Section 1: Yelp Data Challenge - Data Preprocessing

Yiting Luo | Data Science Applied Research - 1

May 2018

Dataset Introduction

Yelp Dataset Challenge (https://www.yelp.com/dataset_challenge)

The Challenge Dataset:

```
4.1M reviews and 947K tips by 1M users for 144K businesses
1.1M business attributes, e.g., hours, parking availability, ambience.
Aggregated check-ins over time for each of the 125K businesses
200,000 pictures from the included businesses
```

Cities:

```
U.K.: Edinburgh
Germany: Karlsruhe
Canada: Montreal and Waterloo
U.S.: Pittsburgh, Charlotte, Urbana-Champaign, Phoenix, Las Vegas, Madison
, Cleveland
```

Files:

```
yelp_academic_dataset_business.json
yelp_academic_dataset_checkin.json
yelp_academic_dataset_review.json
yelp_academic_dataset_tip.json
yelp_academic_dataset_user.json
```

Notes on the Dataset

```
Each file is composed of a single object type, one json-object per-line.
```

Read data from file and load to Pandas DataFrame

In [2]:
```
import json
import pandas as pd
import os
os.getcwd()
```

Out[2]:
```
'/Users/luoyiting/Desktop/yelp dataset'
```

In [3]:
```
file_business, file_checkin, file_review, file_tip, file_user = [
    'dataset/business.json',
    'dataset/checkin.json',
    'dataset/review.json',
    'dataset/tip.json',
    'dataset/user.json'
]
```

Business Data

In [4]:
```
with open(file_business) as f:
    df_business = pd.DataFrame(json.loads(line) for line in f)
```

In [5]:
```
df_business.head(2)
```

Out[5]:

	address	attributes	business_id	categories	
0	4855 E Warner Rd, Ste B9	{'AcceptsInsurance': True, 'ByAppointmentOnly'...	FYWN1wneV18bWNgQjJ2GNg	[Dentists, General Dentistry, Health & Medical...	Ahwat
1	3101 Washington Rd	{'GoodForKids': True, 'WheelchairAccessible': ...	He-G7vWjzVUysIKrfNbPUQ	[Hair Stylists, Hair Salons, Men's Hair Salons...	McMu

In [6]:
```
df_business.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 174567 entries, 0 to 174566
Data columns (total 15 columns):
address         174567 non-null object
attributes      174567 non-null object
business_id     174567 non-null object
categories      174567 non-null object
city            174567 non-null object
hours           174567 non-null object
is_open         174567 non-null int64
latitude        174566 non-null float64
longitude       174566 non-null float64
name            174567 non-null object
neighborhood    174567 non-null object
postal_code     174567 non-null object
review_count    174567 non-null int64
stars           174567 non-null float64
state           174567 non-null object
dtypes: float64(3), int64(2), object(10)
memory usage: 20.0+ MB
```

Checkin Data

In [7]:
```
with open(file_checkin) as f:
    df_checkin = pd.DataFrame(json.loads(line) for line in f)
df_checkin.head(2)
```

Out[7]:

	business_id	time
0	7KPBkxAOEtb3QeIL9PEErg	{'Saturday': {'2:00': 1, '12:00': 1, '16:00': ...
1	kREVIrSBbtqBhlYkTccQUg	{'Saturday': {'21:00': 1, '16:00': 1}, 'Monday...

Review Data

In [8]:
```
with open(file_review) as f:
    df_review = pd.DataFrame(json.loads(line) for line in f)
df_review.head(2)
```

Out[8]:

	business_id	cool	date	funny	review_id	stars
0	0W4lkclzZThpx3V65bVgig	0	2016-05-28	0	v0i_UHJMo_hPBq9bxWvW4w	5
1	AEx2SYEUJmTxVVB18LlCwA	0	2016-05-28	0	vkVSCC7xljjrAI4UGfnKEQ	5

Tip Data

In [9]:
```
with open(file_tip) as f:
    df_tip = pd.DataFrame(json.loads(line) for line in f)
df_tip.head(2)
```

Out[9]:

	business_id	date	likes	text	user_id
0	tJRDll5yqpZwehenzE2cSg	2012-07-15	0	Get here early enough to have dinner.	zcTZk7OG8ovAmh_fenH21g
1	jH19V2I9fIslnNhDzPmdkA	2015-08-12	0	Great breakfast large portions and friendly wa...	ZcLKXikTHYOnYt5VYRO5sg

User Data

In [10]:
```
with open(file_user) as f:
    df_user = pd.DataFrame(json.loads(line) for line in f)
df_user.head(2)
```

Out[10]:

	average_stars	compliment_cool	compliment_cute	compliment_funny	compliment_
0	4.67	0	0	0	0
1	3.70	0	0	0	0

2 rows × 22 columns

Filter data by city and category

Create filters/masks

- create filters that selects business
  - that are located in "Las Vegas"
  - that contains "Restaurants" in their category

In [36]:
```
# Create Pandas DataFrame filters
cond_city = []
cond_category_not_null = []
cond_category_resturant = []
# city Las Vegas bool
cond_city = df_business['city'] == 'Las Vegas'

# isnull bool
cond_category_not_null = ~df_business['categories'].isnull()

# resturant in categories
# first, apply(str) convert categories to strings,
# then check if it contains Resturants
cond_category_resturant = df_business['categories'].apply(str).str.con
tains("Restaurants")
```

In [37]:
```
# Create filtered DataFrame, and name it df_filtered
df_filtered = df_business[cond_city & cond_category_not_null & cond_ca
tegory_resturant]
```

Keep relevant columns

- only keep some useful columns
  - business_id
  - name
  - categories
  - stars

In [38]:
```
selected_features = [u'business_id', u'name', u'categories', u'stars']
```

In [39]:
```
# Make a DataFrame that contains only the abovementioned columns, and
name it as df_selected_business
df_selected_business = df_filtered[selected_features]
```

In [40]:
```
# Rename the column name "stars" to "avg_stars" to avoid naming confli
cts with review dataset
df_selected_business.rename(columns={'stars':'avg_stars'}, inplace = T
rue)
```

```
/Users/luoyiting/anaconda/lib/python3.5/site-packages/pandas/core/fr
ame.py:2844: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/panda
s-docs/stable/indexing.html#indexing-view-versus-copy
  **kwargs)
```

In [41]:
```
# Inspect DataFrame
df_selected_business.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 5899 entries, 52 to 174469
Data columns (total 4 columns):
business_id    5899 non-null object
name           5899 non-null object
categories     5899 non-null object
avg_stars      5899 non-null float64
dtypes: float64(1), object(3)
memory usage: 230.4+ KB
```

In [42]:
```
df_selected_business.head(5)
```

Out[42]:

	business_id	name	categories	avg_stars
52	Pd52CjgyEU3Rb8co6QfTPw	Flight Deck Bar & Grill	[Nightlife, Bars, Barbeque, Sports Bars, Ameri...	4.0
53	4srfPk1s8nlm1YusyDUbjg	Subway	[Fast Food, Restaurants, Sandwiches]	2.5
54	n7V4cD-KqqE3OXk0irJTyA	GameWorks	[Arcades, Arts & Entertainment, Gastropubs, Re...	3.0
91	F0fEKpTk7gAmuSFI0KW1eQ	Cafe Mastrioni	[Italian, Restaurants]	1.5
122	Wpt0sFHcPtV5MO9He7yMKQ	McDonald's	[Restaurants, Fast Food, Burgers]	2.0

Save results to csv files

In [43]:
```
# Save to ./dataset/selected_business.csv for next task
df_selected_business.to_csv("dataset/selected_business.csv", index = False, encoding = "utf-8")
```

In [44]:
```
# Try reload the csv file to check if everything works fine
pd.read_csv("dataset/selected_business.csv", encoding = "utf-8").head()
```

Out[44]:

	business_id	name	categories	avg_stars
0	Pd52CjgyEU3Rb8co6QfTPw	Flight Deck Bar & Grill	['Nightlife', 'Bars', 'Barbeque', 'Sports Bars...	4.0
1	4srfPk1s8nlm1YusyDUbjg	Subway	['Fast Food', 'Restaurants', 'Sandwiches']	2.5
2	n7V4cD-KqqE3OXk0irJTyA	GameWorks	['Arcades', 'Arts & Entertainment', 'Gastropub...	3.0
3	F0fEKpTk7gAmuSFI0KW1eQ	Cafe Mastrioni	['Italian', 'Restaurants']	1.5
4	Wpt0sFHcPtV5MO9He7yMKQ	McDonald's	['Restaurants', 'Fast Food', 'Burgers']	2.0

Use the "business_id" column to filter review data

- We want to make a DataFrame that contain and only contain the reviews about the business entities we just obtained

Load review dataset

In [45]:
```
with open(file_review) as f:
    df_review = pd.DataFrame(json.loads(line) for line in f)
df_review.head(2)
```

Out[45]:

	business_id	cool	date	funny	review_id	stars
0	0W4lkclzZThpx3V65bVgig	0	2016-05-28	0	v0i_UHJMo_hPBq9bxWvW4w	5
1	AEx2SYEUJmTxVVB18LlCwA	0	2016-05-28	0	vkVSCC7xljjrAI4UGfnKEQ	5

Prepare dataframes to be joined, - on business_id

In [46]:
```
# Prepare the business dataframe and set index to column "business_id"
, and name it as df_left
df_left = df_selected_business.set_index("business_id")
```

In [47]:
```
# Prepare the review dataframe and set index to column "business_id",
and name it as df_right
df_right = df_review.set_index("business_id")
```

Join! and reset index

In [48]:
```
# Join df_left and df_right.
df_joined = df_left.join(df_right, how = 'inner')
```

In [49]:
```
# reset the index
df_joined.reset_index(inplace = True)
```

We further filter data by date, e.g. keep comments from last 2 years

- Otherwise laptop may crush on memory when running machine learning algorithms
- Purposefully ignoring the reviews made too long time ago

In [51]:
```
# Make a filter that selects date after 2015-01-20
cond_last_2_years = df_joined['date'] > u'2015-01-20'
```

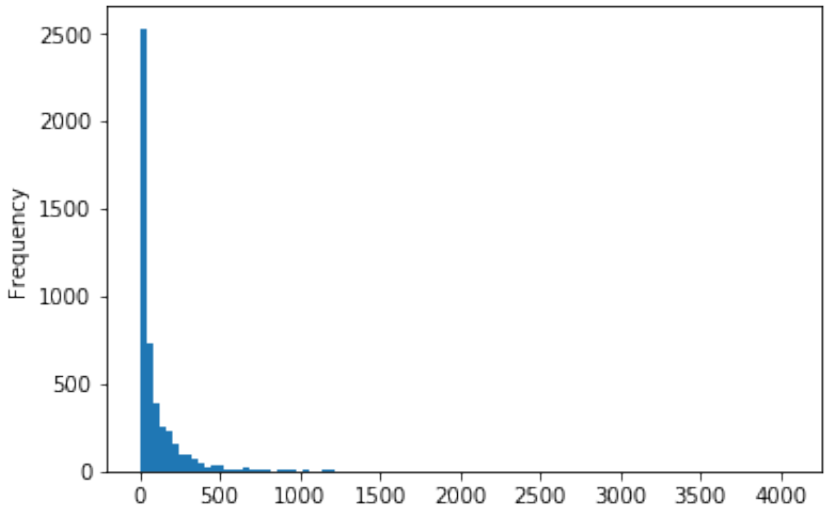
In [52]:
```
# Filter the joined DataFrame and name it as df_final
df_final = df_joined[cond_last_2_years]
```

Take a glance at the final dataset

In [53]:
```
import matplotlib.pyplot as plt

% matplotlib inline
```

In [55]:
```
# calculate counts of reviews per business entity, and plot it
df_final['business_id'].value_counts().plot.hist(bins = 100)
plt.show()
```



Save preprocessed dataset to csv file

In [56]:
```
# Save to ./dataset/last_2_years_restaurant_reviews.csv for next task
df_final.to_csv('dataset/last_2_years_restaurant_reviews.csv', index = False, encoding = 'utf-8')
```

In [ ]:
```

```