Spiegel_Zach - Final Project

August 31, 2024

1 ECON 416 Final Project

1.0.1 Zach Spiegel

1.1 Introduction

There's nothing more American than fast food. Quick service restaurants–from established brands to local eateries–are integral not just to our culture, but also our diets. According to the Centers for Disease Control and Prevention, during the years of 2013 to 2016, 36.6% of adults consumed fast food everyday. Despite some of the health risks that come with large quantities of fast food, notably obesity and diabetes, consumption rates remain high. It is possible that income, time constraints, and availability of food/groceries may factor into consumer's decisions.

Through this project, I hope to find out if the number of fast food restaurants per capita in each United States county can be predicted based on certain metrics, such as obesity rates, diabetes rates, and population density. I will be employing certain machine learning techniques, such as principal component analysis, clustering, and regression analysis to answer this question.

Although seeming promising at the beginning of the project, these methods cannot accurately predict the amount of fast food restaurants each county has.

1.2 Literature Review

According to the USDA, counties with more fast food restaurants per capita are often densely populated or have major tourist attractions (for example, National Parks).

Not much research has been done on this particular topic, although a few studies have used machine learning to predicting food insecure populations.

Machine learning can guide food security efforts when primary data are not available

Machine learning for food security: Principles for transparency and usability

Another study found that "fast food availability was not associated with weekly frequency of fast food consumption in non-urban or low- or high-density urban areas". This finding could make drawing a conclusion difficult for this project.

1.3 Analysis

For this project, I will be using three datasets:

1. USD Food Environment Atlas
    - Food environment statistics per US county

2. County Health Rankings & Roadmaps
    - Recent health statistics per US county
3. US Census Bureau Population Statistics per US County
    - Total population and population density

[1]:
```
# Importing libraries
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import numpy as np
```

[2]:
```
fastfood = pd.read_csv("FoodEnvironmentAtlas-FastFood.csv", converters={'FIPS':␣
↪str})

# converters is needed to keep FIPS (county code) as a string, to prevent␣
↪errors in mapping
# source: https://stackoverflow.com/questions/13250046/
↪how-to-keep-leading-zeros-in-a-column-when-reading-csv-with-pandas

fastfood = fastfood[["FIPS", "State", "County", "FFR11", "FFR16", "FFRPTH11",␣
↪"FFRPTH16", "PC_FFRSALES07", "PC_FFRSALES12", "PCH_FFR_11_16"]]
# Keeping only relevant and fast food related columns
```

[3]:
```
cols_to_keep = [2, 6, 7, 33, 74, 79, 396, 406, 411, 507, 635, 640, 645, 650,␣
↪655, 660, 665, 670, 685]

health = pd.read_csv("analytic_data2021.csv", usecols = cols_to_keep,␣
↪converters={"5-digit FIPS Code": str})

# this dataset is very large and I only need certain columns from the dataset
# https://stackoverflow.com/questions/26063231/
↪read-specific-columns-with-pandas-or-other-python-module
```

```
C:\Users\zachs\AppData\Local\Temp\ipykernel_15972\3108541683.py:3: DtypeWarning:
Columns (6,7,33,74,79,396,406,411,507,635,640,645,650,655,660,665,670,685) have
mixed types. Specify dtype option on import or set low_memory=False.
  health = pd.read_csv("analytic_data2021.csv", usecols = cols_to_keep,
converters={"5-digit FIPS Code": str})
```

[4]:
```
health.drop([0], inplace = True) # row 0 contains descriptions of each column
health = health.dropna(subset=["County Ranked (Yes=1/No=0)"]) # removes any␣
↪non-county row
#https://www.geeksforgeeks.org/
↪how-to-drop-rows-with-nan-values-in-pandas-dataframe/#
health.drop(["County Ranked (Yes=1/No=0)"], axis=1, inplace = True) # column is␣
↪no longer needed
```

```
health.rename(columns={"5-digit FIPS Code": "FIPS"}, inplace = True) # needed␣
↪to match fastfood dataset


health["Premature death raw value"] = health["Premature death raw value"].
↪astype(float)
health["Poor or fair health raw value"] = health["Poor or fair health raw␣
↪value"].astype(float)
health["Adult obesity raw value"] = health["Adult obesity raw value"].
↪astype(float)
health["Food environment index raw value"] = health["Food environment index raw␣
↪value"].astype(float)
health["Diabetes prevalence raw value"] = health["Diabetes prevalence raw␣
↪value"].astype(float)
health["Food insecurity raw value"] = health["Food insecurity raw value"].
↪astype(float)
health["Limited access to healthy foods raw value"] = health["Limited access to␣
↪healthy foods raw value"].astype(float)
health["% below 18 years of age raw value"] = health["% below 18 years of age␣
↪raw value"].astype(float)
health["% 65 and older raw value"] = health["% 65 and older raw value"].
↪astype(float)
health["% Non-Hispanic Black raw value"] = health["% Non-Hispanic Black raw␣
↪value"].astype(float)
health["% American Indian & Alaska Native raw value"] = health["% Non-Hispanic␣
↪Black raw value"].astype(float)
health["% Asian raw value"] = health["% Asian raw value"].astype(float)
health["% Native Hawaiian/Other Pacific Islander raw value"] = health["% Native␣
↪Hawaiian/Other Pacific Islander raw value"].astype(float)
health["% Hispanic raw value"] = health["% Hispanic raw value"].astype(float)
health["% Non-Hispanic White raw value"] = health["% Non-Hispanic White raw␣
↪value"].astype(float)
health["% Rural raw value"] = health["% Rural raw value"].astype(float)

# convert from string to a float, int to prevent any future graphing errors
```

[5]:
```
pop = pd.read_csv("pop_density.csv", usecols = ("GEOID", "B01001_001E",␣
↪"B01001_calc_PopDensity"), converters={"GEOID": str})

pop.rename(columns = {"GEOID" : "FIPS", "B01001_001E" : "Total Pop",␣
↪"B01001_calc_PopDensity" :"Population Density"},
           inplace = True)

pop.head()
```

[5]:
```
    FIPS  Total Pop  Population Density
0  01001      55200           35.853419
1  01003     208107           50.541504
2  01005      25782           11.247981
3  01007      22527           13.973114
4  01009      57645           34.515816
```

[6]:
```
county = pd.merge(pd.merge(fastfood, health, on=["FIPS"]), pop, on = ["FIPS"])␣
 ↪# merge on FIPS code

# https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.merge.html

# 3 way merge: https://stackoverflow.com/questions/23668427/
 ↪pandas-three-way-joining-multiple-dataframes-on-columns
```

[7]:
```
ne = ["CT", "ME", "MA", "NH", "RI", "VT", "NJ", "NY", "PA"]
midw = ["IL", "IN", "MI", "OH", "WI", "IA", "KS", "MN", "MO", "NE", "ND", "SD"]
south = ["DE", "MD", "DC", "AL", "AR", "FL", "GA", "KY", "LA", "MS", "NC",␣
 ↪"SC", "TN", "VA", "WV"]
west = ["AZ", "NM", "OK", "TX", "CO", "ID", "MT", "UT", "WY", "AK", "CA", "HI",␣
 ↪"NV", "OR", "WA"]

# separating states into regions

county["Region"] = np.select(
    [county["State"].isin(ne),
     county["State"].isin(midw),
     county["State"].isin(south),
     county["State"].isin(west)],
    ["Northeast", "Midwest", "South", "West"])

# https://numpy.org/doc/stable/reference/generated/numpy.select.html

def FFany(FFR16):
    if FFR16 == 0:
        out = "No"
    else:
        out = "Yes"
    return out

county["FFany"] = county["FFR16"].map(FFany)

# adding new column to return if a county had any fast food restaurants
```

[8]:
```
county.value_counts("FFany")
```

[8]:
```
FFany
Yes    2986
No      154
dtype: int64
```

[9]:
```
county["FFRPTH16"].describe()
```

[9]:
```
count    3140.000000
mean        0.585381
std         0.307318
min         0.000000
25%         0.419133
50%         0.588938
75%         0.748226
max         5.805515
Name: FFRPTH16, dtype: float64
```

This is used to assign counties their group for fast food restuarants per capita.

[10]:
```
def FFcategory(FFRPTH16):
    if FFRPTH16 < 0.419133:
        out = "Low"
    elif 0.588938 > FFRPTH16 >= 0.41913:
        out = "Medium-Low"
    elif 0.748226 > FFRPTH16 >= 0.588938:
        out = "Medium-High"
    elif FFRPTH16 >= 0.748226:
        out = "High"
    return out

county["FFR Category"] = county["FFRPTH16"].map(FFcategory)
```

[11]:
```
county.value_counts("FFR Category")
```

[11]:
```
FFR Category
High           785
Low            785
Medium-High    785
Medium-Low     785
dtype: int64
```

[12]:
```
countyNoNA = county.copy()

countyNoNA.dropna(inplace = True)

countyNoNA["Median household income raw value"] = countyNoNA["Median household␣
↪income raw value"].astype(int)
```

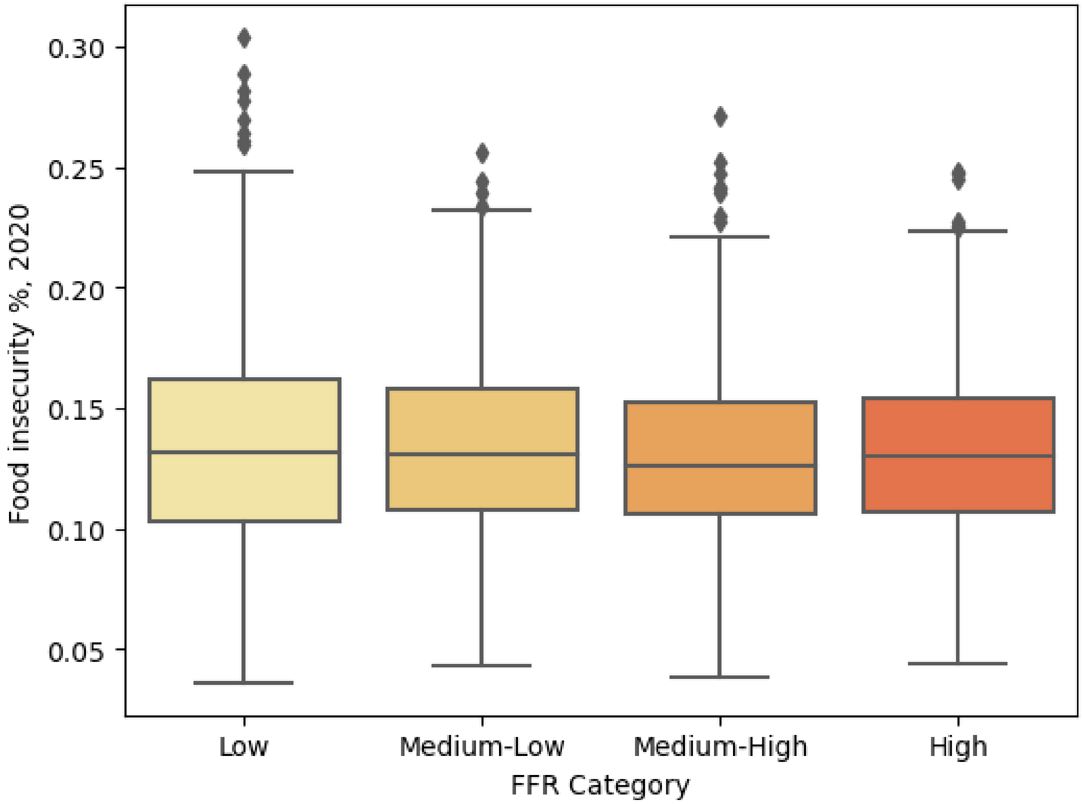
```
# Missing many values for household income
```

1.3.1 Figures

[13]:
```
from urllib.request import urlopen
import json
with urlopen("https://raw.githubusercontent.com/plotly/datasets/master/
↪geojson-counties-fips.json") as response:
    counties = json.load(response)

# this is needed to "translate" FIPS into the geographical location

import plotly.express as px

fig = px.choropleth(county, geojson=counties, locations= "FIPS",␣
↪color="FFRPTH16",
                           color_continuous_scale= px.colors.sequential.OrRd,
                            range_color=(0, 1),
                            labels={"FFRPTH16": "Fast-food restaurants/1,000␣
↪pop, 2016"},
                           scope="usa")

# https://plotly.com/python/builtin-colorscales/

fig.update_traces(marker_line_width=0, marker_opacity=0.8)
fig.update_layout(coloraxis_colorbar=dict(tickvals=[0, 0.2, 0.4, 0.6, 0.8, 1],
    ticktext=["0", "0.2", "0.4", "0.6", "0.8", ">1"]))

# this is needed since some counties are >1, but without changing the ticktext,␣
↪the data is misleading
# # https://stackoverflow.com/questions/63094039/
↪plotly-express-can-you-manually-define-legend-in-px-choropleth

fig.update_geos(showsubunits=True, subunitcolor="black") # shows individual␣
↪states with a black outline
# https://community.plotly.com/t/
↪adding-state-lines-to-county-level-geojson-based-choropleth/36269/2

fig.show()

# https://plotly.com/python/choropleth-maps/
```

[14]:
```
order = ["Low", "Medium-Low", "Medium-High", "High"]

ax = sns.boxplot(data = county, x = "FFR Category", y = "Food insecurity raw␣
↪value", order = order, palette = sns.color_palette("YlOrRd"))
```

```
ax.set(ylabel= "Food insecurity %, 2020");
```

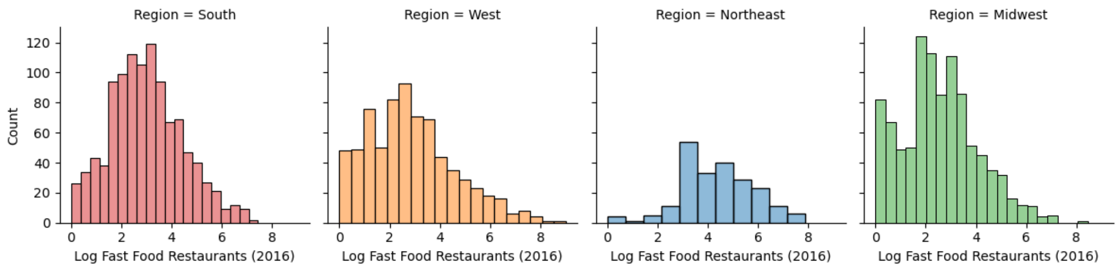


Food insecurity rates doesn't seem to differ based on their fast food restaurant category. Based on the boxplot, it seems that food insecurity rates are higher in counties assigned the "low" category.

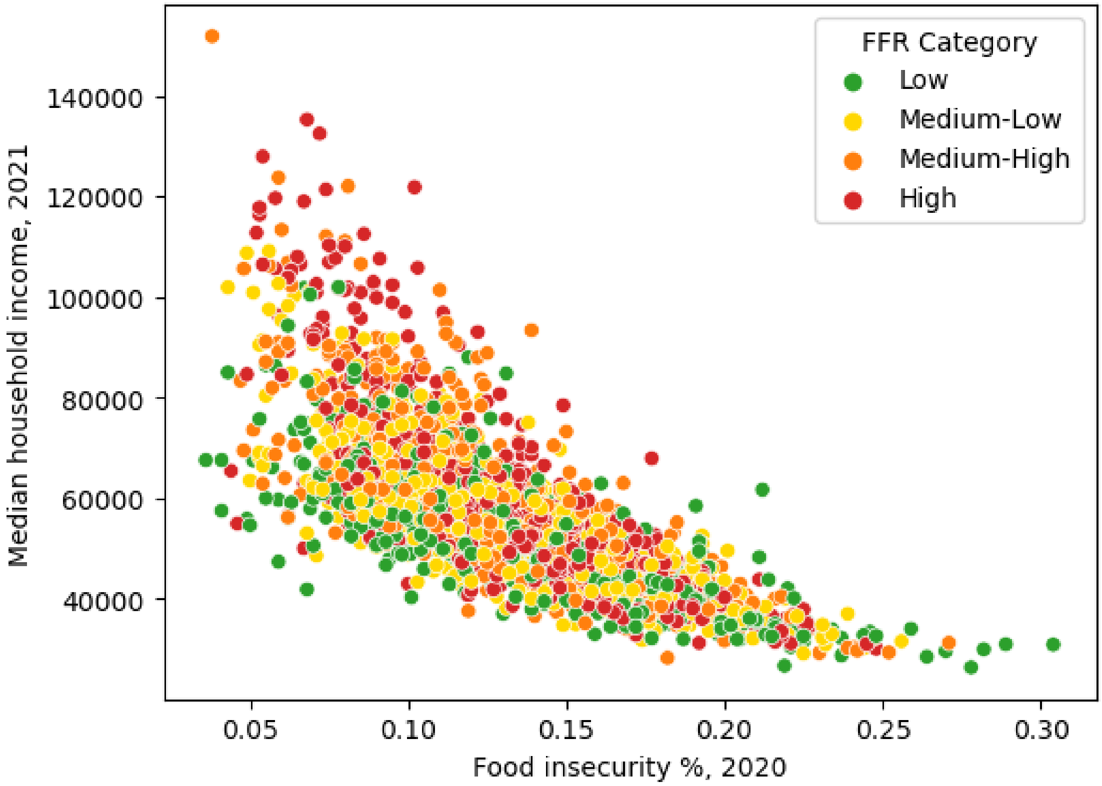
[15]:
```
county["Log FFR16"] = np.log(county["FFR16"]) # Needed to better visualize data

paletteRegion = {"South" : "tab:red",
          "West": "tab:orange",
          "Northeast": "tab:blue",
          "Midwest": "tab:green"}

g = sns.FacetGrid(county, col = "Region")
g.map_dataframe(sns.histplot, x = "Log FFR16", hue = "Region", palette =␣
 ↪paletteRegion)
g.set_xlabels(label = "Log Fast Food Restaurants (2016)");

#https://seaborn.pydata.org/generated/seaborn.FacetGrid.html
```

```
C:\Users\zachs\anaconda3\lib\site-packages\pandas\core\arraylike.py:397:
RuntimeWarning:
```

```
divide by zero encountered in log
```



[16]:
```python
palette = ("tab:green", "gold", "tab:orange", "tab:red")

ax = sns.scatterplot(data = countyNoNA, x = "Food insecurity raw value", y =
↪"Median household income raw value",
                    hue = "FFR Category", hue_order = order, palette = palette);

ax.set(xlabel= "Food insecurity %, 2020", ylabel= "Median household income,
↪2021");
```



As expected, as median household income decreases, food insecurity increases per county. However, it doesn't look like there is a trend with fast food category with these variables.

1.4 Unsupervised Machine Learning:

1.4.1 Principal component analysis: FFR Category

[17]:
```
import scipy
from scipy import stats

math = countyNoNA.drop(columns = ["FIPS", "State", "County", "Region", "FFR␣
↪Category", "FFany"])

normalCounty = (math - math.min())/(math.max()-math.min())

# Normalize data
# https://sparkbyexamples.com/pandas/normalize-columns-of-pandas-dataframe/
```

[18]:
```
from sklearn.decomposition import PCA

variables_county = normalCounty.drop(columns = ["FFR11", "FFR16", "FFRPTH11",␣
↪"FFRPTH16", "PC_FFRSALES07",
                                                      "PC_FFRSALES12",␣
↪"PCH_FFR_11_16", "Poor or fair health raw value",
                                                   "Premature death raw value",␣
↪"Limited access to healthy foods raw value"])
# Removed variables would skew the model

target_county = countyNoNA["FFR Category"]


model = PCA(n_components=2)

model.fit(variables_county)
```

[18]:
```
PCA(n_components=2)
```

[19]:
```
variables_county.head()
```

[19]:
```
   Adult obesity raw value  Food environment index raw value  \
0                 0.459290                          0.666667
1                 0.396660                          0.777778
2                 0.630480                          0.545455
3                 0.551148                          0.757576
4                 0.459290                          0.787879
```

```
   Diabetes prevalence raw value  Food insecurity raw value  \
0                       0.380074                   0.447761
1                       0.291513                   0.347015
2                       0.557196                   0.682836
3                       0.413284                   0.429104
4                       0.450185                   0.373134

   Median household income raw value  % below 18 years of age raw value  \
0                           0.254149                           0.470082
1                           0.267205                           0.415480
2                           0.076711                           0.394951
3                           0.171930                           0.388143
4                           0.211656                           0.461940

   % 65 and older raw value  % Non-Hispanic Black raw value  \
0                  0.208457                        0.230884
1                  0.302332                        0.099717
2                  0.278198                        0.556732
3                  0.221522                        0.244963
4                  0.259622                        0.016991

   % American Indian & Alaska Native raw value  % Asian raw value  \
0                                     0.230884           0.027347
1                                     0.099717           0.024831
2                                     0.556732           0.010944
3                                     0.244963           0.004992
4                                     0.016991           0.007451

   % Native Hawaiian/Other Pacific Islander raw value  % Hispanic raw value  \
0                                           0.008086              0.024483
1                                           0.005373              0.042537
2                                           0.016407              0.040510
3                                           0.009043              0.022300
4                                           0.009025              0.094094

   % Non-Hispanic White raw value  % Rural raw value  Total Pop  \
0                        0.747116           0.420022   0.005379
1                        0.846295           0.422791   0.020523
2                        0.450108           0.677896   0.002466
3                        0.753816           0.683526   0.002143
4                        0.883746           0.899515   0.005621

   Population Density
0            0.001288
1            0.001816
2            0.000404
3            0.000502
```

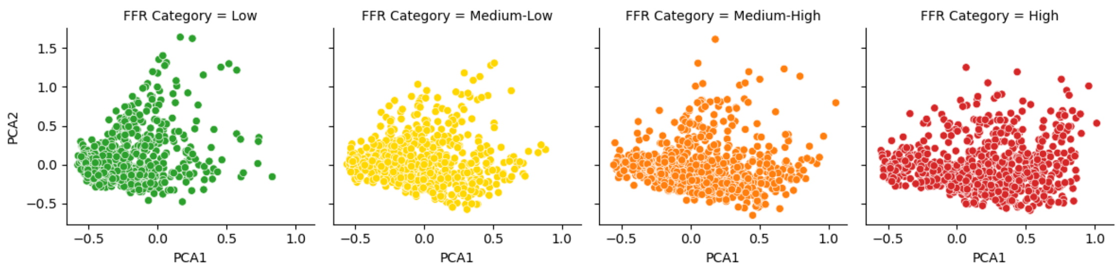
```
4           0.001240
```

[20]:
```
transform = model.transform(variables_county)
```

[21]:
```
countyNoNA["PCA1"] = transform[:,0]
countyNoNA["PCA2"] = transform[:,1]

countyNoNA.head()
```

[21]:
```
    FIPS State   County  FFR11  FFR16  FFRPTH11  FFRPTH16  PC_FFRSALES07  \
0  01001    AL  Autauga     34     44  0.615953  0.795977     649.511367
1  01003    AL  Baldwin    121    156  0.648675  0.751775     649.511367
2  01005    AL  Barbour     19     23  0.694673  0.892372     649.511367
3  01007    AL     Bibb      6      7  0.263794  0.309283     649.511367
4  01009    AL   Blount     20     23  0.347451  0.399569     649.511367

   PC_FFRSALES12  PCH_FFR_11_16  …  % Hispanic raw value  \
0      674.80272      29.411765  …              0.029909
1      674.80272      28.925620  …              0.047188
2      674.80272      21.052632  …              0.045248
3      674.80272      16.666667  …              0.027820
4      674.80272      15.000000  …              0.096531

   % Non-Hispanic White raw value  % Rural raw value  Total Pop  \
0                        0.737708           0.420022      55200
1                        0.832073           0.422791     208107
2                        0.455116           0.677896      25782
3                        0.744083           0.683526      22527
4                        0.867707           0.899515      57645

   Population Density  Region  FFany  FFR Category      PCA1      PCA2
0           35.853419   South    Yes          High  0.174647  0.117574
1           50.541504   South    Yes          High  0.081965 -0.139010
2           11.247981   South    Yes          High  0.146873  0.840137
3           13.973114   South    Yes           Low -0.064325  0.229523
4           34.515816   South    Yes           Low -0.357448  0.001697

[5 rows x 34 columns]
```

[22]:
```
g = sns.FacetGrid(countyNoNA, col = "FFR Category", col_order = order)

# set column order: https://stackoverflow.com/questions/61541776/
↪seaborn-ordering-of-facets

g.map_dataframe(sns.scatterplot, x= "PCA1", y= "PCA2", hue= "FFR Category",
                hue_order = order, palette = palette);
```



"Low" counties are more concentrated on the left of the graph and have a lower PCA1 score.

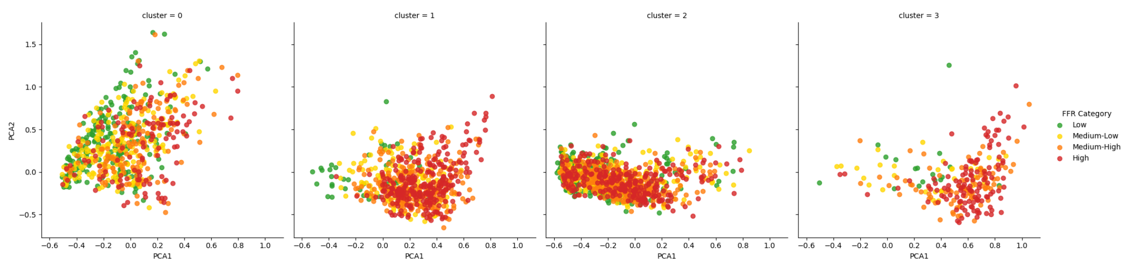
1.4.2 Gaussian Mixture Model Clustering: FFR Category

[23]:
```
from sklearn.mixture import GaussianMixture

model = GaussianMixture(n_components= 4, covariance_type= "full")

model.fit(variables_county)

predict = model.predict(variables_county)

countyNoNA["cluster"] = predict

countyNoNA.head()
```

[23]:
```
    FIPS State   County  FFR11  FFR16  FFRPTH11  FFRPTH16  PC_FFRSALES07  \
0  01001    AL  Autauga     34     44  0.615953  0.795977     649.511367
1  01003    AL  Baldwin    121    156  0.648675  0.751775     649.511367
2  01005    AL  Barbour     19     23  0.694673  0.892372     649.511367
3  01007    AL     Bibb      6      7  0.263794  0.309283     649.511367
4  01009    AL   Blount     20     23  0.347451  0.399569     649.511367

   PC_FFRSALES12  PCH_FFR_11_16  …  % Non-Hispanic White raw value  \
0      674.80272      29.411765  …                        0.737708
1      674.80272      28.925620  …                        0.832073
2      674.80272      21.052632  …                        0.455116
3      674.80272      16.666667  …                        0.744083
4      674.80272      15.000000  …                        0.867707

   % Rural raw value  Total Pop  Population Density  Region  FFany  \
0           0.420022      55200           35.853419   South    Yes
1           0.422791     208107           50.541504   South    Yes
2           0.677896      25782           11.247981   South    Yes
3           0.683526      22527           13.973114   South    Yes
4           0.899515      57645           34.515816   South    Yes
```

```
   FFR Category      PCA1      PCA2  cluster
0          High  0.174647  0.117574        1
1          High  0.081965 -0.139010        1
2          High  0.146873  0.840137        0
3           Low -0.064325  0.229523        0
4           Low -0.357448  0.001697        2

[5 rows x 35 columns]
```

[24]:
```
sns.lmplot(x= "PCA1" ,y= "PCA2",data= countyNoNA, hue= "FFR Category",
        palette = palette, col = "cluster", hue_order = order, fit_reg=False);
```

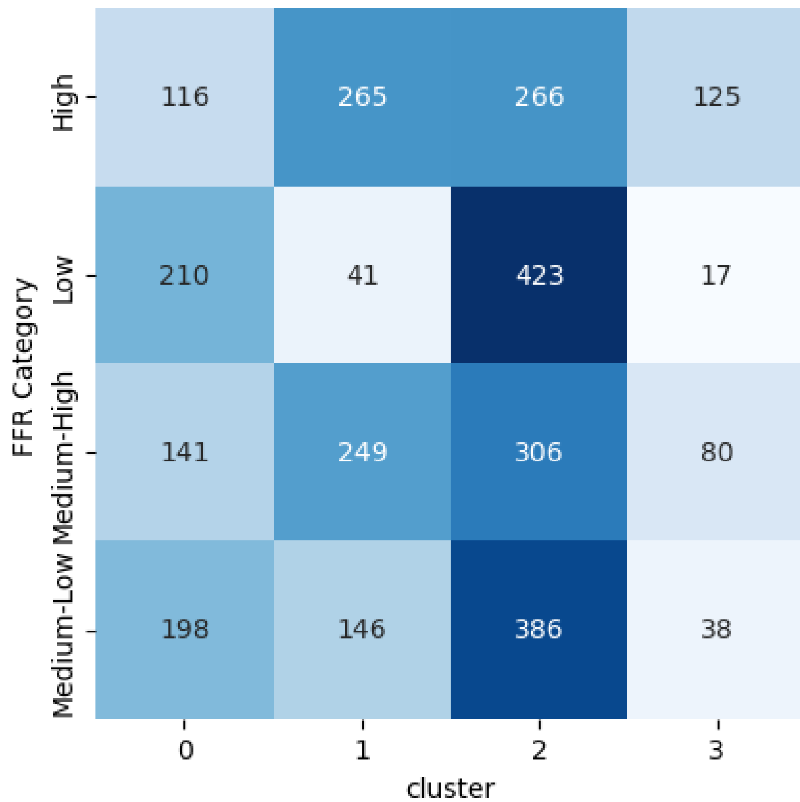


[25]:
```
ct = pd.crosstab(countyNoNA["FFR Category"], countyNoNA["cluster"])

print(ct)
```

```
cluster         0    1    2    3
FFR Category
High          116  265  266  125
Low           210   41  423   17
Medium-High   141  249  306   80
Medium-Low    198  146  386   38
```

[26]:
```
sns.heatmap(ct, square = True, annot = True, cbar = False, fmt = "g", cmap =␣
 ↪"Blues")

# Get rid of scientifc notation
#https://stackoverflow.com/questions/29647749/
 ↪seaborn-showing-scientific-notation-in-heatmap-for-3-digit-numbers
```

[26]: <AxesSubplot:xlabel='cluster', ylabel='FFR Category'>



Clustering was not successful with these classification groups.

1.4.3 K-means Clustering: FFR Category

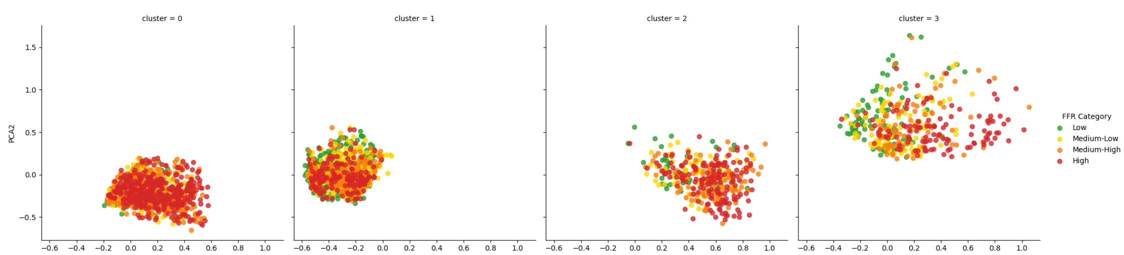
[27]:
```
countyNoNA.drop(columns = ["cluster"], inplace = True)
# drop previous test clusters

from sklearn.cluster import KMeans

kmeans_model = KMeans(n_clusters=4)

kmeans_model.fit(variables_county)

kmeans_predict = kmeans_model.predict(variables_county)

countyNoNA["cluster"] = kmeans_predict

countyNoNA.head()
```

[27]:
```
    FIPS State    County  FFR11  FFR16  FFRPTH11  FFRPTH16  PC_FFRSALES07  \
0  01001    AL   Autauga     34     44  0.615953  0.795977     649.511367
```

```
1  01003     AL  Baldwin     121     156  0.648675  0.751775      649.511367
2  01005     AL  Barbour      19      23  0.694673  0.892372      649.511367
3  01007     AL     Bibb       6       7  0.263794  0.309283      649.511367
4  01009     AL   Blount      20      23  0.347451  0.399569      649.511367

   PC_FFRSALES12  PCH_FFR_11_16  …  % Non-Hispanic White raw value  \
0      674.80272      29.411765  …                        0.737708
1      674.80272      28.925620  …                        0.832073
2      674.80272      21.052632  …                        0.455116
3      674.80272      16.666667  …                        0.744083
4      674.80272      15.000000  …                        0.867707

   % Rural raw value  Total Pop  Population Density  Region  FFany  \
0           0.420022      55200           35.853419   South    Yes
1           0.422791     208107           50.541504   South    Yes
2           0.677896      25782           11.247981   South    Yes
3           0.683526      22527           13.973114   South    Yes
4           0.899515      57645           34.515816   South    Yes

   FFR Category      PCA1      PCA2  cluster
0          High  0.174647  0.117574        0
1          High  0.081965 -0.139010        0
2          High  0.146873  0.840137        3
3           Low -0.064325  0.229523        1
4           Low -0.357448  0.001697        1

[5 rows x 35 columns]
```

[28]:

```python
sns.lmplot(x= "PCA1" ,y= "PCA2",data= countyNoNA, hue= "FFR Category",
        palette = palette, col = "cluster", hue_order = order, fit_reg=False);
```



[29]:

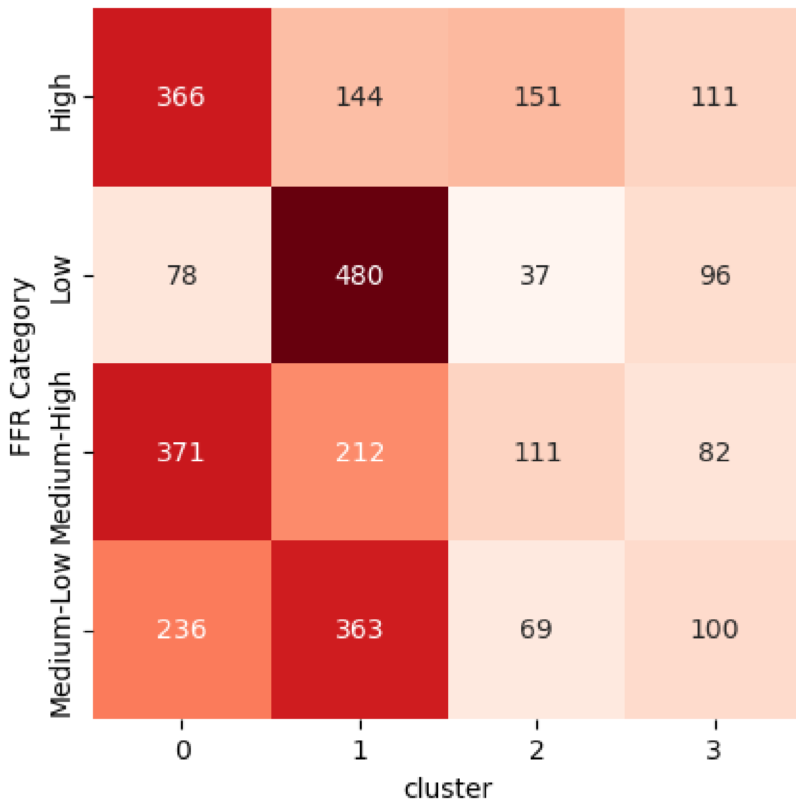
```python
ct = pd.crosstab(countyNoNA["FFR Category"], countyNoNA["cluster"])

print(ct)
```

```
cluster         0    1    2    3
FFR Category
```

```
High          366  144  151  111
Low            78  480   37   96
Medium-High   371  212  111   82
Medium-Low    236  363   69  100
```

[30]:
```
sns.heatmap(ct, square = True, annot = True, cbar = False, fmt = "g", cmap =␣
↪"Reds")
```

[30]:
```
<AxesSubplot:xlabel='cluster', ylabel='FFR Category'>
```



K-means clustering is slightly better, but overall not successful.

1.4.4 Principal component analysis: FFany

[31]:
```
model = PCA(n_components=2)

model.fit(variables_county)
```

[31]:
```
PCA(n_components=2)
```

[32]:
```
transform = model.transform(variables_county)
```

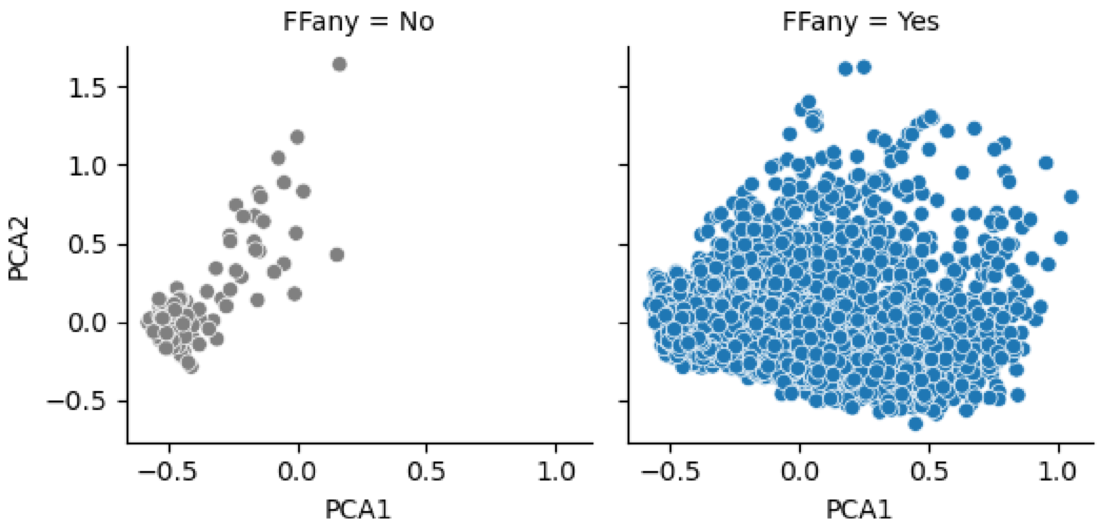
```python
countyNoNA.drop(columns = ["PCA1", "PCA2", "cluster"], inplace = True)
# drop previous test PCAs and clusters

countyNoNA["PCA1"] = transform[:,0]
countyNoNA["PCA2"] = transform[:,1]

countyNoNA.head()
```

[32]:

```
    FIPS State   County  FFR11  FFR16  FFRPTH11  FFRPTH16  PC_FFRSALES07  \
0  01001    AL  Autauga     34     44  0.615953  0.795977     649.511367
1  01003    AL  Baldwin    121    156  0.648675  0.751775     649.511367
2  01005    AL  Barbour     19     23  0.694673  0.892372     649.511367
3  01007    AL     Bibb      6      7  0.263794  0.309283     649.511367
4  01009    AL   Blount     20     23  0.347451  0.399569     649.511367

   PC_FFRSALES12  PCH_FFR_11_16  …  % Hispanic raw value  \
0      674.80272      29.411765  …              0.029909
1      674.80272      28.925620  …              0.047188
2      674.80272      21.052632  …              0.045248
3      674.80272      16.666667  …              0.027820
4      674.80272      15.000000  …              0.096531

   % Non-Hispanic White raw value  % Rural raw value  Total Pop  \
0                        0.737708           0.420022      55200
1                        0.832073           0.422791     208107
2                        0.455116           0.677896      25782
3                        0.744083           0.683526      22527
4                        0.867707           0.899515      57645

   Population Density  Region  FFany  FFR Category      PCA1      PCA2
0           35.853419   South    Yes          High  0.174647  0.117574
1           50.541504   South    Yes          High  0.081965 -0.139010
2           11.247981   South    Yes          High  0.146873  0.840137
3           13.973114   South    Yes           Low -0.064325  0.229523
4           34.515816   South    Yes           Low -0.357448  0.001697

[5 rows x 34 columns]
```

[33]:

```python
NoYes = ["No", "Yes"]

g = sns.FacetGrid(countyNoNA, col = "FFany", col_order = NoYes)

g.map_dataframe(sns.scatterplot, x= "PCA1", y= "PCA2", hue= "FFany", hue_order␣
↪= NoYes,  palette = ("gray", "tab:blue"));
```



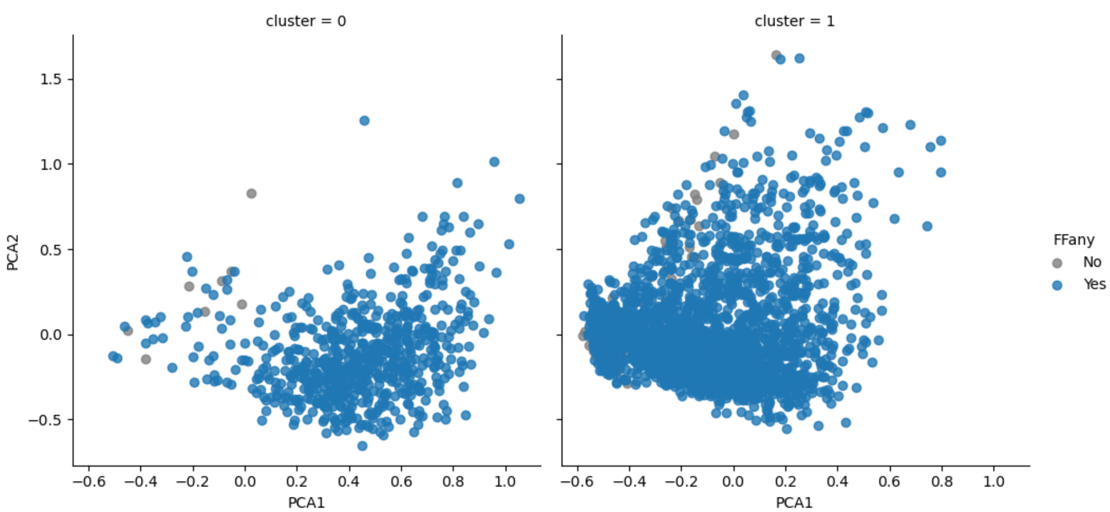
[34]:
```
model = GaussianMixture(n_components=2, covariance_type= "full")

model.fit(variables_county)

predict = model.predict(variables_county)

countyNoNA["cluster"] = predict

countyNoNA.head()
```

[34]:
```
    FIPS State   County  FFR11  FFR16  FFRPTH11  FFRPTH16  PC_FFRSALES07  \
0  01001    AL  Autauga     34     44  0.615953  0.795977     649.511367
1  01003    AL  Baldwin    121    156  0.648675  0.751775     649.511367
2  01005    AL  Barbour     19     23  0.694673  0.892372     649.511367
3  01007    AL     Bibb      6      7  0.263794  0.309283     649.511367
4  01009    AL   Blount     20     23  0.347451  0.399569     649.511367

   PC_FFRSALES12  PCH_FFR_11_16  …  % Non-Hispanic White raw value  \
0      674.80272      29.411765  …                        0.737708
1      674.80272      28.925620  …                        0.832073
2      674.80272      21.052632  …                        0.455116
3      674.80272      16.666667  …                        0.744083
4      674.80272      15.000000  …                        0.867707

   % Rural raw value  Total Pop  Population Density  Region  FFany  \
0           0.420022      55200           35.853419   South    Yes
1           0.422791     208107           50.541504   South    Yes
2           0.677896      25782           11.247981   South    Yes
```

```
3           0.683526      22527           13.973114    South     Yes
4           0.899515      57645           34.515816    South     Yes

   FFR Category       PCA1       PCA2  cluster
0          High   0.174647   0.117574        1
1          High   0.081965  -0.139010        0
2          High   0.146873   0.840137        1
3           Low  -0.064325   0.229523        1
4           Low  -0.357448   0.001697        1

[5 rows x 35 columns]
```

1.4.5 Gaussian Mixture Model Clustering: FFany

[35]:
```python
sns.lmplot(x= "PCA1" ,y= "PCA2", data= countyNoNA, hue= "FFany",
             hue_order = NoYes,  palette = ("gray", "tab:blue"), col =
 "cluster", fit_reg=False);
```

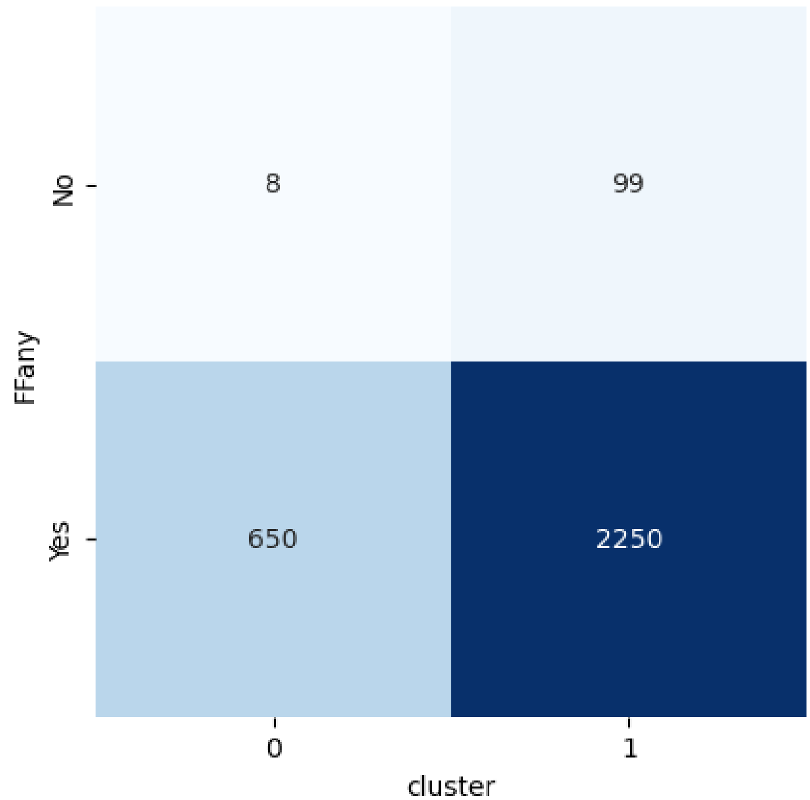


[36]:
```python
ct = pd.crosstab(countyNoNA["FFany"], countyNoNA["cluster"])

print(ct)
```

```
cluster    0     1
FFany
No         8    99
Yes      650  2250
```

[37]:
```python
sns.heatmap(ct, square = True, annot = True, cbar = False, fmt = "g", cmap =
 "Blues")
```

[37]: <AxesSubplot:xlabel='cluster', ylabel='FFany'>



1.4.6 K-means Clustering: FFany

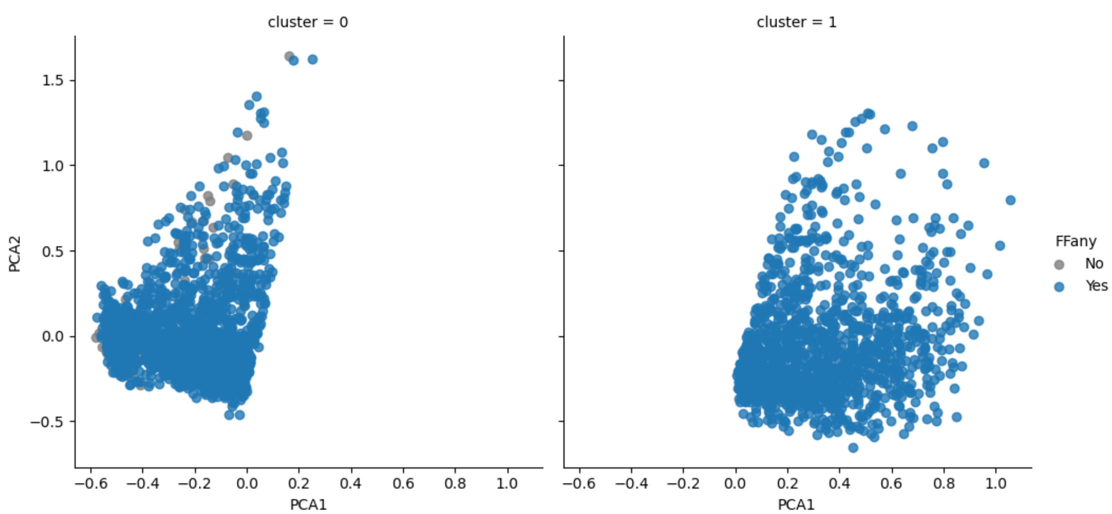
[38]:
```
countyNoNA.drop(columns = ["cluster"], inplace = True)
# drop previous test clusters

kmeans_model = KMeans(n_clusters=2)

kmeans_model.fit(variables_county)

kmeans_predict = kmeans_model.predict(variables_county)

countyNoNA["cluster"] = kmeans_predict

countyNoNA.head()
```

[38]:

	FIPS	State	County	FFR11	FFR16	FFRPTH11	FFRPTH16	PC_FFRSALES07	\
0	01001	AL	Autauga	34	44	0.615953	0.795977	649.511367	
1	01003	AL	Baldwin	121	156	0.648675	0.751775	649.511367	
2	01005	AL	Barbour	19	23	0.694673	0.892372	649.511367	

```
3  01007     AL     Bibb       6       7  0.263794  0.309283      649.511367
4  01009     AL   Blount      20      23  0.347451  0.399569      649.511367

   PC_FFRSALES12  PCH_FFR_11_16  …  % Non-Hispanic White raw value  \
0      674.80272      29.411765  …                        0.737708
1      674.80272      28.925620  …                        0.832073
2      674.80272      21.052632  …                        0.455116
3      674.80272      16.666667  …                        0.744083
4      674.80272      15.000000  …                        0.867707

   % Rural raw value  Total Pop  Population Density  Region  FFany  \
0           0.420022      55200           35.853419   South    Yes
1           0.422791     208107           50.541504   South    Yes
2           0.677896      25782           11.247981   South    Yes
3           0.683526      22527           13.973114   South    Yes
4           0.899515      57645           34.515816   South    Yes

   FFR Category      PCA1      PCA2  cluster
0          High  0.174647  0.117574        1
1          High  0.081965 -0.139010        1
2          High  0.146873  0.840137        0
3           Low -0.064325  0.229523        0
4           Low -0.357448  0.001697        0

[5 rows x 35 columns]
```

[39]:
```
sns.lmplot(x= "PCA1" ,y= "PCA2", data= countyNoNA, hue= "FFany",
              hue_order = NoYes,  palette = ("gray", "tab:blue"), col =␣
 ↪"cluster", fit_reg=False);
```

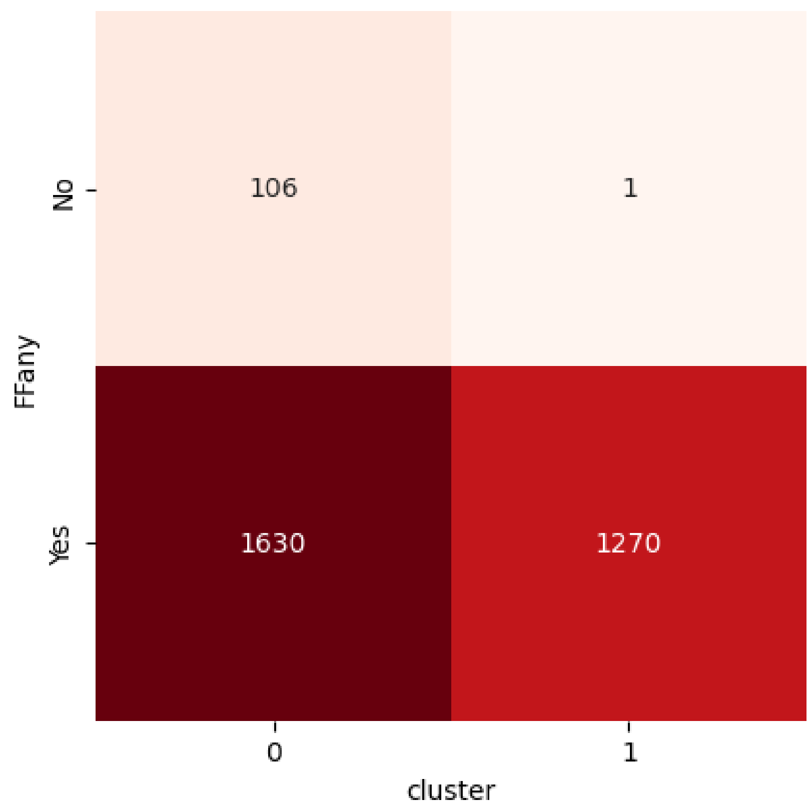


[40]:
```
ct = pd.crosstab(countyNoNA["FFany"], countyNoNA["cluster"])

print(ct)
```

```
cluster     0     1
FFany
No        106     1
Yes      1630  1270
```

[41]:
```
sns.heatmap(ct, square = True, annot = True, cbar = False, fmt = "g", cmap =␣
 ↪"Reds")
```

[41]:
```
<AxesSubplot:xlabel='cluster', ylabel='FFany'>
```



Similar to the previous example, clustering was not accurate and successful.

1.5 Supervised Learning: Linear regression

1.5.1 Predicting Fast Food Restaurants, all Counties

[42]:
```
import statsmodels.api as sm
```

```
[43]: countyLR = countyNoNA.drop(columns = ["FIPS", "State", "County",␣
      ↪"FFR11","FFRPTH11", "PC_FFRSALES07",
                                "PC_FFRSALES12", "PCH_FFR_11_16", "Region",␣
      ↪"FFany", "FFR Category", "FFR11",
                               "PCA1", "PCA2", "cluster", "FFR16", '% Non-Hispanic␣
      ↪Black raw value',
                               '% American Indian & Alaska Native raw value', '%␣
      ↪Asian raw value',
                               '% Native Hawaiian/Other Pacific Islander raw value',
                               '% Hispanic raw value'])

      x = countyLR.drop(columns = ["FFRPTH16"])
      y = countyLR["FFRPTH16"]
```

```
[44]: x = sm.add_constant(x)

      model = sm.OLS(y, x)

      result = model.fit()

      print(result.summary())

      # multiple regression
      # https://www.datarobot.com/blog/multiple-regression-using-statsmodels/
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:               FFRPTH16   R-squared:                       0.290
Model:                            OLS   Adj. R-squared:                  0.286
Method:                 Least Squares   F-statistic:                     87.19
Date:                Sun, 10 Dec 2023   Prob (F-statistic):          4.80e-210
Time:                        16:01:16   Log-Likelihood:                 58.870
No. Observations:                3007   AIC:                            -87.74
Df Residuals:                    2992   BIC:                             2.390
Df Model:                          14
Covariance Type:            nonrobust
================================================================================
=============================
                                                 coef    std err          t
P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
-----------------------------
const                                         -0.1231      1.621     -0.076
0.939      -3.301       3.054
Premature death raw value                   1.062e-05   2.48e-06      4.282
0.000    5.76e-06    1.55e-05
Poor or fair health raw value                 -1.5603      0.202     -7.722
```

```
0.000      -1.956      -1.164
Adult obesity raw value                       -0.5472      0.095     -5.789
0.000      -0.733      -0.362
Food environment index raw value               0.1162      0.150      0.776
0.438      -0.178       0.410
Diabetes prevalence raw value                 -0.0022      0.163     -0.013
0.989      -0.321       0.317
Food insecurity raw value                      4.2161      2.961      1.424
0.155      -1.590      10.022
Limited access to healthy foods raw value      0.6488      1.332      0.487
0.626      -1.962       3.260
Median household income raw value         -1.106e-06   5.56e-07     -1.989
0.047    -2.2e-06   -1.55e-08
% below 18 years of age raw value             -0.0605      0.170     -0.356
0.722      -0.394       0.273
% 65 and older raw value                      -0.0613      0.144     -0.425
0.671      -0.344       0.222
% Non-Hispanic White raw value                 0.0076      0.031      0.246
0.806      -0.053       0.068
% Rural raw value                             -0.4706      0.019    -25.292
0.000      -0.507      -0.434
Total Pop                                 -5.264e-08   1.51e-08     -3.492
0.000   -8.22e-08   -2.31e-08
Population Density                         2.193e-05   6.63e-06      3.307
0.001    8.93e-06    3.49e-05
==============================================================================
Omnibus:                     1884.744   Durbin-Watson:                   1.833
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            57852.584
Skew:                           2.469   Prob(JB):                         0.00
Kurtosis:                      23.913   Cond. No.                     2.96e+08
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 2.96e+08. This might indicate that there are
strong multicollinearity or other numerical problems.
```

Surprisingly, most variables don't have a p-value less then 0.05 and go against the hypothesis.

Increases in poor self rated health and adult obesity % predict decreases in fast food restaurants per capita.

However, decreases in rural area % predict increases in fast food restaurants per capita.

1.6 Discussion and Conclusion

The machine learning methods used in this project were overall not successful in predicting the amount of fast food restuarants each county has. While k-means clustering did a better job than

Gaussian mixture model clustering, there was very low accuracy overall.

For some counties, there was not much of a difference between their principal component analysis 1 and 2 variable scores, even if they had been assigned widely different FFR Category groups. Attempting to mitigate these issues by trying to cluster based off of the FFany variable was slightly better, but not great.

The OLS regression found that most variables were unimportant. Many had p-values above 0.05, and those that had below 0.05 factored very little.

Future work could involve replicating these methods, but on a much smaller scale. The United States is very diverse in its peoples, geography, local economies, and attitudes towards food. Limiting the area of study, such as only examining one particular state, could resolve some issues.

Perhaps fast food based food insecurity is more dependant on the individual consumer, and not the entire county.

1.7 Bibliography

1.7.1 Research Sources

CDC: Fast Food Consumption Among Adults in the United States, 2013–2016

Fuhrman, Joel (MD): The Hidden Dangers of Fast and Processed Food

USDA: Number of fast food restaurants per capita varies across the U.S.

Martini et al.: Machine learning can guide food security efforts when primary data are not available

Zhou et al.: Machine learning for food security: Principles for transparency and usability

Richardson et al.: Neighborhood fast food restaurants and fast food consumption: A national study

1.7.2 Coding Sources

Stack Overflow: Import column as string

Stack Overflow: Import select columns from a csv file

Geeks for Geeks: Drop rows with NA values in one column

Pandas: Merge dataframes

Stack Overflow: Three way dataframe merge

Numpy: numpy.select

Plotly: Built-in colorscales

Stack Overflow: Change legend tick intervals

Plotly: Black outline to choropleth map

Plotly: Choropleth maps

Seaborn: FacetGrid

Spark By Examples: Normalize Data

Stack Overflow: Set Column Order in FacetGrid

Stack Overflow: Remove scientific notation from Seaborn Heatmap

Data Robot: Multiple regression