

US Congress Geojsons doc

Emma Wishneski

Table of contents

```
import pandas as pd
import json

df = pd.read_csv('../iowa_2014_precinct_database.csv')
# sets the data frame to read the cleaned 2014 precinct data

congress = df[df['RaceTitle'].str.startswith('U.S. Rep')].copy()

print(f"US Congress rows: {len(congress)}")
print(f"Unique precincts: {congress['shp_idx'].nunique()}")
print(f"Candidates: {congress['CandidateName'].unique()}")

US Congress rows: 4982
Unique precincts: 1682
Candidates: ['Bryan Jack Holder' 'David Young' 'Edward B. Wright' 'Staci Appel'
'Write-in' 'Pat Murphy' 'Rod Blum' 'Dave Loebsack'
'Mariannette Miller-Meeks' 'Jim Mowrer' 'Steve King']

party_map = {
    'Republican Party': 'Republican',
    'Democratic Party': 'Democratic',
    'Libertarian Party': 'Libertarian',
}
# Rename / standardize the party names, as the dashboard expects them

congress['party'] = congress['PoliticalPartyName'].map(party_map).fillna('Other')

# Anything not listed will become other

print(f"\nParty mapping:")
for orig, mapped in zip(congress['PoliticalPartyName'], congress['party']):
    pass
print(congress.groupby(['PoliticalPartyName', 'party']).size().to_string())
```

```
# Applies the map
```

Party mapping:

PoliticalPartyName	party	
Democratic Party	Democratic	1681
Libertarian Party	Libertarian	374
Republican Party	Republican	1682

```
# build precinct_rshare - R share per precinct  
#   r_share:      Republican votes / total votes * 100  
#   r_twoparty:   Republican votes / (Republican + Democratic) * 100  
#   total_votes:  sum of all votes
```

```
precinct_rshare = {}
```

```
for shp_idx, group in congress.groupby('shp_idx'):  
    total = group['votes'].sum()  
    r_votes = group.loc[group['party'] == 'Republican', 'votes'].sum()  
    d_votes = group.loc[group['party'] == 'Democratic', 'votes'].sum()  
  
    r_share = round(float(r_votes / total * 100), 1) if total > 0 else 0.0  
    r_twoparty = round(float(r_votes / (r_votes + d_votes) * 100), 1) if (r_votes + d_votes) > 0  
  
    precinct_rshare[str(shp_idx)] = {  
        'r_share': r_share,  
        'r_twoparty': r_twoparty,  
        'total_votes': int(total)  
    }  
  
print(f"\nprecinct_rshare: {len(precinct_rshare)} precincts")  
print(f"Example (shp_idx '0'): {precinct_rshare.get('0')}")
```

```
precinct_rshare: 1682 precincts  
Example (shp_idx '0'): {'r_share': 56.9, 'r_twoparty': 56.9, 'total_votes': 202}
```

```
# Build results_district - candidate totals per district  
# Keys will be district numbers as strings ("1"-"4")
```

```
results_district = {}
```

```
for district, group in congress.groupby('congress_district'):  
    district_totals = group.groupby(['CandidateName', 'party'])['votes'].sum().reset_index()
```

```

district_totals = district_totals.sort_values('votes', ascending=False)
total_all = district_totals['votes'].sum()

results_district_list = []
for _, row in district_totals.iterrows():
    results_district_list.append({
        'CandidateName': row['CandidateName'],
        'party': row['party'],
        'votes': int(row['votes']),
        'share': round(float(row['votes'] / total_all * 100), 1)
    })
results_district[str(district)] = results_district_list

print(f"\nresults_district:")
for district, candidates in results_district.items():
    print(f" District {district}:")
    for c in candidates:
        print(f"      {c['CandidateName']} ({c['party']}): {c['votes']:,} votes ({c['share']}%)")

results_district:
District 1.0:
  Rod Blum (Republican): 147,762 votes (51.1%)
  Pat Murphy (Democratic): 141,145 votes (48.8%)
  Write-in (Other): 399 votes (0.1%)
District 2.0:
  Dave Loebsack (Democratic): 143,431 votes (52.5%)
  Mariannette Miller-Meeks (Republican): 129,455 votes (47.4%)
  Write-in (Other): 443 votes (0.2%)
District 3.0:
  David Young (Republican): 145,417 votes (52.4%)
  Staci Appel (Democratic): 117,875 votes (42.5%)
  Edward B. Wright (Libertarian): 8,932 votes (3.2%)
  Bryan Jack Holder (Other): 4,316 votes (1.6%)
  Write-in (Other): 727 votes (0.3%)
District 4.0:
  Steve King (Republican): 169,834 votes (61.6%)
  Jim Mowrer (Democratic): 105,504 votes (38.3%)
  Write-in (Other): 295 votes (0.1%)

# Build results_precinct - candidate totals per precinct
# Same structure as results_district, but keyed by shp_idx

results_precinct = {}

```

```

for shp_idx, group in congress.groupby('shp_idx'):
    total = group['votes'].sum()
    candidates = []
    for _, row in group.sort_values('votes', ascending=False).iterrows():
        candidates.append({
            'CandidateName': row['CandidateName'],
            'party': row['party'],
            'votes': int(row['votes']),
            'share': round(float(row['votes'] / total * 100), 1) if total > 0 else 0.0
        })
    results_precinct[str(shp_idx)] = candidates

print(f"\nresults_precinct: {len(results_precinct)} precincts")
print(f"Example (shp_idx '0'): {results_precinct.get('0')}")

```

results_precinct: 1682 precincts

Example (shp_idx '0'): [{'CandidateName': 'Steve King', 'party': 'Republican', 'votes': 115, 'sha

Export to JSON

office_name = 'us_congress'

```

with open(f'precinct_rshare_{office_name}.json', 'w') as f:
    json.dump(precinct_rshare, f)

```

```

with open(f'results_district_{office_name}.json', 'w') as f:
    json.dump(results_district, f)

```

```

with open(f'results_precinct_{office_name}.json', 'w') as f:
    json.dump(results_precinct, f)

```

```

print(f"\nJSON files created:")
print(f" precinct_rshare_{office_name}.json")
print(f" results_district_{office_name}.json")
print(f" results_precinct_{office_name}.json")
print(f"\nOpen the dashboard and select 'US Congress' from the dropdown to test.")

```

JSON files created:

```

precinct_rshare_us_congress.json
results_district_us_congress.json
results_precinct_us_congress.json

```

Open the dashboard and select 'US Congress' from the dropdown to test.