Knapsack Optimization Problem: Oregon Congressional Apportionment

In this notebook, you'll receive a refresher or learn about US Congressional Apportionment and how the population of each state impacts the representation received through the apportionment. Using the example of Oregon, which is likely to receive a new Congressional District through the 2021 apportionment, the nonpartisan approach below is intended to be an aid for discussion that could be used in many scenarios. As a nonpartisan approach, there will not be any data included from voter registration logs. Instead, county population data, along with mathematical optimization, is used here to align districts through population constraints.

Model notes:

The modeling approach is that of an assignment model, such as a supply chain example where Distribution Centers/Warehouses have a supply of products that they send to customer nodes which have demands for those products.

This assignment model has different constraints and objectives than most assignment models, and this unique complexity makes the problem interesting! Perhaps a similar assignment model might be if a supply node were to want to ship product equitably to multiple demand nodes, rather than based on profit, cost, revenue growth, or other economic objectives. For example, a vaccine distribution model might be have an equitable objective.

Model Objective and Constraints:

Decision Variables

- $\text{assignment}_{i,j} \in [0, 1]$: Whether the county [i] is assigned to the District [j]
- $\text{allocation}_{i,j} \in \mathbb{N}_0$: The non-negative amount of population from County [i] that is allocated to District [j]

Objective Function

- **Assignments**: Minimize the number of counties assigned to districts

$$\text{Minimize} \quad Z = \sum_{(i,j) \in \text{Counties} \times \text{Districts}} \text{assignment}_{i,j}$$

> Objective notes: In order to satisfy the constraints, all 36 counties must be assigned. But counties can be assigned to multiple districts, increasing the upper bound of assignments to [36 counties]*[6 districts] = [216 assignments]. Minimizing the number of assignments while still meeting the constraints ensures that there will not be many counties that are split among multiple districts. Requiring all counties to be assigned to only one district would make the model infeasible given the constraints to ensure the population of each district is close to equal.

Constraints

- **Allocate all population**: Each county must have exactly all population allocated to districts.

$$\sum_{j \in \text{Districts}} \text{assignment}_{i,j} = \text{county\_populations}_i \quad \forall i \in \text{Counties}$$

- **Assignment required for Allocation**: Allocation can only be greater than zero if assignment is greater than zero.

$$\sum_{(i,j) \in \text{Counties} \times \text{Districts}} \text{allocation}_{i,j} \leq \text{M} \times \text{assignment}_{i,j}$$

- **Completeness Constraint 1**: At least 20% of a county population must be allocated to a district if that county is assigned to that district.

$$\text{If assignment}_{i,j} = 1 \text{ then} \sum_{(i,j) \in \text{Counties} \times \text{Districts}} \text{allocation}_{i,j} \geq 0.20 \times \text{county\_population}_{i,j} \times \text{assignment}_{i,j}$$

- **Completeness Constraints 2 and 3**: All counties may be assigned to up to 1 district, but only counties with a population of at least 220,000 may be assigned to up to 2 districts.

$$\text{If county\_populations}_i \leq 220{,}000 \text{ then} \sum_{j \in \text{Districts}} \text{assignment}_{i,j} \leq 1 \quad \forall i \in \text{Counties}$$

$$\text{Else} \sum_{j \in \text{Districts}} \text{assignment}_{i,j} \leq 2$$

In [1]:

```
# initializing useful modules, packages, or libraries
import folium # choropleth maps
import geopandas as gpd # shapefile for Oregon county maps
import numpy as np  # data, np arrays are faster than lists
import pandas as pd # data
from PIL import Image, ImageOps # images
from plotnine import ggplot, aes, geom_map, geom_text, geom_label # pl
ots
from plotnine import * # needed for theme
from pulp import * # for the optimization model with linear programmin
g
```

Oregon's population grew since 2010

- Each of the Districts had significant growth
- Oregon's total population grew by about an estimated 9%, outpacing the US population's 6% growth
- These numbers will be updated once the 2020 Census is completed and public

In [2]:

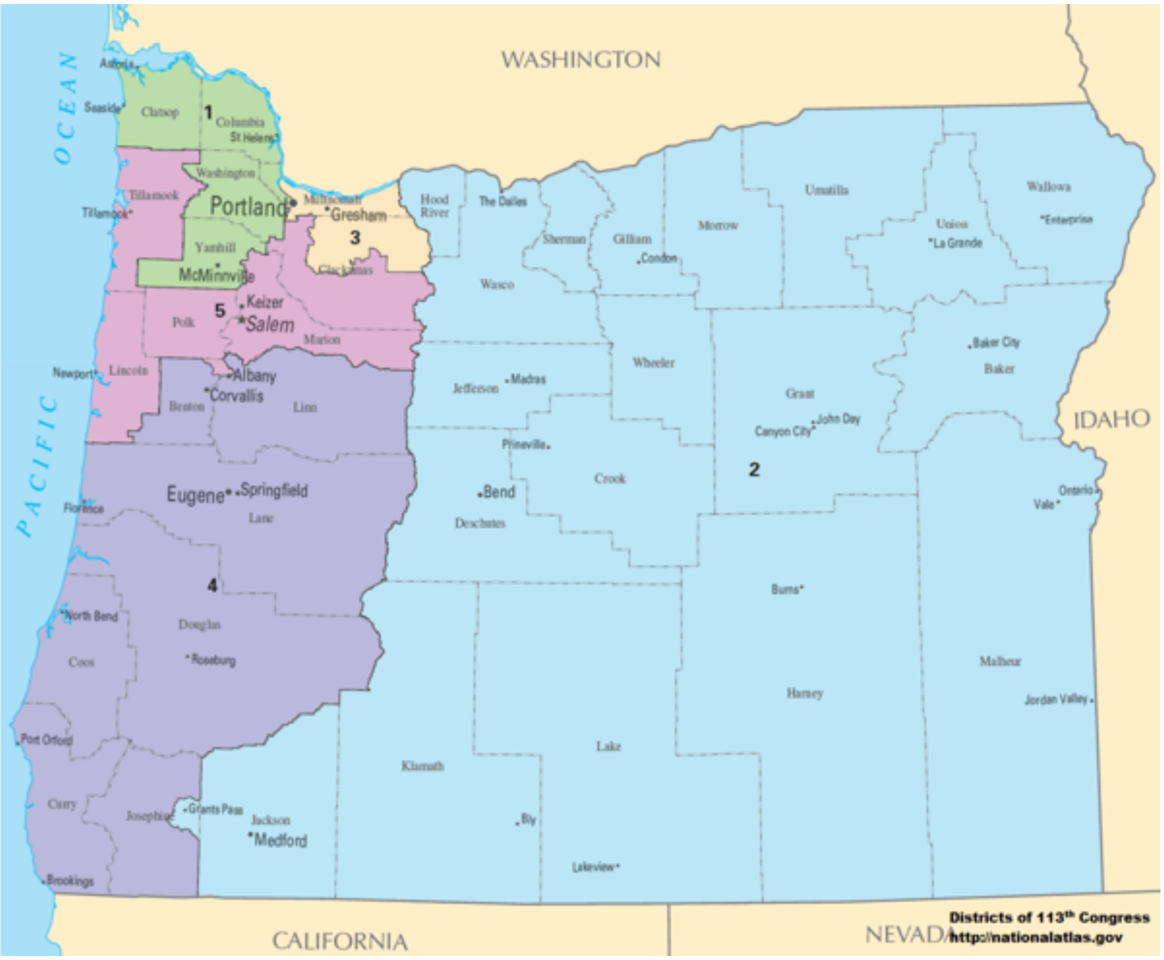
```
district_populations = pd.DataFrame({'District': [1,2,3,4,5],
                                     'Population2018' : [775806,77040
3,782486,770184,772980],
                                     'Population2010' : [858875,84102
2,853116,820504,844220],
                                     'Change': [83069, 70619, 70630, 5
0320, 71240]})
district_populations = district_populations.style.format('{:,}')
district_populations
```

Out[2]:

```
   District  Population2018  Population2010  Change
0         1         775,806         858,875  83,069
1         2         770,403         841,022  70,619
2         3         782,486         853,116  70,630
3         4         770,184         820,504  50,320
4         5         772,980         844,220  71,240
```

In [3]:
```
Image.open('Oregon_Congressional_Districts,_113th.png').resize((600, 4
92))
```

Out[3]:



Oregon is likely to gain the 6th Congressional District in from the 2020 Census

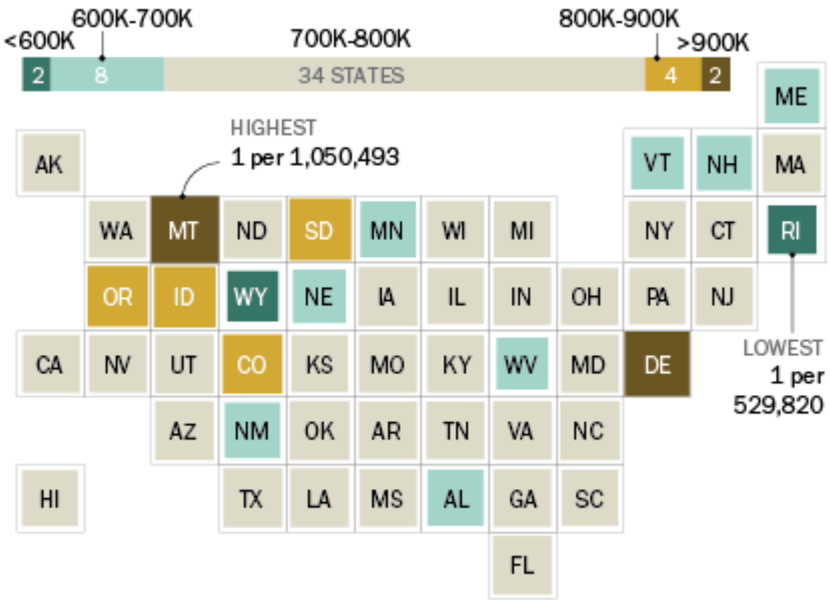
- Oregon was already close to gaining the 6th Congressional District from the 2010 Census, according to the following infographic from the Pew Research Center.
- With the higher than average population growth rate, drawing the 6th District will have representative impact for the next decade and influence beyond the next decade.

```
In [4]: Image.open('FT_18.05.18_RepresentationRatios_states.png')
```

```
Out[4]:
```

Wide range of representation ratios across states

Number of people represented by one lawmaker



Note: Data as of July 1, 2017. Representation ratio calculated as the ratio of voting members of the U.S. House of Representatives to resident population estimates of represented states.
Source: Pew Research Center analysis of U.S. Census Bureau data.

PEW RESEARCH CENTER

Let's review Oregon population by county

In [5]:

```
# county_id = list(range(0, 36))
county_id = np.arange(0, 36)
county_names = np.array(['Baker','Benton','Clackamas','Clatsop','Colum
bia','Coos','Crook','Curry','Deschutes','Douglas','Gilliam','Grant','H
arney','Hood River','Jackson','Jefferson','Josephine','Klamath','Lake
','Lane','Lincoln','Linn','Malheur','Marion','Morrow','Multnomah','Pol
k','Sherman','Tillamook','Umatilla','Union','Wallowa','Wasco','Washing
ton','Wheeler','Yamhill'])
population_by_county = pd.DataFrame({'County_ID': county_id,
                                     'County_Name': county_names,
                                     'Population2018' : [16765,93590,4
19425,39200,51900,63275,22710,22915,188980,111735,1985,7400,7380,2531
0,219200,23560,86395,67960,8115,375120,48210,125575,31925,344035,1188
5,813300,82100,1785,26395,80765,26885,7175,27200,606280,1450,107415],
                                     'Population2010' : [16134,85579,3
75992,37039,49351,63043,20978,22364,157733,107667,1871,7445,7422,2234
6,203206,21720,82713,66380,7895,351715,46034,116672,31313,315335,1117
3,735334,75403,1765,25250,75889,25748,7008,25213,529710,1441,99193],
                                     'Change2010_2018': [631,8011,4343
3,2161,2549,232,1732,551,31247,4068,114,-45,-42,2964,15994,1840,3682,1
580,220,23405,2176,8903,612,28700,712,77966,6697,20,1145,4876,1137,16
7,1987,76570,9,8222],
                                     'Latitude': [44.7346,44.4929,45.3
088,46.1068,45.9189,43.175,44.1533,42.6002,43.9856,43.253,45.4204,44.5
335,43.2214,45.6007,42.4441,44.4914,42.3351,42.5663,42.7821,44.0123,4
4.6733,44.4924,43.9454,44.9367,45.4757,45.5437,44.9262,45.4041,45.395
7,45.726,45.3181,45.5356,45.3856,45.5404,44.7845,45.2256],
                                     'Longitude': [-117.6777,-123.384
4,-122.3999,-123.8773,-122.9863,-124.179,-120.4523,-124.3343,-121.169
9,-123.373,-120.2077,-119.0668,-119.0481,-121.7147,-122.7875,-121.324
6,-123.5119,-121.6302,-120.4691,-123.1668,-123.9267,-122.7806,-117.48
4,-122.7301,-119.6694,-122.5346,-123.3237,-120.7307,-123.8622,-118.74
5,-117.9619,-117.2036,-121.2283,-123.002,-120.02,-123.1982]})
shapefile_oregon = gpd.read_file('orcounty.shp')
map_population_by_county_data = shapefile_oregon.merge(population_by_c
ounty, left_on='NAME', right_on='County_Name',suffixes=('_left', '_rig
ht'))
county_populations = np.array(population_by_county['Population2018'])
state_population = sum(county_populations)
population_by_county.head(3).append(population_by_county.tail(3))
```

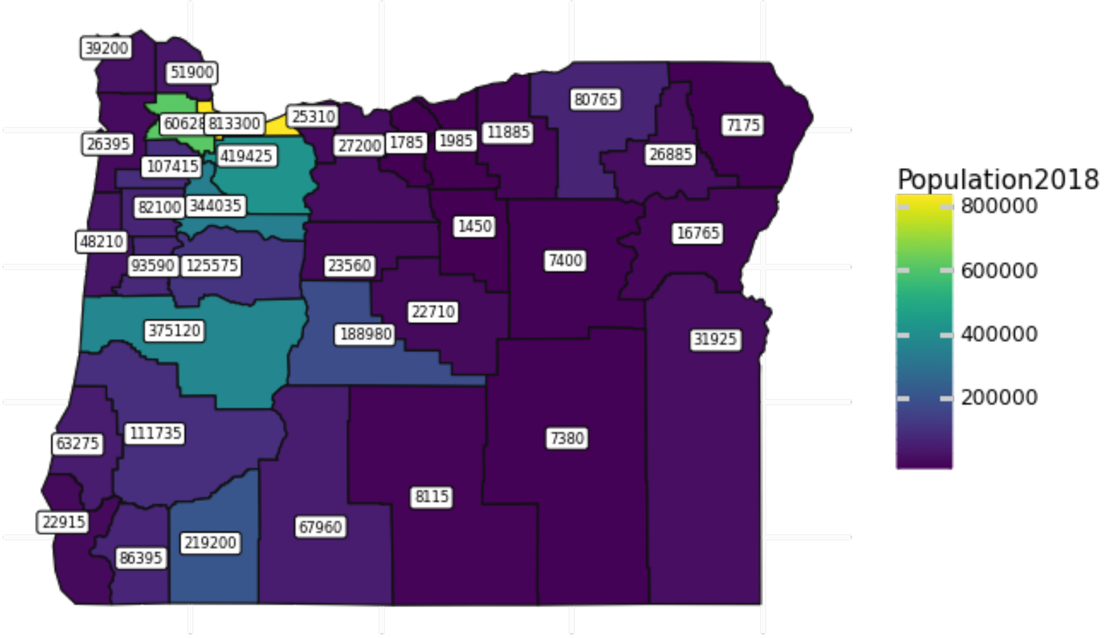
Out[5]:

	County_ID	County_Name	Population2018	Population2010	Change2010_2018	Latitude	Lo
0	0	Baker	16765	16134	631	44.7346	-1
1	1	Benton	93590	85579	8011	44.4929	-1
2	2	Clackamas	419425	375992	43433	45.3088	-1
33	33	Washington	606280	529710	76570	45.5404	-1
34	34	Wheeler	1450	1441	9	44.7845	-1
35	35	Yamhill	107415	99193	8222	45.2256	-1

In [6]:

```
map_population_by_county = (
ggplot(map_population_by_county_data)
+ geom_map(aes(fill='Population2018'))
+ geom_label(aes(x = 'Longitude', y = 'Latitude', label='Population2018',size=2), show_legend=False)
+ theme_minimal()
+ theme(axis_text_x=element_blank(),
        axis_text_y=element_blank(),
        axis_title_x=element_blank(),
        axis_title_y=element_blank(),
        axis_ticks=element_blank(),
        panel_grid_major = element_blank()
       )
)
print('\033[1m'+'Population by County: The Willamette River valley contributes to ~70% of the state population')
map_population_by_county
```

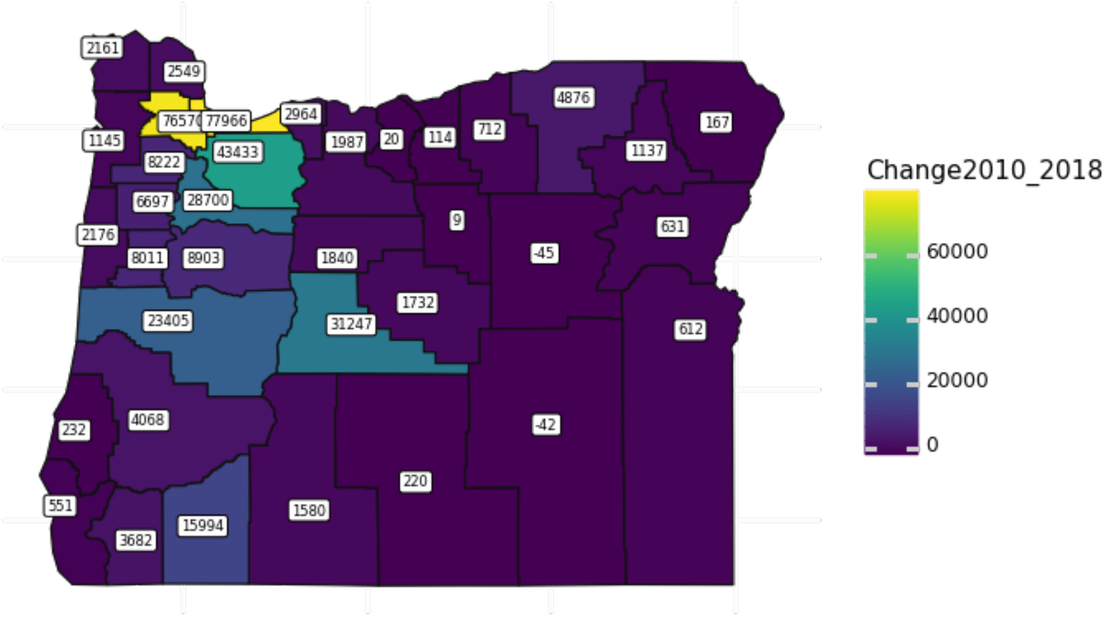
Population by County: The Willamette River valley contributes to ~70% of the state population



Out[6]: `<ggplot: (8774582501367)>`

In [7]:
```
print('\033[1m'+'Population growth from 2010 to 2018 (estimated) occur
red in the most populous counties.')
map_population_change_by_county = (
ggplot(map_population_by_county_data)
+ geom_map(aes(fill='Change2010_2018'))
+ geom_label(aes(x = 'Longitude', y = 'Latitude', label='Change2010_20
18', size=2), show_legend=False)
+ theme_minimal()
+ theme(axis_text_x=element_blank(),
        axis_text_y=element_blank(),
        axis_title_x=element_blank(),
        axis_title_y=element_blank(),
        axis_ticks=element_blank(),
        panel_grid_major = element_blank()
       )
)
map_population_change_by_county
```

```
Population growth from 2010 to 2018 (estimated) occurred in the most
populous counties.
```



Out[7]:
```
<ggplot: (8774592349928)>
```

Optimization Model

The first step in this congressional apportionment modeling process, or algorithm, is to run the following optimization model. After running the initial model, additional constraints will be added to hone in on the solution.

In [8]:

```
n_counties = 36
n_districts = 6
model = LpProblem("Supply-Demand-Problem", LpMinimize) # create model
variable_names = [str(i)+str(j) for j in range(1, n_districts+1) for i in range(1, n_counties+1)]
variable_names.sort() # print("Variable Indices:", variable_names)

# decision variables
# assignment is whether or not the county is assigned to the district
DV_variable_y = LpVariable.matrix("Y",variable_names,cat="Binary")
assignment = np.array(DV_variable_y).reshape(36,6)

# allocation is the amount of population from a county to a district
DV_variable_x = LpVariable.matrix("X",variable_names,cat="Integer",lowBound=0)#upBound N/A
allocation = np.array(DV_variable_x).reshape(36,6)
```

In [9]:

```
# This objective function minimizes the counties are split among multiple districts.
objective_function = pulp.lpSum(assignment)
```

In [10]:

```
# Initial Assignment / Allocation Constraints

# allocate exactly 100% of population from each county
for i in range(n_counties):
    model += lpSum(allocation[i][j] for j in range(n_districts)) == county_populations[i] , "Allocate All " + str(i)

for i in range(n_counties):
    for j in range(n_districts):
        # allocation can only be greater than zero if assignment is greater than zero
        # sum(county_populations) is a big M, which is the Oregon total population
        model += allocation[i][j] <= sum(county_populations)*assignment[i][j] , "Allocation assignment " + str(i) + str(j)
        if assignment[i][j] == 1:
            # at least 20% of population must be allocated to each district for that county
            model += allocation[i][j] >= assignment[i][j]*0.20*county_populations[i] , "Allocation min " + str(i) + str(j)

# Contiguous districts constraints
# e.g. Coos County (5) borders only Curry County (7) or Douglas County (8)
#      Therefore at least (7) or (8) need to be allocated to any district that has (5)
for j in range(n_districts):
    model += assignment[0][j] <= assignment[11][j]+assignment[22][j]+assignment[30][j]+assignment[31][j]
    model += assignment[1][j] <= assignment[19][j]+assignment[20][j]+assignment[21][j]+assignment[26][j]
    model += assignment[2][j] <= assignment[13][j]+assignment[23][j]+assignment[25][j]+assignment[32][j]+assignment[33][j]+assignment[35][j]
    model += assignment[3][j] <= assignment[4][j]+assignment[28][j]
    model += assignment[4][j] <= assignment[3][j]+assignment[25][j]+assignment[33][j]
    model += assignment[5][j] <= assignment[7][j]+assignment[9][j]
    model += assignment[6][j] <= assignment[8][j]+assignment[11][j]+assignment[12][j]+assignment[15][j]+assignment[34][j]
    model += assignment[7][j] <= assignment[5][j]+assignment[9][j]+assignment[16][j]
    model += assignment[8][j] <= assignment[6][j]+assignment[12][j]+assignment[15][j]+assignment[17][j]+assignment[18][j]+assignment[19][j]+assignment[21][j]
    model += assignment[9][j] <= assignment[5][j]+assignment[7][j]+assignment[14][j]+assignment[16][j]+assignment[17][j]+assignment[19][j]
    model += assignment[10][j] <= assignment[24][j]+assignment[27][j]+assignment[32][j]+assignment[34][j]
    model += assignment[11][j] <= assignment[0][j]+assignment[6][j]+assignment[12][j]+assignment[22][j]+assignment[24][j]+assignment[29][j]+assignment[30][j]+assignment[34][j]
    model += assignment[12][j] <= assignment[6][j]+assignment[8][j]+assignment[11][j]+assignment[18][j]+assignment[22][j]
    model += assignment[13][j] <= assignment[2][j]+assignment[25][j]+a
```

```
ssignment[32][j]
    model += assignment[14][j] <= assignment[9][j]+assignment[16][j]+assignment[17][j]
    model += assignment[15][j] <= assignment[6][j]+assignment[8][j]+assignment[21][j]+assignment[23][j]+assignment[32][j]+assignment[34][j]
    model += assignment[16][j] <= assignment[7][j]+assignment[9][j]+assignment[14][j]
    model += assignment[17][j] <= assignment[8][j]+assignment[9][j]+assignment[14][j]+assignment[18][j]+assignment[19][j]
    model += assignment[18][j] <= assignment[8][j]+assignment[12][j]+assignment[17][j]
    model += assignment[19][j] <= assignment[1][j]+assignment[8][j]+assignment[9][j]+assignment[17][j]+assignment[20][j]+assignment[21][j]
    model += assignment[20][j] <= assignment[1][j]+assignment[19][j]+assignment[26][j]+assignment[28][j]
    model += assignment[21][j] <= assignment[1][j]+assignment[8][j]+assignment[15][j]+assignment[19][j]+assignment[23][j]+assignment[26][j]
    model += assignment[22][j] <= assignment[0][j]+assignment[11][j]+assignment[12][j]
    model += assignment[23][j] <= assignment[2][j]+assignment[15][j]+assignment[21][j]+assignment[26][j]+assignment[32][j]+assignment[35][j]
    model += assignment[24][j] <= assignment[10][j]+assignment[11][j]+assignment[29][j]+assignment[34][j]
    model += assignment[25][j] <= assignment[2][j]+assignment[4][j]+assignment[13][j]+assignment[33][j]
    model += assignment[26][j] <= assignment[1][j]+assignment[20][j]+assignment[21][j]+assignment[23][j]+assignment[28][j]+assignment[35][j]
    model += assignment[27][j] <= assignment[10][j]+assignment[32][j]
    model += assignment[28][j] <= assignment[3][j]+assignment[20][j]+assignment[26][j]+assignment[33][j]+assignment[35][j]
    model += assignment[29][j] <= assignment[11][j]+assignment[24][j]+assignment[30][j]+assignment[31][j]
    model += assignment[30][j] <= assignment[0][j]+assignment[11][j]+assignment[29][j]+assignment[31][j]
    model += assignment[31][j] <= assignment[0][j]+assignment[29][j]+assignment[30][j]
    model += assignment[32][j] <= assignment[2][j]+assignment[10][j]+assignment[13][j]+assignment[15][j]+assignment[23][j]+assignment[27][j]+assignment[34][j]
    model += assignment[33][j] <= assignment[2][j]+assignment[4][j]+assignment[25][j]+assignment[28][j]+assignment[35][j]
    model += assignment[34][j] <= assignment[6][j]+assignment[10][j]+assignment[11][j]+assignment[15][j]+assignment[24][j]+assignment[32][j]
    model += assignment[35][j] <= assignment[2][j]+assignment[23][j]+assignment[26][j]+assignment[28][j]+assignment[33][j]

# District size constraints, in order to keep the size of districts by population similar
for j in range(n_districts):
    model += lpSum(allocation[i][j] for i in range(n_counties)) <= 750000 , "District Size Maximum " + str(j)
    model += lpSum(allocation[i][j] for i in range(n_counties)) >= 650000 , "District Size Minimum " + str(j)
```

```
# Only allow counties that meet certain critera to be split among mult
iple districts
# A county must have population > 220,000 to be split among up to two
districts
for i in range(n_counties): # added
    if county_populations[i] <= 220000:
        model += lpSum(assignment[i][j] for j in range(n_districts))
<= 1  , "Unique Assignment " + str(i)
    else:
        model += lpSum(assignment[i][j] for j in range(n_districts))
<= 2  , "Up-to-two Assignments " + str(i)
```

In [11]:

```
model.solve(PULP_CBC_CMD())
print('The model status is: ',LpStatus[model.status])
print('The objective value is: ', pulp.value(objective_function))
```

```
The model status is:  Optimal
The objective value is:  40.0
```

Since there are 36 counties, the unconstrainted lower bound for the objective function would be 36. However, an objective value of 40.0 means that there are 4 occassions that a county was assigned to two districts.

Results (first pass)

The map below will show why more constraints will be added to the model below. Although the constraints have been satisfied, districts have multiple clusters that are not connected to each other. The constraints to eliminate multiple districts are sometimes referred to as Cut constraints (https://en.wikipedia.org/wiki/Cutting-plane_method) in Operations Research applications.

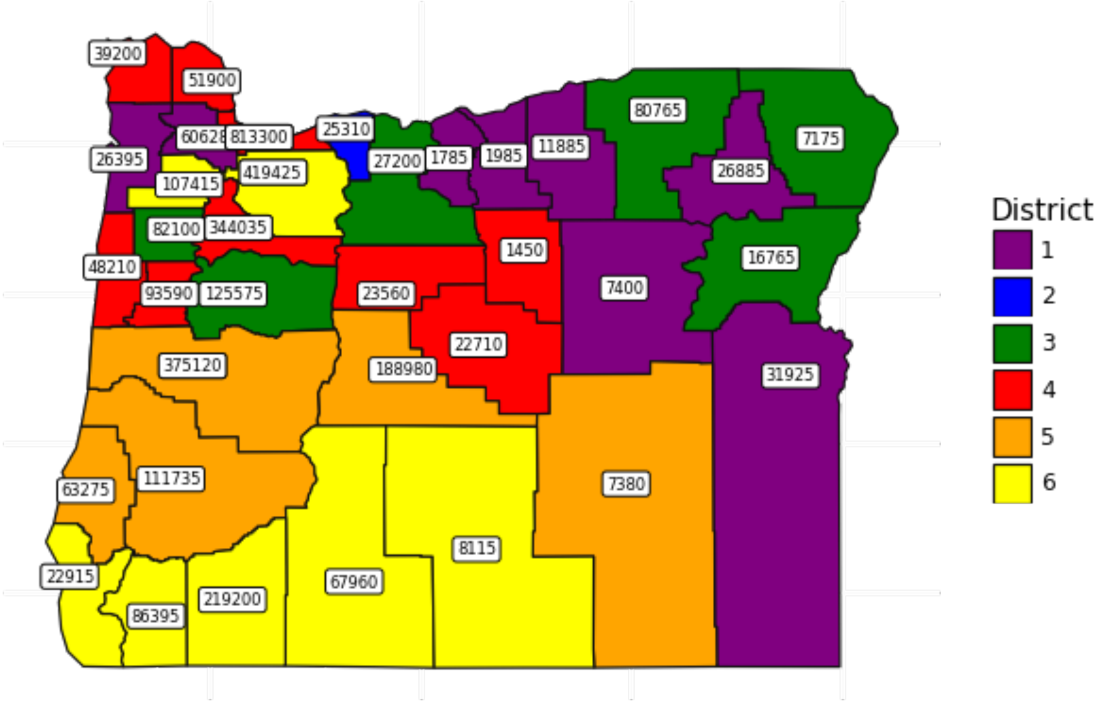
In [12]:

```
# data preparation for mapping the results

df_county_names = pd.DataFrame(county_names, columns = ['County'])
df = pd.DataFrame()
df['County']  = county_names
df['CountySort'] = county_id

output1 = []
output2 = []
output3 = []
output4 = []
output5 = []
output6 = []
for i in range(n_counties):
    for j in range(1):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*1}
    output1.append(var_output)
for i in range(n_counties):
    for j in range(2):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*2}
    output2.append(var_output)
for i in range(n_counties):
    for j in range(3):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*3}
    output3.append(var_output)
for i in range(n_counties):
    for j in range(4):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*4}
    output4.append(var_output)
for i in range(n_counties):
    for j in range(5):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*5}
    output5.append(var_output)
for i in range(n_counties):
    for j in range(6):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*6}
    output6.append(var_output)

df1 = pd.DataFrame.from_records(output1).sort_values(['County', 'District'])
df2 = pd.DataFrame.from_records(output2).sort_values(['County', 'District'])
df3 = pd.DataFrame.from_records(output3).sort_values(['County', 'District'])
df4 = pd.DataFrame.from_records(output4).sort_values(['County', 'District'])
df5 = pd.DataFrame.from_records(output5).sort_values(['County', 'District'])
```

```
df6 = pd.DataFrame.from_records(output6).sort_values(['County', 'District'])
# concatenating all lists at once is less computationally intensive than appending to DF multiple times
assignment_results = pd.concat([df1, df2, df3, df4, df5, df6])

# the following is for the visualization
assignment_results = assignment_results[assignment_results['Assignment'] > 0]
assignment_results.sort_values(['County', 'District'])
assignment_results = assignment_results.merge(df, left_on='County', right_on='CountySort',suffixes=('_ID', '_Name'))
```

```
In [13]: map_first_pass = shapefile_oregon.merge(assignment_results, left_on='N
         AME', right_on='County_Name',suffixes=('_left', '_right'))
         map_first_pass['District'] = map_first_pass['District']+1
         map_first_pass_labels = map_first_pass.merge(population_by_county, lef
         t_on='County_ID', right_on='County_ID',suffixes=('_left','_right'))
         map_first_pass_labels['District'] = map_first_pass_labels['District'].
         astype('category')
         plot_map_first_pass = (
         ggplot(map_first_pass_labels)
         + geom_map(aes(fill='District'))
         + geom_label(aes(x = 'Longitude', y = 'Latitude', label='Population201
         8', size = 2), show_legend=False)
         + theme_minimal()
         + theme(axis_text_x=element_blank(),
                 axis_text_y=element_blank(),
                 axis_title_x=element_blank(),
                 axis_title_y=element_blank(),
                 axis_ticks=element_blank(),
                 panel_grid_major = element_blank()
                )
         + scale_fill_manual(values = ["purple","blue","green","red","orange","
         yellow"])
         )
         print('\033[1m'+'County-District Assignments (some counties may have m
         ore than one district)')
         plot_map_first_pass
```

```
County-District Assignments (some counties may have more than one dis
trict)
```



```
Out[13]: <ggplot: (8774591577411)>
```

Add more constraints and re-run the model

The results satisfy the current constraints, but there can always be improvements by adding more constraints. Particularly, a better result would have each district in only one contiguous cluster. To promote such a result in the model, the following improvement constraints prevent certain counties from being in the same district. The constraints below prevent counties that have significant population centers in between them from being assigned to the same district.

In [14]:

```
# Improvement Assignment / Allocation Constraints (sometimes known as
cuts)

# pairs of counties that are far apart geographically with population
centers in between them
for j in range(n_districts):
    # Baker (0) on the east not to be assigned to the west beyond popu
lation centers in the middle
    model += assignment[0][j] + assignment[1][j] <= 1
    model += assignment[0][j] + assignment[3][j] <= 1
    model += assignment[0][j] + assignment[4][j] <= 1
    model += assignment[0][j] + assignment[5][j] <= 1
    model += assignment[0][j] + assignment[6][j] <= 1
    model += assignment[0][j] + assignment[19][j] <= 1
    model += assignment[0][j] + assignment[20][j] <= 1
    model += assignment[0][j] + assignment[21][j] <= 1
    model += assignment[0][j] + assignment[23][j] <= 1
    model += assignment[0][j] + assignment[25][j] <= 1
    model += assignment[0][j] + assignment[26][j] <= 1
    model += assignment[0][j] + assignment[28][j] <= 1
    model += assignment[0][j] + assignment[33][j] <= 1
    model += assignment[0][j] + assignment[35][j] <= 1
    # same with Grant (11)
    model += assignment[11][j] + assignment[1][j] <= 1
    model += assignment[11][j] + assignment[3][j] <= 1
    model += assignment[11][j] + assignment[4][j] <= 1
    model += assignment[11][j] + assignment[5][j] <= 1
    model += assignment[11][j] + assignment[6][j] <= 1
    model += assignment[11][j] + assignment[19][j] <= 1
    model += assignment[11][j] + assignment[20][j] <= 1
    model += assignment[11][j] + assignment[21][j] <= 1
    model += assignment[11][j] + assignment[23][j] <= 1
    model += assignment[11][j] + assignment[25][j] <= 1
    model += assignment[11][j] + assignment[26][j] <= 1
    model += assignment[11][j] + assignment[28][j] <= 1
    model += assignment[11][j] + assignment[33][j] <= 1
    model += assignment[11][j] + assignment[35][j] <= 1
    # same with Harney (12)
    model += assignment[12][j] + assignment[1][j] <= 1
    model += assignment[12][j] + assignment[3][j] <= 1
    model += assignment[12][j] + assignment[4][j] <= 1
    model += assignment[12][j] + assignment[5][j] <= 1
    model += assignment[12][j] + assignment[6][j] <= 1
    model += assignment[12][j] + assignment[19][j] <= 1
    model += assignment[12][j] + assignment[20][j] <= 1
    model += assignment[12][j] + assignment[21][j] <= 1
    model += assignment[12][j] + assignment[23][j] <= 1
    model += assignment[12][j] + assignment[25][j] <= 1
    model += assignment[12][j] + assignment[26][j] <= 1
    model += assignment[12][j] + assignment[28][j] <= 1
    model += assignment[12][j] + assignment[33][j] <= 1
    model += assignment[12][j] + assignment[35][j] <= 1
    # same with Malheur (22)
    model += assignment[22][j] + assignment[1][j] <= 1
    model += assignment[22][j] + assignment[3][j] <= 1
    model += assignment[22][j] + assignment[4][j] <= 1
    model += assignment[22][j] + assignment[5][j] <= 1
    model += assignment[22][j] + assignment[6][j] <= 1
    model += assignment[22][j] + assignment[19][j] <= 1
    model += assignment[22][j] + assignment[20][j] <= 1
    model += assignment[22][j] + assignment[21][j] <= 1
    model += assignment[22][j] + assignment[23][j] <= 1
    model += assignment[22][j] + assignment[25][j] <= 1
    model += assignment[22][j] + assignment[26][j] <= 1
    model += assignment[22][j] + assignment[28][j] <= 1
    model += assignment[22][j] + assignment[33][j] <= 1
    model += assignment[22][j] + assignment[35][j] <= 1
    # same with Morrow (24)
    model += assignment[24][j] + assignment[1][j] <= 1
    model += assignment[24][j] + assignment[3][j] <= 1
    model += assignment[24][j] + assignment[4][j] <= 1
    model += assignment[24][j] + assignment[5][j] <= 1
    model += assignment[24][j] + assignment[6][j] <= 1
    model += assignment[24][j] + assignment[19][j] <= 1
    model += assignment[24][j] + assignment[20][j] <= 1
    model += assignment[24][j] + assignment[21][j] <= 1
    model += assignment[24][j] + assignment[23][j] <= 1
    model += assignment[24][j] + assignment[25][j] <= 1
    model += assignment[24][j] + assignment[26][j] <= 1
    model += assignment[24][j] + assignment[28][j] <= 1
    model += assignment[24][j] + assignment[33][j] <= 1
    model += assignment[24][j] + assignment[35][j] <= 1
    # same with Umatilla (29)
    model += assignment[29][j] + assignment[1][j] <= 1
    model += assignment[29][j] + assignment[3][j] <= 1
    model += assignment[29][j] + assignment[4][j] <= 1
    model += assignment[29][j] + assignment[5][j] <= 1
    model += assignment[29][j] + assignment[6][j] <= 1
    model += assignment[29][j] + assignment[19][j] <= 1
    model += assignment[29][j] + assignment[20][j] <= 1
    model += assignment[29][j] + assignment[21][j] <= 1
    model += assignment[29][j] + assignment[23][j] <= 1
    model += assignment[29][j] + assignment[25][j] <= 1
    model += assignment[29][j] + assignment[26][j] <= 1
    model += assignment[29][j] + assignment[28][j] <= 1
    model += assignment[29][j] + assignment[33][j] <= 1
    model += assignment[29][j] + assignment[35][j] <= 1
    # same with Union (30)
    model += assignment[30][j] + assignment[1][j] <= 1
    model += assignment[30][j] + assignment[3][j] <= 1
    model += assignment[30][j] + assignment[4][j] <= 1
    model += assignment[30][j] + assignment[5][j] <= 1
    model += assignment[30][j] + assignment[6][j] <= 1
    model += assignment[30][j] + assignment[19][j] <= 1
    model += assignment[30][j] + assignment[20][j] <= 1
    model += assignment[30][j] + assignment[21][j] <= 1
    model += assignment[30][j] + assignment[23][j] <= 1
    model += assignment[30][j] + assignment[25][j] <= 1
    model += assignment[30][j] + assignment[26][j] <= 1
    model += assignment[30][j] + assignment[28][j] <= 1
    model += assignment[30][j] + assignment[33][j] <= 1
    model += assignment[30][j] + assignment[35][j] <= 1
    # same with Wallowa (31)
    model += assignment[31][j] + assignment[1][j] <= 1
    model += assignment[31][j] + assignment[3][j] <= 1
    model += assignment[31][j] + assignment[4][j] <= 1
    model += assignment[31][j] + assignment[5][j] <= 1
    model += assignment[31][j] + assignment[6][j] <= 1
    model += assignment[31][j] + assignment[19][j] <= 1
    model += assignment[31][j] + assignment[20][j] <= 1
    model += assignment[31][j] + assignment[21][j] <= 1
    model += assignment[31][j] + assignment[23][j] <= 1
    model += assignment[31][j] + assignment[25][j] <= 1
    model += assignment[31][j] + assignment[26][j] <= 1
    model += assignment[31][j] + assignment[28][j] <= 1
    model += assignment[31][j] + assignment[33][j] <= 1
    model += assignment[31][j] + assignment[35][j] <= 1
    # southwest counties (5,6,9,19) shouldn't be in the same district
as north counties
    # Coos County (5)
    model += assignment[5][j] + assignment[3][j] <= 1
    model += assignment[5][j] + assignment[4][j] <= 1
    model += assignment[5][j] + assignment[6][j] <= 1
    model += assignment[5][j] + assignment[10][j] <= 1
    model += assignment[5][j] + assignment[11][j] <= 1
    model += assignment[5][j] + assignment[15][j] <= 1
    model += assignment[5][j] + assignment[24][j] <= 1
    model += assignment[5][j] + assignment[27][j] <= 1
    model += assignment[5][j] + assignment[28][j] <= 1
    model += assignment[5][j] + assignment[34][j] <= 1
    # Curry County (7)
    model += assignment[7][j] + assignment[3][j] <= 1
    model += assignment[7][j] + assignment[4][j] <= 1
    model += assignment[7][j] + assignment[6][j] <= 1
    model += assignment[7][j] + assignment[10][j] <= 1
    model += assignment[7][j] + assignment[11][j] <= 1
    model += assignment[7][j] + assignment[15][j] <= 1
    model += assignment[7][j] + assignment[24][j] <= 1
    model += assignment[7][j] + assignment[27][j] <= 1
    model += assignment[7][j] + assignment[28][j] <= 1
    model += assignment[7][j] + assignment[34][j] <= 1
    # Douglas County (9)
    model += assignment[9][j] + assignment[3][j] <= 1
    model += assignment[9][j] + assignment[4][j] <= 1
    model += assignment[9][j] + assignment[6][j] <= 1
    model += assignment[6][j] + assignment[10][j] <= 1
    model += assignment[9][j] + assignment[11][j] <= 1
    model += assignment[9][j] + assignment[15][j] <= 1
    model += assignment[9][j] + assignment[24][j] <= 1
    model += assignment[9][j] + assignment[27][j] <= 1
    model += assignment[9][j] + assignment[28][j] <= 1
    model += assignment[9][j] + assignment[34][j] <= 1
    # Lane County (19)
    model += assignment[19][j] + assignment[3][j] <= 1
    model += assignment[19][j] + assignment[4][j] <= 1
    model += assignment[19][j] + assignment[6][j] <= 1
    model += assignment[19][j] + assignment[10][j] <= 1
    model += assignment[19][j] + assignment[11][j] <= 1
    model += assignment[19][j] + assignment[15][j] <= 1
    model += assignment[19][j] + assignment[24][j] <= 1
    model += assignment[19][j] + assignment[27][j] <= 1
    model += assignment[19][j] + assignment[28][j] <= 1
    model += assignment[19][j] + assignment[34][j] <= 1
    # northwest counties (3,4,28) shouldn't be in the same district as counties on other side of population centers
    model += assignment[3][j] + assignment[1][j] <= 1
    model += assignment[3][j] + assignment[6][j] <= 1
    model += assignment[3][j] + assignment[8][j] <= 1
    model += assignment[3][j] + assignment[10][j] <= 1
    model += assignment[3][j] + assignment[21][j] <= 1
    model += assignment[3][j] + assignment[27][j] <= 1
    model += assignment[4][j] + assignment[1][j] <= 1
    model += assignment[4][j] + assignment[6][j] <= 1
    model += assignment[4][j] + assignment[8][j] <= 1
    model += assignment[4][j] + assignment[10][j] <= 1
    model += assignment[4][j] + assignment[21][j] <= 1
    model += assignment[4][j] + assignment[27][j] <= 1
    model += assignment[28][j] + assignment[1][j] <= 1
    model += assignment[28][j] + assignment[6][j] <= 1
    model += assignment[28][j] + assignment[8][j] <= 1
    model += assignment[28][j] + assignment[10][j] <= 1
    model += assignment[28][j] + assignment[21][j] <= 1
    model += assignment[28][j] + assignment[27][j] <= 1
    # multnomah
    model += assignment[6][j] + assignment[25][j] <= 1
    model += assignment[7][j] + assignment[25][j] <= 1
    model += assignment[10][j] + assignment[25][j] <= 1
    model += assignment[16][j] + assignment[25][j] <= 1
    model += assignment[17][j] + assignment[25][j] <= 1
    model += assignment[18][j] + assignment[25][j] <= 1
    model += assignment[19][j] + assignment[25][j] <= 1
    model += assignment[20][j] + assignment[25][j] <= 1
    model += assignment[21][j] + assignment[25][j] <= 1
    model += assignment[23][j] + assignment[25][j] <= 1
    model += assignment[26][j] + assignment[25][j] <= 1
    model += assignment[27][j] + assignment[25][j] <= 1
    model += assignment[28][j] + assignment[25][j] <= 1
    model += assignment[34][j] + assignment[25][j] <= 1
    # these constraints from practice iterations
    model += assignment[3][j] + assignment[32][j] <= 1
    model += assignment[3][j] + assignment[34][j] <= 1
    model += assignment[4][j] + assignment[22][j] <= 1
    model += assignment[4][j] + assignment[32][j] <= 1
    model += assignment[4][j] + assignment[34][j] <= 1
    model += assignment[7][j] + assignment[33][j] <= 1
    model += assignment[8][j] + assignment[26][j] <= 1
    model += assignment[8][j] + assignment[28][j] <= 1
    model += assignment[8][j] + assignment[33][j] <= 1
    model += assignment[9][j] + assignment[28][j] <= 1
    model += assignment[9][j] + assignment[33][j] <= 1
    model += assignment[10][j] + assignment[28][j] <= 1
    model += assignment[14][j] + assignment[28][j] <= 1
    model += assignment[14][j] + assignment[33][j] <= 1
    model += assignment[16][j] + assignment[33][j] <= 1
    model += assignment[17][j] + assignment[33][j] <= 1
    model += assignment[18][j] + assignment[28][j] <= 1
    model += assignment[18][j] + assignment[33][j] <= 1
    model += assignment[24][j] + assignment[28][j] <= 1
    model += assignment[28][j] + assignment[29][j] <= 1
```

In [15]:

```
model.solve(PULP_CBC_CMD())
print('The model status is: ',LpStatus[model.status])
print('The objective value is: ', pulp.value(objective_function))
```

```
The model status is:  Optimal
The objective value is:  40.0
```

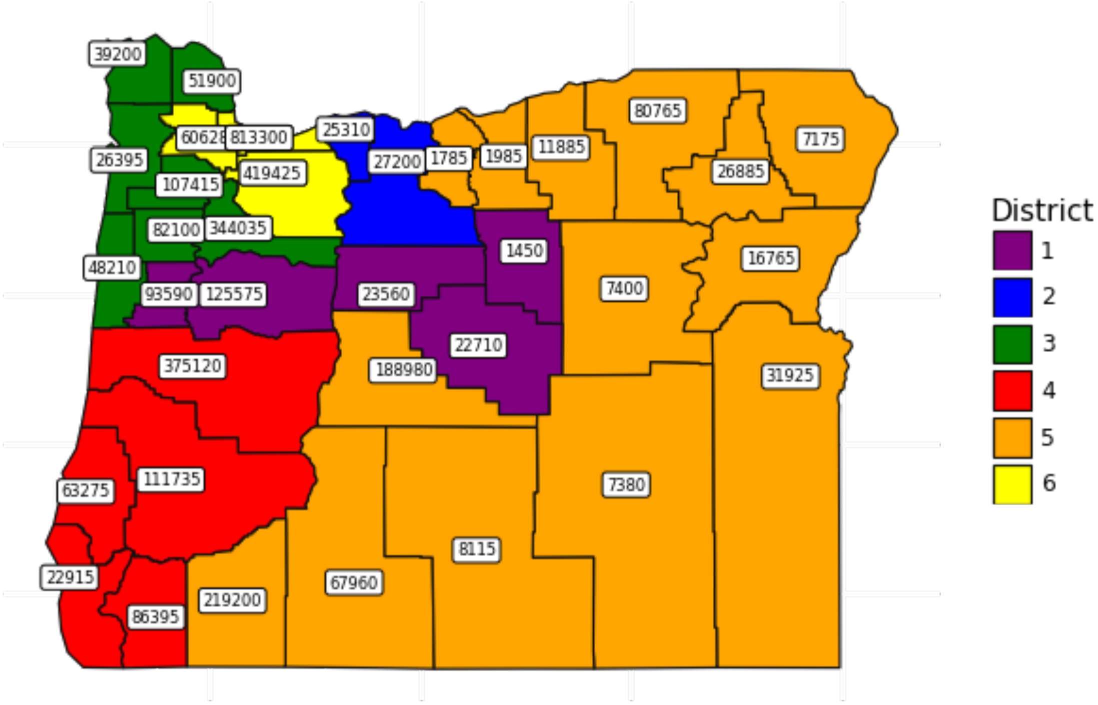
In [16]:

```
# data preparation for mapping the results

df_county_names = pd.DataFrame(county_names, columns = ['County'])
df_county_names
df = pd.DataFrame()
df['County']  = county_names
df['CountySort'] = county_id

output1 = []
output2 = []
output3 = []
output4 = []
output5 = []
output6 = []
for i in range(n_counties):
    for j in range(1):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*1}
    output1.append(var_output)
for i in range(n_counties):
    for j in range(2):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*2}
    output2.append(var_output)
for i in range(n_counties):
    for j in range(3):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*3}
    output3.append(var_output)
for i in range(n_counties):
    for j in range(4):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*4}
    output4.append(var_output)
for i in range(n_counties):
    for j in range(5):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*5}
    output5.append(var_output)
for i in range(n_counties):
    for j in range(6):
        var_output = {'County': i,'District': j,'Assignment': assignment[i][j].value()*6}
    output6.append(var_output)

df1 = pd.DataFrame.from_records(output1).sort_values(['County', 'District'])
df2 = pd.DataFrame.from_records(output2).sort_values(['County', 'District'])
df3 = pd.DataFrame.from_records(output3).sort_values(['County', 'District'])
df4 = pd.DataFrame.from_records(output4).sort_values(['County', 'District'])
df5 = pd.DataFrame.from_records(output5).sort_values(['County', 'Distr
ict'])
df6 = pd.DataFrame.from_records(output6).sort_values(['County', 'Distr
ict'])
assignment_results = pd.concat([df1, df2, df3, df4, df5, df6])

# the following is for the visualization
assignment_results = assignment_results[assignment_results['Assignment
'] > 0]
assignment_results.sort_values(['County', 'District'])
assignment_results = assignment_results.merge(df, left_on='County', ri
ght_on='CountySort',suffixes=('_ID', '_Name'))
```

In [17]:

```
map_second_pass = shapefile_oregon.merge(assignment_results, left_on='NAME', right_on='County_Name',suffixes=('_left', '_right'))
map_second_pass['District'] = map_second_pass['District']+1
map_second_pass_labels = map_second_pass.merge(population_by_county, left_on='County_ID', right_on='County_ID',suffixes=('_left','_right'))
map_second_pass_labels['District'] = map_second_pass_labels['District'].astype('category')
plot_map_second_pass = (
ggplot(map_second_pass_labels)
+ geom_map(aes(fill='District'))
+ geom_label(aes(x = 'Longitude', y = 'Latitude', label='Population2018', size = 2), show_legend=False)
+ theme_minimal()
+ theme(axis_text_x=element_blank(),
        axis_text_y=element_blank(),
        axis_title_x=element_blank(),
        axis_title_y=element_blank(),
        axis_ticks=element_blank(),
        panel_grid_major = element_blank()
       )
+ scale_fill_manual(values = ["purple","blue","green","red","orange","yellow"])
)
print("\033[1m"+"Now the map has compact district in singular clusters.")
print("\033[1m"+"However, let's see the breakdown below to understand how some counties are split between two districts.")
plot_map_second_pass
```

```
Now the map has compact district in singular clusters.
However, let's see the breakdown below to understand how some countie
s are split between two districts.
```



```
Out[17]: <ggplot: (8774592354594)>
```

As you can see below, Clackamas, Multnomah, and Washington counties (Yellow Color for District 6) are split among District 1 (Purple), District 2 (Blue), and District 3 (Green) respectively.

Additionally, Marion county is split between District 1 (Purple) and District 3 (Green). Since essentially Marion county is partly purple, that is what enables a piece of Clackamas county to be purple as well. Adding more improvement constraints can still improve the solution. Adding crosshatching colors to the map will help improve the visualization.

In [18]:

```
# Which counties are assigned to each district, and total the populati
ons
print('State Population: ', f"{state_population:,.0f}")
print('Assigned Population: ', f"{pulp.value(lpSum(allocation[i][j] fo
r i in range(n_counties) for j in range(n_districts))):,.0f}", '\n')
def thousands(x):
    try:
        return '{:,}'.format(int(x))
    except ValueError as e:
        return x
f_thousands = np.vectorize(thousands)
for j in range(n_districts):
    district_totals = lpSum(round(allocation[i][j].value()) for i in r
ange(n_counties))
    print("District", str(j+1), "Population: " , f"{pulp.value(distric
t_totals):,.0f}", "\n")
    County_Assigned_Population = list([0]*36) # initialize list
    for i in range(n_counties):
        x_dataframe = pd.DataFrame()
        x_dataframe['County_ID'] = county_id
        x_dataframe['County_Name'] = county_names
        if allocation[i][j].value() != 0.0:
            County_Assigned_Population[i] = f"{pulp.value(allocation
[i][j].value()):,.0f}"
        x_dataframe['County_Assigned_Population'] = County_Assigned_Po
pulation
        x_dataframe['County_Total_Population'] = f_thousands(county_po
pulations)
    x_dataframe = x_dataframe[x_dataframe['County_Assigned_Population
'] != 0]

    print(x_dataframe, "\n")
```

```
State Population:  4,195,300
Assigned Population:  4,195,300

District 1 Population:  750,000

    County_ID County_Name County_Assigned_Population County_Total_Pop
ulation
1           1      Benton                     93,590
93,590
2           2   Clackamas                    335,540
419,425
6           6       Crook                     22,710
22,710
15         15   Jefferson                     23,560
23,560
21         21        Linn                    125,575
125,575
23         23      Marion                    147,575
344,035
34         34     Wheeler                      1,450
1,450

District 2 Population:  703,150

    County_ID County_Name County_Assigned_Population County_Total_Pop
ulation
13         13  Hood River                     25,310
25,310
25         25   Multnomah                    650,640
813,300
32         32       Wasco                     27,200
27,200

District 3 Population:  750,000

    County_ID County_Name County_Assigned_Population County_Total_Pop
ulation
3           3     Clatsop                     39,200
39,200
4           4    Columbia                     51,900
51,900
20         20     Lincoln                     48,210
48,210
23         23      Marion                    196,460
344,035
26         26        Polk                     82,100
82,100
28         28   Tillamook                     26,395
26,395
33         33  Washington                    198,320
606,280
35         35     Yamhill                    107,415
107,415
```

```
District 4 Population:  659,440

    County_ID County_Name County_Assigned_Population County_Total_Pop
ulation
5           5        Coos                     63,275
63,275
7           7       Curry                     22,915
22,915
9           9     Douglas                    111,735
111,735
16         16   Josephine                     86,395
86,395
19         19        Lane                    375,120
375,120

District 5 Population:  678,205

    County_ID County_Name County_Assigned_Population County_Total_Pop
ulation
0           0       Baker                     16,765
16,765
8           8   Deschutes                    188,980
188,980
10         10     Gilliam                      1,985
1,985
11         11       Grant                      7,400
7,400
12         12      Harney                      7,380
7,380
14         14     Jackson                    219,200
219,200
17         17     Klamath                     67,960
67,960
18         18        Lake                      8,115
8,115
22         22     Malheur                     31,925
31,925
24         24      Morrow                     11,885
11,885
27         27     Sherman                      1,785
1,785
29         29    Umatilla                     80,765
80,765
30         30       Union                     26,885
26,885
31         31     Wallowa                      7,175
7,175

District 6 Population:  654,505

    County_ID County_Name County_Assigned_Population County_Total_Pop
ulation
2           2   Clackamas                     83,885
419,425
```

```
25          25   Multnomah                       162,660
813,300
33          33  Washington                       407,960
606,280
```

In [19]:

```
# The following is a work-in-process to do color-striping on counties
that are in two districts
map_second_pass_labels.set_crs(epsg=4326, inplace=True)

m = folium.Map(location=[43, -121], zoom_start=6, width = '70%')

folium.Choropleth(
    geo_data=map_second_pass_labels,
    name="any name",
    data=map_second_pass_labels,
    columns=['NAME', 'District'],
    key_on="feature.properties.NAME",
    fill_color="YlGnBu", # yellow green blue
    fill_opacity=0.9,
    line_opacity=0.9,
    legend_name="Population",
).add_to(m)

folium.LayerControl().add_to(m)

m

# will want to create a new variable to capture those counties and use
a stripepattern
# from folium.plugins import StripePattern
# sp = StripePattern(angle=45, color='grey', space_color='white')
# sp.add_to(m)
# m
```

Out[19]: Make this Notebook Trusted to load map: File -> Trust Notebook

In [ ]: