Computational Gerrymandering

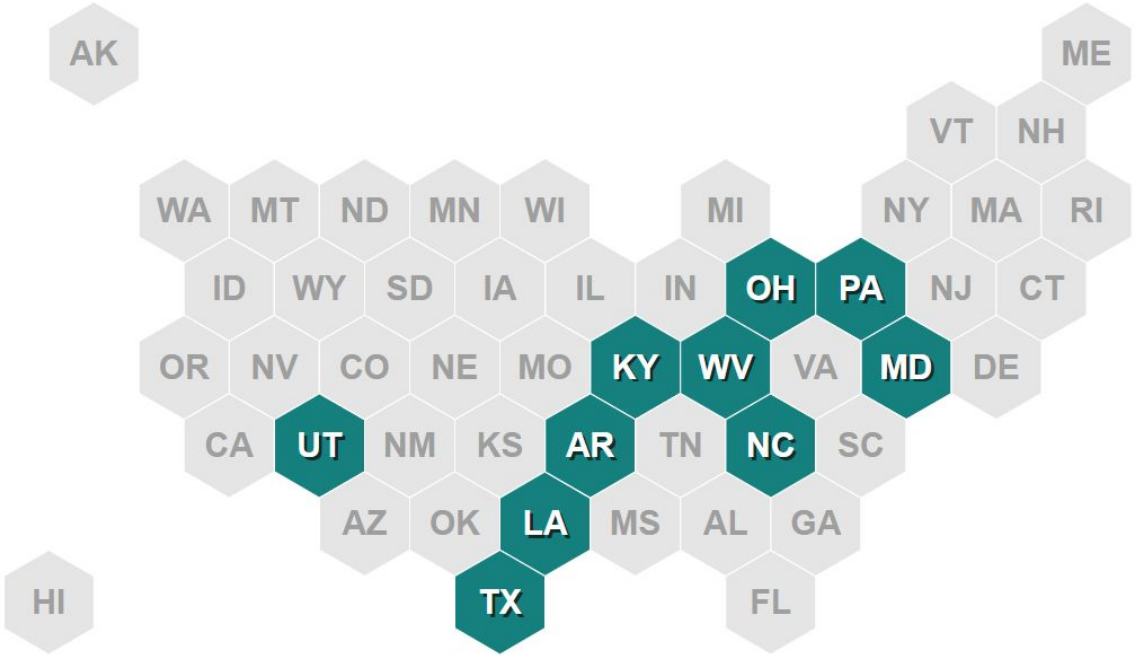
The Future Politicians (The Evil Politicians):

Mac, Jessica, Casmali, Liz

Motivation

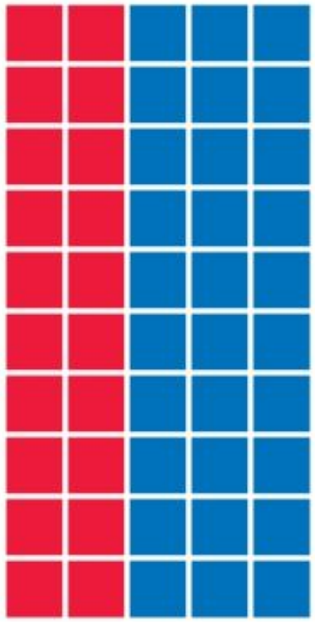
- What is gerrymandering?
- Why does it matter?

Most Gerrymandered States

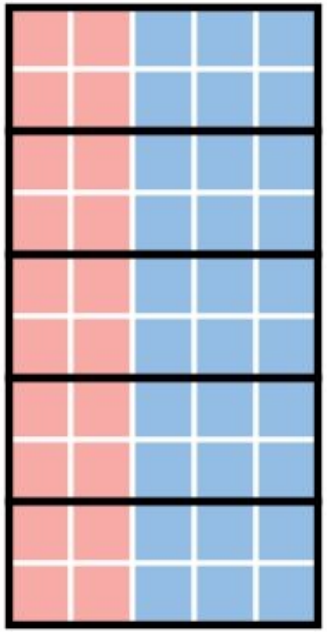


- North Carolina: racial gerrymandering
- Maryland: partisan gerrymander (benefit Democrat)
- Pennsylvania: urban areas split into rural districts
- Kentucky: Places urban populations in rural districts.
- Louisiana: partisan gerrymander (benefit Republicans)
- Utah: urban areas split into rural districts
- Texas: racial gerrymandering
- Arkansas: urban areas split into rural districts
- Ohio: partisan gerrymandering

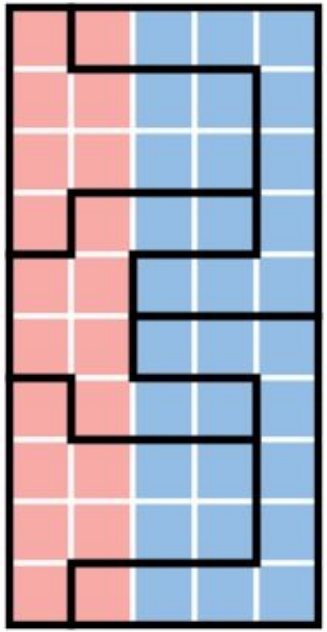
HOW TO STEAL AN ELECTION



50 PRECINCTS
60% BLUE
40% RED

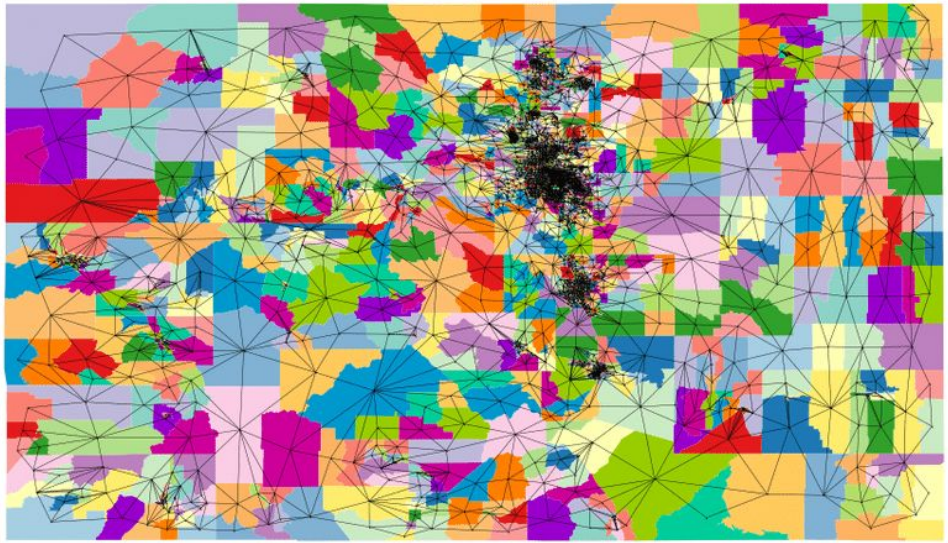


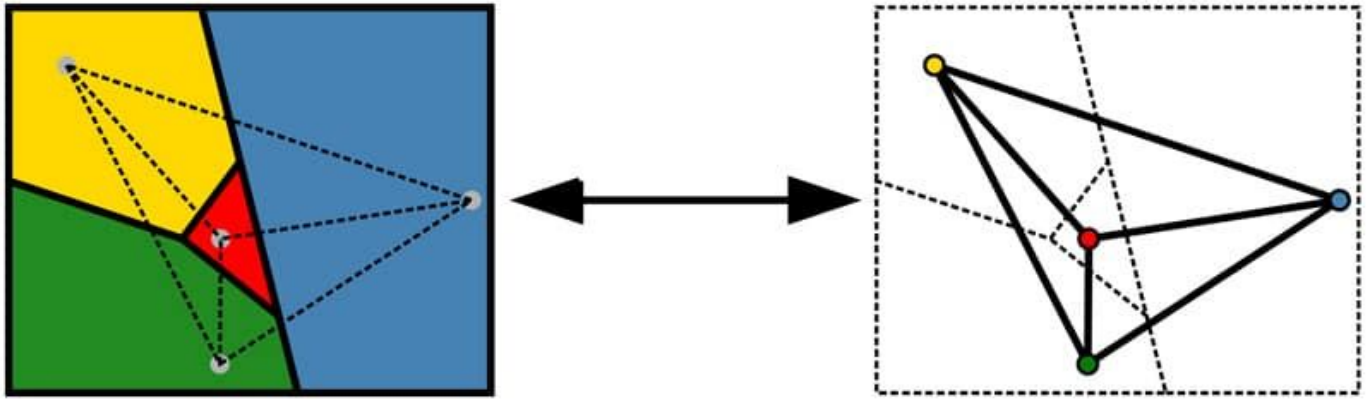
5 DISTRICTS
5 BLUE
0 RED
BLUE WINS



5 DISTRICTS
3 RED
2 BLUE
RED WINS

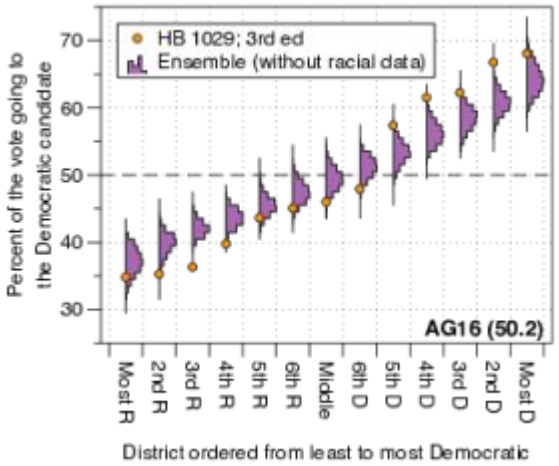
Maps and Planar Graphs

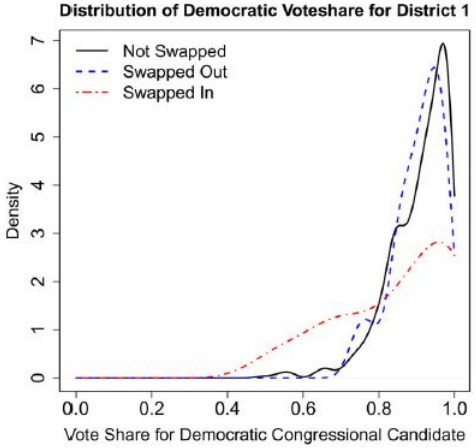


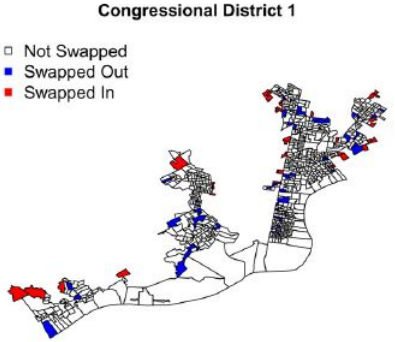


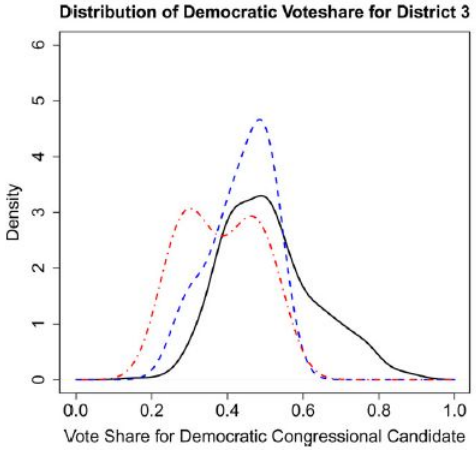
Markov Chain Monte Carlo

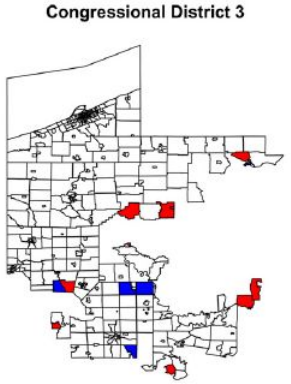
Ensemble Analysis









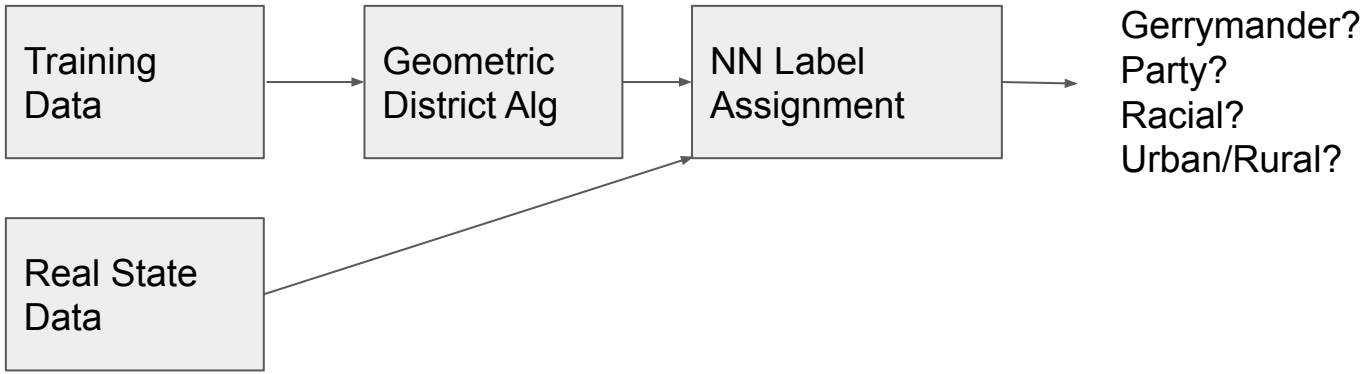


Projet Background

- Our goals
    - Create un gerrymandered and gerrymandered graphs
    - Train the algorithm on those graphs
    - Load in Colorado data
    - See what our algorithm labels the state

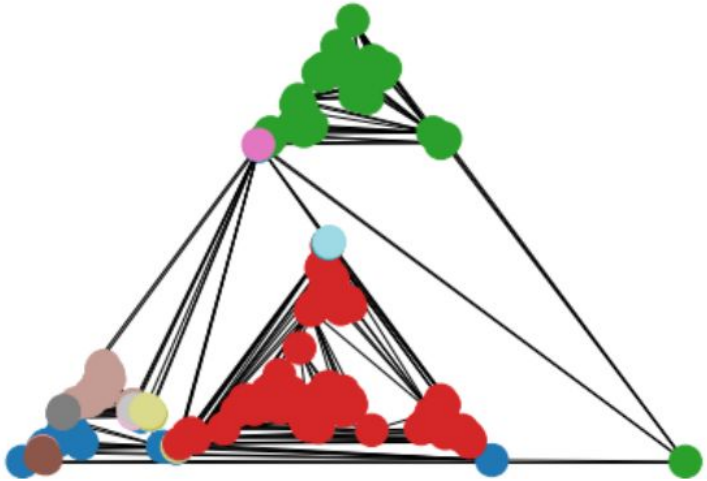
- Our approach
    - Two attempts
        - Planar graph glueing and outside detection
        - Grid graph

Application Diagram



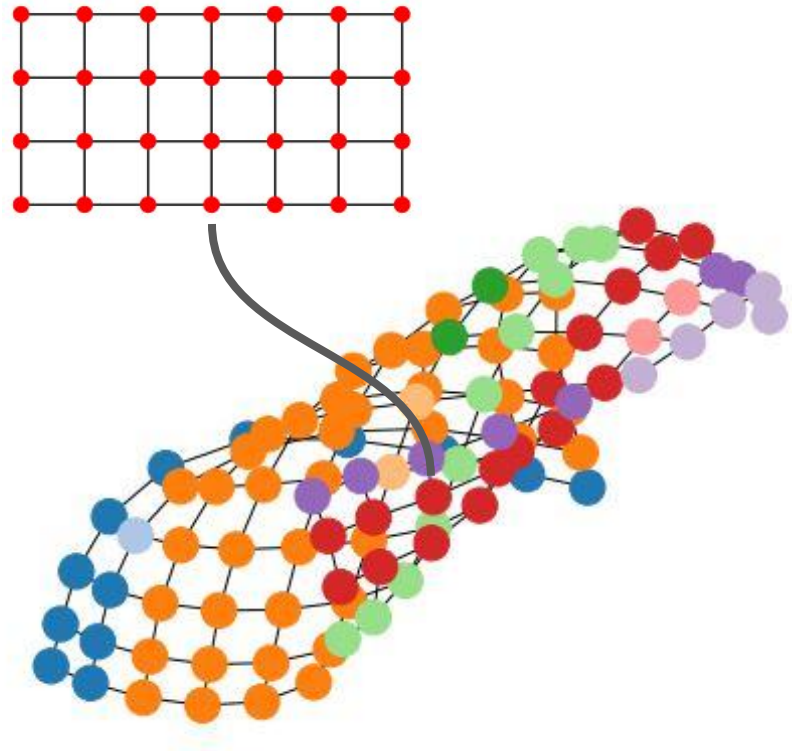
Project Mechanics - Map Creation

Original process

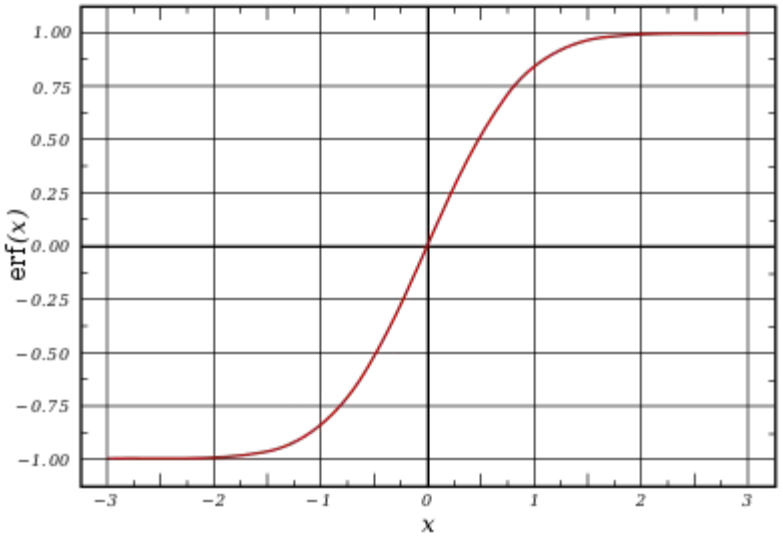


```
[[0.1009981  0.10075226 0.00024585 0.05104856 0.00195703 0.04799251]
 [0.09980574 0.08832797 0.01147777 0.05681942 0.02562083 0.01736549]
 [0.09905345 0.09678507 0.00226838 0.03029625 0.01197653 0.05678066]
 [0.09953821 0.09780275 0.00173546 0.07364988 0.00196123 0.0239271 ]
 [0.10045301 0.0992288  0.00122421 0.04163477 0.01365784 0.0451604 ]
 [0.09998666 0.09815374 0.00183291 0.02357023 0.02351368 0.05290275]
 [0.10056347 0.09895077 0.0016127  0.00400391 0.03546815 0.06109141]
 [0.09998858 0.09911259 0.00087598 0.04009564 0.01524763 0.04464531]
 [0.0995083  0.09906462 0.00044368 0.0463141  0.00156904 0.05162516]
 [0.10010449 0.09571771 0.00438678 0.0795281  0.01061299 0.0099634 ]]
[1.         0.97389629 0.02610371 0.44696085 0.14158496 0.41145419]
```

Secondary Approach



Gerrymandering Algorithm



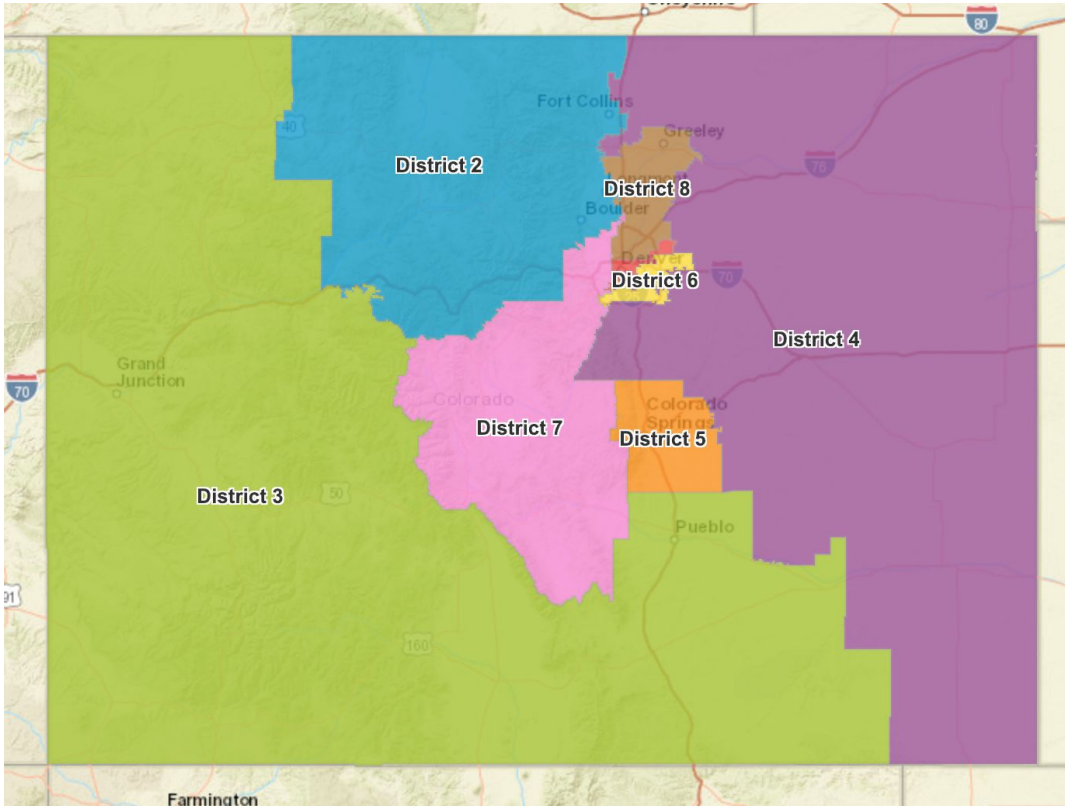
sum(fn(((Assignment.T)@partisanship@StateData)@T)

Where Assignment@Adjacency is 0, Freeze Gradient

Optimize gerrymander! (I’m sure we could sell this to someone….)

Geometric District Algorithm

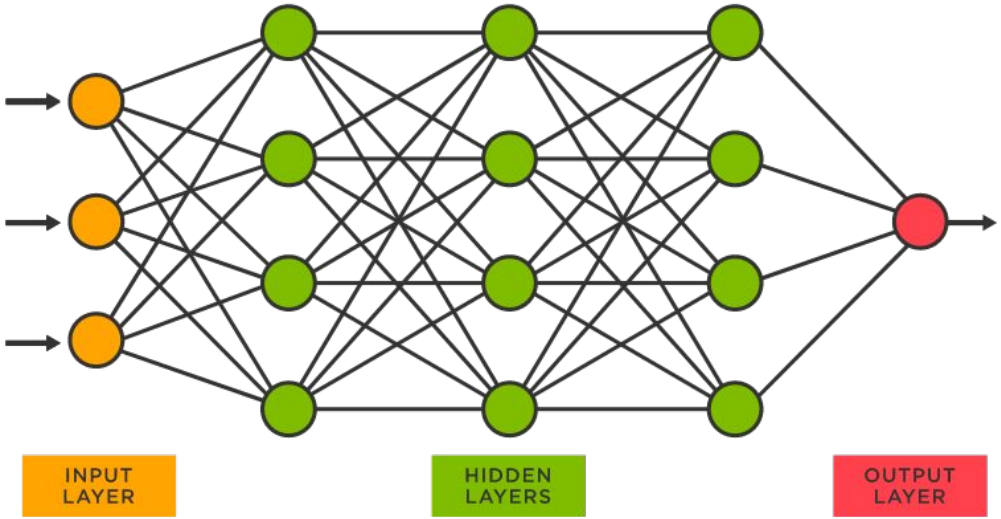
PYTORCH



Learning to Represent Clusterings

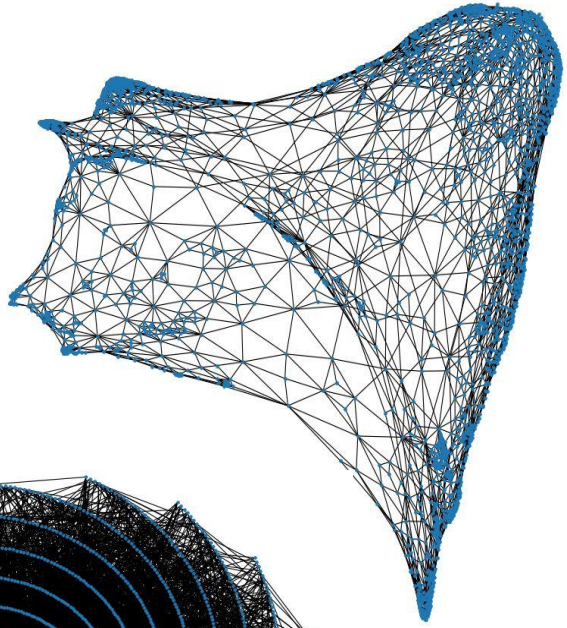
```
def forward(self):
  return softmax(mm(mm(self.A, self.X), self.hypothesis))

def h_loss(self, predicted, actual):
  return torch.sum(torch.square(torch.sub(predicted, actual)))
```

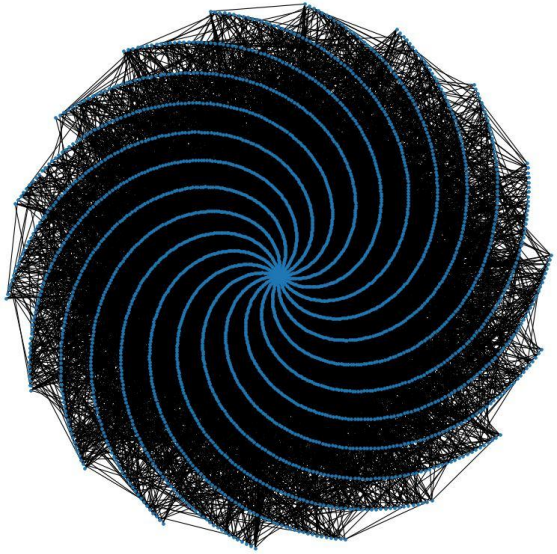
NN Label Assignment!

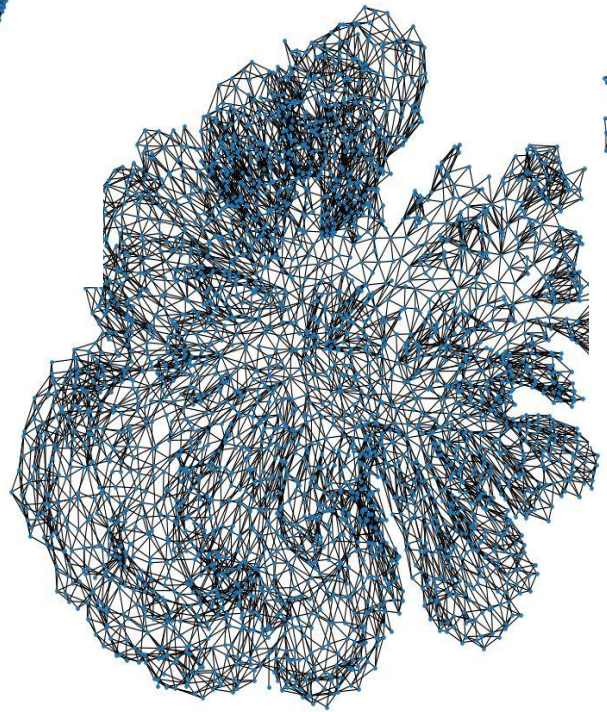


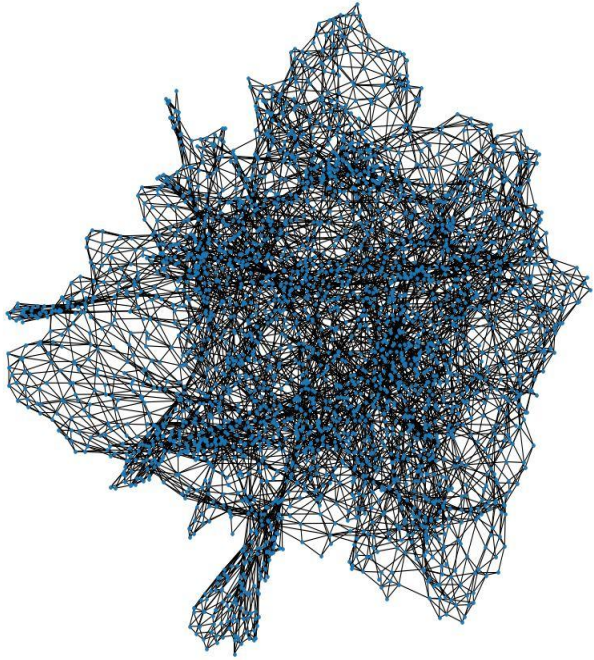
Your Governor Should Be Arrested….

Colorado Data









Results

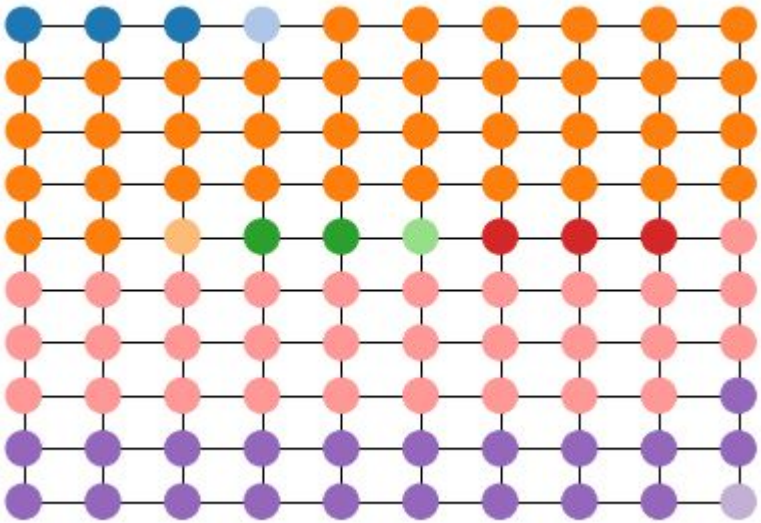
```
In [419]: proportions

Out[419]: array([1.        , 0.48698391, 0.51301609, 0.26150467, 0.526446  ,
                 0.21204933])
```

```
In [418]: majority_outcomes(state, proportions)

          [[0.0944383  0.03576582 0.05867248 0.00558752 0.079381   0.00946978]
           [0.1079178  0.05947359 0.04844421 0.04880748 0.03837716 0.02073316]
           [0.09705339 0.05587512 0.04117827 0.01464982 0.07258844 0.00981513]
           [0.09667101 0.04154762 0.05512339 0.01309335 0.07594579 0.00763187]
           [0.10848442 0.06018822 0.0482962  0.0368041  0.03956868 0.03211164]
           [0.10668739 0.04591928 0.06076811 0.0069595  0.07807447 0.02165341]
           [0.09771226 0.06733887 0.03037339 0.00619066 0.08428445 0.00723715]
           [0.09333961 0.02852738 0.06481223 0.03478451 0.03147564 0.02707945]
           [0.10713551 0.05602135 0.05111417 0.06187735 0.01570069 0.02955747]
           [0.09056032 0.03632667 0.05423364 0.03275038 0.01104968 0.04676025]]
          [0. 5. 5. 3. 6. 1.]

Out[418]: array([0. , 0.5, 0.5, 0.3, 0.6, 0.1])
```

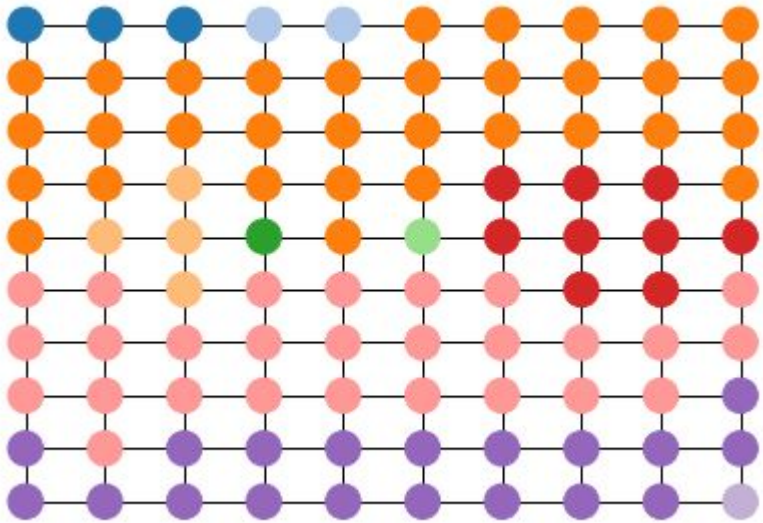
```
In [427]: majority_outcomes(optimized, proportions)

          [[1.46578262e-01 4.11436561e-02 1.05434606e-01 4.30978953e-02
            8.79148334e-02 1.55655331e-02]
           [9.70559583e-02 5.58767412e-02 4.11792171e-02 1.46514357e-02
            7.25887331e-02 9.81578949e-03]
           [5.57778426e-02 5.40957574e-02 1.68208522e-03 1.12971035e-02
            2.98433280e-02 1.46374111e-02]
           [9.66710082e-02 4.15476220e-02 5.51233862e-02 1.30933503e-02
            7.59457867e-02 7.63187127e-03]
           [1.08481846e-01 6.01865924e-02 4.82952540e-02 3.68024835e-02
            3.95683826e-02 3.21109803e-02]
           [1.55824473e-01 5.12458007e-02 1.04578672e-01 2.30709971e-02
            8.86632891e-02 4.40901867e-02]
           [9.77122574e-02 6.73388666e-02 3.03733908e-02 6.19065598e-03
            8.42844538e-02 7.23714763e-03]
           [4.42025232e-02 2.32008519e-02 2.10016713e-02 1.86730161e-02
            2.08868262e-02 4.64268092e-03]
           [1.67822102e-01 8.64888702e-02 8.13332322e-02 9.34738829e-02
            2.66314690e-02 4.77167504e-02]
           [2.98737268e-02 5.85915110e-03 2.40145757e-02 1.15385297e-03
            1.18897748e-04 2.86009761e-02]]
          [0. 6. 4. 1. 8. 1.]

Out[427]: array([0. , 0.6, 0.4, 0.1, 0.8, 0.1])
```

State partisan split [0.54527269 0.45472731]

(0.6, 0.4)



(0.5, 0.5)



Classification Results and Limitations

- Model had difficulty classifying data
- Computational limitations
  - Hard to generate enough data
- Limited information in representation
  - Can we extract feature data, spectral data ect. to learn more?
- Time restrictions
  - Could use more time to tweak structures and parameters

Theoretical Next Steps

- Run on Colorado map!
- Get more state data
- Measure different things from the graph to train on
- Do more tricky gerrymandering
- Try to do multi-level gerrymandering
- Uncover “whys” in background (both for gerrymandering and identifying it)
- Compare accuracy to MCMC
- Improve Pytorch net