Final project

TripAdvisor European Restaurants

DaQuest Group

Rasha Alharthi
Samar Alosaimi
Jawhara Almulhim
Fatima Almulhim
Maha Alharbi
Areej Alhuthali

AGENDA:

- Introduction
- Data preprocessing
- Dashboard
- EDA
- Pig Latin questions
- PySpark Machine Learning
- Kafka implementation
- Conclusion and future work

Introduction

- **Business Problem:**
  - **Predict the average rating**

- **Saudi's 2030 Vision and TripAdvisor:**
  - **Quality of Life Program**

Presented By : Rasha Alharthi

Data Review

- This is the TripAdvisor dataset of 1,083,397 restaurants from the main European countries.
- The data has been scraped in early May 2021 from the TripAdvisor publicly available website.
- The data contains 1,083,397 rows and 42 columns.

Presented By : Rasha Alharthi

Data Review

restaurant_link (object)	unique TripAdvisor restaurant link
restaurant_name (object)	name of the restaurant on TripAdvisor
original_location (object)	original location displayed on TripAdvisor
country (object)	country name retrieved from original_location
region (object)	region name retrieved from original_location
province (object)	province name retrieved from original_location
city (object)	city name retrieved from original_location
address (object)	address displayed on TripAdvisor
latitude (float64)	latitude coordinate
longitude (float64)	longitude coordinate
claimed (object)	restaurant business claimed on TripAdvisor
awards (object)	award names
popularity_detailed (object)	popularity detailed ranking
popularity_generic (object)	popularity generic ranking

Presented By : Rasha Alharthi

Data Review

top_tags (object)	tag names
price_level (object)	level of prices in current currency
price_range (object)	range of prices in current currency
meals (object)	types of meal
cuisines (object)	types of cuisine
special_diets (object)	types of special diets
features (object)	restaurant features
vegetarian_friendly (object)	is the restaurant vegetarian friendly?
vegan_options (object)	does the restaurant offer vegan options?
gluten_free (object)	does the restaurant gluten-free options?
original_open_hours (object)	original open hours on TripAdvisor
open_days_per_week (float64)	number of open days per week retrieved from original_open_hours.
open_hours_per_week (float64)	number of open hours per week retrieved from original_open_hours.
working_shifts_per_week (float64)	number of working shifts per week retrieved from original_open_hours.

Data Review

avg_rating (float64)	average restaurant rating
total_reviews_count (float64)	total reviews count
default_language (object)	default language displayed while scraping
reviews_count_in_default_language (float64)	total reviews count in default language.
excellent (float64)	excellent reviews count in default language
very_good (float64)	very good reviews count in default language
average (float64)	average reviews count in default language
poor (float64)	poor reviews count in default language
terrible (float64)	terrible reviews count in default language
food (float64)	food rating
service (float64)	service rating
value (float64)	value rating
atmosphere (float64)	atmosphere rating
keywords (object)	popular keywords

Presented By : Rasha Alharthi

DATA PREPROCESSING

Data Preprocessing

```
# check null values
european_restaurants_sub.isnull().sum()
```

restaurant_link	0
restaurant_name	0
original_location	0
country	0
region	34132
province	236188
city	286925
address	0
latitude	4973
longitude	4973
claimed	1337
awards	531187
popularity_detailed	26795
popularity_generic	28762
top_tags	0
price_level	451
meals	253077
cuisines	53600
special_diets	454768
features	506619
features	506619
vegetarian_friendly	0
vegan_options	0
gluten_free	0
original_open_hours	290110
open_days_per_week	290110
open_hours_per_week	290110
working_shifts_per_week	290110
avg_rating	27760
total_reviews_count	14628
default_language	26590
reviews_count_in_default_language	26590
excellent	26590
very_good	26590
average	26590
poor	26590
terrible	26590
food	208433
service	204330
value	205672
atmosphere	533779
keywords	684538
price_range	0

Data Cleaning

- Drop columns with a large number of null values

```
#These columns contain a large number of null values and are unnecessary columns

european_restaurants_sub.drop(['price_level', 'special_diets', 'restaurant_link', 'original_location', 'original_open_hours',
                               'keywords', 'features', 'awards', 'meals', 'province', 'address', 'top_tags'], axis=1, inplace= True)
```

- Handling null values in other columns

```
#sorte data by country
european_restaurants_sub = european_restaurants_sub.sort_values(by=['country'])
```

```
#fill null city and region value using the previous row value
european_restaurants_sub['city'] = european_restaurants_sub['city'].ffill()
european_restaurants_sub['region'] = european_restaurants_sub['region'].ffill()
```

Presented By : Rasha Alharthi

Data Cleaning

- **Handling null values in other columns cont.**

```
#fill null values in numeric columns using mean

# 1- Rating columns
european_restaurants_sub['average'].fillna(round(european_restaurants_sub['average'].mean(),1), inplace=True)
european_restaurants_sub['very_good'].fillna(round(european_restaurants_sub['very_good'].mean(),1), inplace=True)
european_restaurants_sub['excellent'].fillna(round(european_restaurants_sub['excellent'].mean(),1), inplace=True)
european_restaurants_sub['value'].fillna(round(european_restaurants_sub['value'].mean(),1), inplace=True)
european_restaurants_sub['atmosphere'].fillna(round(european_restaurants_sub['atmosphere'].mean(),1), inplace=True)
european_restaurants_sub['service'].fillna(round(european_restaurants_sub['service'].mean(),1), inplace=True)
european_restaurants_sub['food'].fillna(round(european_restaurants_sub['food'].mean(),1), inplace=True)
european_restaurants_sub['terrible'].fillna(round(european_restaurants_sub['terrible'].mean(),1), inplace=True)
european_restaurants_sub['poor'].fillna(round(european_restaurants_sub['poor'].mean(),1), inplace=True)

# 2- count columns
european_restaurants_sub['total_reviews_count'].fillna(round(european_restaurants_sub['total_reviews_count'].mean(),1), inplace=True)
european_restaurants_sub['reviews_count_in_default_language'].fillna(round(european_restaurants_sub['reviews_count_in_default_language'].mean(),1), inplace=True)

# 3- opening informations
european_restaurants_sub['open_days_per_week'].fillna(european_restaurants_sub['reviews_count_in_default_language'].median(), inplace=True)
european_restaurants_sub['open_hours_per_week'].fillna(european_restaurants_sub['reviews_count_in_default_language'].median(), inplace=True)
european_restaurants_sub['working_shifts_per_week'].fillna(european_restaurants_sub['reviews_count_in_default_language'].median(), inplace=True)
```

- drop null values in target column
- Replace the remaining null with an unknown
- clean text in popularity_detailed and restaurant_name using the patterns to replace the values.

Presented By : Rasha Alharthi

Data Preprocessing

- Map each Y to 1 and N to 0, in the gluten-free, vegan_options, and vegetarian_friendly columns.

```
# Drop price_range column, we will replace this column by mid or cheap categories from top_tags column
european_restaurants.drop(['price_range'], axis=1, inplace= True)
```

```
# here we will split the values in top_tags column to separate columns and save them in top_tags variable
top_tags = european_restaurants['top_tags'].str.split(',', expand=True)
```

```
#creat a price_range column take the first element of top_tags
european_restaurants['price_range'] = top_tags[0]
```

```
# selecting rows based on mid or cheap range
european_restaurants_sub = european_restaurants[european_restaurants['price_range'].isin(['Mid-range', 'Cheap Eats'])]
```

Presented By : Rasha Alharthi

Data cleaning

- Check Null values after cleaning:

restaurant_name	0
country	0
region	0
city	0
latitude	0
longitude	0
claimed	0
popularity_detailed	0
popularity_generic	0
cuisines	0
vegetarian_friendly	0
vegan_options	0
gluten_free	0
open_days_per_week	0
open_hours_per_week	0
working_shifts_per_week	0

avg_rating	0
total_reviews_count	0
default_language	0
reviews_count_in_default_language	0
excellent	0
very_good	0
average	0
poor	0
terrible	0
food	0
service	0
value	0
atmosphere	0
price_range	0

The Data Shape is: (749544, 30)

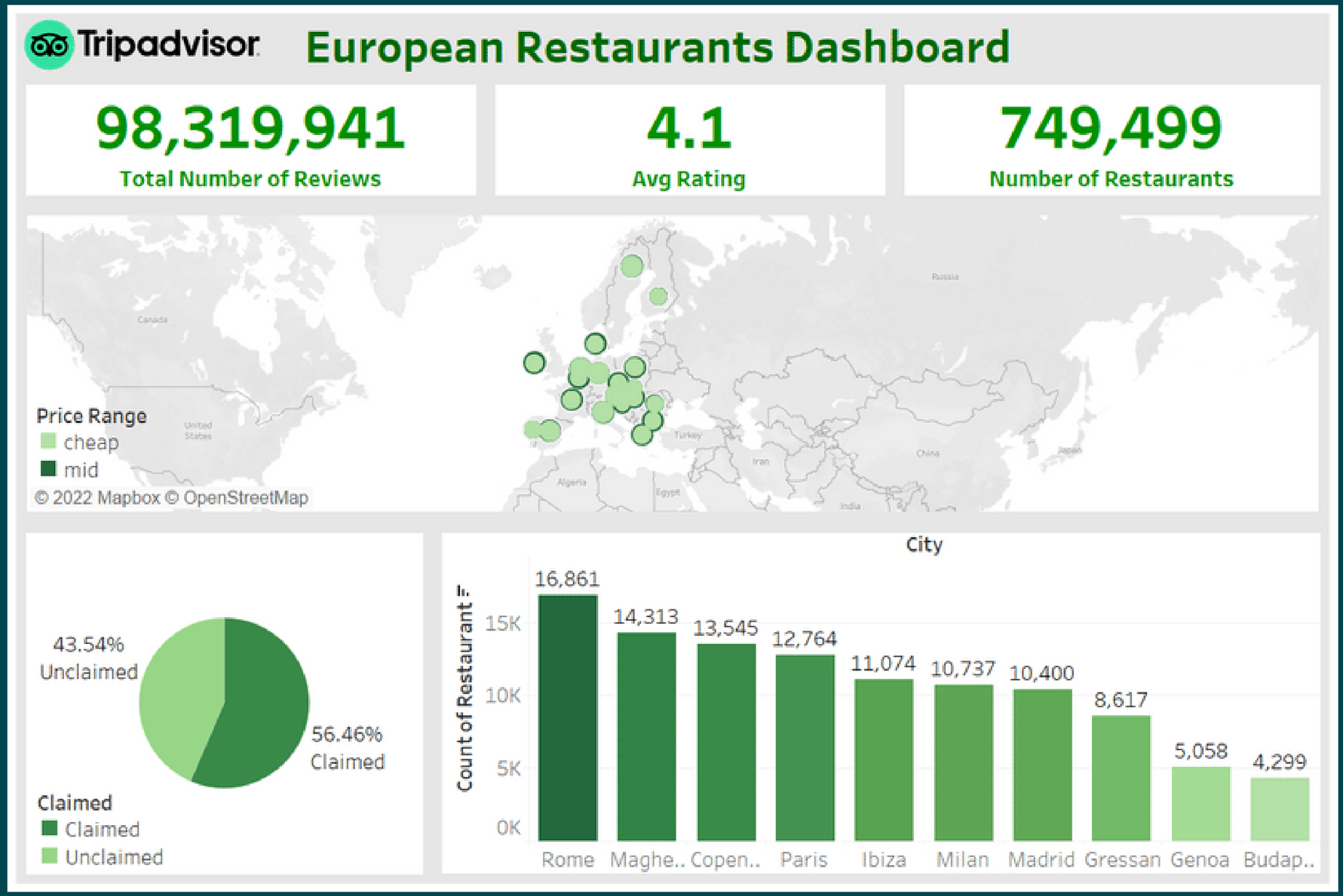
- save the cleaning to new csv file

Presented By : Rasha Alharthi

DASHBOARD

Dashboard

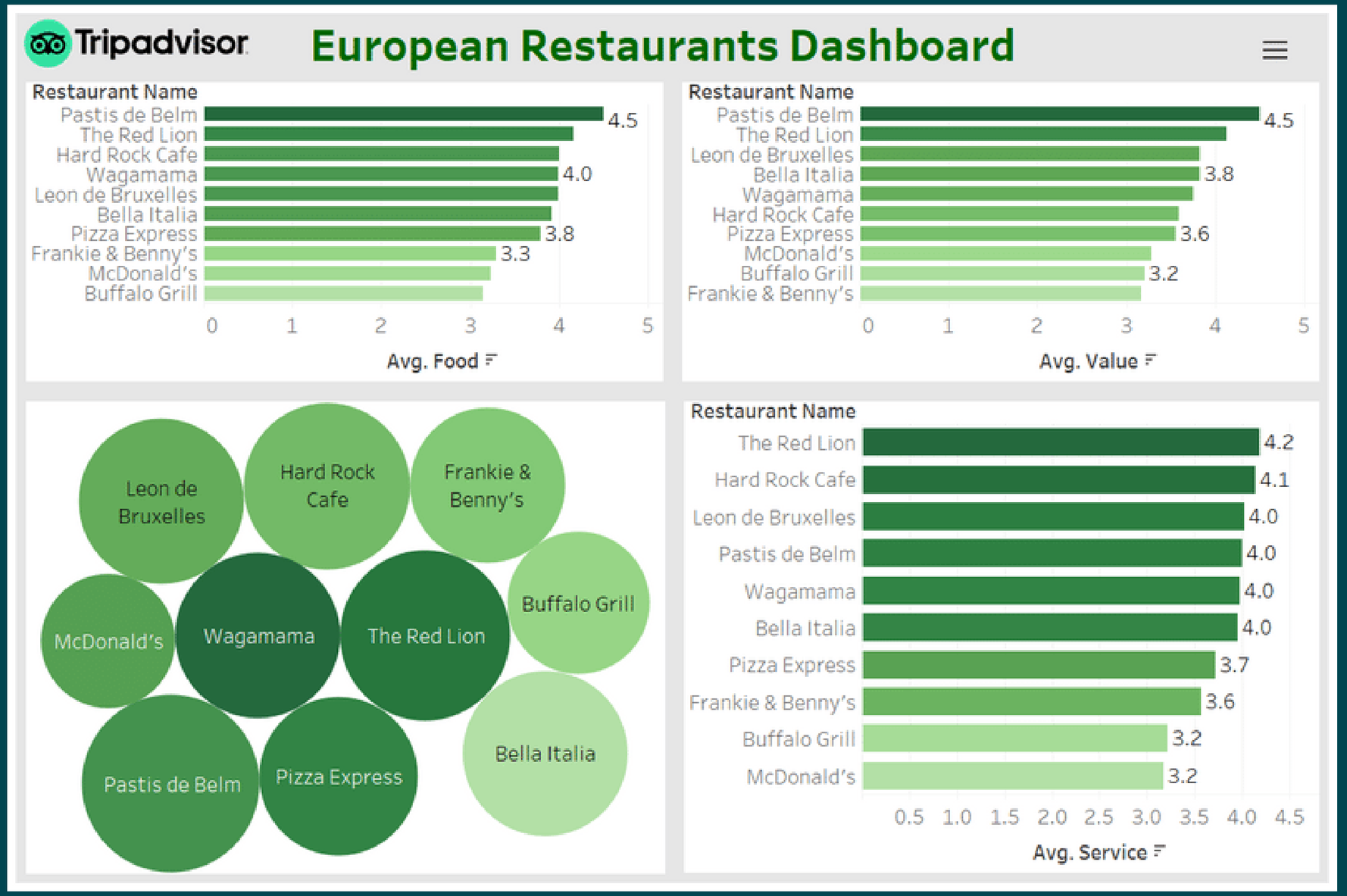
TripAdvisor European Restaurants Overview Dashboard



Presented By : Samar Alosaoimi

Dashboard

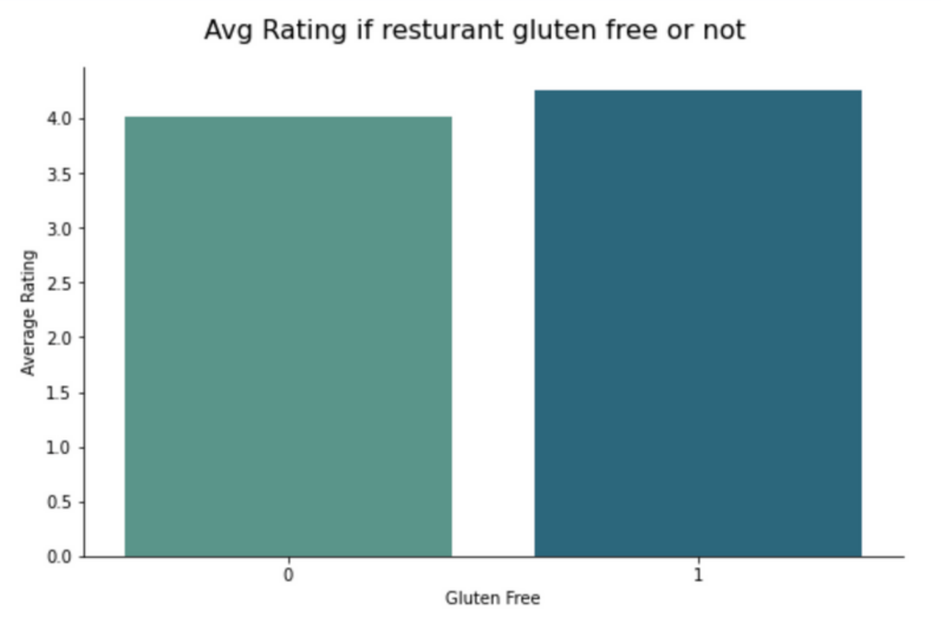
TripAdvisor European Restaurants Ratings Overview Dashboard

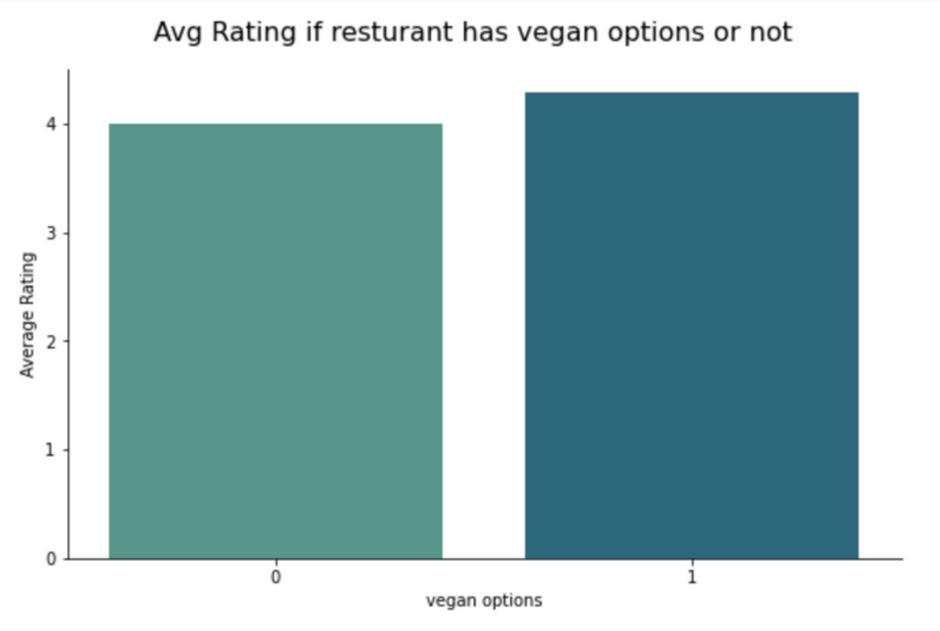


Presented By : Samar Alosaoimi

EXPLORATORY DATA ANALYSIS (EDA)

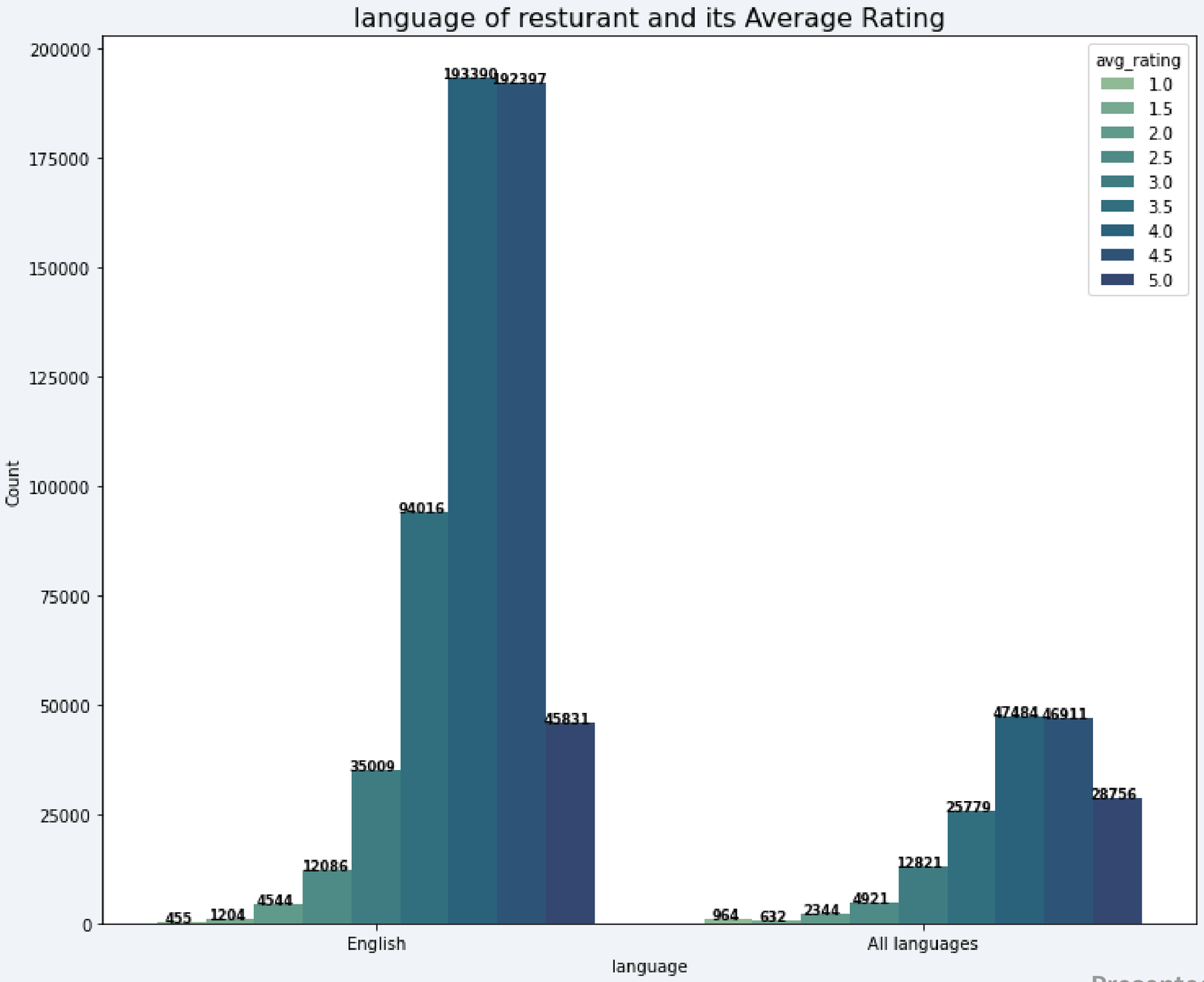
1st and 2nd PLOT



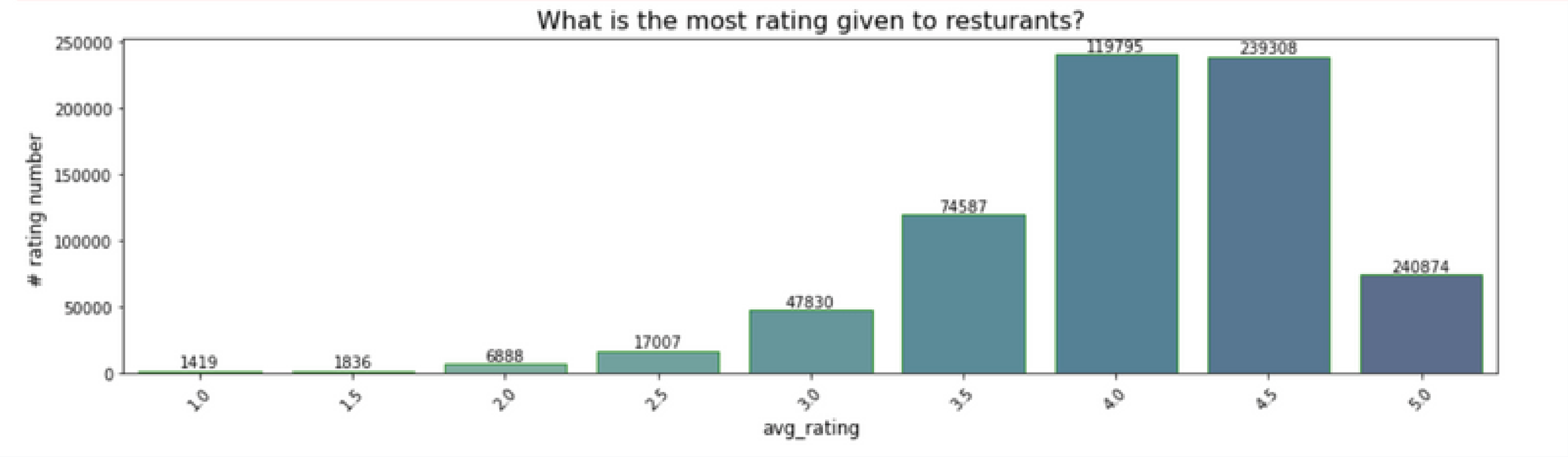


Presented By : Samar Alosaoimi

3rd PLOT


language of resturant and its Average Rating
avg_rating
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Count
200000
175000
150000
125000
100000
75000
50000
25000
0
455
1204
4544
12086
35009
94016
193390
192397
45831
964
632
2344
4921
12821
25779
47484
46911
28756
English
All languages
language

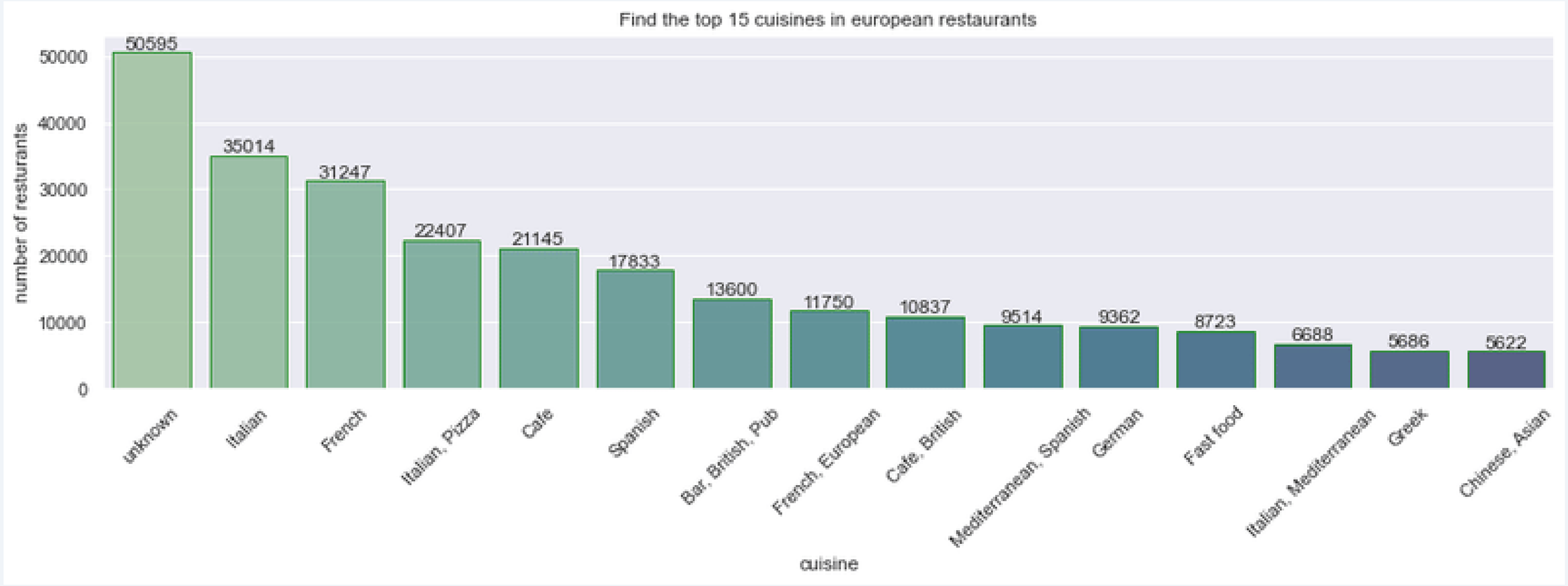
4th PLOT


What is the most rating given to resturants?
rating number
avg_rating
250000
200000
150000
100000
50000
0
1419
1836
6888
17007
47830
74587
119795
239308
240874
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0

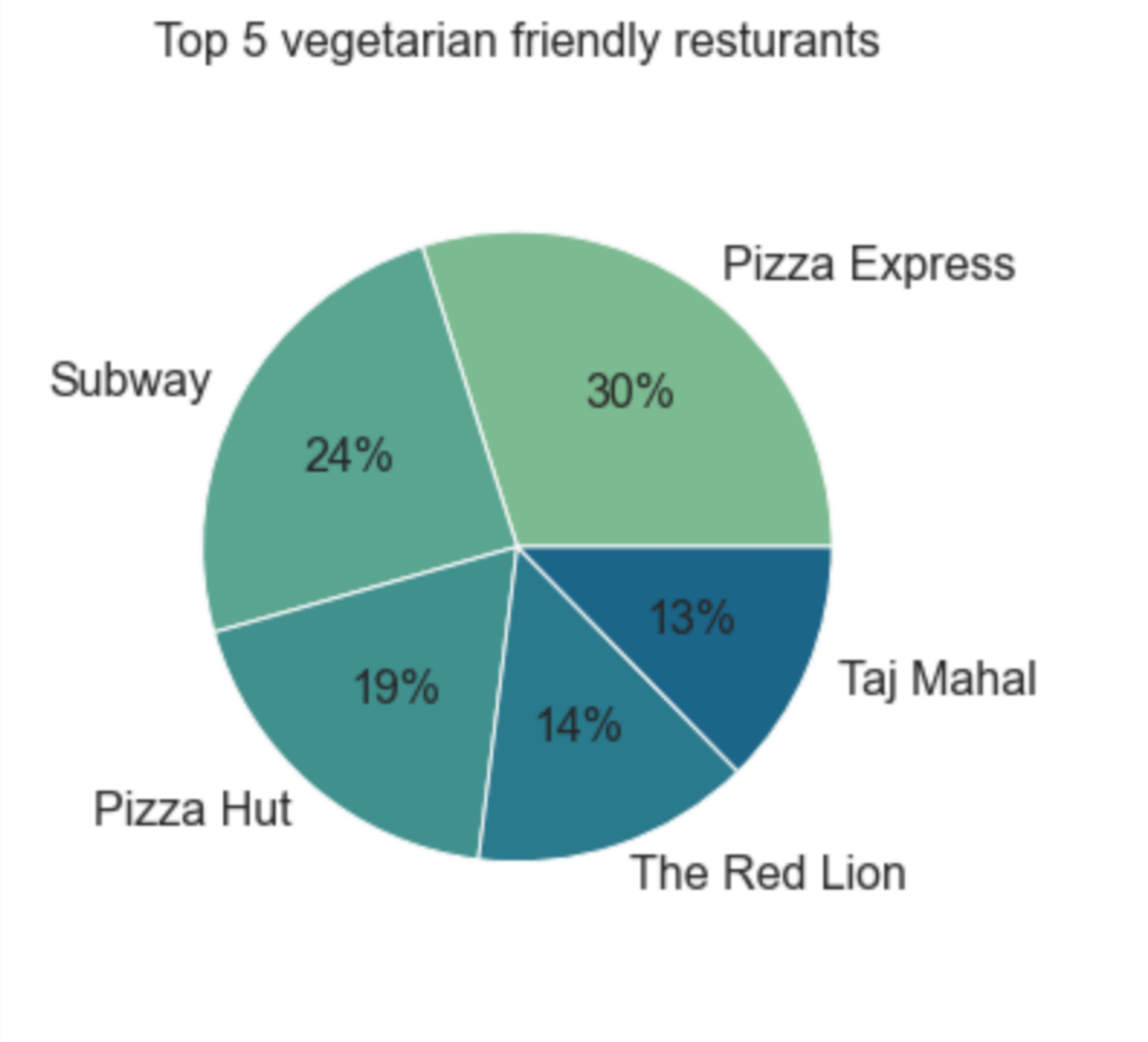
5th PLOT


price range for european resturants
500000
400000
300000
200000
100000
0
count
mid
cheap
price range

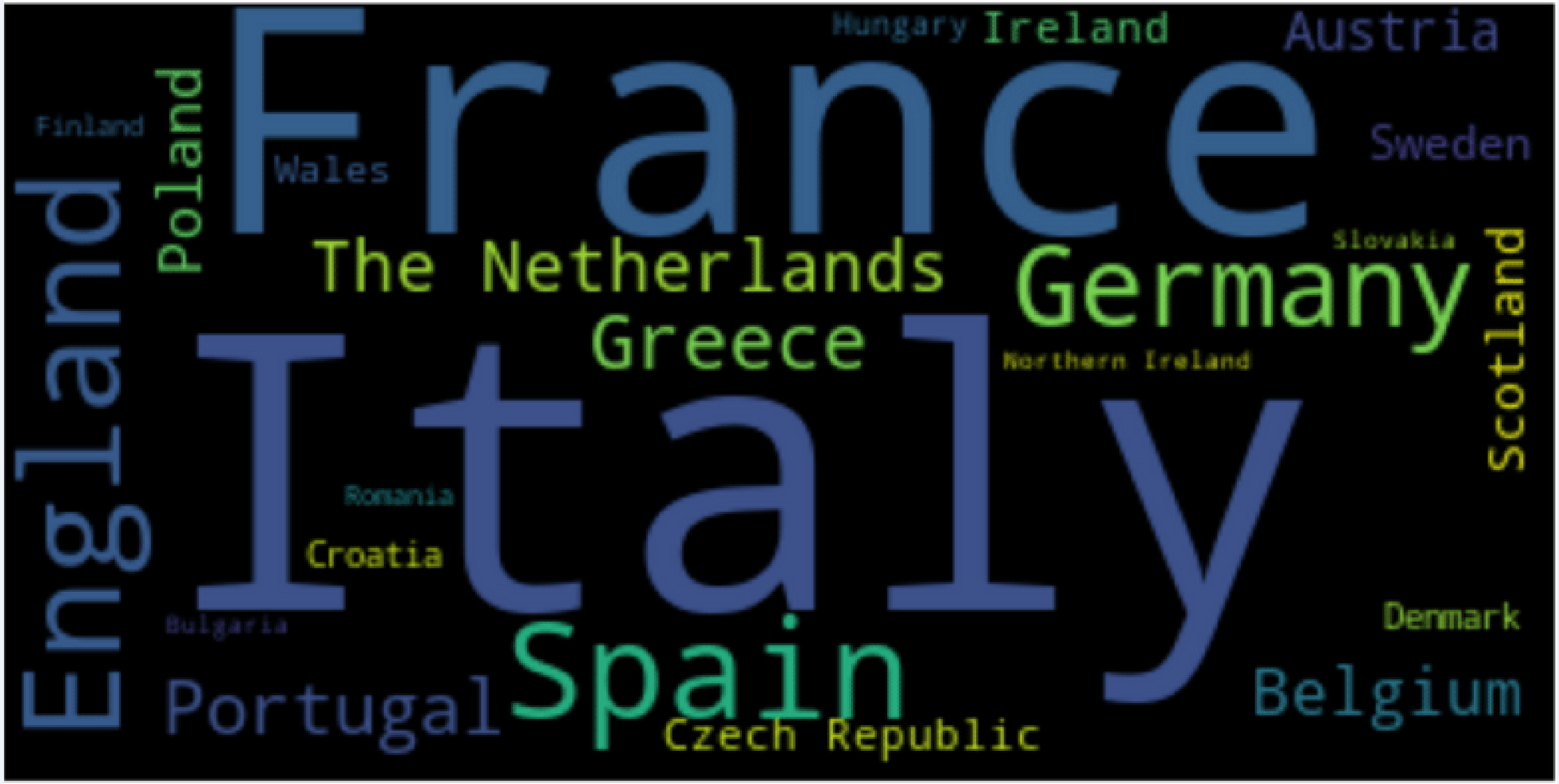
7th PLOT


Find the top 15 cuisines in european restaurants
number of resturants
0
10000
20000
30000
40000
50000
50595
35014
31247
22407
21145
17833
13600
11750
10837
9514
9362
8723
6688
5686
5622
unknown
Italian
French
Italian, Pizza
Cafe
Spanish
Bar, British, Pub
French, European
Cafe, British
Mediterranean, Spanish
German
Fast food
Italian, Mediterranean
Greek
Chinese, Asian
cuisine

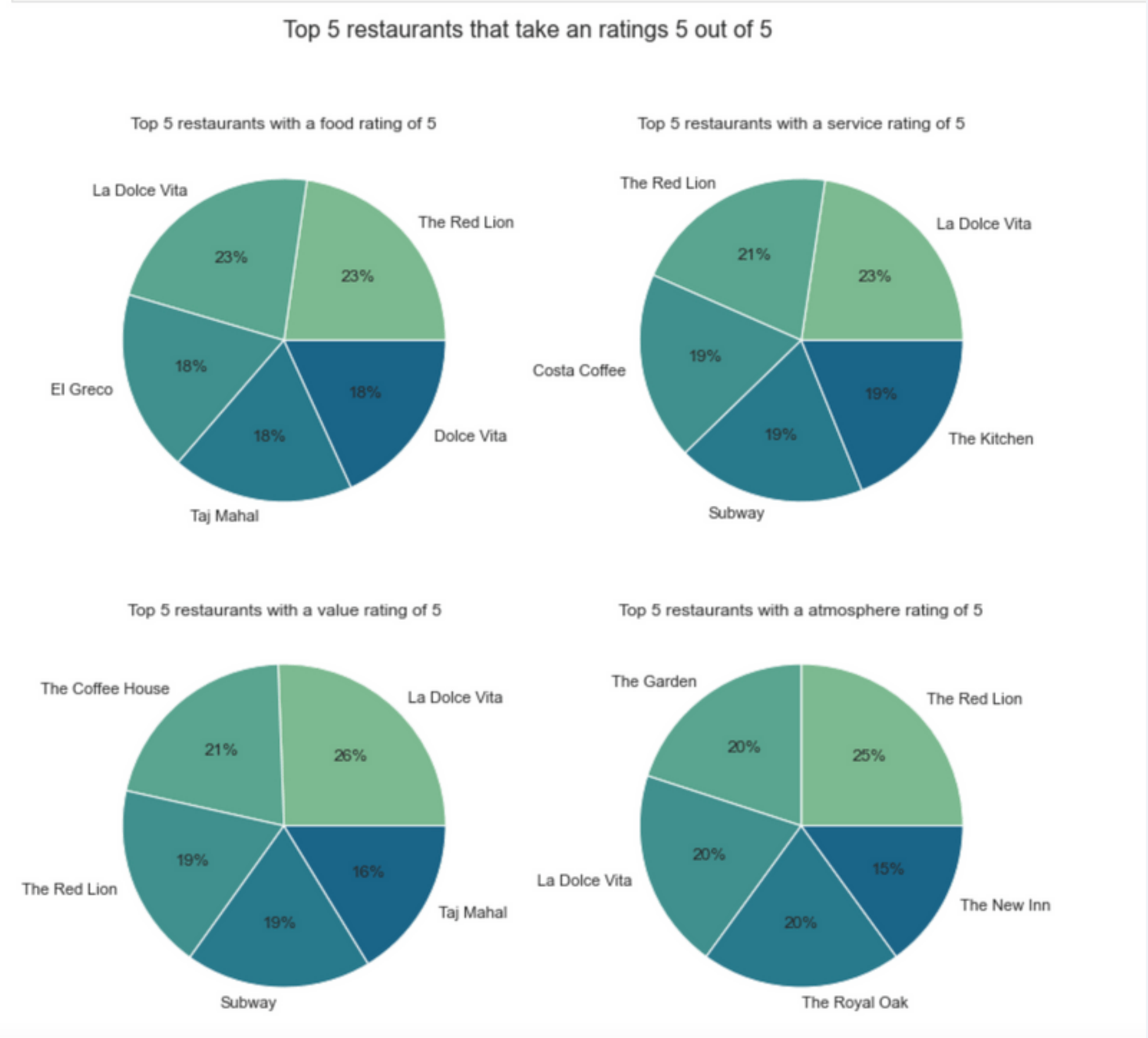
8th PLOT


Top 5 vegetarian friendly resturants
Pizza Express
30%
Subway
24%
13%
Taj Mahal
19%
14%
Pizza Hut
The Red Lion

9th PLOT


Hungary
Ireland
Austria
France
Finland
Poland
Wales
Sweden
Slovakia
The Netherlands
Germany
Greece
Northern Ireland
Scotland
England
Italy
Romania
Croatia
Bulgaria
Denmark
Portugal
Spain
Belgium
Czech Republic

10th PLOT

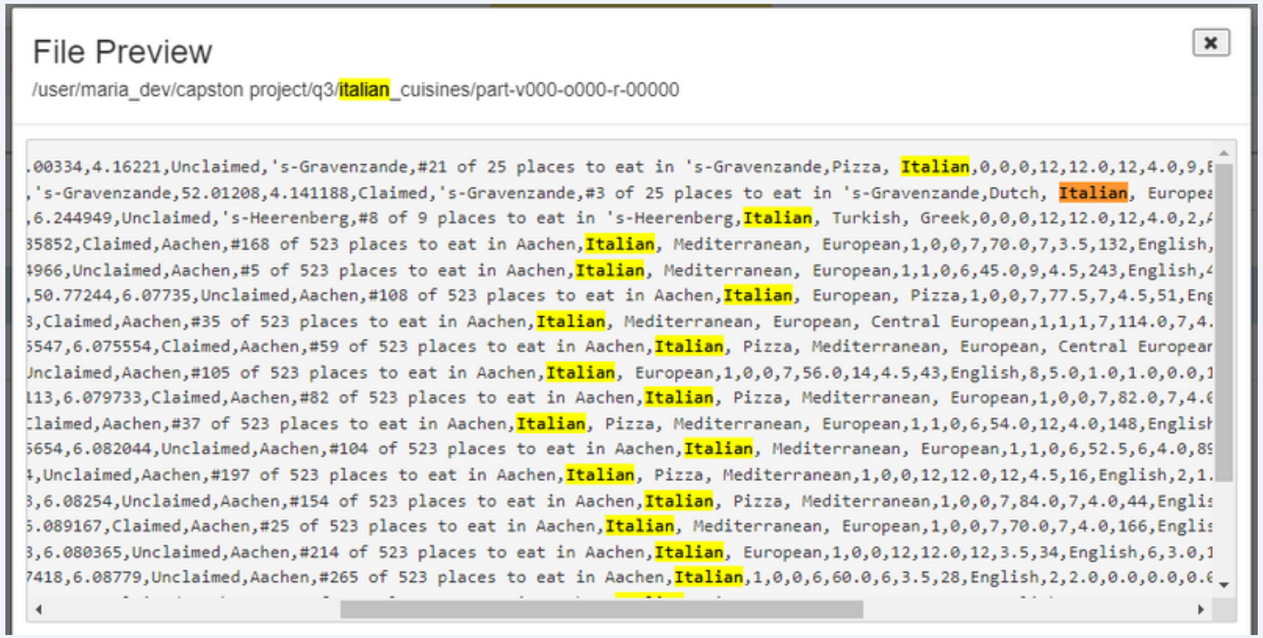


Presented By : Samar Alosaoimi

PIG LATIN QUESTIONS

1

Split the dataset based on cuisines (french or Italian) and store each one in a separate dataset



Presented By : Jawhara Almulhim

2

List the France restaurants that are vegetarian and vegan and have food ratings greater than 4.5

Results

Download

```
(Le Pizzaiolo)
(La Marina)
(Mon Coin d'Asie)
(Bistro d'Ailhon)
(L'Atelier)
(Izumi)
(Manlio's)
(Tita)
(Chez Ama)
(Bunny's Bubble Tea)
(Le Pt' Tha)
(Les petits plats de Trinidad)
(Aux Petits Oignons)
(Kabbaz)
(La Cabane en Ville)
(L'Atelier des Saisons)
```

3

List the top 10 cities and order them by total reviews and save the results

Results

Download

```
(Lisbon,52404)
(Madrid,33731)
(Rome,31144)
(Munich,30142)
(Milan,29273)
(Rome,24671)
(Madrid,22364)
(Rome,19856)
(Rome,19167)
(Vienna,18971)
```

4

Find the Maximum, Minimum total reviews of restaurants located in France and have an average rating of 4.0, and save the results

Results

```
(15188,1)
```

5

Count the restaurants that are gluten-free and have average ratings higher than 4 in each country, and save the results

```
(Italy,14764)
(Spain,10920)
(Wales,1348)
(France,1571)
(Greece,3371)
(Poland,504)
(Sweden,441)
(Austria,660)
(Belgium,567)
(Croatia,652)
(Denmark,163)
(England,18791)
(Finland,166)
(Germany,1954)
(Hungary,353)
(Ireland,1704)
(Romania,180)
(Bulgaria,189)
(Portugal,2303)
(Scotland,2188)
(Slovakia,94)
(Czech Republic,316)
(The Netherlands,1544)
(Northern Ireland,440)
```

6

Find the maximum and minimum food ratings of the restaurants that are located in Italy and vegan friendly, and save the results

Results

```
(5.0,2.0)
```

7

Find the total reviews in English for each city that are located in Greece, and save the results

Results

Download

```
(Fri,1)
(Fry,3)
(Ios,45)
(Kea,17)
(Kos,74)
(Oia,73)
(Rio,12)
(Vai,2)
(Zia,13)
(Agia,2)
(Arta,12)
(Arvi,1)
(Avia,2)
(Axos,2)
```

8

Find the average rating of the atmosphere of all the restaurants that are located in Poland and gluten-free

Results

```
(4.07258527725935)
```

9

Find the maximum reviews that have all languages as the default language of the restaurants that have an excellent rating higher that one thousand, and save the results

Results

(2087)

10

Lists the restaurant name that is located in Spain and vegan and has a cheap prices, and save the results

Results

Download

```
(Ash,7)
(Bow,1)
(Box,4)
(Ely,57)
(Eye,6)
(Ham,1)
(Ide,2)
(Kew,21)
(Lea,2)
(Old,1)
(Par,12)
(Rye,48)
(Wem,6)
(Wye,7)
(Acle,7)
(Alne,1)
(Aust,1)
(Bath,284)
(Beer,11)
```

11

Find the number of restaurants that are located in England and have mid-price range in each city, and save the results

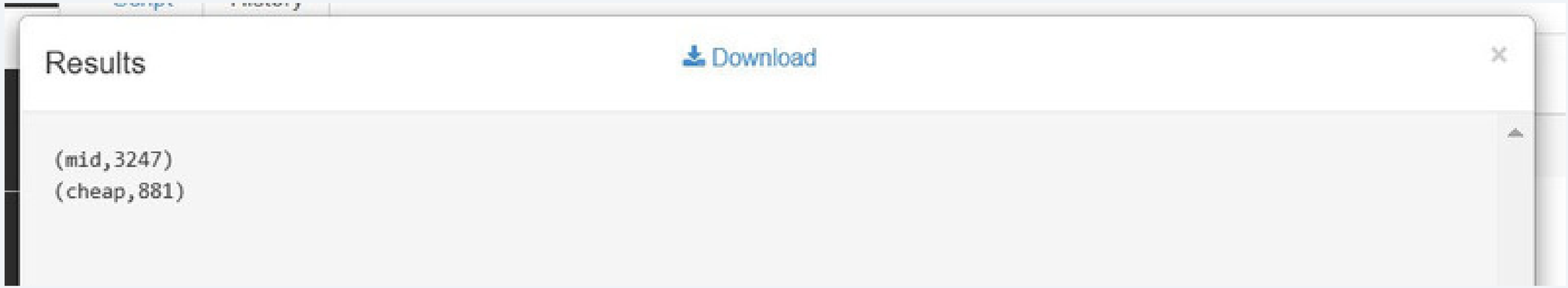
Results

Download

```
(B13,1)
(Bow,1)
(JOY,1)
(Koi,1)
(No3,1)
(Wok,1)
(Amun,1)
(Anil,1)
(Arti,1)
(Aura,1)
(Bold,1)
(Eden,1)
(Ikea,1)
(Kali,1)
(Kint,1)
(Knos,1)
(MAMA,1)
(Maoz,1)
(MoMo,1)
(Ninn,1)
(Popa,1)
(Rooq,1)
(Sopa,1)
(Yoma,1)
(Zuum,1)
(Anavi,1)
(Antic,1)
(Arpit,1)
```

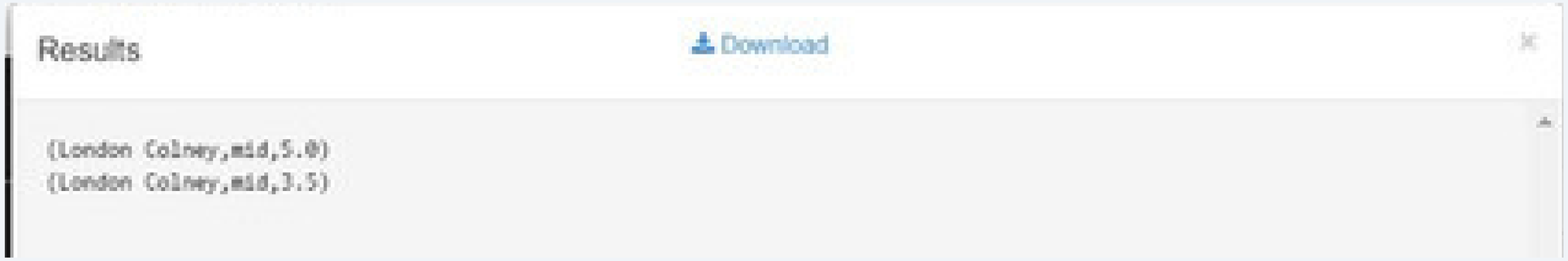
12

Count the restaurants that are vegetarian and located in Paris of each price range , and save the results



13

List the restaurants that are located in London Colney and have a mid-price range, and order them by the Avg ratings , and save the results



14

List the restaurants that are located in Spain and average ratings = 5, and service rating= 4.5, and save the results

Results

Download

```
(Sabroso Bistro,Spain,4.5,5.0)
(Restaurante Hotel Villa de Abalos,Spain,4.5,5.0)
(Recuncho de Gredos,Spain,4.5,5.0)
(Restaurante Napoli Di Notte,Spain,4.5,5.0)
(Abatec C B,Spain,4.5,5.0)
(Sandy  Sandwich,Spain,4.5,5.0)
(Restaurante Vegetariano Mezze,Spain,4.5,5.0)
(Tasca El Caon,Spain,4.5,5.0)
(AMA Taberna,Spain,4.5,5.0)
(Salsamora Boutique,Spain,4.5,5.0)
(Bar Cafeteria La MaGia,Spain,4.5,5.0)
(Restaurante Gines Aguilas,Spain,4.5,5.0)
(Nigella Bistro,Spain,4.5,5.0)
(Restaurante Ordio,Spain,4.5,5.0)
(Trattoria Italia,Spain,4.5,5.0)
(Naturale,Spain,4.5,5.0)
(La Gelateria da Piero,Spain,4.5,5.0)
(Meson De Castro,Spain,4.5,5.0)
(Mum,Spain,4.5,5.0)
(El Terrat,Spain,4.5,5.0)
```

15

What is the cuisines for the most popularity generic restaurant located in Elne city

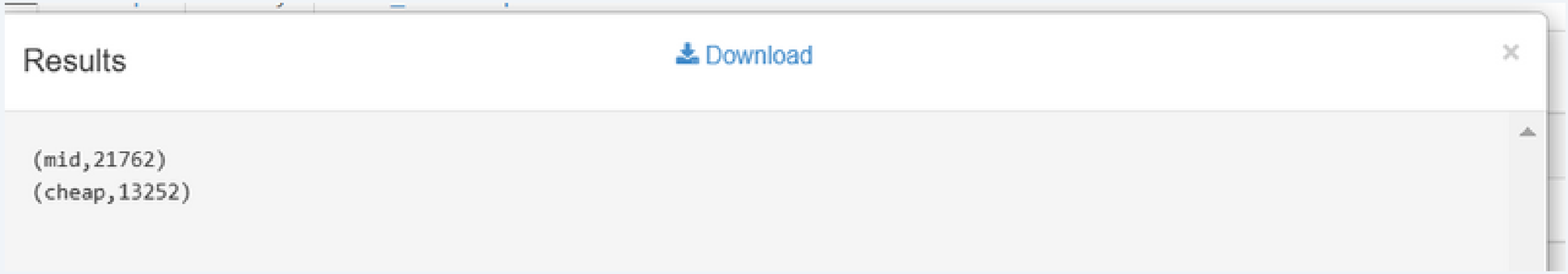
Results

Download

```
(French, Mediterranean, European, Catalan)
```

16

How many Italian cuisine restaurants have a cheap and mid-price range



17

Count the claimed and unclaimed restaurants per country and save them on a separate file

Results

Download

```
(Italy,100674,61004)
(Spain,61202,46551)
(Wales,4222,2592)
(France,70224,45438)
(Greece,14742,9902)
(Poland,7855,6429)
(Sweden,3904,6020)
(Austria,5118,7368)
(Belgium,7739,9502)
(Croatia,3143,2905)
(Denmark,2746,3094)
(England,67461,38610)
(Finland,1926,2039)
(Germany,23362,45939)
(Hungary,2535,2383)
(Ireland,5031,3008)
(Romania,2427,1848)
(Bulgaria,1371,1276)
(Portugal,12968,10225)
(Scotland,6732,4026)
(Slovakia,1177,1107)
(Czech Republic,4023,4362)
(The Netherlands,11217,9616)
(Northern Ireland,1376,1124)
```

Presented By : Fatima Almulhim

18

Find the number of restaurants that have food, service, value, and atmosphere ratings higher than 4, and have mid-price range in each country.

Results

Download

```
(Italy,7588)
(Spain,4676)
(Wales,695)
(France,4946)
(Greece,3123)
(Poland,378)
(Sweden,235)
(Austria,560)
(Belgium,634)
(Croatia,460)
(Denmark,231)
(England,9330)
(Finland,83)
(Germany,1864)
(Hungary,239)
(Ireland,1012)
(Romania,111)
(Bulgaria,144)
(Portugal,1141)
(Scotland,977)
(Slovakia,76)
(Czech Republic,291)
(The Netherlands,851)
(Northern Ireland,226)
```

19

Count the restaurants that open more than 150 hours per week and have average ratings equal to 5, in each country.

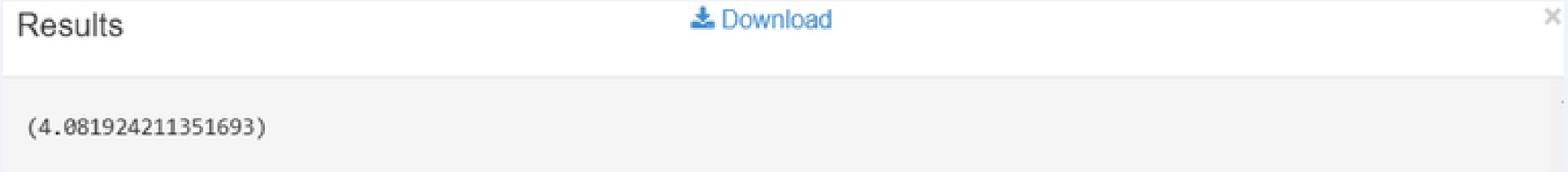
Results

Download

```
(Italy,135)
(Spain,63)
(Wales,7)
(France,64)
(Greece,271)
(Poland,8)
(Austria,3)
(Belgium,1)
(Croatia,13)
(Denmark,1)
(England,45)
(Finland,1)
(Germany,12)
(Hungary,2)
(Ireland,4)
(Romania,6)
(Bulgaria,8)
(Portugal,13)
(Scotland,5)
(Slovakia,2)
(Czech Republic,2)
(The Netherlands,8)
(Northern Ireland,1)
```

20

Find the average for food rating for all restaurants where the price range is cheap



PYSPARK MACHINE LEARNING

1

READ DATASET AS PANDAS THEN CONVERTE IT TO SPARK

```
pandadf = pd.read_csv('/content/drive/MyDrive/cleaning_data.csv') #read csv file as pandas
```

```
#convert from pandas to spark
df = spark.createDataFrame(pandadf)
```

2

combine avg_rating values to 3 categories

```
spark = data.replace( [1.0,1.5,2.0,2.5] , 0, subset=["avg_rating"]) #replace avg_rating values 1.0-2.5  to 0
spark = spark.replace([3,3.5,4.0] , 1, subset=["avg_rating"]) #replace avg_rating  values 3-4.0 to 1
spark = spark.replace([4.5,5.0] , 2, subset=["avg_rating"]) #replace avg_rating values 4.5-5 to 2
```

3

convert type of columns

```
spark = spark.withColumn("vegetarian_friendly", spark["vegetarian_friendly"].cast(IntegerType()))
spark = spark.withColumn("avg_rating", spark["avg_rating"].cast(IntegerType()))
spark = spark.withColumn("vegan_options", spark["vegan_options"].cast(IntegerType()))
spark = spark.withColumn("reviews_count_in_default_language", spark["reviews_count_in_default_language"].cast(IntegerType()))
```

oversampling unbalanced dataset

```
major_df = spark.filter(col("avg_rating") == 2)
minor_df = spark.filter(col("avg_rating") == 0)
major2_df = spark.filter(col("avg_rating") == 1)

'''we will calculate the ratio to determine the difference
between the number of avg_rating 2 and avg_rating 0 transactions.'''
ratio = int(major_df.count()/minor_df.count())
print("ratio: {}".format(ratio))
```

```
ratio: 11
```

```
#create a range on this ratio and store it in variable a
a = range(ratio)

#duplicate the minority rows
oversampled_df = minor_df.withColumn("dummy", explode(array([lit(x) for x in a]))).drop('dummy')
```

oversampling count is : 298650

5

Encoded unnumbric columns

```
#This step will label encode all the categorical columns and store them in different columns with the same name + '_idx',
#so category will become category_idx
cat_cols = ['price_range','cuisines','claimed'] #cuisines and price_range

#StringIndexer() is equivalent to LabelEncoder()
for c in cat_cols:
    indexer = StringIndexer(inputCol=c, outputCol=c+'_idx') #we pass the columns from the list as input one by one
    combined_df = indexer.fit(combined_df).transform(combined_df) #here we fit and transform the data altogether

final_df2 = combined_df.drop(*cat_cols) #we will drop all the categorical columns we defined earlier
```

6

Vector assembler

```
cols = final_df.columns #extract the column names from the dataframe
cols.remove('avg_rating') #remove avg_rating -> we need this to be our label

#vector assembler will take all the columns and convert them into one column called features
assembler = VectorAssembler(inputCols=cols, outputCol='features')

#the .transform will apply the changes here
final_df = assembler.transform(final_df)
```

7

Split Data

```
# We will now create a new dataframe only consisting of the features column and the label column (actually avg_rating column but renamed)
df_data = final_df.select(col('features'), col('avg_rating').alias('label'))

#simple data splitting
df_train, df_test = df_data.randomSplit([0.8, 0.2])
```

8

Models Train

Random Forst Tree

Decision Tree

Logistic Regression

NaiveBayes

Result after Evaluate Models

	Accuracy	F1-Score	Weighted Precision	Weighted Recall
Decision Tree	0.788288	0.784283	0.790955	0.788288
Random Forest	0.804779	0.801209	0.807106	0.804779
Logistic Regression	0.402184	0.231629	0.453494	0.402184
nive	0.479239	0.428802	0.641618	0.479239

Conclusions

The best model is Forest Random Tree
with accuracy .80
and the worst model is Logistic Regression with 0.40

Presented By : Maha Alharbi

Tuning best model

```
#initialize  grid -> we are using variation with only one parameter called maxIter (maximum iteration)
grid = ParamGridBuilder().addGrid(rf.maxDepth, [5,6]).build()
cv = CrossValidator(estimator=rf, estimatorParamMaps=grid, evaluator=evaluator_A, parallelism=2)
#lets fit again on our training set
cvModel = cv.fit(df_train)
```

```
# accuracy of our model on the testing set
evaluator_A.evaluate(cvModel.transform(df_test))
```

```
0.8252858399296394
```

Pipeline best model

```
pipeline = Pipeline(stages=[rf])

# Fit the pipeline to training documents.
model = pipeline.fit(df_train)
```

```
# accuracy of our model on the testing set
evaluator_A.evaluate(model.transform(df_test))
```

```
0.8019790846733386
```

KAFKA IMPLEMENTATION

1st application

- gluten free restaurants

```
#necessay imports
from pykafka import KafkaClient
import csv
```

```
#initialize KafkaClient
client = KafkaClient(hosts="localhost:9092")
# we use this command to access the topic by its name we created earlier
#Topic name is gfree
topic = client.topics['gfree']
```

```
# read TripAdvisor data
TripAdvisor_data=[]
with open('cleaning_data.csv',newline='')as csvfile:
    spamreader=csv.DictReader(csvfile,delimiter=',')
    for row in spamreader:
        TripAdvisor_data.append(row)
```

```
#access the producer
producer =topic.get_sync_producer()
#we want to filter the data by gluten free restaurant
#and we want to return the restaurants name and country and city and It's avg rating
for i in TripAdvisor_data:
    if i['gluten_free'] == '1':
        data = "Nestaurant Name: "+ i['restaurant_name'] +", country: "+ i['country']
        + ", city: "+ i['city'] + ", Average Rating: "+ i['avg_rating']
        producer.produce(bytes(data, 'utf-8'))
```

```
# we can access the consumer using the following statement
consumer = topic.get_simple_consumer()
# we will now use a simple loop to get all the values one by one
for message in consumer:
    if message is not None:
        print (message.offset, message.value)
```

Results

```
 0 b"Nestaurant Name: Restaurant Vlaar, country: The Netherlands, city: 's-Graveland, Average Rating: 4.0"
 1 b"Nestaurant Name: De Colonie, country: The Netherlands, city: 's-Graveland, Average Rating: 4.0"
 2 b"Nestaurant Name: 't Swaentje, country: The Netherlands, city: 's-Graveland, Average Rating: 4.5"
 3 b"Nestaurant Name: BeachLine Beach & Events s-Gravenzande, country: The Netherlands, city: 's-Gravenzande, Average
Rating: 4.5"
 4 b"Nestaurant Name: Bistro de Herberg, country: The Netherlands, city: 's-Gravenzande, Average Rating: 4.5"
 5 b'Nestaurant Name: Ave Dourada, country: Portugal, city: A dos Cunhados, Average Rating: 4.5'
 6 b'Nestaurant Name: Cafe Butler, country: Denmark, city: Aabenraa, Average Rating: 4.5'
 7 b'Nestaurant Name: Misaki, country: Germany, city: Aachen, Average Rating: 4.5'
 8 b'Nestaurant Name: Ferbers, country: Germany, city: Aachen, Average Rating: 3.5'
 9 b'Nestaurant Name: LivingRoom, country: Germany, city: Aachen, Average Rating: 4.0'
10 b'Nestaurant Name: Pfannenzauber, country: Germany, city: Aachen, Average Rating: 4.0'
11 b'Nestaurant Name: Konak, country: Germany, city: Aachen, Average Rating: 3.5'
12 b'Nestaurant Name: Kanpai Running Sushi & Lounge, country: Germany, city: Aachen, Average Rating: 4.5'
13 b'Nestaurant Name: Polonia, country: Germany, city: Aachen, Average Rating: 4.5'
14 b'Nestaurant Name: Degustino, country: Germany, city: Aachen, Average Rating: 4.0'
15 b'Nestaurant Name: Mei Choi, country: Germany, city: Aachen, Average Rating: 5.0'
16 b'Nestaurant Name: Kaiserwetter, country: Germany, city: Aachen, Average Rating: 4.0'
17 b'Nestaurant Name: Sausalitos, country: Germany, city: Aachen, Average Rating: 3.5'
```

2nd application

- vegetarian friendly restaurants

jupyter Topec۲ Last Checkpoint: منذ ٢١ دقيقة (unsaved changes) Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3 (ipykernel)

Run Code

In [10]:
```
#necessay imports
from pykafka import KafkaClient
import csv
```

In [6]:
```
#initialize KafkaClient
client = KafkaClient(hosts="localhost:9092")
# we use this command to access the topic by its name we created earlier
#Topic name is gfree
topic = client.topics['vegetarian']
```

In [7]:
```
# read TripAdvisor data
TripAdvisor_data=[]
with open('/Users/areejalhuthali/cleaning_data.csv',newline='')as csvfile:
    spamreader=csv.DictReader(csvfile,delimiter=',')
    for row in spamreader:
        TripAdvisor_data.append(row)
```

In [8]:
```
#access the producer
producer =topic.get_sync_producer()
#we want to filter the data by gluten free restaurant
#and we want to return the restaurants name and country and city and It's avg rating
for i in TripAdvisor_data:
    if i['vegetarian_friendly'] == '1':
        data = "Nestaurant Name: "+ i['restaurant_name'] +", country: "+ i['country']+ ", city: "+ i['city'] + ", Avera
        producer.produce(bytes(data, 'utf-8'))
```

In [9]:
```
# we can access the consumer using the following statement
consumer = topic.get_simple_consumer()
# we will now use a simple loop to get all the values one by one
for message in consumer:
    if message is not None:
        print (message.offset, message.value)
```

```
0 b"Nestaurant Name: Restaurant Nonna, country: The Netherlands, city: 's Gravendeel, Average Rating: 4.0"
```

CONCLUSIONS AND FUTURE WORK

Presented By : Areej Alhuthali

THANK YOU!

ANY QUESTIONS?

Rasha Samar Jawhara

Maha Fatima Areej