Recommender System

Team members: Yingyin Xiao, Dun Lin, Janathon Lin

Introduction

We are trying to build a recommender system that recommends business to users. To develop a recommendation system for users, essentially we are looking at their preferences in different categories, which are presumably dependent on the rating and location. Since users are the target, we would generate a model to predict the users' preference specifically.

First of all, we imported the datasets and did visualizing analysis.

In [1]:
```
# import libraries
import os
import io
import json
import gzip
from collections import defaultdict # shoutout
import numpy as np
import random
import matplotlib.pyplot as plt
import seaborn as sns
```

In [2]:
```
# list current directory
os.listdir()
```

Out[2]:
```
['.DS_Store',
 'places.clean.json.gz',
 'sample.reviews.json',
 'users.clean.json.gz',
 'project.ipynb',
 'reviews.clean.json.gz',
 'users.clean.json',
 'Categories.ipynb',
 '.ipynb_checkpoints',
 'yelp-category.csv']
```

Read in a gz file

- since file is too large, we extract the first 200 rows

In [3]:
```
def readGz(fname):
    gz = gzip.open(fname, 'rb')
    f = io.BufferedReader(gz)
    data = []

    counter = 0
    for l in f.readlines():
        if counter > 2000:
            break
        else:
            counter += 1
            data.append(eval(l))

    gz.close()
    return data

places = readGz("places.clean.json.gz")
```

In [5]:
```
users = readGz("users.clean.json.gz")
```

In [6]:
```
# load sample reviews file instead
reviews = []
with open('sample.reviews.json') as f:
    reviews = json.load(f)
    print(len(reviews))
```

```
500000
```

In [7]:
```
from pandas.io.json import json_normalize
import pandas as pd
places_df = pd.DataFrame.from_dict(json_normalize(places), orient='colum
ns')
users_df = pd.DataFrame.from_dict(json_normalize(users), orient='column
s')
reviews_df = pd.DataFrame.from_dict(json_normalize(reviews), orient = 'c
olumns')
```

In [8]:
```
reviews_df.groupby('gPlusUserId')['rating'].value_counts().head()
```

Out[8]:
```
gPlusUserId            rating
100000021336848867366  5.0       1
100000032416892623125  4.0       1
100000036174088924566  5.0       1
100000042779388982190  5.0       1
100000059843227870895  3.0       1
Name: rating, dtype: int64
```

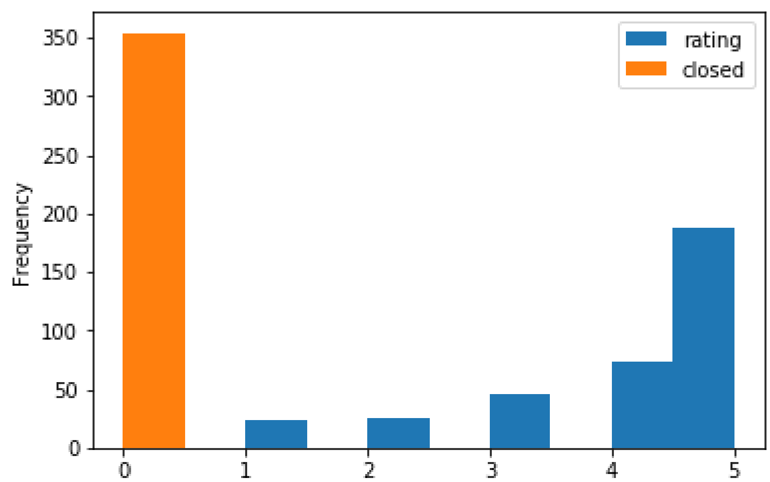
Examine if the column "closed" affects the rating of the place. If it does, then it should be taken into the consideration of one of the features that determines the recommender system.

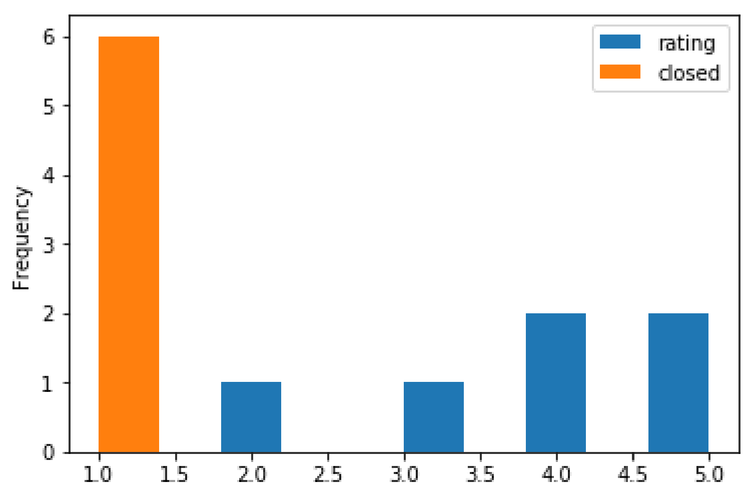
In [9]:
```
combo = reviews_df.merge(places_df, how = 'inner', on = 'gPlusPlaceId')
display((combo.closed == True).mean())
```

```
0.016666666666666666
```

In [10]:
```
def change(r):
    if r == True:
        return 1
    elif r == False:
        return 0
    else:
        return
combo['closed'] = combo['closed'].apply(change)
(
   combo[['rating','closed']]
    .groupby('closed')
    .plot(kind = 'hist', legend = True))
```

Out[10]:
```
closed
0    AxesSubplot(0.125,0.125;0.775x0.755)
1    AxesSubplot(0.125,0.125;0.775x0.755)
dtype: object
```





We could not tell whether business closing would affect the rating, since the difference is not as obvious, so we move to "review time". First, we examine if the year of review affects rating.

In [15]:
```
reviews_df['reviewTime'] = pd.to_datetime(reviews_df['reviewTime'])
```

In [16]:
```
sorted_rev = reviews_df.sort_values(by = 'reviewTime')
sorted_rev['reviewYear'] = reviews_df.reviewTime.dt.year
```

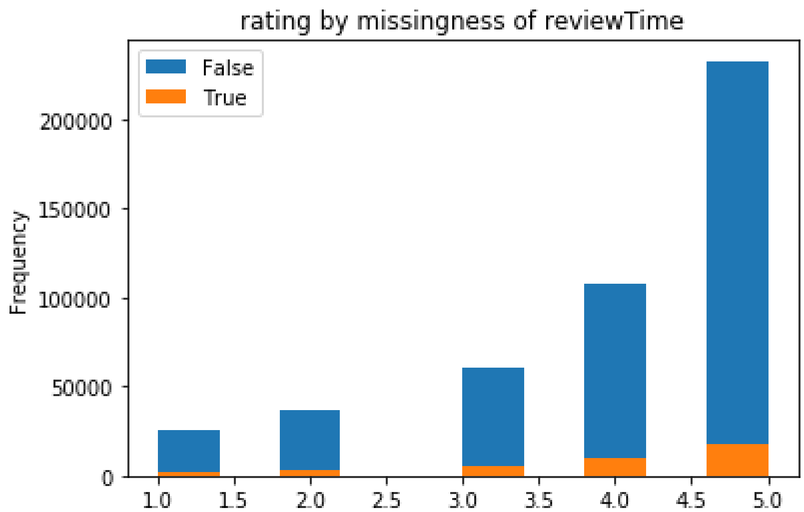
In [17]:
```
sorted_rev[['reviewYear', 'rating']].groupby('reviewYear')['rating'].mea
n()
```

Out[17]:
```
reviewYear
1990.0    4.072464
2001.0    4.250000
2002.0    4.043478
2003.0    3.972222
2004.0    3.924138
2005.0    3.969231
2006.0    3.900383
2007.0    3.908696
2008.0    3.806998
2009.0    3.707412
2010.0    3.871208
2011.0    3.981289
2012.0    4.097034
2013.0    4.044226
2014.0    4.094523
Name: rating, dtype: float64
```

From year 1990 to 2014, we see that the mean of the rating doesn't change too much, which means review of the year is not a big factor that influences people's rating.

In [18]:
```
(
    sorted_rev
    .assign(is_null=sorted_rev.reviewYear.isnull())
    .groupby('is_null')
    .rating
    .plot(kind='hist', legend=True, title='rating by missingness of revi
ewTime')
)
```

Out[18]:
```
is_null
False    AxesSubplot(0.125,0.125;0.775x0.755)
True     AxesSubplot(0.125,0.125;0.775x0.755)
Name: rating, dtype: object
```



Though the mean tells us the difference is trivial, missingness of the reviewTime vs. ratings tells us reviewTime is not missing completely by random, indicating that they have correlation, so we put reviewTime into one of the features.

validation set should be the top 5 businesses recommended. Now I am putting the information into one table, so we could use the model.

Approach

Categories

- Based on a user's previous reviews, we want to recommend businesses that relate to their categories Using a Yelp list of possible categories, we can categorize businesses into 22 types. Then make recommendations using businesses that categorized into the same type.

In [20]:
```
def separate(lst):
    output = []
    if lst is None:
        return
    for i in lst:
        output += i.split()
    return output
```

In [21]:
```
# Separate categories
df1 = reviews_df
df1['categories'] = df1['categories'].apply(separate)

# Import
df3 = pd.read_csv('yelp-category.csv')
df3['Secondary Categories'] = df3['Secondary Categories'].apply(lambda x
: x.split())

# Separate elements of a list into one list on whitespace; i.e. ['a b',
 'c d'] --> ['a', 'b', 'c', 'd']
df3 = df3.groupby('Primary Categories').agg({'Secondary Categories': 'su
m'}).reset_index()

# Identify Group Categories
primary_list = df3['Primary Categories'].value_counts().index.tolist()

# Category List; Key is Primary Category, Value is List of Elements of S
econd Category
cat_list = dict(zip(df3['Primary Categories'], df3['Secondary Categorie
s']))

# Introduce New Columns into original df
for col in primary_list:
    print(col) # progress bar
    df1[col] = pd.Series([(lambda x: any(i in cat_list[col] for i in x))
(x) \
                          for x in df1['categories'] if x is not None])
# output
df1
```

```
Food
Arts Entertainment
Nightlife
Health Medical
Shopping
Local Services
Financial Services
Real Estate
Beauty Spas
Pets
Education
Event Planning Services
Restaurants
Religious Organizations
Mass Media
Public Services Government
Home Services
Professional Services
Automotive
Active Life
Hotels Travel
```

Out[21]:

	categories	gPlusPlaceId	gPlusUserId	rating	reviewText	rev
0	[Liquor, Store]	114925821516688138194	116425610241236691097	5.0	Very nice and friendly. Good prices period!	20
1	[Restaurant]	114682145364484181515	103860393153033344115	3.0	None	20
2	[Air, Duct, Cleaning, Service]	115669271055428738535	113081498166076457035	5.0	These guys know their stuff. They went above a...	20
3	[Pizza, Restaurant, European, Restaurant, Ital...	111941167660939978882	106153641177787026098	5.0	I love this place. It is the best pizza place ...	20
4	[Stores, and, Shopping]	110391771625004866921	117155856496626773171	5.0	None	20
5	[Hotel, Conference, Center, Wedding, Venue]	109852713940297521047	103770677869830976765	3.0	This hotel receives a 3. Everything was ther...	20
6	[Barber, Shop]	107662653277549107610	104586295404256608323	5.0	Thoroughly Impressive. The art of a shave and ...	20
7	[Sushi, Restaurant]	106125283893529710008	112227921873451203379	5.0	Best sushi in Toledo, hands down!!!	20
8	[Dental, Clinic, Cosmetic, Dentist]	116541407946218420572	106696123708364240150	5.0	I highly recommend McKinney Dentist to all my ...	20
9	[Department, Store]	105604475535252952337	100788862244810273226	3.0	Seriously! I know there are still employees in...	20
10	[Restaurant]	110128953026710713434	111887420408060199060	4.0	I had never been to Killarney and was recommen...	20
11	[Pan-Latin, Restaurant, Latin, American, Resta...	108362455980329688201	115529531005064321048	2.0	This Mexican-Asian fusion place takes up a goo...	20
12	[Sandwich, Shop, Fast, Food, Restaurant, Hambu...	105476053511679800709	115152155256434284655	5.0	One of my favorite places to indulge in a sand...	20

	categories	gPlusPlaceId	gPlusUserId	rating	reviewText	rev
13	[Mexican, Grocery, Store]	103434248426747675089	107135843791159394677	5.0	This little hole in the wall has some of the s...	20
14	[Flooring, Store, Professional, Services]	115405695169303363511	108069241587088560650	5.0	We had been thinking about restoring the appea...	20
15	[American, Restaurant]	105890867174623506993	104317602234883483517	4.0	Glad I didn't read the reviews before we went....	20
16	[Furniture, Store]	103070084553231792489	101192543446872830305	5.0	Best purchase experience! Will definitely be b...	20
17	[Diner, Fast, Food, Restaurant, Seafood, Resta...	101498825316457513167	117209672961921000573	5.0	None	20
18	[Art, Center, Performing, Arts, Theater]	100439308203621824539	112780603331056741628	1.0	.	20
19	[Chinese, Restaurant, Asian, Restaurant]	118416572844859238945	108767763873912925492	5.0	These guys have the best Asian food on the wes...	20
20	[Marketing, Consultant]	111483424280797950177	102029173329517841566	5.0	As an owner of an SEO Firm, I work with a LOT ...	20
21	[Baby, Store, Baby, Clothing, Store]	116467659180871085419	107513231393652284700	1.0	Personnel très désagréable. On ne s occupe pas...	20
22	[Transit, Line, Shipping, and, Mailing]	117437830878542871720	114302578042916920895	4.0	We took an autoprogresso tour when my family w...	20
23	[Pub]	115377910616059612171	104240391260588088694	5.0	Our local love it	20
24	[Brewery, Beer, Store, Brewing, Supply, Store]	116255695206537406043	107354940521199286060	5.0	None	20
25	[Bagel, Shop]	112512224575729516572	107255357678978458660	5.0	Very delicious bagels. Great service. Recommen...	20

	categories	gPlusPlaceId	gPlusUserId	rating	reviewText	rev
26	[Asian, Restaurant, Noodle, Shop]	101498399682617416769	105035095475091037591	5.0	This is quite simply the best Lak Sa you will ...	20
27	[Advertising, Agency, Website, Designer, Graph...	107644525659465239839	102669726718204864717	5.0	Wir haben unsere Webseite komplett neue design...	20
28	[Liquor, Store]	109776549491915256664	102863165178545312172	4.0	Staff is friendly and knowledgeable. Decent be...	20
29	[Pet, Store]	111418443395287914317	115621663235829950364	3.0	I love going to any pet store because I just l...	20
...	...	...	...	...	...	
499970	[Family, Restaurant, Catering, Pizza, Restaurant]	100516535045772111439	111006942665943136632	4.0	best in chicago	20
499971	None	118379564434385983105	113094657754931703381	4.0	best location right to tour ticket	
499972	[Pawn, Shop]	111427551961520907476	114243877905544018118	1.0	None	20
499973	[Beauty, Salon]	107737092105236079701	100293724178989280178	5.0	The best haircuts on the island for the best p...	20
499974	[Buddhist, Temple]	116686016694162035048	117432297272919267798	4.0	Peace!	20
499975	[Restaurant]	111407322802028102625	103647758334709097817	5.0	Molto bello, servizio ottimo ma soprattutto il...	20
499976	[Indoor, Lodging]	102073830472602805068	114194868950412774323	5.0	None	20
499977	[Courier, Service]	116708440932355561016	106517386391012661896	5.0	Left the sweaty FedEx near Bloor/Spadina and w...	20
499978	[Mosque]	112250362529412896072	110460857021855102442	5.0	Ok	20
499979	[Restaurant]	100848932410910153105	114395399737128477553	3.0	C	20
499980	[Beach, Resort]	107596216512320683390	112608504335987307898	3.0	Nice sea view	20

	categories	gPlusPlaceId	gPlusUserId	rating	reviewText	rev
499981	None	116590996231594025781	106654503918907830147	4.0	More non-dairy and vegan options you could eve...	
499982	[Auto, Repair, Shop, Car, Detailing, Service, ...	110227571133581002749	112564077131535025875	1.0	Went in looking for a specific car from the we...	20
499983	[カレー]	109595490650194636288	110398175014933586565	5.0	旨いです	20
499984	[Computer, Training, School]	102553644633183578748	116397861309469141300	2.0	They are bitch.... totally bitch.. Sala 8 mont...	20
499985	None	104709503962627854418	102883593640621964041	5.0	None	20
499986	None	118181698030035480636	110599527049716261391	5.0	Uma ótima empresa, levei meu notebook la e rec...	
499987	[Fast, Food, Restaurant, Hamburger, Restaurant]	105236743510615797456	102541896073371530085	3.0	Way over price should not cost over .80 per. H...	20
499988	[Ramen, Restaurant, Noodle, Shop]	112180403705878358769	118227885384517039280	4.0	700円で並・中・大選べるのが良いです。ネギ飯も美味しかったです。	20
499989	[Auto, Body, Shop]	108722380094766206203	115942816109313667879	5.0	I had scraped my car pulling in too close to t...	20
499990	[Seafood, Restaurant, Seafood, Wholesaler]	111055690129361822246	104681711536002944997	3.0	None	20
499991	[Japanese, Restaurant, Asian, Restaurant]	113147685766037059498	114054720139410468034	4.0	None	20
499992	[Furniture, Store]	102295032017336340760	105171433773205930884	5.0	None	20
499993	[Restaurant, Breakfast, Restaurant, Caterer]	113671722973515207546	105812269051240397309	1.0	Terrible Experience, 15 to get acknowledge by ...	20

	categories	gPlusPlaceId	gPlusUserId	rating	reviewText	rev
499994	[Honda, Dealer, Auto, Repair, Shop, Used, Car,...	115260856856615896091	104196153381047444780	5.0	We'd been to a few car dealerships (for a vari...	20
499995	[Ford, Dealer]	112417904085749987687	116583583954333537204	5.0	My experience at Galpin was excellent. Jeff Le...	20
499996	[Movie, Theater]	110081618419347485364	104723189161600817863	4.0	Not easy to get to (need to travel up 3 long e...	20
499997	[Hotel, Motel]	102590848144467070611	112724086223739446476	4.0	Stopped here on our way to Cabot Trail staff w...	20
499998	[Bed, &, Breakfast, Hostel]	114687148170012563644	116550451315359170311	5.0	Ferien dort sind ein Traum, das Anwesen ist ei...	20
499999	[Bar, Seafood, Restaurant]	117757161834484259080	102604452078876976373	5.0	F-ing Amazing!!!! Char grilled oysters-foodgas...	20

500000 rows × 29 columns

From here we know what does the categories belong to.

GPS

- By merging data from where a user makes reviews, we can see the GPS location average of where to make recommendations
    - Ideally with extra time, we can employ a clustering algorithm and avoid outliers such as one time travel locations for each user
- Then by using a k-NN algorithm, we should be able to find businesses that are closest to the user

In [35]:
```
%%capture
sample_reviews = reviews_df
places = places_df
users = users_df
sample_reviews_gps = sample_reviews[['gPlusPlaceId', 'gPlusUserId', 'rat
ing', 'reviewTime']]
places_gps = places[['closed', 'gPlusPlaceId', 'gps']]
places_gps['gPlusPlaceId'] = places_gps['gPlusPlaceId'].astype(float)
users_gps = users[['gPlusUserId']]
users_gps['gPlusUserId'] = users_gps['gPlusUserId'].astype(float)
```

In [36]:
```
%%capture
sample_reviews_gps.gPlusPlaceId = sample_reviews_gps['gPlusPlaceId'].ast
ype(float)
sample_reviews_gps.gPlusUserId = sample_reviews_gps['gPlusUserId'].astyp
e(float)
```

In [29]:
```
merged = sample_reviews_gps.merge(places_gps, on='gPlusPlaceId')
merged.head()
```

Out[29]:

	gPlusPlaceId	gPlusUserId	rating	reviewTime	closed	gps
0	1.130389e+20	1.159144e+20	5.0	2014-03-20	False	[27.813431, -82.60796]
1	1.090884e+20	1.023256e+20	5.0	2013-03-18	False	[40.051802, -75.236426]
2	1.006022e+20	1.061090e+20	5.0	2011-09-29	False	[46.717622, 11.65158]
3	1.105433e+20	1.163070e+20	4.0	NaT	False	[23.427299, 72.654368]
4	1.014607e+20	1.008819e+20	5.0	2013-05-10	False	[21.252132, 81.649393]

In [30]:
```
from math import cos, sin, atan2, sqrt

def geolocate(geolocations):

    x=0
    y=0
    z=0
    for geo in geolocations:
        if geo is None:
            continue
        #Convert lat/lon (must be in radians) to Cartesian coordinates f
or each location.
        lat = geo[0] * math.pi/180
        lon = geo[1] * math.pi/180
        X = cos(lat) * cos(lon)
        Y = cos(lat) * sin(lon)
        Z = sin(lat)

        #Compute average x, y and z coordinates.
        x += X
        y += Y
        z += Z

    x /= len(geolocations)
    y /= len(geolocations)
    z /= len(geolocations)

    #Convert average x, y, z coordinate to latitude and longitude.
    Lon = atan2(y, x)
    hyp = sqrt(x * x + y * y)
    Lat = atan2(z, hyp)

    return (Lat *  180/math.pi, Lon *  180/math.pi)
```

In [31]:
```
list_of_gps = merged.groupby('gPlusUserId').gps.apply(list)
```

In [33]:
```
import math
locations = []
for i in range(len(list_of_gps)):
    if list_of_gps.iloc[i] is None:
        continue
    locations.append(geolocate(list_of_gps.iloc[i]))

locations
```

```
Out[33]: [(40.581469, -73.961767),
          (32.860754, -97.320848),
          (59.418126, 10.484576000000002),
          (36.29582899999999, 139.981826),
          (33.918259, -117.24491),
          (9.742561000000002, -63.16708400000002),
          (23.277928, -106.467908),
          (33.744288000000005, 73.068239),
          (40.815224, -73.95811599999999),
          (41.543575999999995, -96.136931),
          (48.208718, 16.369795000000003),
          (23.974806999999995, 121.611822),
          (45.457688, -73.567618),
          (10.198515999999998, -64.692918),
          (41.951976, -73.994132),
          (21.252131999999996, 81.649393),
          (26.062206, -80.17411799999999),
          (45.535176, -122.862242),
          (53.09542199999999, -0.7603900000000001),
          (8.310219999999997, -62.71572500000001),
          (26.471814000000002, 74.60818100000002),
          (33.004670000000004, -96.986088),
          (-3.0489310000000005, -60.015761000000005),
          (39.075762, -77.136983),
          (-3.0489310000000005, -60.015761000000005),
          (34.07813800000001, -83.918605),
          (35.659839, 139.699568),
          (56.15929899999999, 10.209891),
          (37.796345, -122.21405599999999),
          (29.77631, -95.751877),
          (52.575185, -0.24284599999999995),
          (40.74835099999999, -73.985143),
          (38.94777, -77.08099),
          (-20.432558, -54.594466),
          (25.054277000000003, 121.514633),
          (1.31219, 103.895915),
          (40.663925, -73.698498),
          (52.22829900000001, 20.967753999999996),
          (24.676048000000005, 121.769089),
          (45.726438, -122.65264199999999),
          (32.860754, -97.320848),
          (47.502029, -122.25098),
          (21.559276, 39.142304),
          (32.860754, -97.320848),
          (-8.25013, 111.37026999999999),
          (40.051801999999995, -75.236426),
          (49.879086, 14.757129000000004),
          (42.631098, 18.117668),
          (44.058973, -79.461552),
          (35.87334100000001, -78.62385),
          (53.762775, -0.29208199999999999),
          (51.577272, -0.12333999999999999),
          (35.624915, 139.719712),
          (9.943973000000002, -84.146303),
          (29.653288999999994, -95.009795),
          (51.01900799999999, 7.842272999999999),
          (53.582547, -113.45913),
```

```
(45.514622, -122.45354699999999),
(50.840212, 4.355341999999999),
(43.66693, -79.388095),
(24.99999999787196, -53.9999999986315),
(48.200061, 13.645716000000002),
(29.889588, -95.640896),
(29.629662999999997, -82.370113),
(40.425163999999995, -86.90811),
(29.629662999999997, -82.370113),
(20.913757, -156.322226),
(13.021778, 77.597558),
(-22.877173, -43.446314),
(45.514622, -122.45354699999999),
(40.024738, -75.221245),
(39.595999, -104.902224),
(40.729842, -74.003515),
(39.96253999999999, -75.607102),
(28.526235, -81.377123),
(41.928643000000015, -87.66844699999999),
(29.92572, -95.597082),
(45.535176, -122.862242),
(-37.888627, 145.057059),
(39.595999, -104.902224),
(38.149722, -79.070978),
(52.377036, -2.315574),
(51.523036, -0.717845),
(35.074745, -78.92597299999998),
(42.069674, -80.096553),
(20.68889, -101.35707199999999),
(60.60501800000001, 16.770125999999994),
(41.939534, -87.64465599999998),
(53.343062999999994, -6.270012),
(38.872882, -77.245892),
(52.28268500000001, 17.011398),
(53.058393, 14.283423),
(29.895741, -81.312805),
(51.75413499999999, -1.2537400000000003),
(38.903727, -77.019589),
(25.054277000000003, 121.514633),
(39.96253999999999, -75.607102),
(35.224978, 139.08838200000002),
(26.561170000000004, -82.00776800000001),
(33.81630700000001, -118.343058),
(53.375605, -2.214905),
(38.050151, -87.274153),
(41.87235, 12.576360999999999),
(41.63923900000001, -70.616299),
(25.054277000000003, 121.514633),
(35.607074000000004, 139.647438),
(51.060806, -114.158098),
(41.63581400000001, -70.953055),
(45.41454, -75.695751),
(35.812964, -78.618618),
(42.298349999999985, -71.448611),
(46.717622, 11.65158),
(20.68889, -101.35707199999999),
(44.146868000000005, 12.01434),
```

```
(39.034285000000004, -76.909304),
(49.870939, 8.648039),
(51.86732399999999, -2.1043679999999996),
(32.062453000000005, 45.24343699999999),
(50.49326899999999, 30.557709999999997),
(37.841221999999995, -122.20910099999998),
(51.50949899999999, -0.13576200000000002),
(59.603734, 15.210024),
(35.75089799999999, -5.832795999999999),
(20.467836999999996, 85.868856),
(-24.71783, -53.754114),
(24.71933400000001, 46.603584),
(38.898033000000005, -77.020917),
(40.74496, -73.98767599999998),
(51.50949899999999, -0.13576200000000002),
(52.22048299999999, -1.7620709999999997),
(45.61500999999999, -94.225669),
(40.729842, -74.003515),
(-2.9762560000000007, 104.742662),
(49.12363800000001, 8.598834999999998),
(54.71015, 20.528694),
(19.088684, 72.86459),
(54.167, -1.2457889999999998),
(51.45080000000001, -2.5861),
(47.17510999999998, 27.597456000000005),
(49.23999999999999, 5.94),
(41.06543400000001, 1.0585790000000002),
(19.372609, -99.155994),
(41.117309, -85.121534),
(39.129092, -76.549345),
(41.71172, -88.067319),
(33.81630700000001, -118.343058),
(17.44479, 78.376785),
(38.211728, -122.255909),
(40.024738, -75.221245),
(50.840212, 4.355341999999999),
(45.52648, -122.540741),
(-34.865778, 138.592478),
(35.624915, 139.719712),
(40.759569, -73.987139),
(12.195880999999998, 109.227042),
(32.851383999999996, -96.82401699999998),
(41.90176300000001, 12.483788000000002),
(40.36150400000001, -74.968345),
(42.03809099999999, -82.73583599999999),
(26.471814000000002, 74.60818100000002),
(35.224978, 139.08838200000002),
(39.13770099999999, -121.617162),
(24.233752000000003, 120.55896200000001),
(41.71172, -88.067319),
(51.45080000000001, -2.5861),
(52.037926000000006, 4.425891000000001),
(40.759569, -73.987139),
(43.656914, -79.435345),
(51.060806, -114.158098),
(38.898033000000005, -77.020917),
(32.860754, -97.320848),
```

```
(-23.779121999999997, -46.530770000000004),
(39.96253999999999, -75.607102),
(33.779138, -84.410614),
(40.687214, -73.990375),
(33.779138, -84.410614),
(57.44975300000001, 42.131538),
(47.17510999999998, 27.597456000000005),
(40.69935199999999, -73.939962),
(13.021778, 77.597558),
(52.36148099999998, 4.888531),
(-11.99372, -77.061142),
(41.712694, -72.58510400000002),
(-26.833512, -65.228259),
(39.96253999999999, -75.607102),
(35.268905, -75.539598),
(36.517095, -4.8777539999999995),
(39.827411999999995, -75.15157499999998),
(41.928643000000015, -87.66844699999999),
(51.55555699999999, -0.283149),
(36.30884, -86.682728),
(39.91700399999999, -82.993176),
(22.452679, 114.188403),
(27.782902999999997, -82.71329900000002),
(50.49326899999999, 30.557709999999997),
(36.06650799999999, -80.320007),
(45.516127999999995, -122.556546),
(18.527521, 73.85177499999999),
(27.334944000000004, -80.379045),
(21.547386, 39.14298600000001),
(37.69277, -97.442578),
(37.905974000000015, 127.054622),
(40.729842, -74.003515),
(40.463307, -3.670675),
(39.96253999999999, -75.607102),
(36.80798200000001, -119.84811700000002),
(29.779978000000003, -95.952012),
(29.92572, -95.597082),
(45.940495999999996, 9.110862999999998),
(51.220915999999995, 6.789948),
(50.07311399999999, 14.340184),
(58.970805, 5.732119),
(39.536907, -107.32602799999998),
(11.567582, 104.931099),
(22.315431, 73.162886),
(52.502138999999985, 13.420115),
(40.448643, -109.50847),
(43.22776, 44.762726),
(25.054277000000003, 121.514633),
(-15.121941000000001, 39.263124),
(52.575185, -0.24284599999999995),
(-34.58749400000001, -58.429806),
(50.733094, 7.096869),
(53.79844, -1.541406),
(33.950614, -84.551148),
(41.90523999999999, 12.475500999999996),
(42.310471, -71.927318),
(49.209626, 18.744665000000005),
```

```
(28.61216, -81.454273),
(44.36606, -79.650098),
(40.75322400000001, -73.978366),
(32.83123199999999, -117.131351),
(35.816723, -78.621614),
(-27.092610000000004, -52.61738099999999),
(30.833598000000002, 75.157558),
(43.92652499999999, -78.909483),
(41.882712000000005, -87.632876),
(59.95374699999999, 30.217254000000004),
(39.595999, -104.902224),
(45.095875, -93.441674),
(37.758789, -122.414823),
(44.73320899999999, -85.59188499999999),
(26.281059000000003, 73.04769299999998),
(40.691092999999995, -73.925074),
(38.597106, -121.45266699999998),
(44.73320899999999, -85.59188499999999),
(32.167993, 13.009485999999999),
(21.215174, 81.357902),
(52.502138999999985, 13.420115),
(44.465205, 26.099542999999993),
(-27.471143000000005, -58.823150000000005),
(22.375188999999995, 114.112972),
(22.65451, 120.293644),
(38.903727, -77.019589),
(45.014033, -93.321766),
(40.572876, -122.36219800000002),
(38.02294700000001, 12.547262000000003),
(33.744288000000005, 73.068239),
(36.238402, -121.813096),
(36.732498, -108.238964),
(29.701985000000004, -98.096009),
(29.702664, -98.123941),
(39.61643899999999, -86.11042999999998),
(32.103212000000006, 74.822629),
(30.018052, -95.437272),
(40.728618, -73.999826),
(59.305993, 18.029004),
(21.376684000000008, 74.241373),
(34.581427, -92.575302),
(44.72802699999998, 37.758385000000004),
(-34.580672, -58.414654),
(29.702664, -98.123941),
(34.246596, -118.58807800000002),
(41.120362, -101.715814),
(40.394269, -86.852994),
(1.31219, 103.895915),
(49.780637999999996, 6.331773000000001),
(38.898033000000005, -77.020917),
(39.184629, -78.165839),
(32.860754, -97.320848),
(51.01900799999999, 7.842272999999999),
(51.71318699999999, 6.5724409999999995),
(36.059993, 136.211135),
(13.021778, 77.597558),
(33.621265, 130.67052199999998),
```

```
(-23.925265999999993, -46.40852),
(40.641639, -88.782818),
(-2.9762560000000007, 104.742662),
(-37.800695999999995, 144.943057),
(50.49326899999999, 30.557709999999997),
(29.653288999999994, -95.009795),
(39.827411999999995, -75.15157499999998),
(40.653268, -75.54513700000001),
(33.950614, -84.551148),
(40.32837, -3.7592520000000005),
(20.402262, 85.80609900000002),
(42.631098, 18.117668),
(19.176716, 72.838107),
(26.281059000000003, 73.04769299999998),
(36.932625, -76.253179),
(25.054277000000003, 121.514633),
(52.037926000000006, 4.425891000000001),
(41.822511, -87.61656999999998),
(53.343062999999994, -6.270012),
(27.813430999999998, -82.60796),
(39.847421000000004, -86.167959),
(47.28673400000001, -2.391725),
(42.63333699999999, 141.597163),
(43.22776, 44.762726),
(35.19212600000001, -101.88115900000001),
(53.326708, -3.8312449999999996),
(26.281059000000003, 73.04769299999998),
(40.74496, -73.98767599999998),
(30.225620000000003, -97.77461),
(23.427298999999998, 72.654368),
(30.315605999999992, -97.871338),
(-7.969881, 112.625004),
(53.546181, 9.957089),
(38.448119, -78.86904099999998),
(41.822511, -87.61656999999998),
(13.006007, 80.258023),
(26.281059000000003, 73.04769299999998),
(41.853006, 12.496779000000002),
(19.171910999999998, 73.02235900000001),
(43.66536399999999, -79.409445),
(33.539796, -111.88324699999998),
(43.522996, -80.267583),
(61.18382100000001, -149.869489),
(32.707466, -117.159719),
(30.421812000000006, -97.945193),
(42.279602000000004, -83.745162),
(37.758789, -122.414823),
(30.018052, -95.437272),
(35.401841, -97.530368),
(24.776916999999997, 121.02355800000001),
(38.123743999999995, -122.259512),
(41.896891, -87.627336),
(40.143163, -8.817754000000003),
(26.062206, -80.17411799999999),
(49.24921299999998, -122.867114),
(41.613914, -93.76948899999998),
(32.860754, -97.320848),
```

```
 (38.448119, -78.86904099999998),
 (30.057238999999996, 31.368079999999996),
 (36.30884, -86.682728),
 (39.163852000000006, -86.475694),
 (25.054277000000003, 121.514633),
 (-16.819201, 145.63862),
 (29.629662999999997, -82.370113),
 (39.692899, -84.136173),
 (-2.9762560000000007, 104.742662),
 (51.341199, 12.375349)]
```

In [ ]:

```

```

Reviews

- From the k-NN algorithm, we want to balance distance with quality of the business.
  - We can do this in the future by looking at the travel distances of the user's past reviews to determine weighting for how far to recommend
- Future ideas:
  - Recommend businesses that have similar number of reviews as the user has been reviewing (this is to avoid suggesting only super popular places to someone who enjoys finding smaller known places)
  - Increase the efficiency of cleaning and smoothing the categories

What we will do if we have more time

- we are creating a modeling pipeline to predict the top businesses that users will go to with knn classifier

In [37]:

```
from sklearn.model_selection import train_test_split

from sklearn.pipeline import Pipeline
from sklearn.compose import ColumnTransformer
from sklearn.base import BaseEstimator, ClassifierMixin

from sklearn.preprocessing import FunctionTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.impute import SimpleImputer

from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression

from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
```

In [39]:
```
num_feat = ['rating']
num_transformer = Pipeline(steps=[
    ('scaler', StandardScaler())
])

# Categorical columns and associated transformers
cat_feat = ['categories', 'reviewYear']
cat_transformer = Pipeline(steps=[
    ('onehot', OneHotEncoder())
])

# preprocessing pipeline (put them together)
preproc = ColumnTransformer(transformers=[('num', num_transformer, num_feat), ('cat', cat_transformer, cat_feat)])

pl = Pipeline(steps=[('preprocessor', preproc), ('regressor', LinearRegression())])
```

Conclusion:

Validation

- Using first n years to run the test data, and last few years comparing the recommended restaurant and people's choice in restaurants for testing accuracy

Future ideas

- Recommend businesses that have similar number of reviews as the user has been reviewing (this is to avoid suggesting only super popular places to someone who enjoys finding smaller known places)
- Increase the efficiency of cleaning and smoothing the column "categories"
- Using Omnisci Geograph to display the concentration of the businesses in relation to the location of the user to allow closer proximity suggestions
- Check the frequency of the user's reviews in individual categories and respective reviews to approximate the people's preference across categories and adjust the respective weighting of categories

In [ ]:
```

```