Final Part 2 - Percent Delinquent Maps

March 15, 2021

0.1 Final Part 2: Percent Delinquent Analysis

This notebook will focus on my analysis of the risk indicators I found thorugh my regressions that are statistically significant and have an impact on the percent of delinquent households per zip code.

0.1.1 Load Libraries

[1]:
```python
import pandas as pd
import numpy as np
import geopandas as gpd
import contextily as ctx
import matplotlib.pyplot as plt
import plotly.express as px

import seaborn as sns
```

```
/opt/conda/lib/python3.8/site-packages/geopandas/_compat.py:106: UserWarning:
The Shapely GEOS version (3.8.1-CAPI-1.13.3) is incompatible with the GEOS
version PyGEOS was compiled with (3.9.0-CAPI-1.16.2). Conversions between both
will be slow.
  warnings.warn(
```

0.1.2 Load Data

The same datasets I've been working with are below, except I have added a file with latitude and longitudes for the state of California so I can create a scatterplot of the zip codes to save some memory on my interactive maps/

[2]:
```python
acs = pd.read_csv('../Data/Updated Bill Data 2_22.csv')

zips = gpd.read_file('../Data/ca_california_zip_codes_geo.min.json')

latlon = pd.read_csv('../Data/CA Zip Lat and Lon - Sheet1.csv')
```

0.1.3 Clean Data

From working with this data for several weeks now, I know there are N/A values I need to drop.

[3]:
```python
acs = acs.dropna()
```

Next, I only keep my columns of interest.

[4]:
```python
refined_columns = ['Zip Codes',
 'Sum of Less than $100',
 'Sum of $100-$200',
 'Sum of $200-$300',
 'Sum of $300-$400',
 'Sum of $400-$500',
 'Sum of $500-$600',
 'Sum of $600-$700',
 'Sum of $700-$800',
 'Sum of $800-$900',
 'Sum of $900-$1000',
 'Sum of More than $1000',
 'Sum of Total number of delinquent residential accounts',
 'pop',
 'mhhi',
 'pct_nhw',
 'pct_black',
 'pct_hisp',
 'pct_asian',
 'pct_povt',
 'pct_overcrowded',
 'pct_no_veh_hh',
 'pct_broadband',
 'pct_no_broadband',
 'pct_uninsured_19_64',
 'pct_noncitizen',
 'pct_immigrants',
 'pct_lep_hh',
 'pct_no_hins',
 '% Renter Pop',
  '% Owner Pop']
```

Then I save these over the old dataframe.

[5]:
```python
acs = acs[refined_columns]
```

0.1.4 Normalize the Data

Next, I use the for loops I created a few weeks ago to normalize my data so that my debt buckets are in percents and my demographic factors are multipled by 100 so they are in a more readable percent format and not decimals

[6]:

```
acs['Percent Delinquent'] = acs['Sum of Total number of delinquent residential␣
↪accounts']/acs['pop']*100


pct_debt_buckets = ['Percent Less than $100', 'Percent $100-$200', 'Percent␣
↪$200-$300' ,
                'Percent $300-$400', 'Percent $400-$500', 'Percent $500-$600',␣
↪'Percent $600-$700',
                'Percent $700-$800', 'Percent $800-$900', 'Percent $900-$1000',␣
↪'Percent More than $1000']

debt_buckets =  ['Sum of Less than $100', 'Sum of $100-$200', 'Sum of␣
↪$200-$300' ,
                'Sum of $300-$400', 'Sum of $400-$500', 'Sum of $500-$600',␣
↪'Sum of $600-$700',
                'Sum of $700-$800', 'Sum of $800-$900', 'Sum of $900-$1000',␣
↪'Sum of More than $1000']

sum_total = 'Sum of Total number of delinquent residential accounts'

demographics = ['pct_nhw','pct_black',␣
↪'pct_hisp','pct_asian','pct_povt','pct_overcrowded','pct_no_veh_hh',
                'pct_broadband','pct_no_broadband',␣
↪'pct_uninsured_19_64','pct_noncitizen','pct_immigrants',
                'pct_lep_hh','pct_no_hins','% Renter Pop','% Owner Pop']

for pct, debt in zip(pct_debt_buckets, debt_buckets):
    acs[pct] = acs[debt] / acs[sum_total]*100

for dem in demographics:
    acs[dem] = acs[dem]*100
```

Now I drop the columns I no longer need (the sum columns, since I hasve now converted my columns of interest into percents)

[7]:
```
columns_drop = ['Sum of Less than $100',
 'Sum of $100-$200',
 'Sum of $200-$300',
 'Sum of $300-$400',
 'Sum of $400-$500',
 'Sum of $500-$600',
 'Sum of $600-$700',
 'Sum of $700-$800',
 'Sum of $800-$900',
 'Sum of $900-$1000',
 'Sum of More than $1000',
 'Sum of Total number of delinquent residential accounts']
```

```
acs = acs.drop(columns_drop, axis = 1)
```

Now that I've done these calculations, I want to make sure all new N/A values have also been dropped

[8]:
```
acs = acs.dropna()
```

Finally, I know from working with the data, that there are some outliers of over 100% that also need to be removed.

[9]:
```
acs.drop(acs[acs['Percent Delinquent'] > 100].index, inplace = True)
acs.drop(acs[acs['Percent $400-$500'] > 100].index, inplace = True)
```

Next, I need to clean up the zip code shapefile.

0.1.5 Clean Zip Codes and Merge

[10]:
```
zips_keep = ['ZCTA5CE10', 'geometry']
```

I know from working with this file a lot that my two columns of interest are ZCTA5CE10 and geometry

[11]:
```
zips = zips[zips_keep]
```

Now I will over-write these onto the orginial zips file

[12]:
```
zips.columns = ['Zip Codes', 'geometry']
```

Then I will rename ZCTA5CE to 'Zip Codes' so it can merge with my ACS file with the same column name.

[13]:
```
merged = acs.merge(zips,
                   on='Zip Codes')
```

I save this file as a geodataframe.

[14]:
```
merged = gpd.GeoDataFrame(merged, geometry='geometry')
```

Then I want to make sure my "Zip Codes" file is saved as an integer so I can merge with the Latitude and Longitude file.

[15]:
```
merged['Zip Codes'] = merged['Zip Codes'].astype(int)
```

0.1.6 Latitude and Longitude Data

Now I just need to merge latitude and longitude data to my merged data so I can plot zip code locations. First, I need to look at the data to get a sense of which columns to keep.

[16]:
```
latlon.head()
```

[16]:
```
     Zip              City State   Latitude  Longitude  Timezone  \
0  95717          Gold Run    CA  39.177026 -120.84510        -8
1  94564            Pinole    CA  37.997509 -122.29208        -8
2  91605  North Hollywood    CA  34.208142 -118.40110        -8
3  91102          Pasadena    CA  33.786594 -118.29866        -8
4  95019           Freedom    CA  36.935552 -121.77972        -8

   Daylight savings time flag               geopoint
0                           1   39.177026, -120.8451
1                           1  37.997509, -122.29208
2                           1   34.208142, -118.4011
3                           1  33.786594, -118.298662
4                           1  36.935552, -121.77972
```

My columns of interest are Zip, Latitude, and Longitude

[17]:
```
latlon_keep = ['Zip', 'Latitude', 'Longitude']
```

Now I will over-write those columns on the original dataset.

[18]:
```
latlon = latlon[latlon_keep]
```

Now I change the column names so I can merge

[19]:
```
latlon.columns = ['Zip Codes', 'Latitude', 'Longitude']
```

And now my second merged column is merged_2

[20]:
```
merged_2 = merged.merge(latlon,
                   on='Zip Codes')
```

0.1.7 Maps: Percent Delinquent Inidcators

First, I am going to map my risk indicators for percent delinquent per zip code: Percent Black, Percent Overcrowded, and Percent Immigrants. Each of these factors showed a positive, statistically significant relationship to the percent of delinquent housesholds per zip code - as these demographic factors increase, so does the percent of households per zip code with water bill debt.

I want to map the zip codes that are most at risk of having water bill debt based on these demographic indicators, but to do this, I want to isolate the highest quartile for each indicator, thus, I will map the zip codes that are in the 75th percentile for each risk indicator

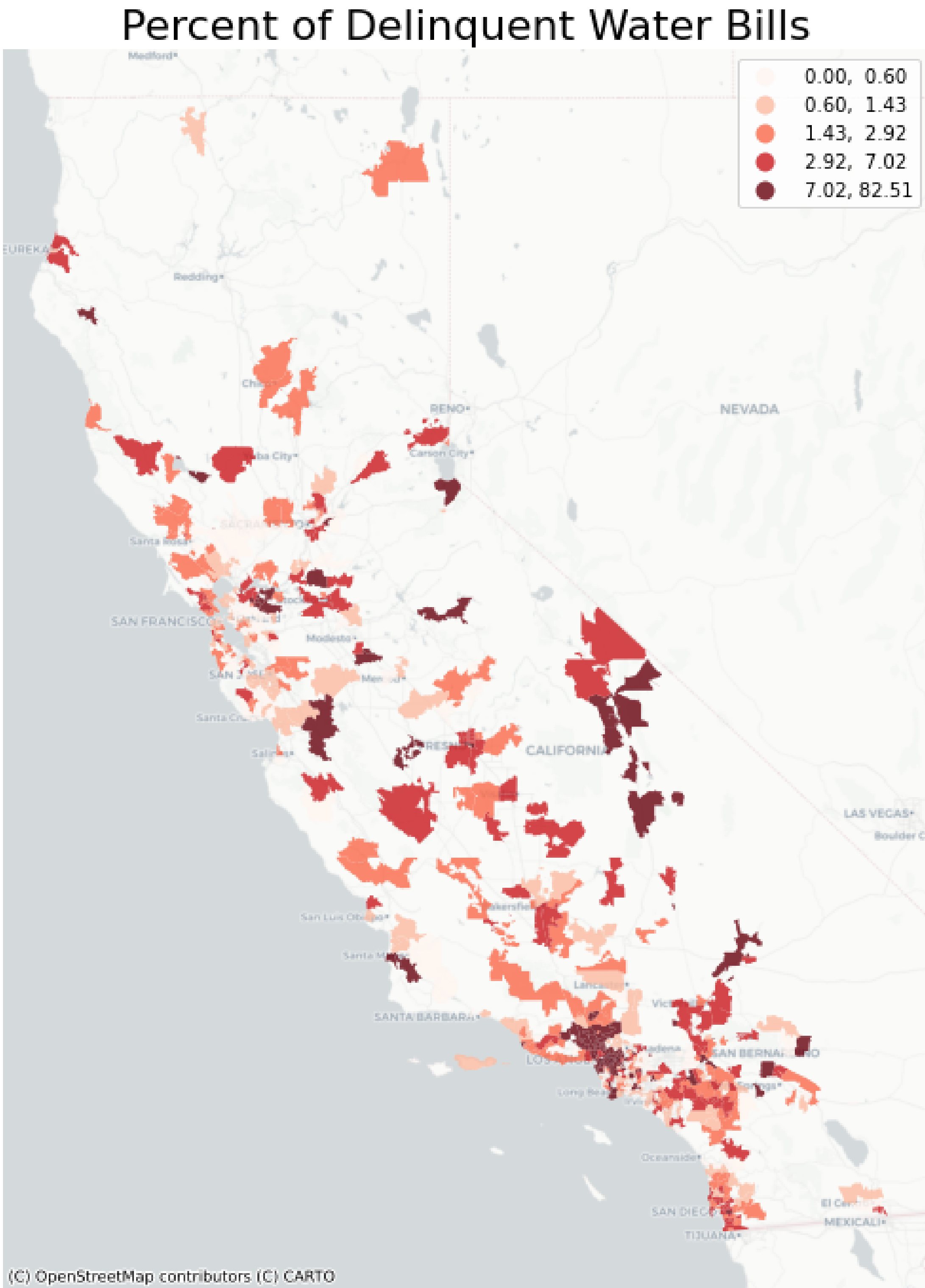
First, I need to get the web mercator projection.

[21]:
```
merged_2 = merged_2.to_crs(epsg=3857)
```

Then I plot the distribution of water bill debt in quantiles.

[22]:
```
fig,ax = plt.subplots(figsize=(12,12))
```

```
merged_2.plot(ax=ax,
        column='Percent Delinquent',
        legend=True,
        alpha=0.8,
        cmap='Reds',
        scheme='quantiles')

ax.axis('off')
ax.set_title('Percent of Delinquent Water Bills',fontsize=22)
ctx.add_basemap(ax,source=ctx.providers.CartoDB.Positron)
```



We can see from the map there is a high contentration of extreme percent debt in/around LA and also closer to the Nevada border (sort of in the Death Valley area). Let's map the risk indicators and see if they also appear in those areas.

First, we want to create new colums in the dataset and print which quartile each zip code falls under for my variables of interest. I am interested in which zip codes have %black %overcrowded, and %immigrant in the 4th quartile or 75th percentile.

[23]:
```
merged_2['black_quartile'] = pd.
↪qcut(merged_2['pct_black'],4,labels=['Q1','Q2','Q3','Q4'])
```

[24]:
```
merged_2['over_quartile'] = pd.
↪qcut(merged_2['pct_overcrowded'],4,labels=['Q1','Q2','Q3','Q4'])
```

[25]:
```
merged_2['imm_quartile'] = pd.
↪qcut(merged_2['pct_immigrants'],4,labels=['Q1','Q2','Q3','Q4'])
```

[26]:
```
merged_2_quart = merged_2[(merged_2['black_quartile']=='Q4') |
          (merged_2['over_quartile']=='Q4') |
          (merged_2['imm_quartile']=='Q4')
         ]
```

Now I have instructed my code to spit out Q1, 2, 3, and 4 based on the percent of each variable of interest.

I want to get a sense of how many zip codes fall into quartile 4 for at least 1 of my variables of interest.

[27]:
```
merged_2_quart.shape
```

[27]:
```
(398, 37)
```

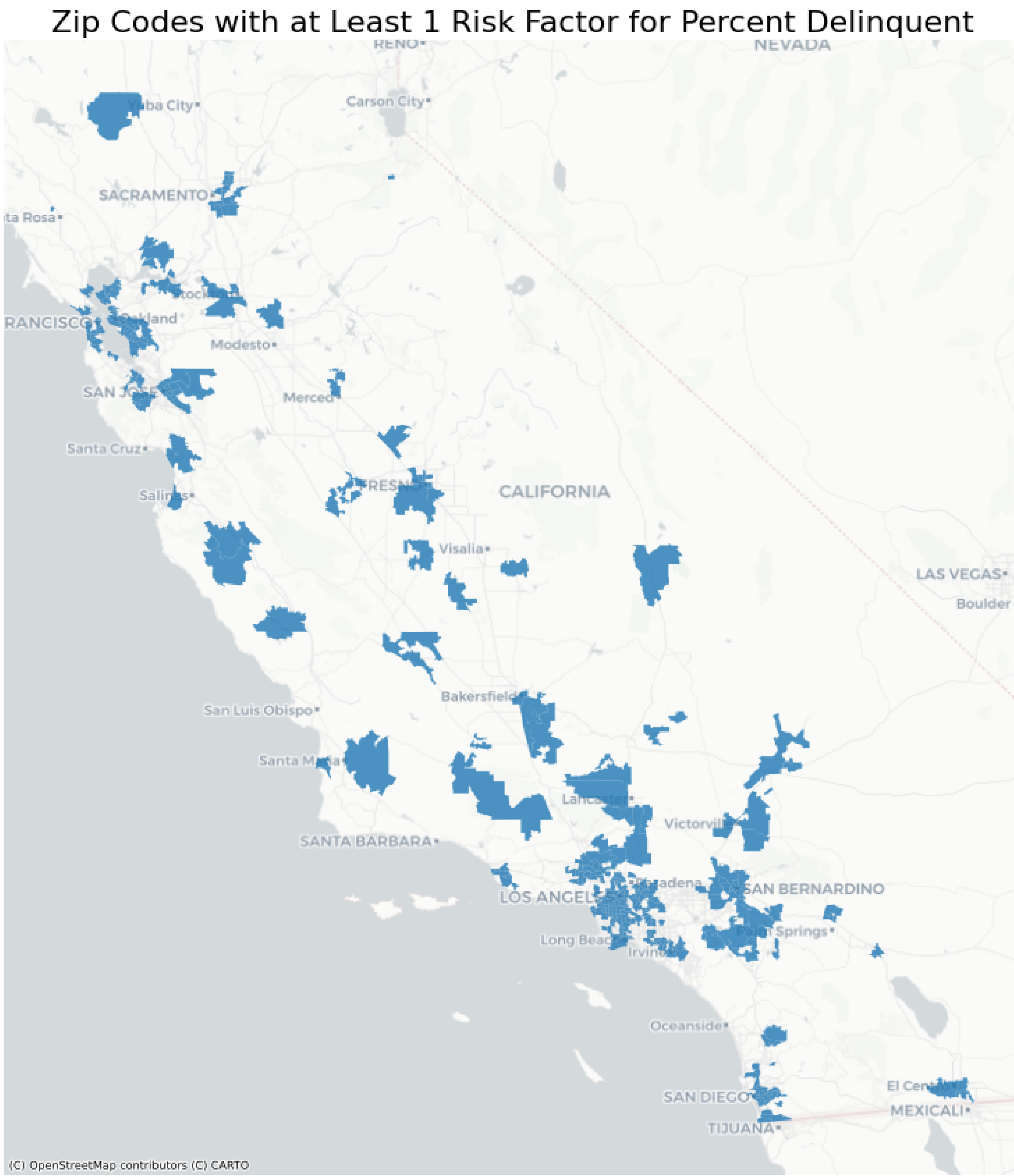
I see there are 398 zip codes in my dataset that have at least one of my 3 variables of interest (percent black, percent overcrowded, or percent immigrant) in the 75th percentile.

Now I want to map only those zip codes under my variables of interest that fall into the 75th percentile. First,I project to the web mercator.

[28]:
```
merged_2_quart=merged_2_quart.to_crs('epsg:3857')
```

Next, we plot the basemap.

[29]:
```
fig,ax = plt.subplots(figsize=(15,15))

merged_2_quart.plot(ax=ax,alpha=0.8)

ax.axis('off')
ax.set_title('Zip Codes with at Least 1 Risk Factor for Percent␣
↪Delinquent',fontsize=22)

ctx.add_basemap(ax,source=ctx.providers.CartoDB.Positron)
```



Above, you'll see a basemap with each of the 398 zip codes of interest. There is a high concentration of zip codes in the 75th percentile for %black, %overcrowded, and %immigrant mostly around LA, but a smattering up the coast.

The next thing I want better unsderstand is if there are certain zip codes that fall under 1 or more of these 75th percentile variables of interest. I am going to make an interactive map to better visualize this.

The first thing I do is define a new function, "color_code," and I create a function using numpy that sums up all the instances where %black, %overcrowded, and %immigrant is in the 4th Quartile for each zip code (or each line of data).

[30]:
```
def color_code(x):
    return np.sum([x['black_quartile']=='Q4', x['over_quartile']=='Q4',␣
↪x['imm_quartile']=='Q4'])

merged_2_quart['color_code'] = merged_2_quart.apply(color_code, axis=1)
```

Once this function runs through all my data, I add it to my existing dataset, "merged_2_quart" as a new column, "color_code."

Next, I want to clean up the names.

[31]:
```
merged_2_quart = merged_2_quart.rename(columns={'black_quartile': 'Percent␣
↪Black',
                                        'over_quartile': 'Percent Overcrowded',
                                        'imm_quartile': 'Percent Immigrant'})
```

I want the names of each of my variables of interest to appear a little cleaner on my map, so I changed the names to Percent Black, Percent Overcrowded, and Percent Immigrant

Next, I'll create the interactive map.

[32]:
```
fig = px.scatter_mapbox(merged_2_quart,
                        lat="Latitude",
                        lon="Longitude",
                        color="color_code",
                        hover_name = 'Zip Codes',
                        hover_data = {'Percent Black': True,
                                      'Percent Overcrowded': True,
                                      'Percent Immigrant': True,
                                      'Latitude': False,
                                      'Longitude': False
                                     },
                        size = 'pop',
                        labels={'color_code':'Risk Level'}
                       )

fig.update_layout(mapbox_style="stamen-terrain")

fig.show()
```

[33]:
```
fig.write_html('../Final/debtriskdelin.html')
```

You can see from the above map that I have color coded the zip codes that overlap with 1-3 of the 75th percentile variables of interest, and these Zip codes are labeled as Risk Level 1, 2, or 3. The size of the circles depends on the population of the zip code, which ideally helps better allocate resources to the majority of Californians who may live in an "at risk" area for high bill delinquency. When you hover over a circle, it will tell you which quartile the demograpgic variable of interst falls under and give you the zip code of that area.

0.1.8 Point Pattern Analysis

Next I want to do a point pattern analysis to see where the highest concentration of Zip Codes within each risk level exist.

[34]:
```
merged_2_quart = merged_2_quart.to_crs('EPSG:3857')
```

First I connected to the Web Mercator

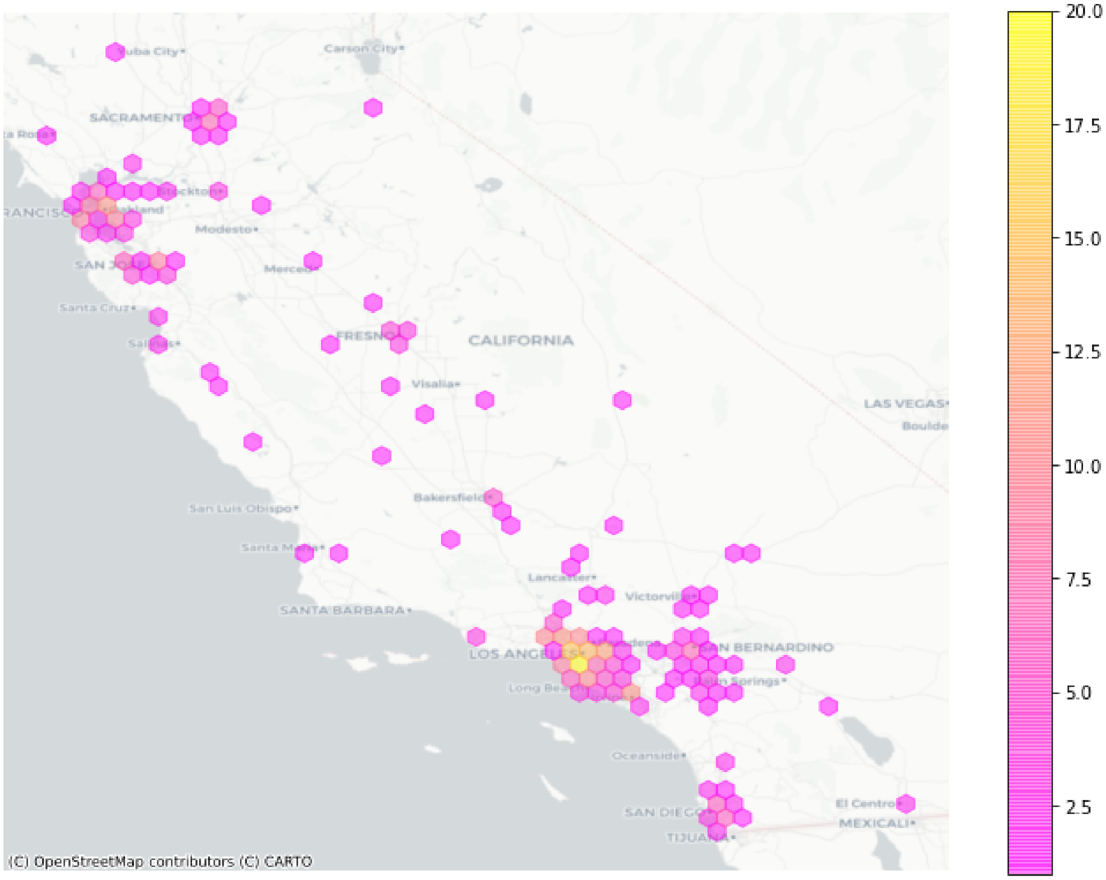
Then I changed my Longitude and Latitude variable names to x and y.

[35]:
```
merged_2_quart = merged_2_quart.rename(columns={'Longitude': 'x',
                                                'Latitude': 'y',})
```

Then I created hex bins and added them to a base map.

[36]:
```
f, ax = plt.subplots(figsize=(12, 9))

hb = ax.hexbin(
    x = merged_2_quart['x'],
    y = merged_2_quart['y'],
    gridsize=50,
    linewidths=1,
    alpha=0.5,
    mincnt=1,
    cmap='spring'
)


ctx.add_basemap(
    ax,
    crs='epsg:4326',
    source=ctx.providers.CartoDB.Positron
)


plt.colorbar(hb)


ax.axis('off')
```

[36]:
```
(-123.1352505079858, -115.1494594920142, 32.22710005, 39.46870695)
```

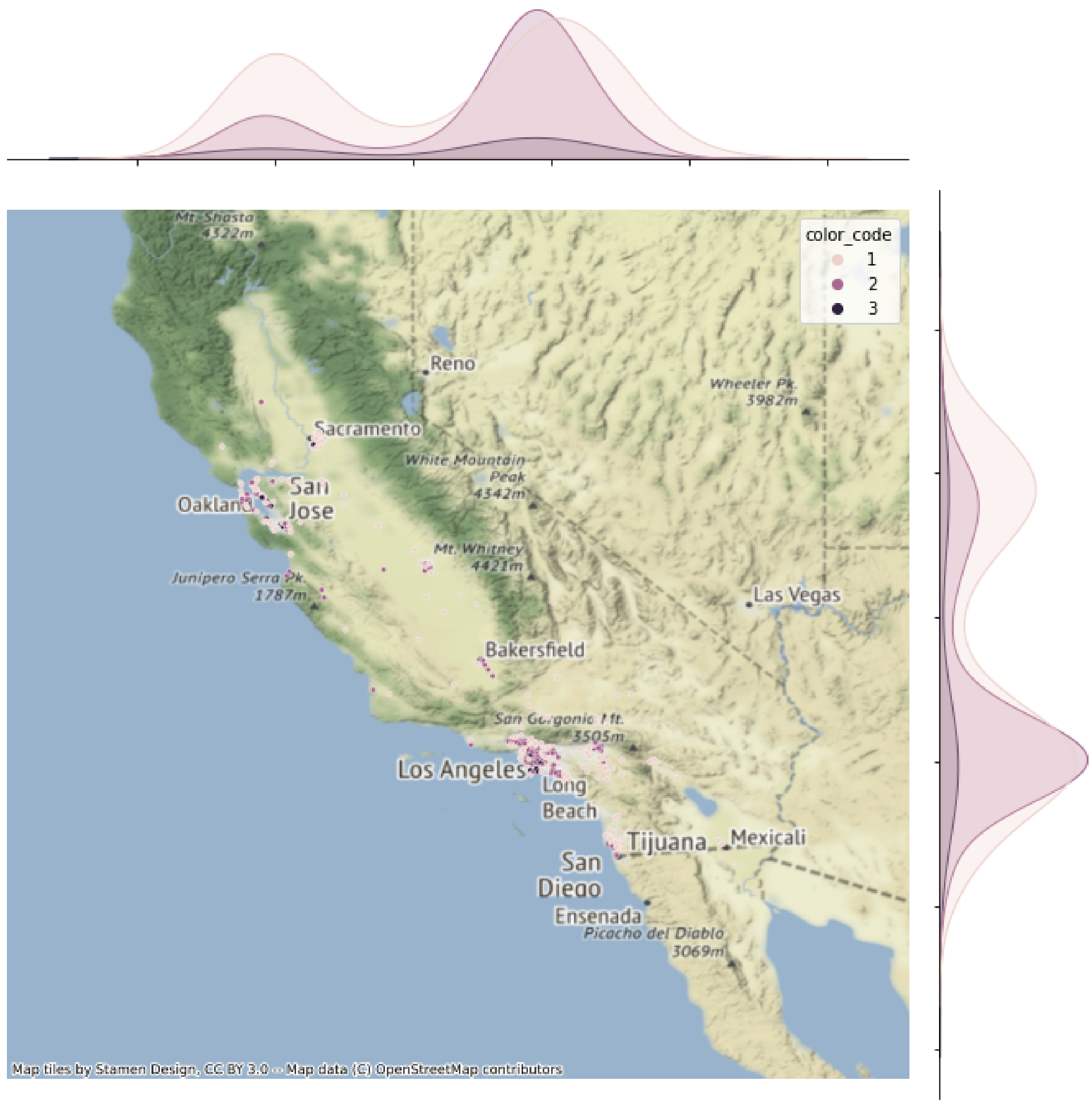


The cell shows a lot of zip codes around LA meet at least 1 criteria for risk based on my previous analysis. The next highest conentration of risky zip codes is near San Francisco.

[37]:
```
data_color = merged_2_quart[merged_2_quart.color_code.isin(['1','2','3'])]
```

Next I subsetted my data based just on my color_code column, and then mapped it.

[38]:
```
g = sns.jointplot(data = data_color,
                  x='x',
                  y='y',
                  hue='color_code',
                  s=10,
                  height=10)

# Clean axes
g.ax_joint.set_axis_off()

# add basemap
ctx.add_basemap(g.ax_joint,
                crs='epsg:4326')
```



Based on the map, it seems there is a high concentration of zip codes with 2 risk factors in/around the LA area and a fairly high concentration of zip codes with 1 risk factor sort of in the middle of the state, around the Oakland area. This seems to align with what the interactive map showed.

If I were the water Board and I had to focus on an area, I would focus on debt relief in LA. First, I would target zip codes with all three risk factors, then two.

[ ]: