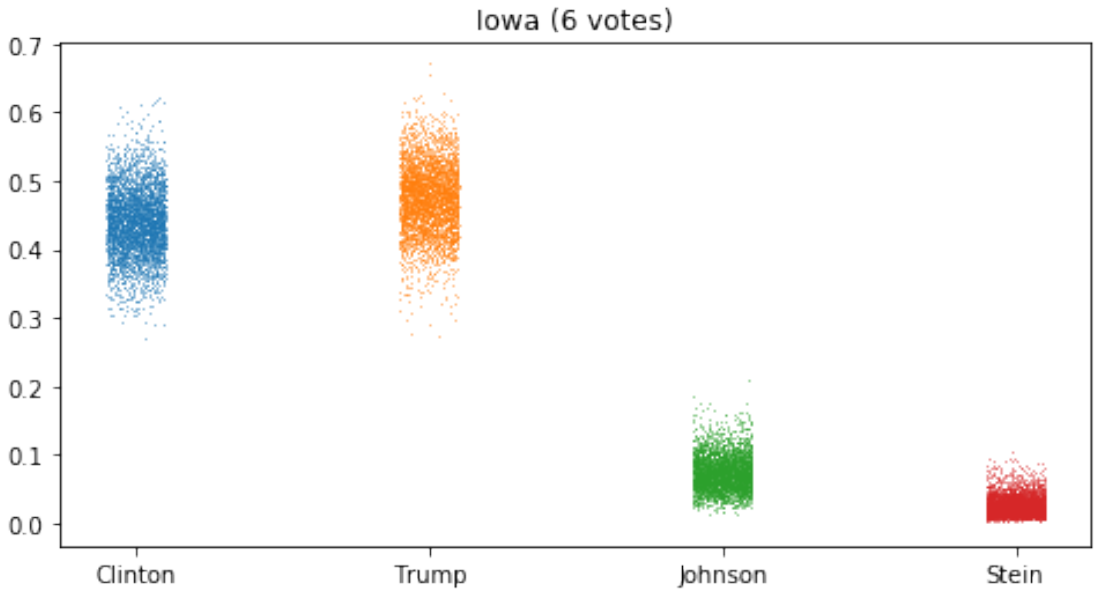

Georgia (16 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

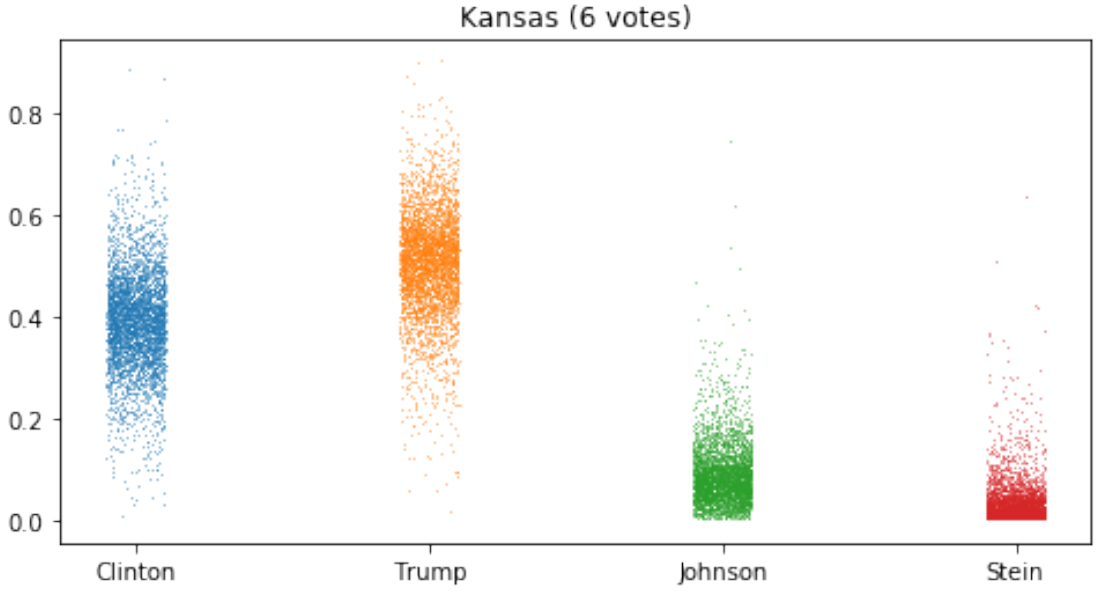

Hawaii (4 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

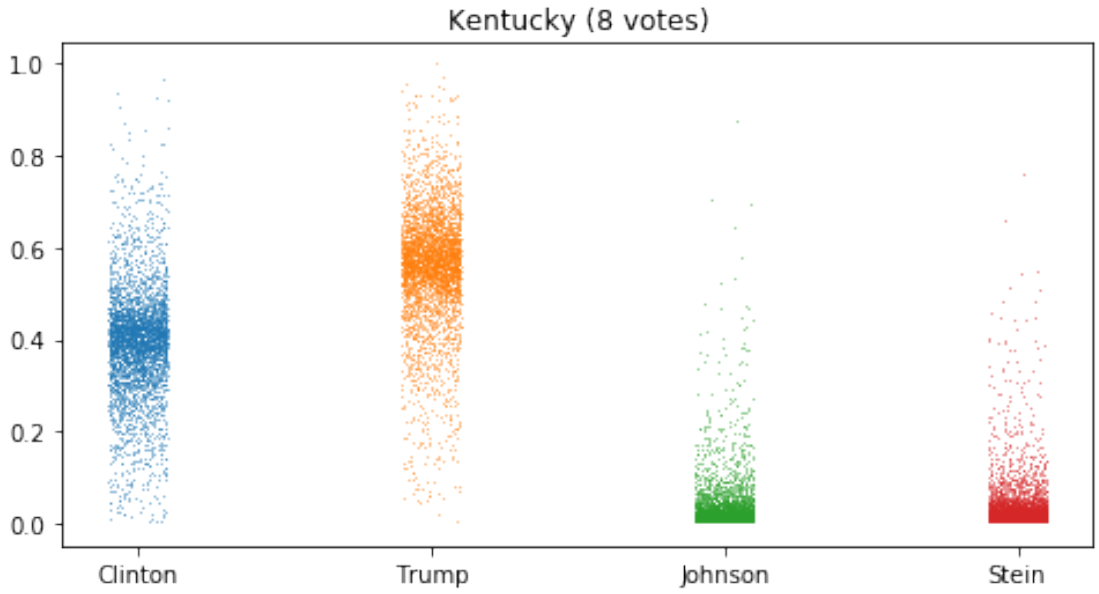

Idaho (4 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

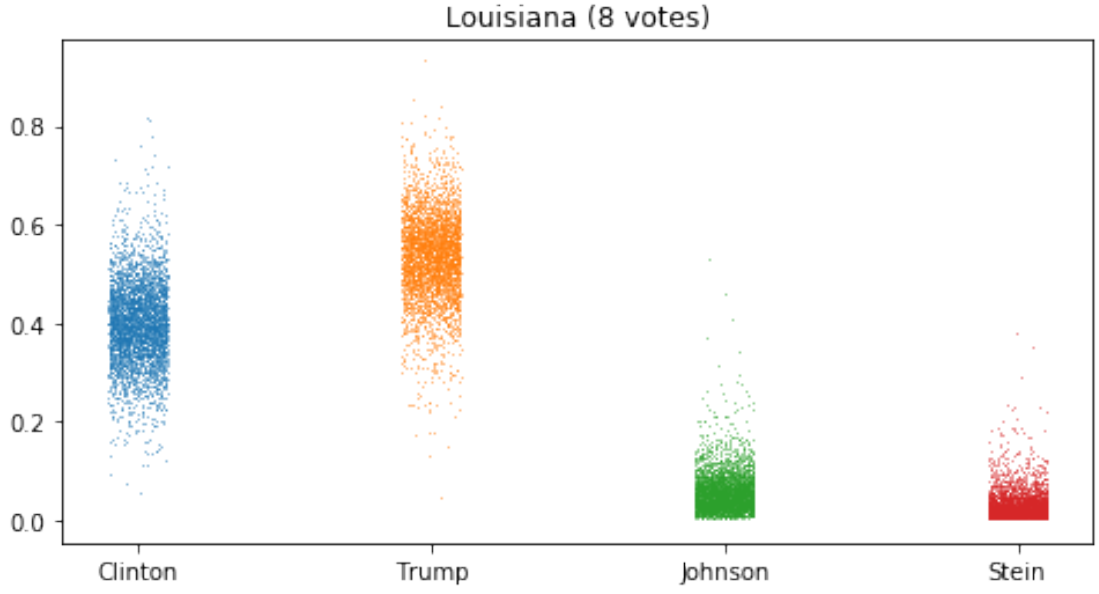

Illinois (20 votes)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

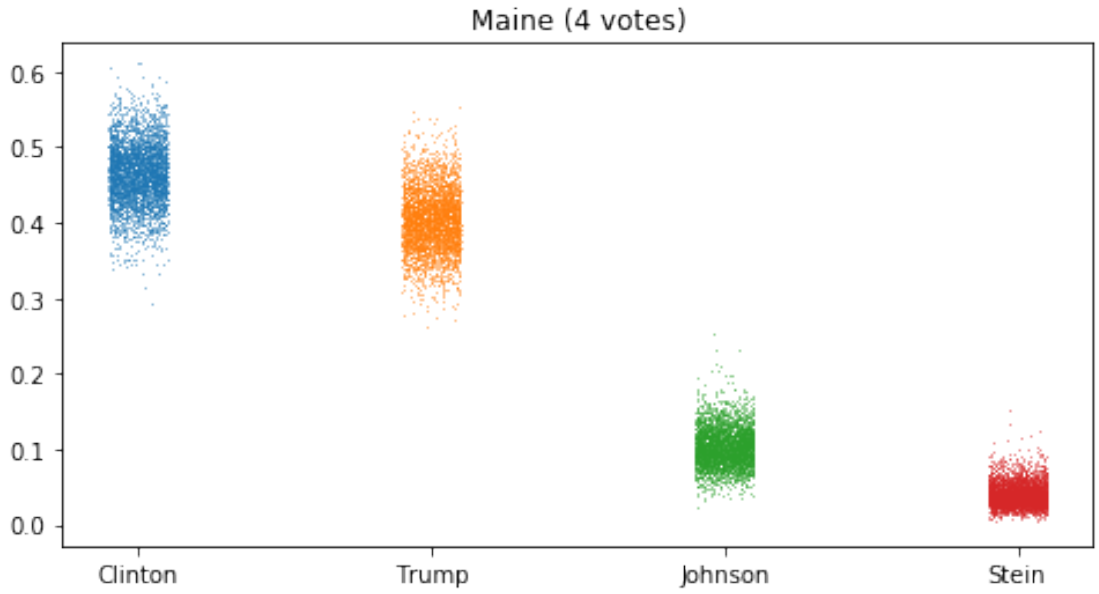

Indiana (11 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

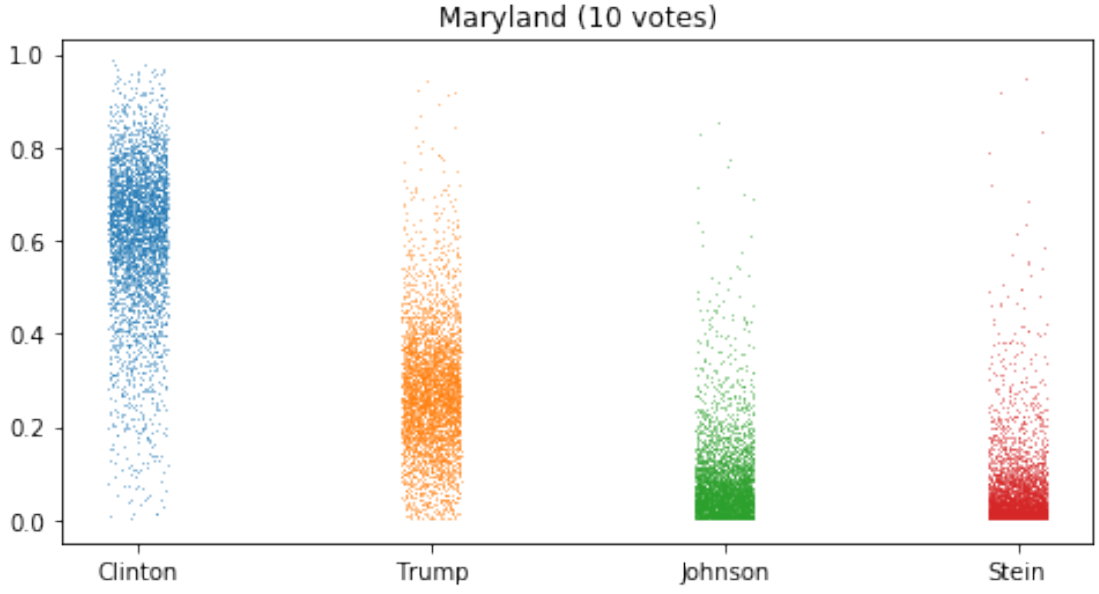

Iowa (6 votes)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

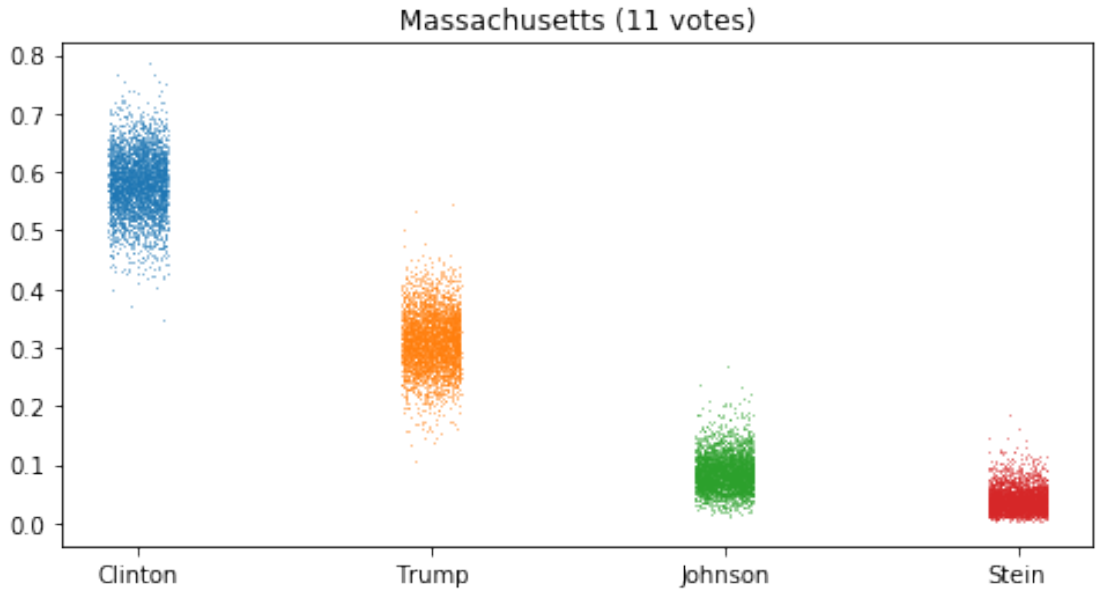

Kansas (6 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

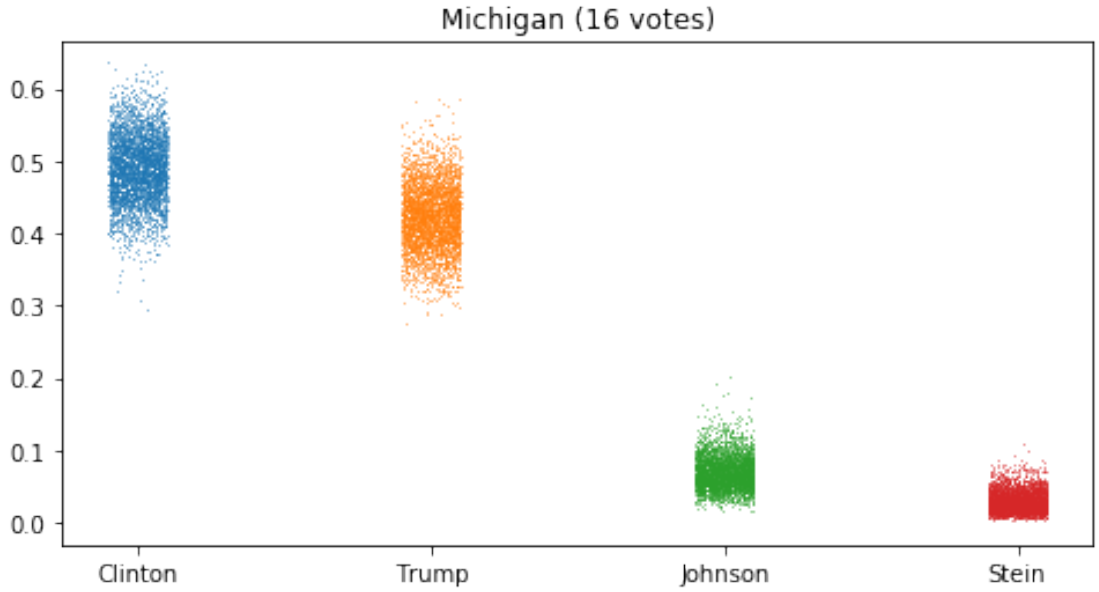

Kentucky (8 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

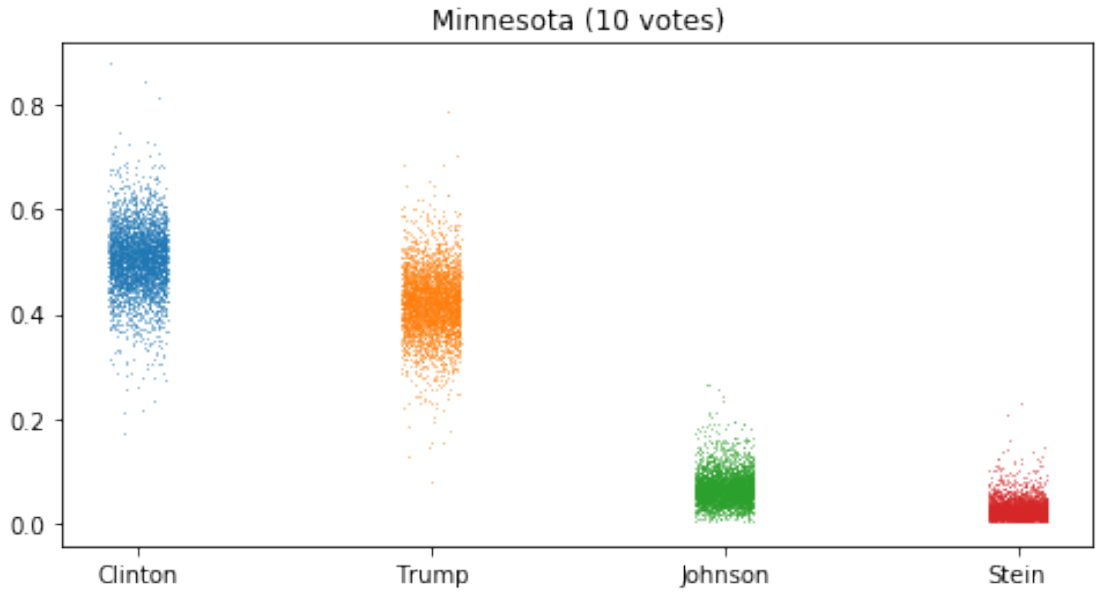

Louisiana (8 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

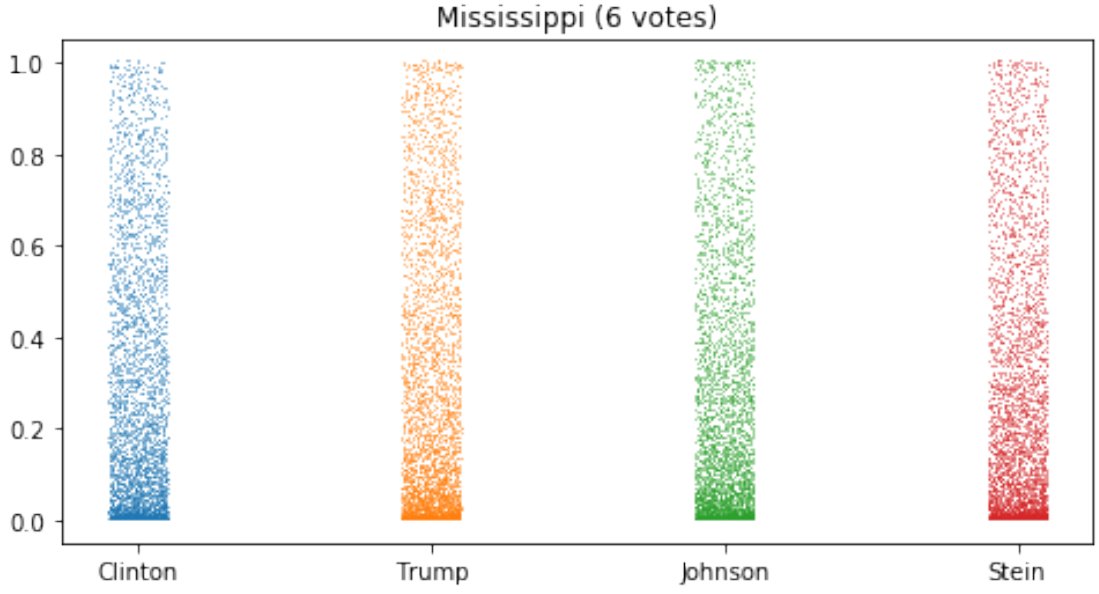

Maine (4 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

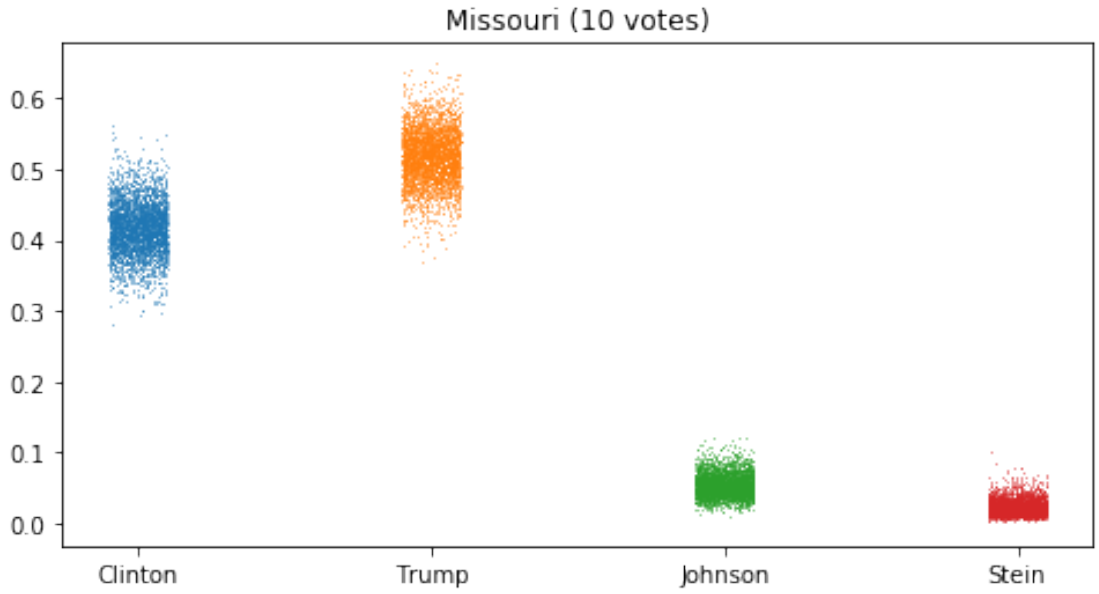

Maryland (10 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

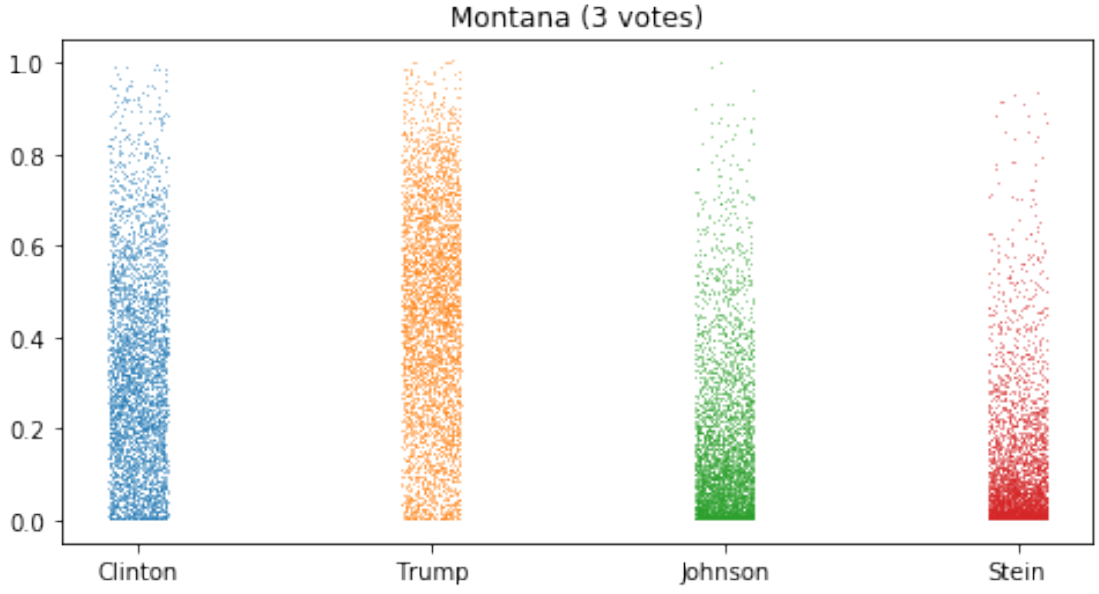

Massachusetts (11 votes)
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

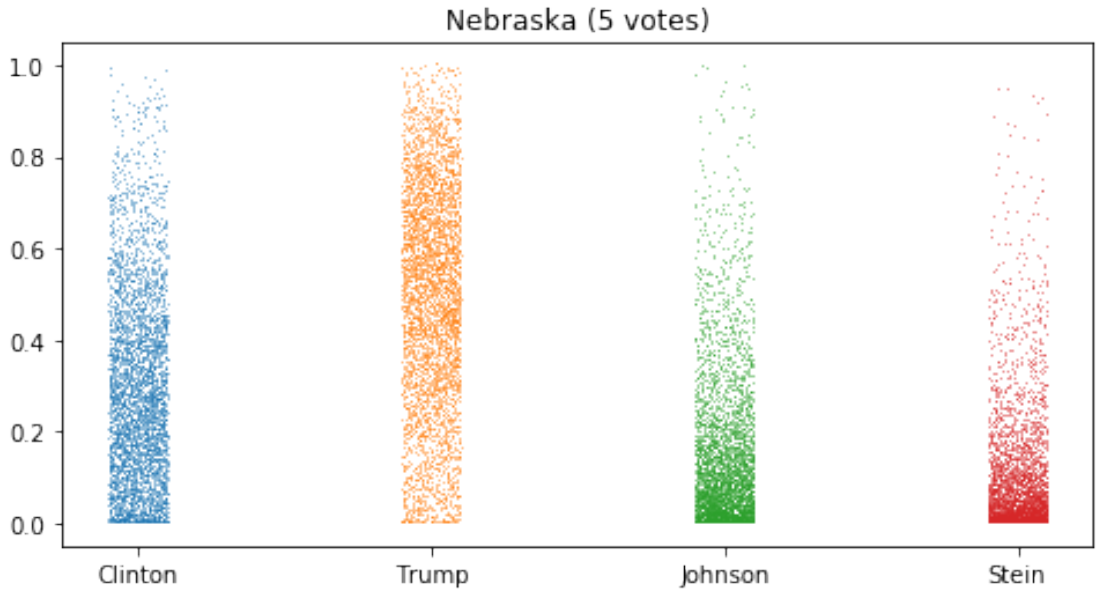

Michigan (16 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

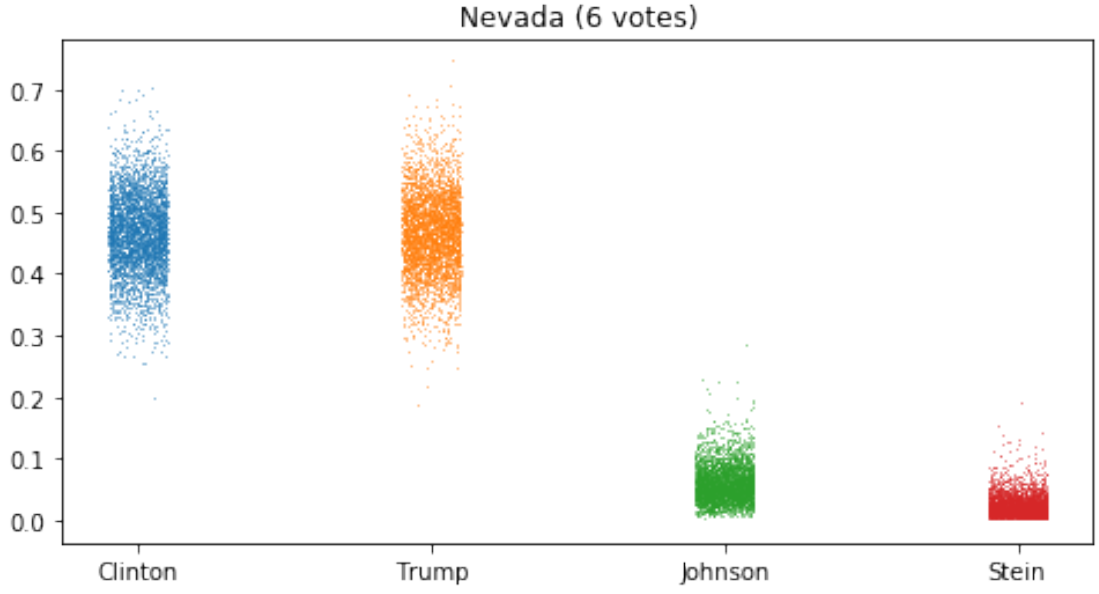

Minnesota (10 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

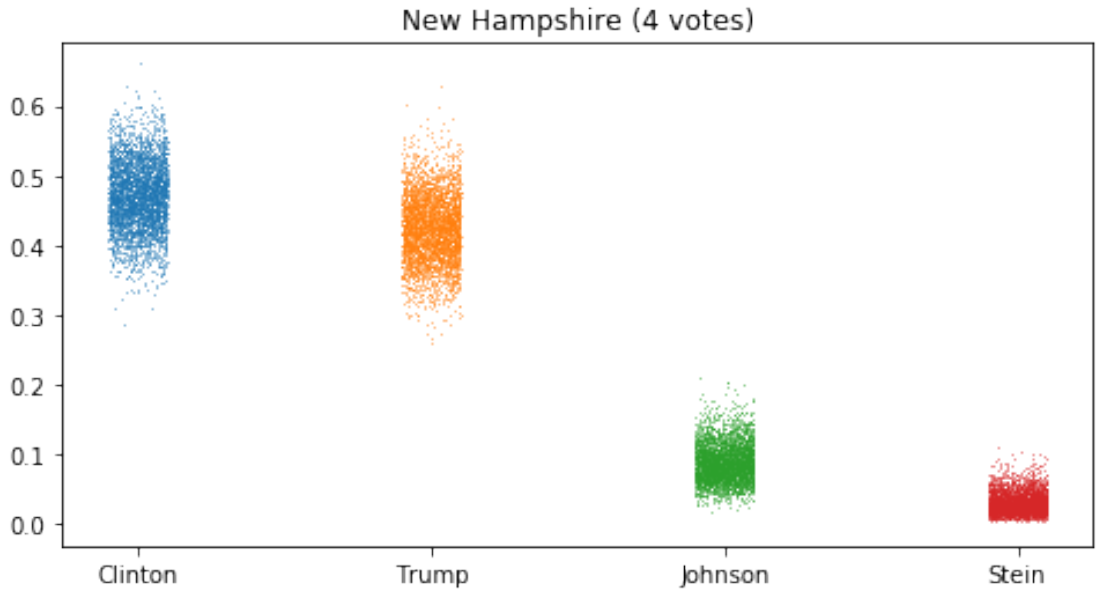

Mississippi (6 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

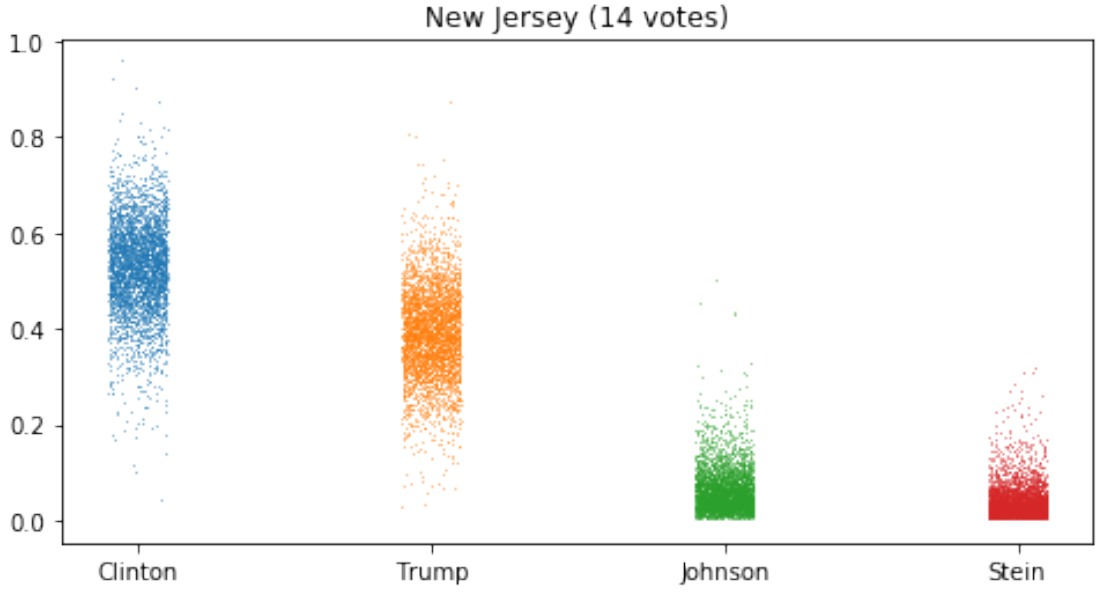

Missouri (10 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

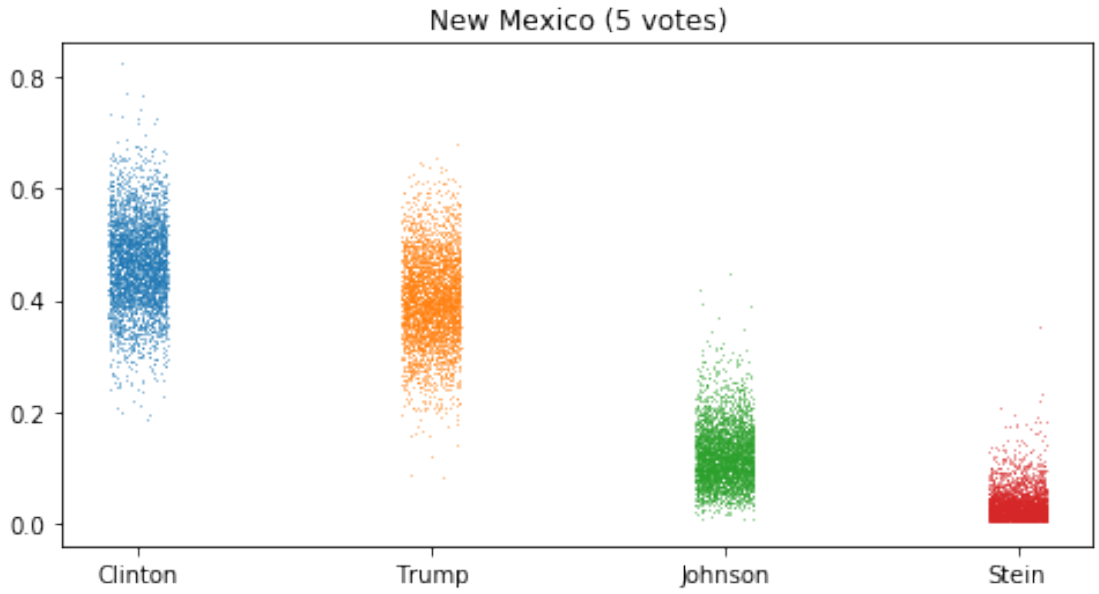

Montana (3 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

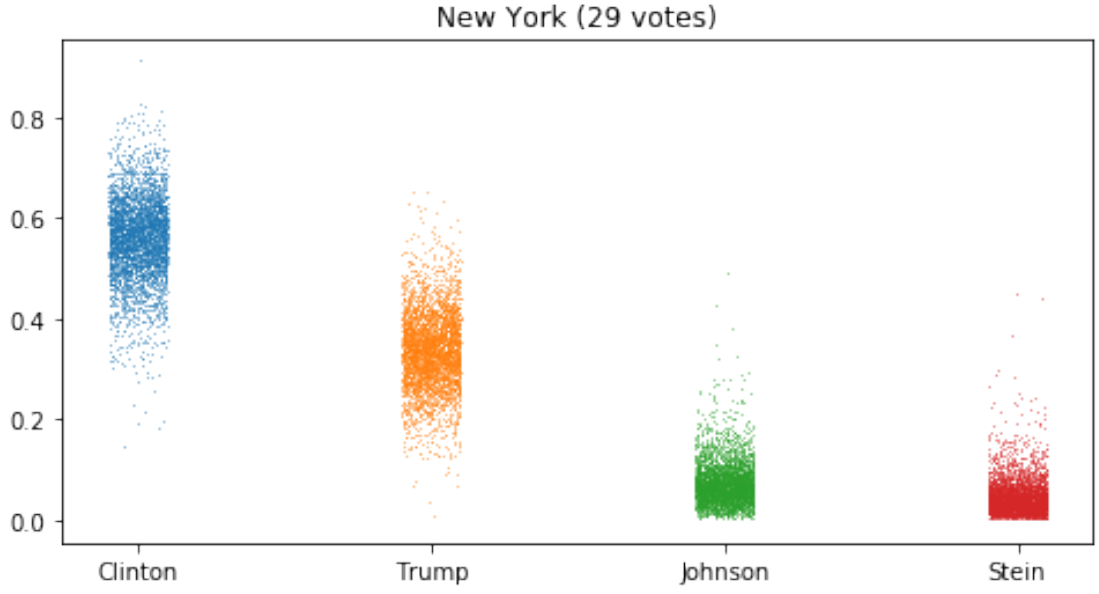

Nebraska (5 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

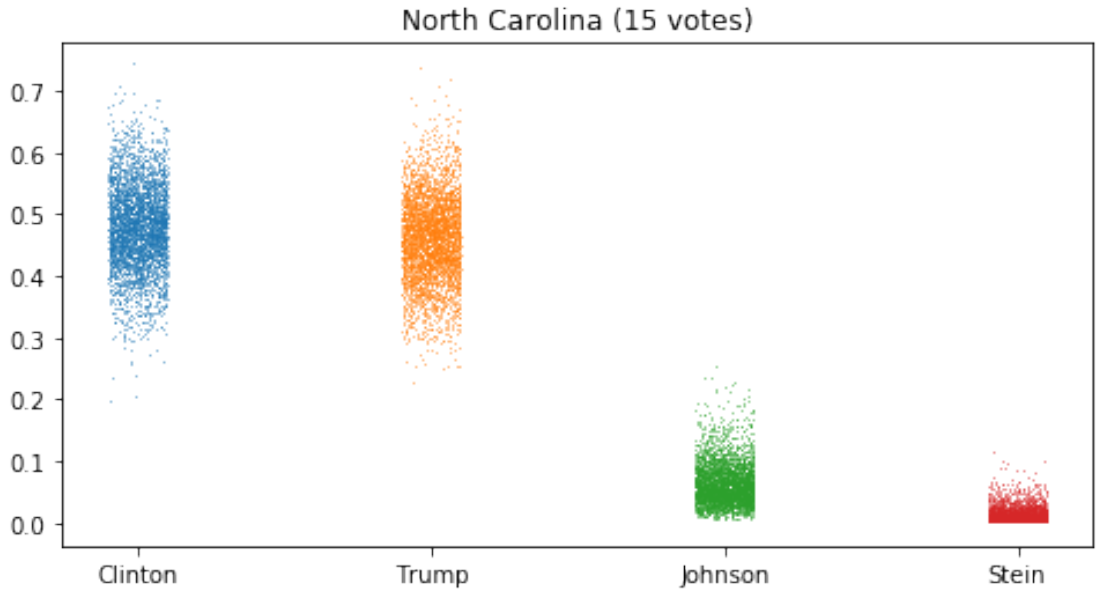

Nevada (6 votes)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

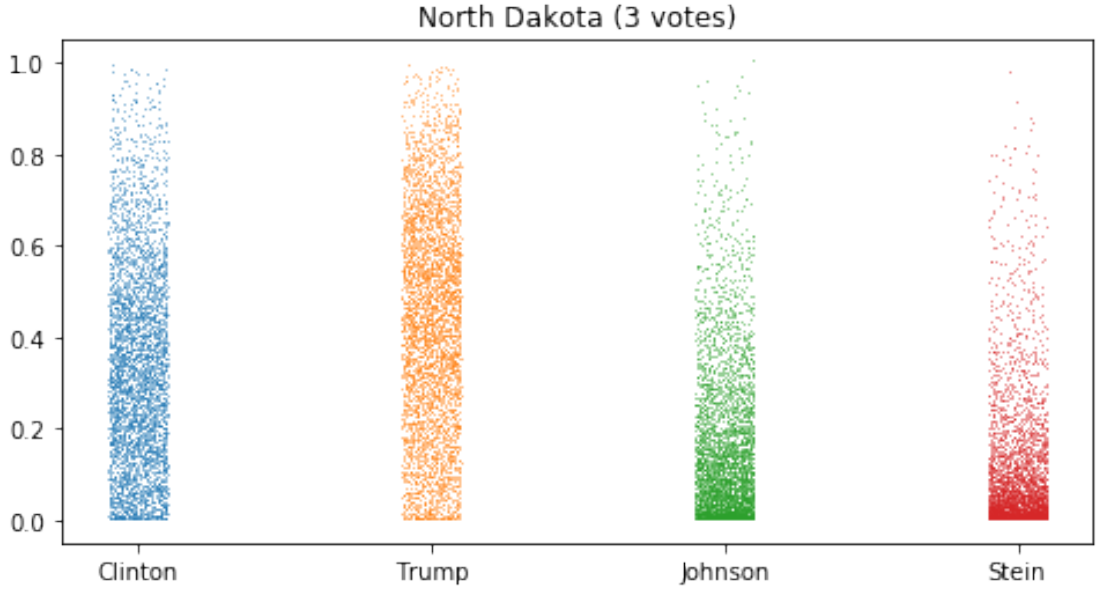

New Hampshire (4 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

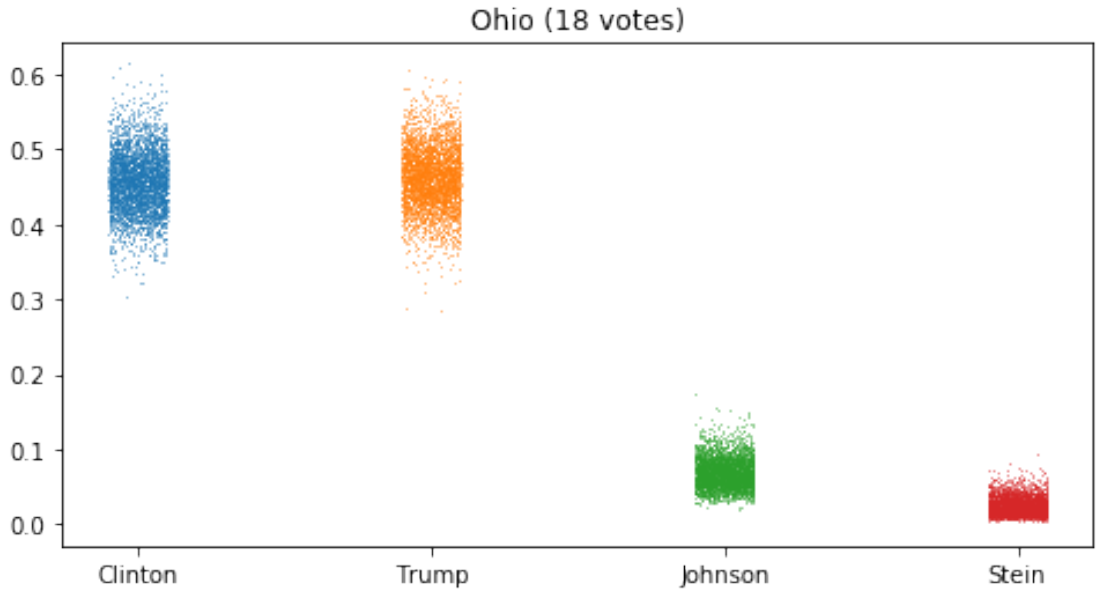

New Jersey (14 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

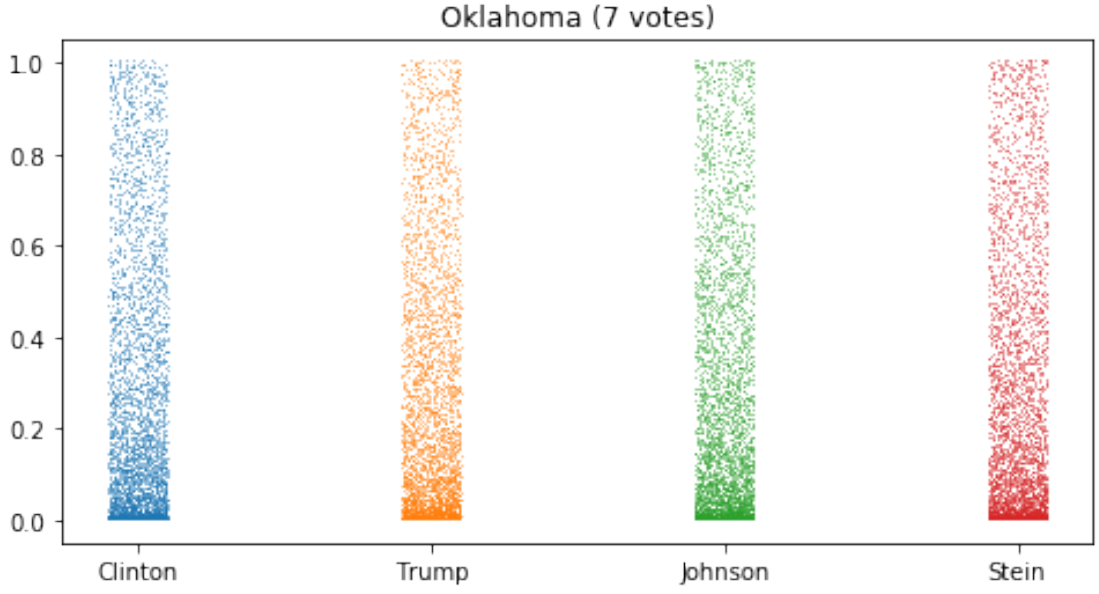

New Mexico (5 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

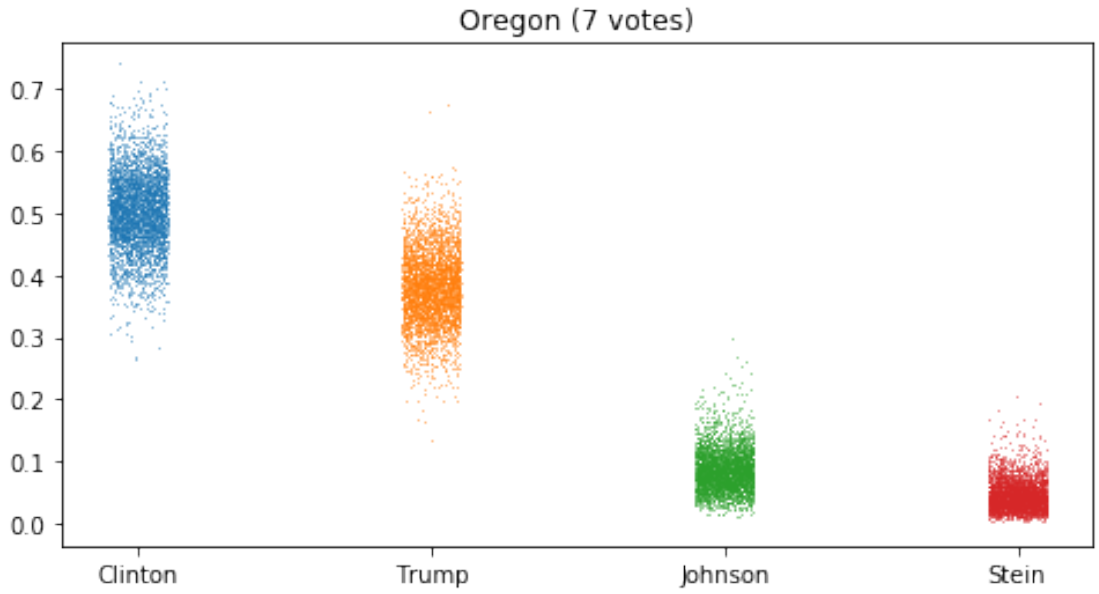

New York (29 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

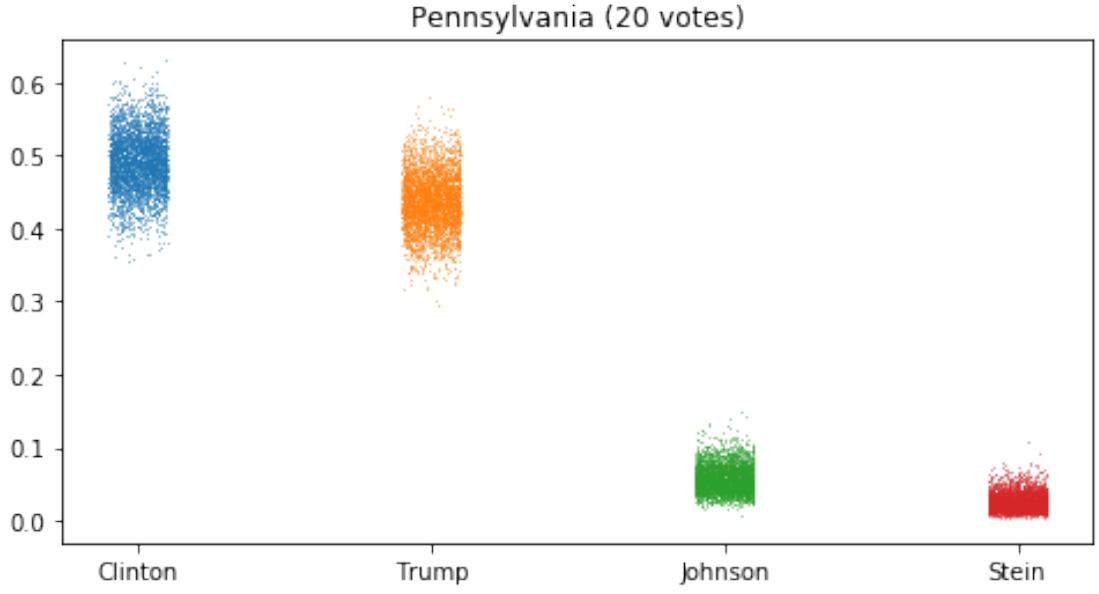

North Carolina (15 votes)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

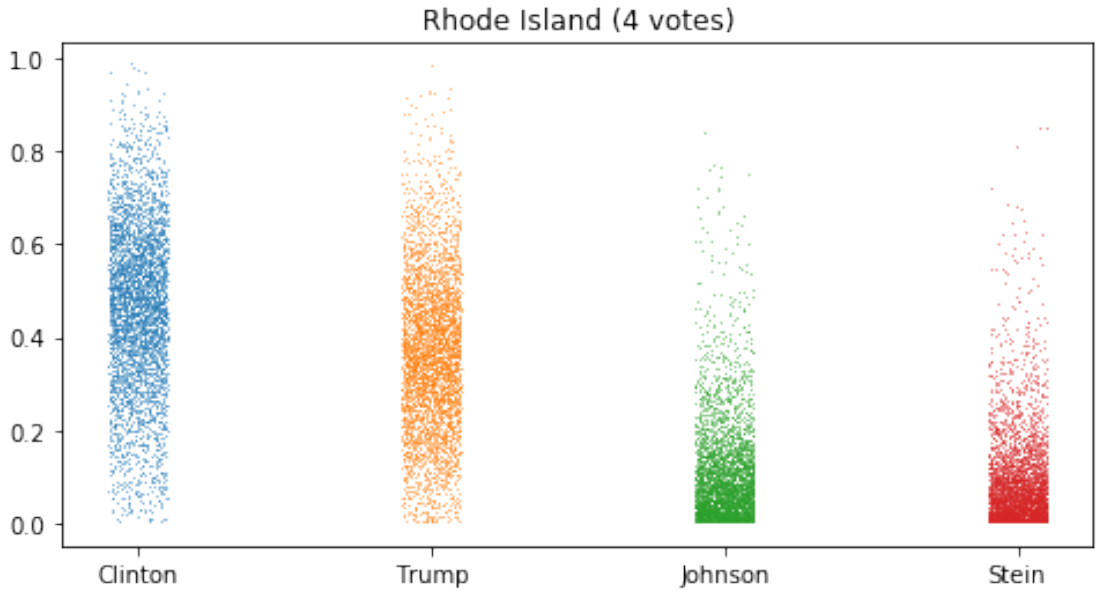

North Dakota (3 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

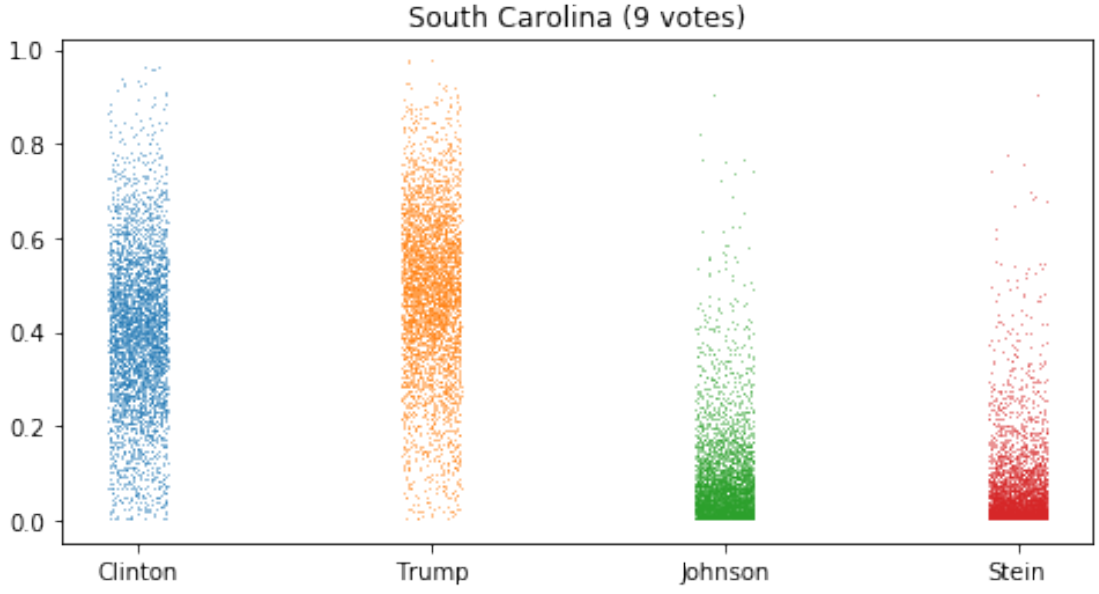

Ohio (18 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

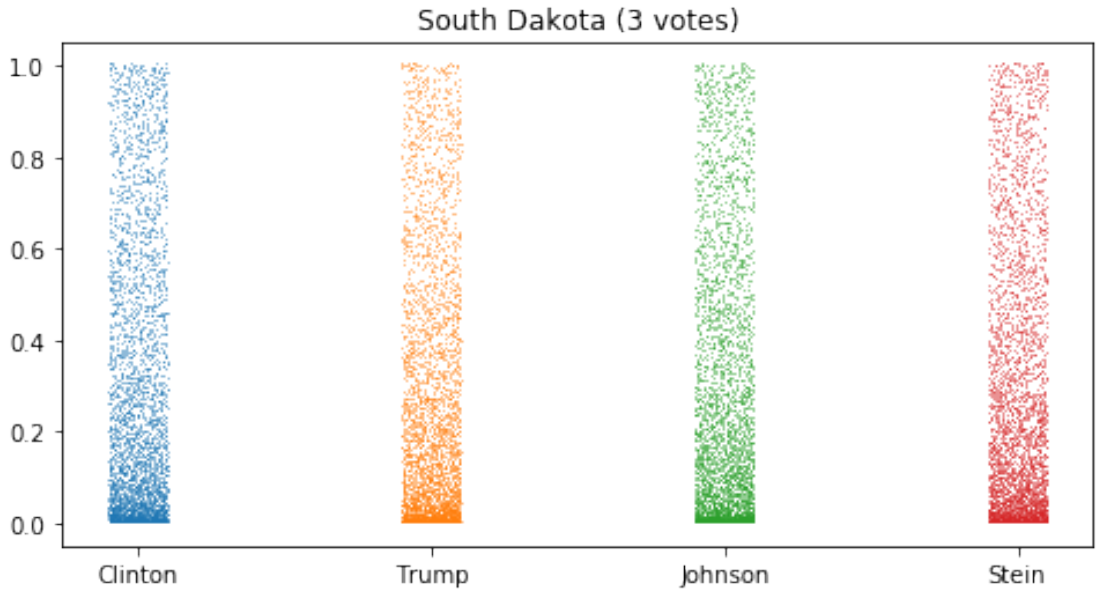

Oklahoma (7 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

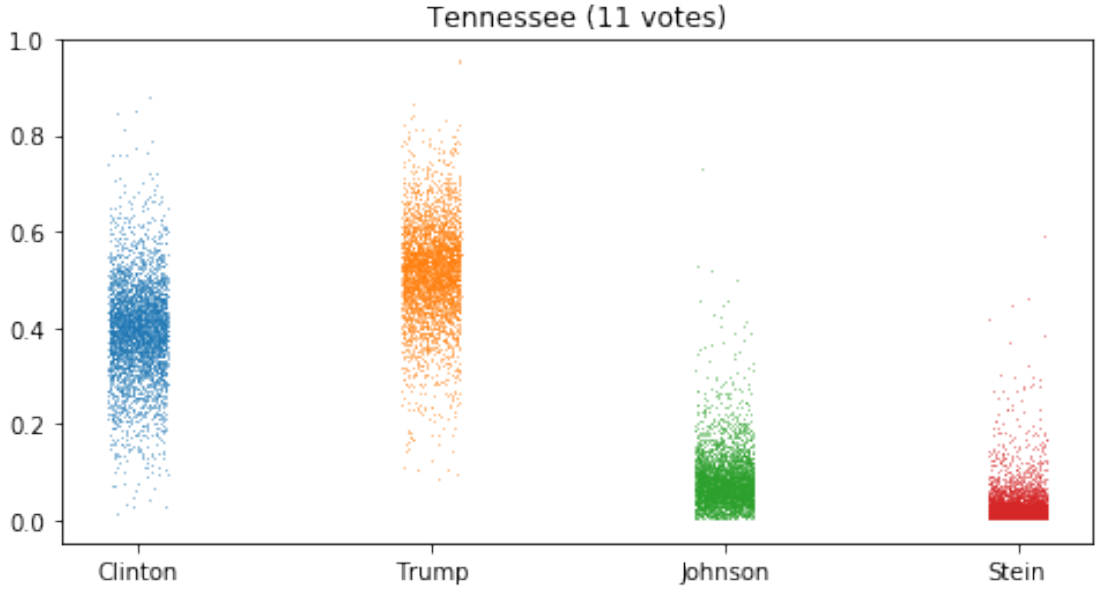

Oregon (7 votes)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

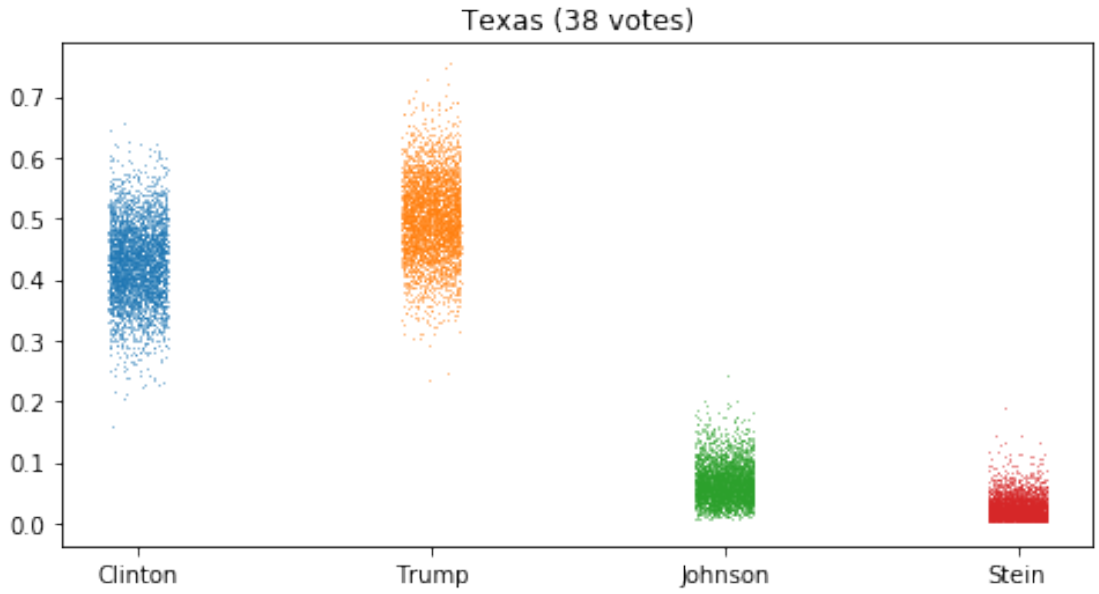

Pennsylvania (20 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

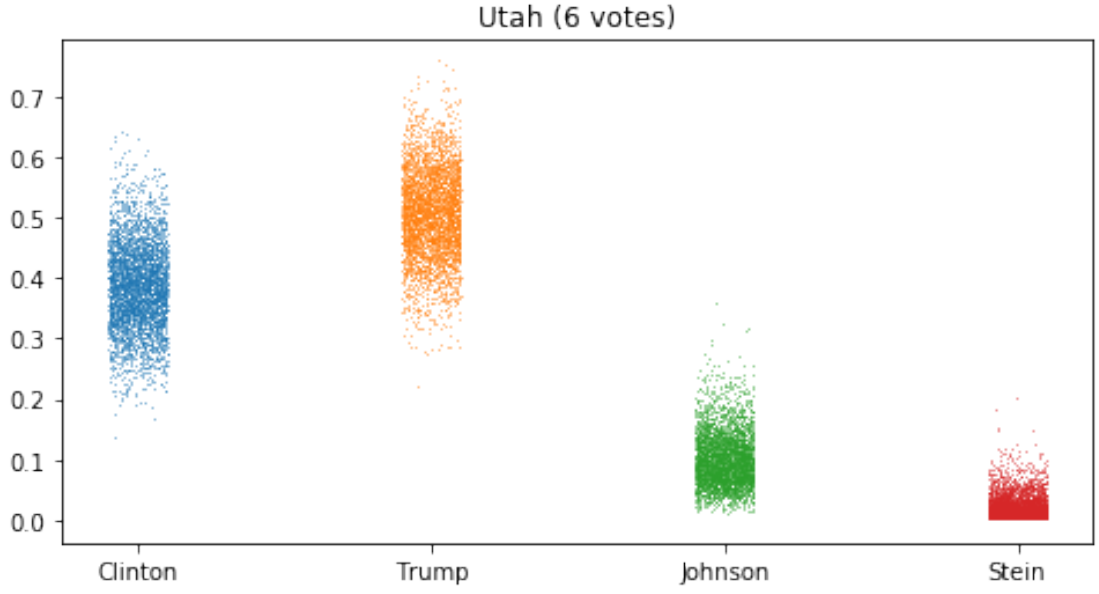

Rhode Island (4 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

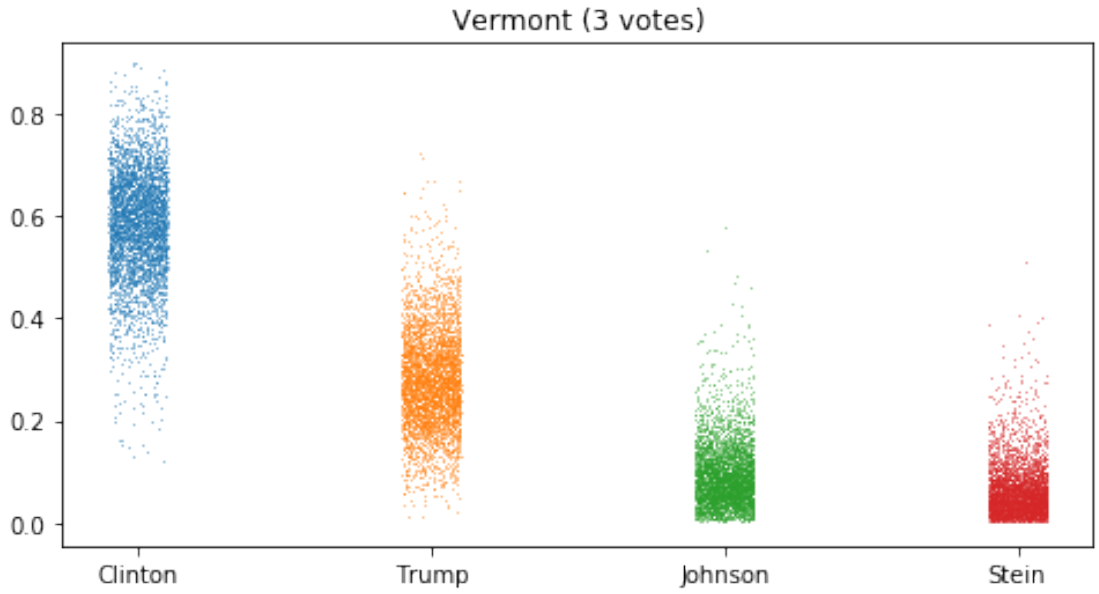

South Carolina (9 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

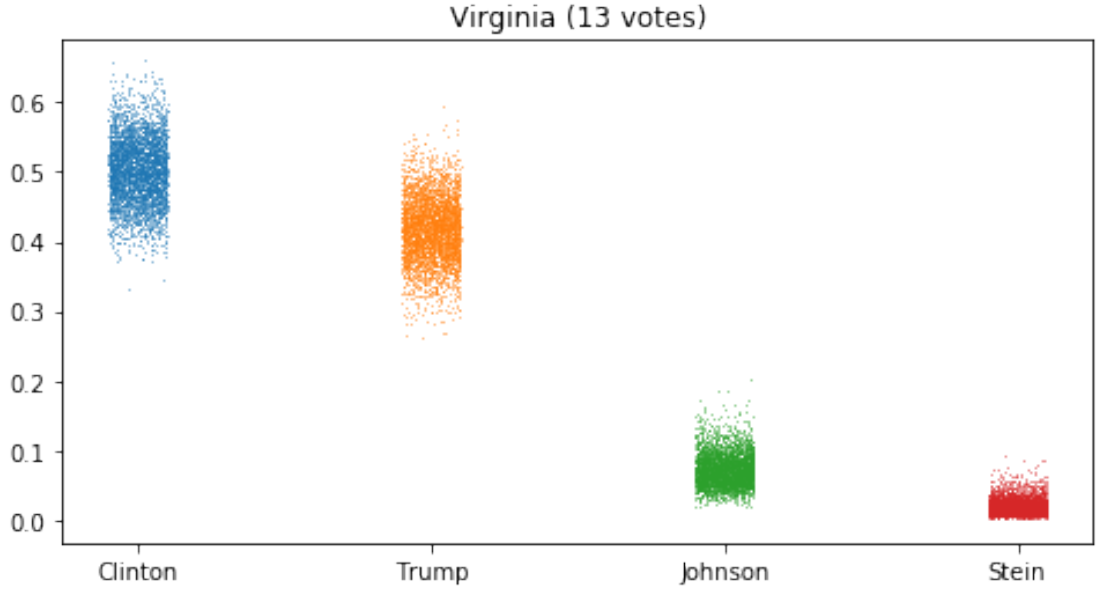

South Dakota (3 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

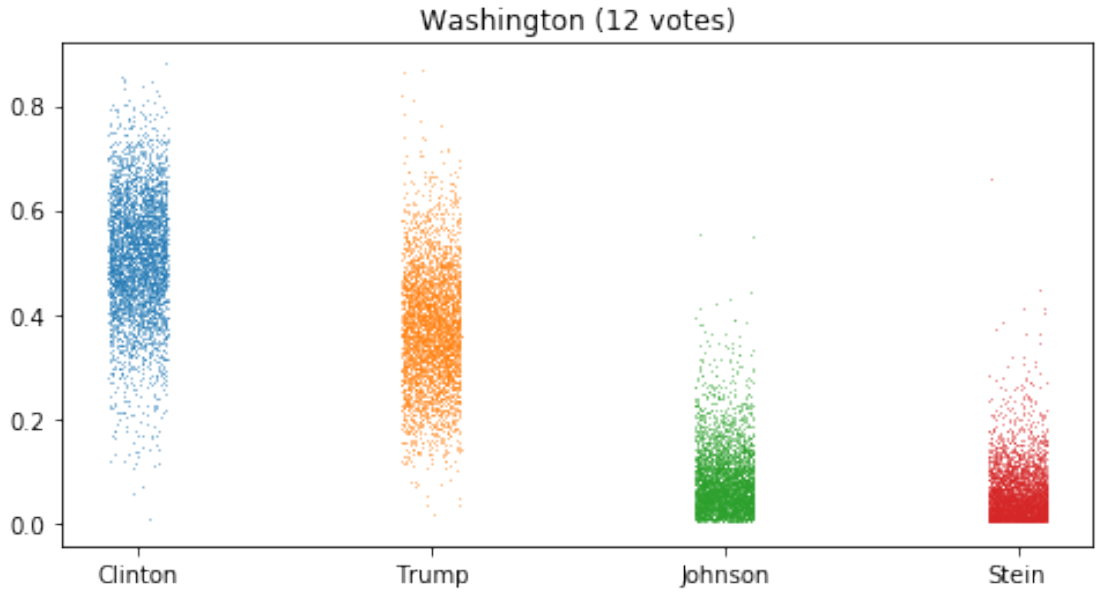

Tennessee (11 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein

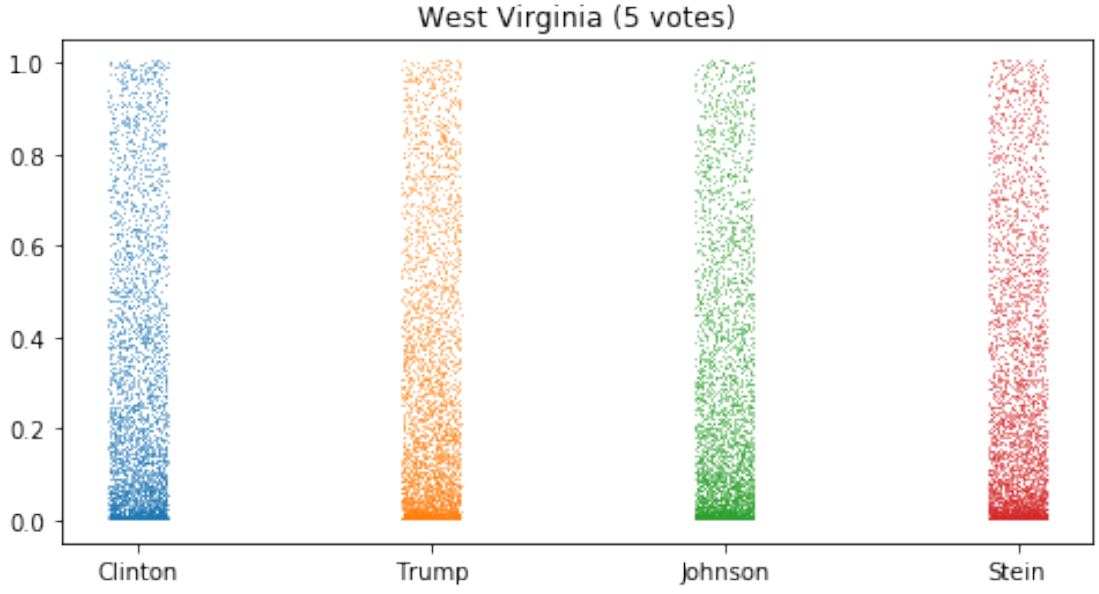

Texas (38 votes)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

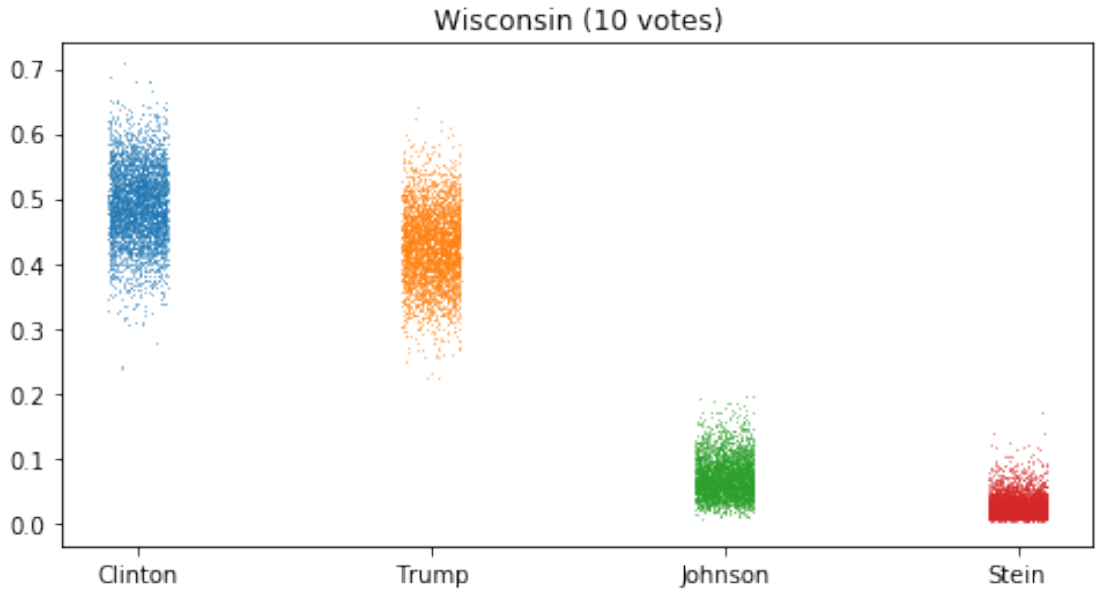

Utah (6 votes)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein

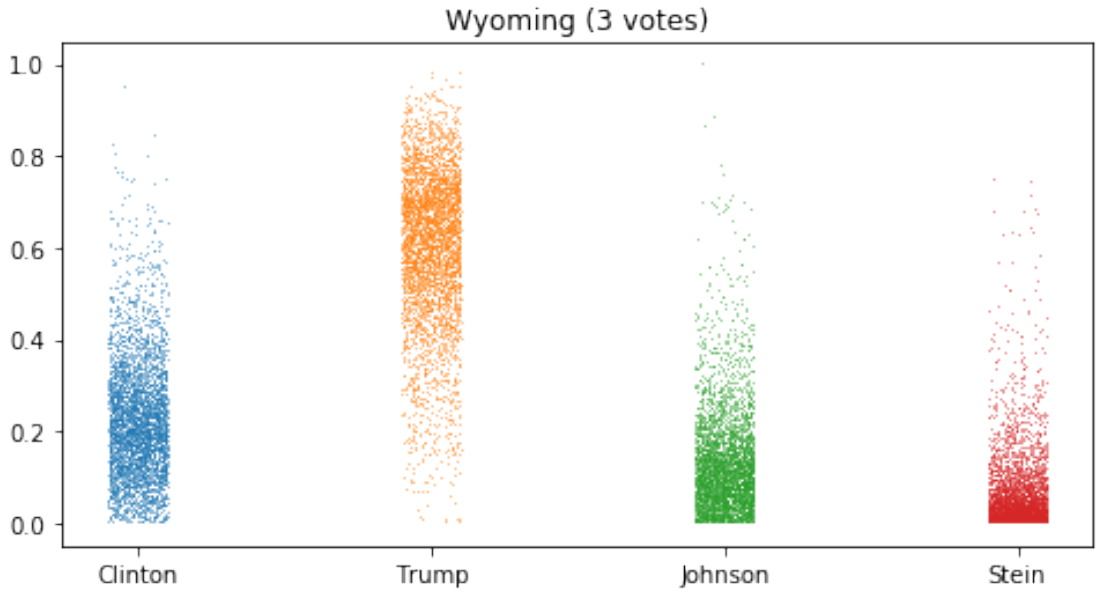

Vermont (3 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein


Virginia (13 votes)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Clinton
Trump
Johnson
Stein


Washington (12 votes)
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein


West Virginia (5 votes)
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein



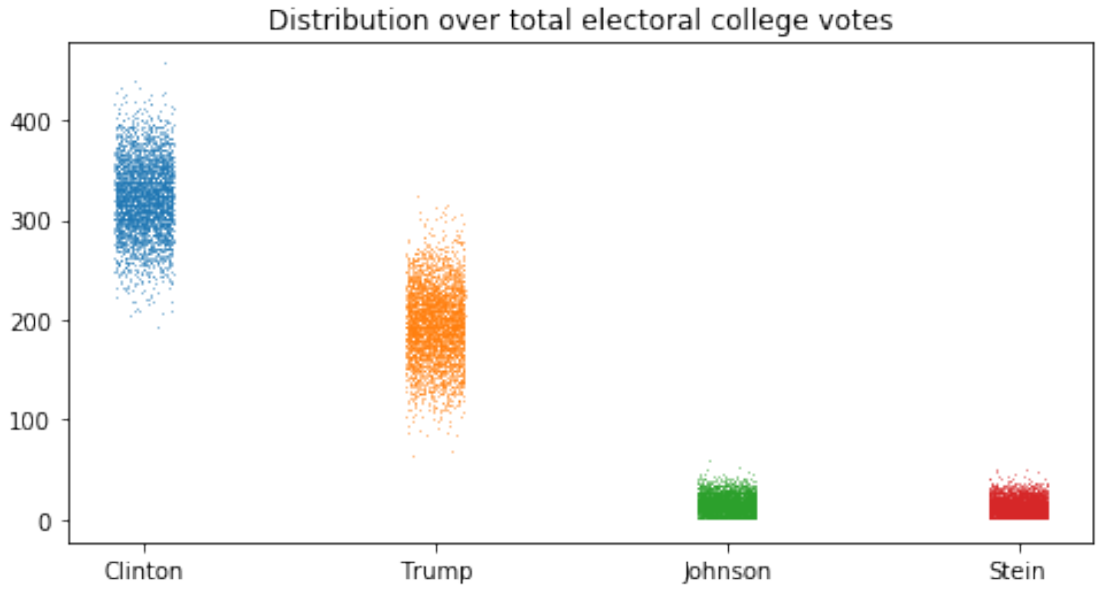


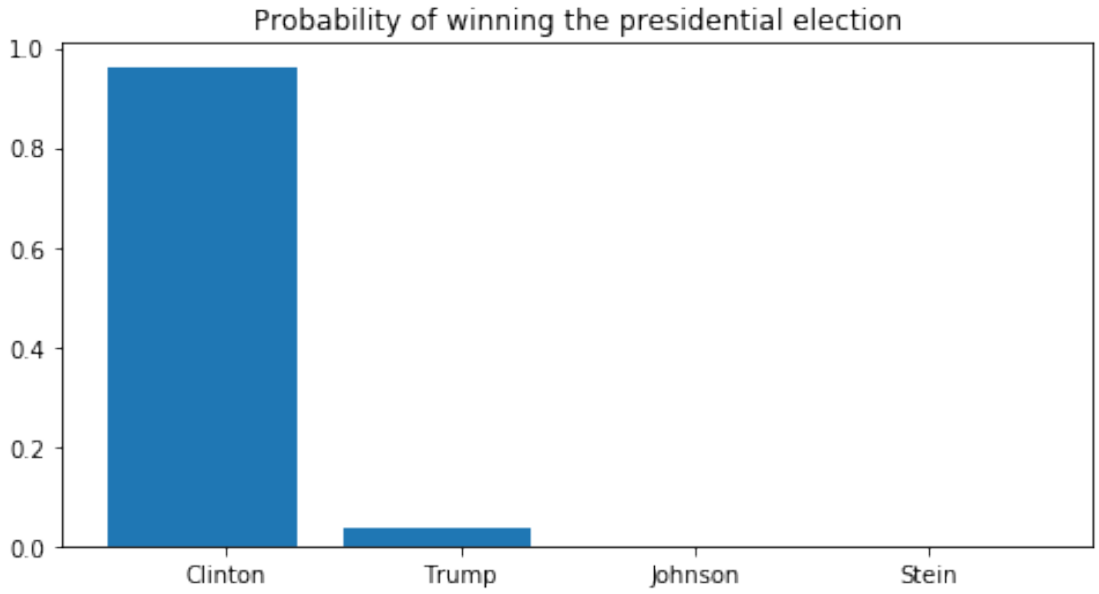

```
In [6]: # Generating winner simulation using SciPy
        samples = {state: results[state].extract() for state in states}
        total_votes = np.zeros((4000, 4))
        overall_winner = np.zeros(4000)
        for s in range(4000):
            for state in states:
                alpha = samples[state]['alpha'][s]
```

```
            p_predicted = stats.dirichlet.rvs(alpha)
            winner = p_predicted.argmax()
            total_votes[s, winner] += electoral_votes[state]
        overall_winner[s] = total_votes[s].argmax()

plt.figure(figsize=(8, 4))
for i in range(4):
    plt.plot(stats.uniform.rvs(loc=i-0.1, scale=0.2, size=4000), total_votes[:,i], ','
plt.xticks([0, 1, 2, 3], ['Clinton', 'Trump', 'Johnson', 'Stein'])
plt.title('Distribution over total electoral college votes')

plt.figure(figsize=(8, 4))
plt.hist(overall_winner, bins=[-0.5, 0.5, 1.5, 2.5, 3.5], width=0.8, align='mid', dens
plt.xticks([0, 1, 2, 3], ['Clinton', 'Trump', 'Johnson', 'Stein'])
plt.title('Probability of winning the presidential election')
plt.show()
```




Probability of winning the presidential election
1.0
0.8
0.6
0.4
0.2
0.0
Clinton
Trump
Johnson
Stein