RMDS Restaurant Dashboard Project

Strong Woman Team (Finka Marisa Geanandа Sufie and Olivia Destri Riana)

```
# Necessary imports for general purpose
import pandas as pd
import numpy as np
```

```
# Pandas function that reads an excel sheet as a dataframe
detail_df = pd.read_excel('competition_data/Q3_competition_detail_dataset.xlsx')
```

```
# pandas option to display all the columns of a dataframe for the first 5 rows
pd.set_option('display.max_columns', None)
detail_df.head()
```

	id	name	is_claimed	is_closed	phone	review
0	nzgC5hhlnSq2DYbJbtH5MQ	Foxy's Landing & Restaurant	True	False	1.661949e+10	
1	i-2aG9_PQBEy7LrsRv0Ivg	Mosman's Steakhouse	True	False	1.661949e+10	
2	DJoeogRsOW5s9MzgveHQ2A	El Tamarindo	True	False	1.661723e+10	
3	hwWfv3sSxV3a47UAdSVT5w	Subway	True	False	1.661730e+10	
4	TxU0fwF2N2nVhCpzokc1Pg	Little Caesars	True	False	1.661946e+10	

```
# dataframe info method that displays some useful information for data cleaning (null coun
detail_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10710 entries, 0 to 10709
Data columns (total 21 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   id              10710 non-null  object
 1   name            10710 non-null  object
 2   is_claimed      10710 non-null  bool
 3   is_closed       10710 non-null  bool
 4   phone           9993 non-null   float64
 5   review_count    10710 non-null  int64
 6   categories01    10710 non-null  object
 7   categories02    7434 non-null   object
 8   categories03    4034 non-null   object
 9   rating          10710 non-null  float64
 10  price           8374 non-null   object
 11  transactions    10710 non-null  object
 12  zip_code        10703 non-null  float64
 13  city            10710 non-null  object
 14  address         10710 non-null  object
 15  restaurant_url  10710 non-null  object
 16  image_url       10350 non-null  object
 17  latitude        10710 non-null  float64
 18  longitude       10710 non-null  float64
 19  photos          10710 non-null  object
 20  cross_streets   0 non-null      float64
dtypes: bool(2), float64(6), int64(1), object(12)
memory usage: 1.6+ MB
```

```
# dataframe drop method to remove the cross_streets column from our dataframe
detail_df = detail_df.drop('cross_streets', axis=1)
```

```
# Check for duplicates in each column
for col in detail_df.columns:
    boolean = not detail_df[col].is_unique
    print(col + ':')
    print(boolean)
    print()

# TODO: MAYBE pick out specific columns to check dupes for cuz this list is kinda long.
```

```
id:
False

name:
True

is_claimed:
True

is_closed:
True
```

```
phone:
True

review_count:
True

categories01:
True

categories02:
True

categories03:
True

rating:
True

price:
True

transactions:
True

zip_code:
True

city:
True

address:
True

restaurant_url:
False

image_url:
True

latitude:
True

longitude:
True

photos:
```

```
# this simple query shows that Subway has at least 2 different entries in this dataframe
detail_df[detail_df['name'] == 'Subway'].head(2)
```

	id	name	is_claimed	is_closed	phone	review_co
3	hwWfv3sSxV3a47UAdSVT5w	Subway	True	False	1.661730e+10	

```
# Pandas function to query rows with duplicates in the address column
pd.concat(g for _, g in detail_df.groupby("address") if len(g) > 1).head()
```

	id	name	is_claimed	is_closed	phone	rev
8811	O9w_-6yJaOzXqCJf7yt4mA	Rock & Brews - LAX Southwest Terminal 1	True	False	1.424702e+10	
8825	MdOKg7Nkw4Tusys0cnSfhQ	Panda Express	False	False	1.424751e+10	
8885	CJC1pycIOA-ocBWawAOXow	Blue Window	False	False	NaN	
8813	IeEq0IQZ-ie39KeP91_07g	California Pizza Kitchen	False	False	1.866820e+10	
8829	dP6HQtMwyBhKH00nmkBQlw	Cassell's Hamburgers	True	False	NaN	

```
# Checks for duplicate longitude and latitude values using groupby
gps_indexed = detail_df.groupby(['latitude', 'longitude']).first()
gps_indexed[gps_indexed.index.duplicated()]
```

		id	name	is_claimed	is_closed	phone	review_count	categorie
latitude	**longitude**							

```
# Display unique values of select columns (commented out for brevity)

#for col in ['is_claimed', 'is_closed', 'review_count', 'categories01', 'categories02', 'c
#    print(col + ':')
#    print(detail_df[col].unique())
#    print()
```

```
print(detail_df['categories01'].unique()[:10])
```

```
     ['breakfast_brunch' 'bars' 'salvadoran' 'sandwiches' 'pizza' 'tacos'
      'chinese' 'burgers' 'foodtrucks' 'hotdogs']
```

```
#
def clean_categories01(category):
    if category in ['bars', 'beer_and_wine', 'wine_bars', 'cocktailbars', 'beerbar', 'brew
                    'speakeasies', 'gaybars', 'beergardens', 'distilleries', 'winetastingr
        return 'alchoholic beverages'
    else:
        return category
    #if category in ['']
```

```
detail_df['categories01'] = detail_df['categories01'].apply(clean_categories01)
```

```
# Aggregates the different categories into a new features of type: list
detail_df['categories'] = detail_df[['categories01', 'categories02', 'categories03']].valu
```

```
from ast import literal_eval
```

```
# Convert transactions string value to list
detail_df['transactions'] = detail_df['transactions'].apply(literal_eval)
```

```
detail_df['transactions']
```

```
     0                        []
     1                        []
     2                        []
     3        [delivery, pickup]
     4                        []
                     ...
     10705    [delivery, pickup]
     10706                    []
     10707            [delivery]
     10708    [delivery, pickup]
     10709                    []
     Name: transactions, Length: 10710, dtype: object
```

```
# Set transactions with empty list (assumed to be physical) to a new df and transactions w
physical = detail_df[detail_df.transactions.str.len().eq(0)]
non_physical = detail_df[~detail_df.transactions.str.len().eq(0)]
```

```
# Show top 10 most reviewed categories for physical restaurants
# Show top 10 most reviewed categories for physical restaurants
physical[['categories01', 'review_count']].groupby('categories01').sum().sort_values('revi
```

categories01	review_count
korean	19966
mexican	15531
steak	14292
newamerican	13917
seafood	12715
burgers	11492
alchoholic beverages	11183
bakeries	10685
pizza	10590
foodtrucks	9205

```
# Show top 10 most reviewed categories for non-physical restaurants
non_physical[['categories01', 'review_count']].groupby('categories01').sum().sort_values('
```

categories01	review_count
mexican	201524
newamerican	163360
sushi	140113
korean	139912
japanese	120714
pizza	115398
italian	112978
burgers	95279
chinese	94883
ramen	81655

```
physical_review_counts = physical[['categories01', 'review_count']].groupby('categories01'
physical_review_props = physical_review_counts / physical_review_counts['review_count'].su
physical_review_props = physical_review_props.rename(columns={'review_count': 'review_prop
```

```
non_physical_review_counts = non_physical[['categories01', 'review_count']].groupby('categ
non_physical_review_props = non_physical_review_counts / non_physical_review_counts['revie
non_physical_review_props = non_physical_review_props.rename(columns={'review_count': 'rev
```

```
review_differences = physical_review_props.merge(non_physical_review_props, how='outer', l
```

```
review_differences = review_differences.fillna(0)
review_differences['difference'] = abs(review_differences['review_prop_x'] - review_differ
review_differences.sort_values('difference', ascending=False)[:10]
```

categories01	review_prop_x	review_prop_y	difference
steak	0.055477	0.012206	0.043271
sushi	0.012309	0.052545	0.040237
foodtrucks	0.035731	0.002245	0.033487
bakeries	0.041476	0.013137	0.028339
ramen	0.004844	0.030622	0.025778
korean	0.077502	0.052470	0.025032
alchoholic beverages	0.043409	0.020700	0.022709
chicken_wings	0.030386	0.008718	0.021668
seafood	0.049356	0.028229	0.021126
buffets	0.018275	0.000442	0.017833

```
review_df = pd.read_excel('competition_data/Q3_competition_review_dataset.xlsx')
```

```
# https://www.iflexion.com/blog/sentiment-analysis-python
review_df.head()
```

	id	review_id	review_text	review_rating
0	cal0Wpupxj9c_AV7WzDXsw	AyueC5Vq_5lUKJFqSzXWWw	Slightly turned off by the hostess. She wasn't...	3.0
1	cal0Wpupxj9c_AV7WzDXsw	yaH4AmHUz9b3Ywv4VtvU5g	Wish I would have known about no brunch at the...	3.0

```
detail_review = detail_df.merge(review_df, how='outer', left_on='id', right_on='id')
```

```
detail_review = detail_review[detail_review['name'].notna()]
```

```
import nltk
```

```
# nltk.download()
# Uncomment to download nltk packages


from nltk.sentiment import SentimentIntensityAnalyzer


sia = SentimentIntensityAnalyzer()


def analyze_sentiment(review):

    if type(review) != str:
        return review

    score = sia.polarity_scores(review)['compound']
    if score >= 0.05:
        return 'positive'
    elif score >= -0.05:
        return 'neutral'
    else:
        return 'negative'


detail_review['sentiment'] = detail_review['review_text'].apply(analyze_sentiment)


detail_review = detail_review.drop(detail_review[detail_review['review_text'] == 0].index)


detail_review['review_rating'].value_counts()
```

```
5.0    14549
1.0     6165
4.0     5017
3.0     2747
2.0     2200
Name: review_rating, dtype: int64
```

```
detail_review = detail_review[detail_review['sentiment'].notna()]
```

What type of restaurants would customers prefer? (pick-up/delivery, dine-in)

[ ] ↳ *8 cells hidden*