95-869: Big Data and Large-Scale Computing Homework 5

Yelp Dataset Assignment

This project delves into exploratory analysis and building predictive models using the Yelp academic dataset. It is an opportunity for you to explore machine learning tasks in the context of a real-world data set using big data analysis tools. In order to use the dataset and finish this project, you must agree to the dataset's terms of use provided here .

We have chosen a subset of the Yelp academic dataset for you to work with. This subsampled data is loaded into RDDs in part (0). The complete dataset is available from Yelp's website here.

This assignment will cover:

- *Part 1 (30 Points):* Exploratory Data Analysis Link Link
- *Part 2 (15 Points):* Prediction using tree ensemble methods Link
- *Part 3 (20 Points):* Collaborative filtering for recommendation Link
- *Part 4 (15 Points):* Topic modeling for text reviews Link
- *Part 5 (10 Points):* Word2Vec for text reviews Link
- *Part 6 (10 Points):* Frequent pattern mining using FP-Growth algorithm Link
- *Part 7 (Bonus: 20 Points):* Any additional and insightful exploratory data analysis or machine learning tasks you want to do.

Note that, for reference, you can look up the details of the relevant Spark methods in Spark's Python API and the relevant NumPy methods in the NumPy Reference

Code of Conduct

Please follow the following guidelines with respect to collaboration:

- By using the dataset, you agree to Yelp's terms of use available here .
- You are free to use the Web, APIs, ML toolkits, etc. in this project to your best benefit. Please cite any online or offline sources (even casual sources like StackOverflow) if you use them in the project.
- The assignment is to be done individually. No collaboration is allowed between students. No discussion is allowed about the project with anyone else except the class instructors.

Submission Instructions:

You will submit both an html and a Jupyter Notebook on Canvas. You will submit a PDF on Gradescope and assign the pages to their corresponding questions. No printout submission is expected. The submission issues below will result in points deduction:

1. If the student doesn't make the submission on both Canvas and Gradescope: -5%
2. If the student submits an empty jupyter notebook/version obviously different from the submitted PDF: -5%
3. If the student does't assign pages to questions on Gradescope: -3%

Rename the notebook from "hw5_yelp_student.ipynb" to "studentid_hw5_yelp_student.ipynb" where "studentid" is your Andrew ID. Complete the assignment, execute all cells in the completed notebook, and make sure all results show up. Export the contents of the notebook by choosing "File > Download as > HTML" and saving the resulting file as "studentid_hw5_yelp_student.html" Convert the exported HTML file to PDF by using a feature such as the "Save as PDF" feature on Mac. Another way to get the PDF file is to do ctrl(or command) + P and save as a PDF. Submit the PDF solution to gradescope. Submit the IPython and exported html solution files on Canvas.

Part 0: Load the datasets required for the project

We will load four datasets for this project. In addition to the four datasets, we will also load two lists which contain names by gender. These lists are helpful in assigning a gender to a Yelp user by their name, since gender is not available in the Yelp dataset.

Let's first start by creating the SparkContext.

In [1]:
```
import sys
sys.path.append("/opt/packages/spark/latest/python/lib/py4j-0.10.9-src.zip")
sys.path.append("/opt/packages/spark/latest/python/")
sys.path.append("/opt/packages/spark/latest/python/pyspark")
from pyspark import SparkConf, SparkContext
from pyspark.sql import SparkSession
import nltk
spark = SparkSession.builder.master('local[*]').config("spark.driver.memory", "25g").getOrCreate()
sc = spark.sparkContext
sc
```

```
23/04/30 22:27:31 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where appli
cable
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
```

Out[1]: **SparkContext**

Spark UI

Version	`v3.0.1`
Master	`local[*]`
AppName	`pyspark-shell`

In [2]:
```
import json
import os
import sys
import os.path
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
%matplotlib inline

# helper function to load a JSON dataset from a publicly accessible url
def get_rdd_from_path(path):
    file_reader = open(path, 'r')
    str_contents = file_reader.readlines()
    json_contents = [json.loads(x.strip()) for x in str_contents]
    rdd = sc.parallelize(json_contents,  numSlices=500)
    return rdd
```

The first dataset we are going to load is information about Yelp businesses. The information of each business will be stored as a Python dictionary within an RDD. The dictionary consists of the following fields:

- "business_id":"encrypted business id"
- "name":"business name"
- "neighborhood":"hood name"
- "address":"full address"
- "city":"city"
- "state":"state -- if applicable --"
- "postal code":"postal code"
- "latitude":latitude
- "longitude":longitude
- "stars":star rating, rounded to half-stars
- "review_count":number of reviews
- "is_open":0/1 (closed/open)

- "attributes":["an array of strings: each array element is an attribute"]
- "categories":["an array of strings of business categories"]
- "hours":["an array of strings of business hours"]
- "type": "business"

In [3]:
```
# load the data about Yelp businesses in an RDD
# each RDD element is a Python dictionary parsed from JSON using json.loads()
businesses_rdd = get_rdd_from_path('/ocean/projects/cis220071p/shared/data/yelp_academic_dataset_business.json')
print (businesses_rdd.count())
print (businesses_rdd.take(2))
```

```
[Stage 0:====================>                                (203 + 128) / 500]
61184
[{'business_id': 'vcNAWiLM4dR7D2nwwJ7nCA', 'full_address': '4840 E Indian School Rd\nSte 101\nPhoenix, AZ 85018', 'hours': {'Tuesday':
{'close': '17:00', 'open': '08:00'}, 'Friday': {'close': '17:00', 'open': '08:00'}, 'Monday': {'close': '17:00', 'open': '08:00'}, 'Wedne
sday': {'close': '17:00', 'open': '08:00'}, 'Thursday': {'close': '17:00', 'open': '08:00'}}, 'open': True, 'categories': ['Doctors', 'He
alth & Medical'], 'city': 'Phoenix', 'review_count': 9, 'name': 'Eric Goldberg, MD', 'neighborhoods': [], 'longitude': -111.983758, 'stat
e': 'AZ', 'stars': 3.5, 'latitude': 33.499313, 'attributes': {'By Appointment Only': True}, 'type': 'business'}, {'business_id': 'UsFtqoB
l7naz8AVUBZMjQQ', 'full_address': '202 McClure St\nDravosburg, PA 15034', 'hours': {}, 'open': True, 'categories': ['Nightlife'], 'city':
'Dravosburg', 'review_count': 4, 'name': "Clancy's Pub", 'neighborhoods': [], 'longitude': -79.88693, 'state': 'PA', 'stars': 3.5, 'latit
ude': 40.350519, 'attributes': {'Happy Hour': True, 'Accepts Credit Cards': True, 'Good For Groups': True, 'Outdoor Seating': False, 'Pri
ce Range': 1}, 'type': 'business'}]
```

The second dataset we are going to load is information about Yelp users. Each user's information will be stored as a Python dictionary within an RDD. The dictionary consists of the following fields:

- "user_id":"encrypted user id"
- "name":"first name"
- "review_count":number of reviews
- "yelping_since": date formatted like "2009-12-19"
- "friends":["an array of encrypted ids of friends"]
- "useful":"number of useful votes sent by the user"
- "funny":"number of funny votes sent by the user"
- "cool":"number of cool votes sent by the user"
- "fans":"number of fans the user has"
- "elite":["an array of years the user was elite"]
- "average_stars":floating point average like 4.31
- "compliment_hot":number of hot compliments received by the user
- "compliment_more":number of more compliments received by the user
- "compliment_profile": number of profile compliments received by the user
- "compliment_cute": number of cute compliments received by the user

- "compliment_list": number of list compliments received by the user
- "compliment_note": number of note compliments received by the user
- "compliment_plain": number of plain compliments received by the user
- "compliment_cool": number of cool compliments received by the user
- "compliment_funny": number of funny compliments received by the user
- "compliment_writer": number of writer compliments received by the user
- "compliment_photos": number of photo compliments received by the user
- "type":"user"

In [4]:
```
# load the data about Yelp users in an RDD
# each RDD element is a Python dictionary parsed from JSON using json.loads()
users_rdd = get_rdd_from_path('/ocean/projects/cis220071p/shared/data/yelp_academic_dataset_user.json')
print (users_rdd.count())
print (users_rdd.take(2))
```

366715
[{'yelping_since': '2004-10', 'votes': {'funny': 166, 'useful': 278, 'cool': 245}, 'review_count': 108, 'name': 'Russel', 'user_id': '18k
Pq7GPye-YQ3LyKyAZPw', 'friends': ['rpOyqD_893cqmDAtJLbdog', '4U9kSBLuBDU391x6bxU-YA', 'fHtTaujcyKvXglE33Z5yIw', '8J4IIYcqBlFch8T90N923A',
'wy6l_zUo7SN0qrvNRWgySw', 'HDQixQ-WZEV0LVPJlIGQeQ', 'T4kuUr_iJiywOPdyM7gTHQ', 'z_5D4XEIlGAPjG3Os9ix5A', 'i63u3SdbrLsP4FxiSKP0Zw', 'pnrGw4
ciBXJ6U5QB2m0F5g', 'ytjCBxosVSqCOQ62c4KAxg', 'r5uiIxwJ-I-oHBkNY2Ha3Q', 'niWoSKswEbooJC_M7HMbGw', 'kwoxiKMyoYjB1wTCYAjYRg', '9A8OuP6XwLwnN
b9ov3_Ncw', '27MmRg8LfbZXNEHkEnKSdA', 'Bn4sJUTtKFZQt0FKHF2Adw', 'uguXfIEpI65jSCH5MgUDgA', '6VZNGc2h2Bn-uyuEXgOt5g', 'AZ8CTtwr-4sGM2kZqF6q
ig', 'S742m-AuQicMSLDdErrLZQ', 'uGmQ6ab4iVpWn5m61VFhkQ', 'GJYJX4SujVj3BR8v2F9PDQ', '3shjifK-vZkIHciyy_KbYA', '4lc_H2Cf7CO0tCgyA3aSVQ', 'T
unkp_F1R_uFBJQTsDxD4g', 'B9pKfr27czBbCoAIircZdQ', 'pePGMO6EbDpbaZ7D2m6HIg', 'XRM8W6HUoXbrYKR3BCj9Rg', '8DqIWXsKXOipfduYEfFpNw', 'dvRVX54Z
9f7Om51NsTRX1w', 'CM0saLQmk4oAB17UmQTV-g', 'HANb8-8InWnju-XzBQQSBw', 'JuJeZeQJgv7bUreY7a1SlQ', '2NVVEEYhhoVELdaPILFrDQ', 'e4M9_S-ASmRys3D
vKQfotw', 'XA109o963exKgoVGcg9z7w', 'tVAKdax3Jbf24R7OJB99JQ', 'OMWT-Z6OnJLcg44lCuDuhw', 'd82F_FFtowYtjGtxRySehA', 'pW91HUnVz6ssLZ4dY-zty
Q', 'ojQYtstkGXtcryk5I9GTYA', '4hxVql33ldY_gkf3rG8_3w', '0arQ82n4mvrc42U8KuoE6g', 'w6Vv-kldGpmvSGqXvTbAdQ', 'NZeCINmoOJ8vsQvYkZAp9w', 'n9
ltC5DxMjefffeMfBgcXQ', 'QBj2AL66bEAmK3ULkrn1Gw', 'ayZlWyV1M2k_UWx1dreIDg', 'AG1KRDkIa8QCCPsMnINEOg', 'P-lfV8cdAgEOZJkw2dJpsw', '7Pef9EA21
szXXOPJo6Gb5w', 'YvtRBqCqhSIhGGREqcqnQw', '8M7I2-s5nQ8LSkuCvN0Xqg', '-ETxy7f37BBQXhw5zJfkrQ', 'TjPt2hD56jfpmtoa6kVVRg', 'uZgFEPEUIWJMaaqv
MlxYoA', 'YKppbE7ogNmbtVrluxEF4A', 'D7uXj1_0pE_Fa2YeLkOkyw', 'eVV99D3EZvkuswIvIySKiw', 'PhwgVPqpJlxu40AxAOUBbg', 'gWcz4QFTW76q8VtpIlZEH
Q', 'wpcoiQNbO5x0xkZwKFVUFw', 'Ot-QjYpIdVsluuxcqidjkA', 'E3C_gEhktux9Ca-Z2fmNTw', 'ClaGLOr72DPJlQhRc9HHyQ', 'sc6wS6YOFIgtN2XuJI8_WQ', 'dD
5mA6kwVlBHvrNtX9EyZQ', 'qwvgycPC5u4JZv6DfawPqw', 'As_oVE27fiE-0CbDJwUDiA', 'SuzSrMWoycf4DXL_DMMdNg', 'y0SRKlzVuvj-mbQtbsWx-A', 'qbfQRHLvZ
k5WSkKY0l_lMw', 'D6A0rn5MIg7AVRrVeouABg', 'KUqpJ9eKREuvlnlitJJRYg', 'gOvKWYXobX_Pgzsz-K8auA', 'xzdZTsR2KKcQfjF4OjJeOw', 'DyfA7aPzRQ81kJnb
kuIiJQ', 'tA2suyZG_qWIuNB_HR-YRA', 'R5QmtpSsrQ9mqIxjZIv3uw', '5T8SdBwiH0jdwatUJ7KFUg', '1Gfn4nlqPEIvRX4cnctJlg', 'A4_fxqSj4tgYet8OE3jTy
A', 'oPD_t6BVpfhvu1-TS4FNzA', 'gn3Y0mWduWrvT1hvu7fv4A', 'NUeLSLukiPPCSK6EVdUo_Q', 'rtrh46NJfksxR4TgljYg9w', '5zpVMWD-q6nEYapiWrSyXw', 'oo
Z7TV6IuxuqbFgnqaoSsQ', '8phGuOZ0NIfeJiTX-ZDZLA', '8L1p_zlmAB_uWyaW0QTaCg', 's7ZSag-bHo0C4fSBDJeBDw', '5ANDwUyQqsRhEl7QBKRFiw', 'KMGF6KAnP
RoOTFVXBy9jtQ', 'yk-lz4JUKREUVw9hxk-VZw', '6uM5lx0DCX4ctQlebtxraw', 'Hc6agWZ9jGCxsPAqQwlf2w', 'z39ulrgtvEMB2oeokx0-mw', 'wLt28qh-46LFhKOe
ByWZ0w', 'R9FFFqYOUmvWjaHqODfzyw', 'p4Jg_rO_AsCWZGMd_w8xjw', 'pJppmQyW74ejj8xCa2SHuA', 'hzaqnSxfRYv_Wbp6CMKtQw', '0QwV8PJk9PJDFaMNw1Fcs
w', '6pC5j-hfP4xLZB2W0Wi1vg', '8nAj60xsBjcHXLLSs7cPqw', 'NhbV8AbaQiKNzEU11cbxcQ', '5Szwd1CSepRXwAFH12nPzQ', 'ifD7o8Mi67yNYwgWL5CbKg', 'Aa
ZdXn0I6F5bdIVwGpxdDA', 'YwcacxA7WCB9fryDHDxF8w', 'HBFtzjRarXxDH8xZRSNsPA', 'zi0kxtr5gq0Flg7jYdIlMA', 'oWEdmTYl13VHh1A_Mkhh2g', 'n1GSQ7VT9
sTL5MlCWolbNw', 'PO8XNxGUizzzKGHmn24fQQ', 'JTRxzA1AZ8Zfwj7FpH2CnA', 'AixLPxra_4HirfchjPkiWA', 'xOK3i0wgmkpGu54BuIcwQA', 'F9T6m1YdRFreyKDu
fcyoOQ', '-cscgA-lVPipPwEX4RmcgQ', '1UJbOl-jKNIUjroeC4xWxg', 'ACLuEsNHV7lfq1O7CHuFaA', 'UyOtYjnavsr-Dtjo6Xy91g', 'J11gYVC1u9S5jqEg3oh-g
Q', '9LZP7Ll7N9zkuWfA5DUAYw', 'SbM1m5nejHDD1pJePsar9g', 'wF7_-zf7R8Rri7Kq-a0tlw', 'B5o7JC4iRJ7bKPRnoYEIHg', 'SAPz30NHFJ95SBlvz5cTqw', 'fz
kO1lGkx8MHhT8S4Rbg3w', 'tlSSQwfHYJany7wPoTH46A', '2VfhMrTJ9bZ0O4PZmRmxKA', 'odeoZZHRQp6TkeoEQk3eeA', '2XCsVpAMsGUykrM-BBFEXA', 'qVePixO2k
uK0Dr03upCWFg', 'D9CIW04wrcQzZNdCEoASAg', 'G-iuXcbJnX3sA3lhB6nFhQ', '3utEqyNzFcxH8E8pm7ByPA', 'T5TZfn9_9X-LTbRtaMAgUw', 'fczQCSmaWF78toLE
mb0Zsw', 'Hyhmpb3UnbjtSEXvgviCow', 'FI-Fjxr3fUBx2q7icnqcfA', 'S8ZY1HEDoECqKrIUY34JOg', '3JtdBvnY6Wzp22gt5zFwDg', '470dZswdTHz5aaGPvIYNp
g', 'JT5kM6wRwh7cNt8IAuX_sw', 'r7V9Nhes-Eg-lH0cqfpFFw', '2m3lE8p8T2nFRDxIUaRXew', 'bZFRqP7s0Vszxeu8_IwYow', 'snd45oedwXxF9tsHUX_6dg', '8j
EK8gxHYg9w6sINujO0MQ', 'LTWY2Ee-SYu8ItVOebD3CQ', '7zpDhrRZRTGCkAh3SHbEww', 'NSg7k2UFt_HqM2mvsw-n0w', 'egdj4-wXAzYzT5bmTAHYqQ', 'wxhf-RSEx
CGECZuz_iVV7w', 'C6IOtaaYdLIT5fWd7ZYIuA', 'jntSRFnN_O7QfFzZo24rvA', 'yfQwWLYtqwHKiky1hiTBew', '6SxnRYFR3GUzGREKSttxJg', 'WZgoM8UuzxjZf1Aa
Wx3qZQ', 'cNj8rheuldPDX9zF6q2yaA', 'OqOCMRpVNvvdPDMhoG_DCA', '9ZonzxW0DAqIUO_5tDqZig', 'rLQNlHnIKdtoqoU77CbEUQ', 'LbgQK5B_5IkN77FgRJHhr
g', 'it23EnutJ6fOjeYQBDKJug', '1XWzXgFvcu5pU2wj1ONQ_Q', '92j9xiUMOsM3HviIGoR8Yg', 'AP6udBIStvWLY0T5rS_24A', 'G_PCzMHcOLoMyyEsxW1DFQ', 'Zu
Q87LQYWRw3AHj-jEiQ1Q', 'zdutsCEMdvc0B8s3_XN3nQ', 'fRbbPPUdMCmL9o39yshi2g', 'pjgE1d3csesAkgn6rQI_RQ', 'C7nUczj6n0ouxWqp1I7cNQ', 'D8EE4gbAi
XS3DF_HNRJfAw', 'Milv8k4zZsoCaNFZugKtJw', 'GALRNo-AbcAHxEdmqKl--w', 'XOe-I25-W_FnbYfQyafylw', '-weTB55ulUzv0E_4laXsqQ', 'kuP0AMnSGgkLfNSt
0wOAgA', 'EwMgdjuPXLAuxxu3IOUyKA', 'Mx-vxv_V-SQCe76w4RmUfA', '6xxVdJBdMREj1fCqjm6UtA', 'm61SjiXAIlOY1qAXOHEk4Q', 'R4sYowLqQQEpxWjQbqaFv
g', 'OCkge7lzOw33bqMs1244Yw', 'rjJJ8QQg-eqb7mgMxArgBA', 'T_wjLgPOPXry7Bea4MzoVQ', 'hBwbBaBqofCgvkJ3PatuVw', 'epTMlecM8SZNjDhBsd64Jg', 'PU
MROYfbVgKypcfJNxs1mw', '82ZqgHOfUMV1xtpG2KfisA', 'IN2HcBun5PpyfQK8OC68iw', 'yQZL8MTulNh3d3de-N0oug', 'nZFcWvgPU8T4JCjOVvJU8Q', 'FDPjfyyHN
fy9tqyxo27QeQ', 'jIuc3nM9R0KWX3v-8OyLsQ', 'F1oGgK3NtTYnyRMxupiiuA', 'nGYaT5sAuqKkd48dqX_2Kw', 'NzP9OoNP8vBMRUSs0UXBNw', 'ZTs-go-xYRQOp3PS
w_XGDg', 'gc4rNAgbGydNMAPJ85FZag', 'KYuljD4Tovti3J1pdBnT4g'], 'fans': 69, 'average_stars': 4.14, 'type': 'user', 'compliments': {'profil
e': 8, 'cute': 15, 'funny': 11, 'plain': 25, 'writer': 9, 'note': 20, 'photos': 15, 'hot': 48, 'cool': 78, 'more': 3}, 'elite': [2005, 20
06]}, {'yelping_since': '2004-10', 'votes': {'funny': 6849, 'useful': 12642, 'cool': 9837}, 'review_count': 1233, 'name': 'Jeremy', 'user
_id': 'rpOyqD_893cqmDAtJLbdog', 'friends': ['18kPq7GPye-YQ3LyKyAZPw', '4U9kSBLuBDU391x6bxU-YA', 'fHtTaujcyKvXglE33Z5yIw', 'SIBCL7HBkrP4ll
olm4SC2A', '8J4IIYcqBlFch8T90N923A', 'ysYmC-ufbdmVEX9yAv-VEQ', 'UTS9XcT14H2ZscRIf0MYHQ', '1blidZhgxDVSBuJ_dbTU6g', 'HDQixQ-WZEV0LVPJlIGQe
Q', '91ovYNfGfPiqe6GbTcPwTg', '8VJUJbrfPYJC0vMAtXGR_g', 'T4kuUr_iJiywOPdyM7gTHQ', 'anTtulQ9G2b3SipCEwGVHg', 'Re447krbp0VQVX3Dk4Tsog', 'Xe
jvgD4CrkR7FeIkjwAjtQ', 'Ym2-9Rdeb_PPher2F6qksQ', 'i63u3SdbrLsP4FxiSKP0Zw', 'T7t5UdadpHua-YnmiFpcZw', 'pnrGw4ciBXJ6U5QB2m0F5g', 'ytjCBxosV
SqCOQ62c4KAxg', 'K4FAia2Iy5MVnmBLfS-mCg', 'o-6ZHG2iUO3oooMXQawd9g', 'r5uiIxwJ-I-oHBkNY2Ha3Q', 'niWoSKswEbooJC_M7HMbGw', 'kwoxiKMyoYjB1wTC
YAjYRg', 'F4sfffJ8-VCNFLUT0q6Gtg', 'TNJRTBrl0yjtpAACr1Bthg', 'W0M8csYKD2ryUsr6OsACyA', 'POQZrdj2otEnb0Le1oLWjg', 'HKvXA96cbqAawKCU7xyGS
w', 'EoTooNBcxu0Hy0-XeZI8aw', '9A8OuP6XwLwnNb9ov3_Ncw', '27MmRg8LfbZXNEHkEnKSdA', 'WsPgO3ycZjRp7wFh18JdMQ', 'Bn4sJUTtKFZQt0FKHF2Adw', 'AC

84oF7Ciuw7M52RszgWUA', 'uguXfIEpI65jSCH5MgUDgA', '6VZNGc2h2Bn-uyuEXgOt5g', 'AZl5xknQ2QzJGUfd0V_mjQ', 'pz97SxRe1Vk-5_K6EB9OSA', 'AZ8CTtwr-
4sGM2kZqF6qig', 'xaPwzbtkselbxI472WIjsA', '4ohJEFpIk99NGGNdSa2Uig', 'q1dyK6mo5QkHI0ZBPXeCQQ', 'xRpSYEOS4MN_fnfjBPy5dw', 'Gcb3QKjGjvH_GdzL
pM-fAg', 'uGmQ6ab4iVpWn5m61VFhkQ', 'GJYJX4SujVj3BR8v2F9PDQ', 'Eh9SSFnh7xOF8N3X6ESPOA', 'DglSIRnl69GUqHvVnG3QaA', '3shjifK-vZkIHciyy_KbY
A', 'xafscyoSZPnchPkzGyDAWw', '4lc_H2Cf7CO0tCgyA3aSVQ', 'Tunkp_F1R_uFBJQTsDxD4g', 'yAIA9J3FQmg6XwAfXMLh-g', 'IsVIVpgV7VralvYD4DEQYg', '-6
k_vUBna1kzXBjyBOCeqg', 'B9pKfr27czBbCoAIircZdQ', '7WGcHwBkXKRCTL2AG--3CA', 'ORXFUXpvHloPYVyQ4C5oqA', 'Rir-YRPPClKXDFQbc3BsVw', 'wj-UBmM6p
-TkfD4Xg_zY7A', 'G4LfUOgAglHfZLILESSb0w', '-1Q1s_NMGjBLBULA8z_npg', 'UDIR9Aojno4FzqYfOuwunQ', 'nw-QVHwrhhwZpVuR5DWdpg', 'qaeKxVt-KnFKKzKx
KVrKDA', 'uyf9AeK95dh4oxs36bMQkw', 'lxZSVeJz6KEBW1nlA3JKJg', 'zTWH9b_ItSdLOK9ypeFOIw', 'pePGMO6EbDpbaZ7D2m6HIg', 'MKkaAk3EZumQVvnMjd_av
Q', 'XRM8W6HUoXbrYKR3BCj9Rg', '7FuLnS_-b79GG-33mwLaMg', 'o5KEicAZaO72mHgSYVZ8bw', '8DqIWXsKXOipfduYEfFpNw', 'GZ9kx3IQe15JWrVGrAQZ8A', '7u
xXuCcpw9-mUS3OJVw8aQ', 'p64irD26M0-tgKj0z1jgKw', 'dvRVX54Z9f7Om51NsTRX1w', 'K2PDI4_GJLUTwjq0jA2viQ', 'WF4Bz2Z37AGdFZSVqOObOg', 'CM0saLQmk
4oAB17UmQTV-g', 'jIJXDhJ08iCh5KxZjIuRPQ', 'HANb8-8InWnju-XzBQQSBw', 'CvQhlr_VtwGqQBbiFOjOaw', 'frz1uzgKdxjOt8bWZCMmVQ', '-jXrmFgORGmiTEfx
aKlA6w', 'S6sAyi61H7ecAj6qfNMKNQ', 't51FxYOLqwbTSBeyPbHHVQ', 'e4M9_S-ASmRys3DvKQfotw', 'weLT87ID6D0yPWg5d3atcQ', 'OMWT-Z6OnJLcg44lCuDuh
w', 'd82F_FFtowYtjGtxRySehA', 'pW91HUnVz6ssLZ4dY-ztyQ', 'ojQYtstkGXtcryk5I9GTYA', 'n0vqc-DbRK84XOLKHp5wjw', '5StJwPvmlJj0yMHBj1PATw', '8w
23JX96mElm3ViwtS_WKQ', '4hxVql33ldY_gkf3rG8_3w', '9V14_mnEi28kNnN_aVhuSA', 'kxkSjuIHutMTS88y1-b_Ew', '0arQ82n4mvrc42U8KuoE6g', 'CvWhDNP2p
LVYdlHg_e4DXQ', 'WGf64PeXIZtdoqkBj0-_zA', 'YA6LcxRxXEyMN-jpFSBarA', 'ZZ5hV2jcxchiZo6C9AXq_Q', 'lDnXkR-Hc5FyPZ7T5oPyRQ', '3_VAhzONbrThOlZr
_6crpQ', 'NZeCINmoOJ8vsQvYkZAp9w', 'hfdh6M4M8cOqG4McE1aoYA', 'LroJKi7_BWqDT7lbHvt6kg', '0FqVxW9PVqh66vMbNJ4Y1A', 'eB_WpbBxBTABkzes-ZQTR
w', 'QBj2AL66bEAmK3ULkrn1Gw', 'ayZlWyV1M2k_UWx1dreIDg', '1--cOFkdo345uvepQwfGEA', 'hGyKkRaJkOpl6jagEZAWcg', 'JguVDOCjcNicW5zCwuqehg', 'ms
9P6qj-mKUGkBJRY0yAHA', 'I38kTTMpIQfm6CZQIeZKgg', 'mj2X5F2xh2xxPTaw2HsnGg', 'qq57IKwroZux85aC6hM38w', 'KUtzHIs-Y2Va3STDlWa0iA', 'Gi62agSW3
4oYrCcEdB9UGg', 'ZQf-UQL8ChAmTfN9SrUBfA', 'P-lfV8cdAgEOZJkw2dJpsw', '7Pef9EA21szXXOPJo6Gb5w', 'Qcn-UlswGM8JmrYbf4WMLA', 'Q4rDqPKe1Iko7Kld
j9ADBw', 'MfSPSNoF2iWWt7ZdCA8T8Q', '4RK2egxIKfC55TGg2ToulA', 'HHCn5JGkJc5Lp3EaGw6baA', 'hlyE3S6w-949MA-u8TXsjA', 'hDuhc9RQOf3ydg1PxLkRx
Q', 'ZQduSjrBcIYjO1d8IaXxVw', 'PuuRW29sMLIN0wuoZAllFA', 'KLIgcDSCque_hMrtjnUcow', '8dwO6Y8Vsss1hGEeXm1acw', 'Q1lOqbip5AspTHHBJ4VDlA', 'AR
e8Nr_YehB2ubsGJhZ-hg', 'oDYf8_HV9WpSxrLmJ1j5JA', '_YI2xCPY90YqTazh-xL_Pg', '-ETxy7f37BBQXhw5zJfkrQ', 'oUa5MYdA6FdEJO-HBB0DaA', 'jDCblyKGE
1UrulT7Dyzb0Q', '8wQ8fugCcUEMO7lXg8yDGQ', 'BDx8yOOvNE5UWs_xsIOXwA', 'Wwl_HlxMyfnPhH4iKbzdOA', '49pqdhT4su49QnwmXNdyIA', 'uZgFEPEUIWJMaaqv
MlxYoA', 'gvqh39oy-m10H7aNyVBL8g', 'YKppbE7ogNmbtVrluxEF4A', 'qB_WxOFVPKlK8DmbhPXGhw', 'Gwj-__sLYJUXO3S2ajHUDw', 'MofhftznXiuBxID1GNOSY
A', '9dmIMMQT6XJ19RSsWioU5Q', 'D7uXj1_0pE_Fa2YeLkOkyw', 'zzWUzYAj8SXwohidVLax9Q', 'eVV99D3EZvkuswIvIySKiw', 'PhwgVPqpJlxu40AxAOUBbg', 'Li
Aq4ejWXt9ooo9X5wHQRg', 'ajCb2CmRkd5V0OvzsjYC5w', 'CkY48UaWOmLhsdc_NTfFBA', 'Z78lp4o03tZbGRMwLZw0nA', 'J4xFxIXDH6nXdF5rTYiBCw', 'pbUs7oReA
S4lhMvEzwNVhA', 'AtFGH7Spf7QMNm-BFcshWQ', '_cDnFQHMrwo5Xg_1Z8QQNA', '-e0zsU6ootfv9ImTZ7wb2A', 'wpcoiQNbO5x0xkZwKFVUFw', 'eSAmN-5RAu8y1igS
vz_yJw', 'Upt8QGZF8LFWRSw3LUopIg', 'wOSwzy3p0cuwbv5xjZW1bA', 'Ot-QjYpIdVsluuxcqidjkA', 'E3C_gEhktux9Ca-Z2fmNTw', 'HpuonncxvaKSixI3bM5Of
w', 'mGSyq3buf9AggxuXuZRkyQ', 'ClaGLOr72DPJlQhRc9HHyQ', 'YX-O9N2rg1DNH-IbyD1GLw', 'wHr9izcKk8uLc6tucnXskQ', 'sc6wS6YOFIgtN2XuJI8_WQ', 'PM
gE5Yqv7QL_cTjjInyYIg', 'zdhIXcofp-N9WfC0tiQ5jg', '6HJzAxgP9XB-JhyjeOEFaA', 'tn5AIW1NVjTI4oNdrpqwiA', 'dThnQfCZCpbLXwqRqnQhFg', 'bUjVCdMEd
lHUZTj-uiBLUQ', 'yWeouzBJp_kwplPtwcl_ig', 'hO4UkN8RniS204O1PdxoBg', 'X1YVsT7fadktjP0WvraqtA', 'zlta0Zhv3qrlPT4wgmx1zQ', '0BxWPocgVIAzZUBQ
_Nrimw', 'Cz6cUmulaEii__wUcQ6QVw', 't1h9wl-Q4zGnt3JX860Ztw', '4YhwHzvNj-Rg8IhLntmhCw', 'OwcvQW8hbLjcU8eW-1xCIw', '-QjeoLnIeKStNfGIq5KGD
w', 'iR1pShxMP5Aiwo1DZzGJlg', 'Mg0G8_Wa_FoNQO3Kv7RDNw', 'G5PsRwFSjF4Sql7ME0ENsw', 'SuzSrMWoycf4DXL_DMMdNg', 'niQLnZVEz-E9KtbqFRKWcA', 'dN
BfuG6I4nadibbjFX7SiA', 'y0SRKlzVuvj-mbQtbsWx-A', '2hBQmM99bJWBc4TLXyFCxg', '-E6yYsngVZqIUHlX856sbw', 'hpAFdPtU6Am_agXJT856gw', 'n7WUEZeNK
CYgC950r3rbvA', 'BuaNITlWZwJskEhBbz-H6w', 'qbfQRHLvZk5WSkKY0l_lMw', '6kJ9OXgotJzYBdWWTOjpCg', '10O3dYXmF3R_OMcXBnTkWg', 'D6A0rn5MIg7AVRrV
eouABg', 'KUqpJ9eKREuvlnlitJJRYg', 'a0pG_Dl4sKb3yNps7I_hMQ', 'x7_Wyt9kjqn3itW9y0DHbA', 'tAAi5iUrQY1OKYnuChDJGQ', '5axKCCwwpcfPTzv7fS819
A', 'BgqGtOskxyKzP2oGR8uQ8g', 'dTyeHgCDwu9binCwjw7d5g', 'MMCiQSRLdhyUzkHyksST_w', 'YqXkPv4dittwSxctmZbqkw', 'gOvKWYXobX_Pgzsz-K8auA', '8q
oEnuFvMAQWv7qmhKyulw', '6JXh-O4gFMm-ugGad6PvQQ', 'brRr5SRQyzYa6S623Kvd5Q', 'b0bhYMBU9vk_smKjby-u_A', 'muSuq2Ep_PXjvLYXHNStig', '77s2qg6JO
qXy69stUku6ew', 'y4F30s5qp4VBInqmt4Upyg', 'WTvD6w7arH4S74rAlefC9A', '2176sNPv5tchjgJcV-2yiA', 'DyfA7aPzRQ81kJnbkuIiJQ', 'NhP0IQNpZkw6tQ0d
rfd-qQ', 'fA1-M_FoudCU9xxpauJBaw', 'tA2suyZG_qWIuNB_HR-YRA', 'C3edE64xBOfWC6HJbxRJxQ', 'PzYWzSKJ_HcwNKPD_f1cXw', 'VPon9k-yAGmeSw3sUfMyv
w', 'VJjlRu2Hg2VgaN1P2GVdbA', 'SCw9iirijegjhSfK26ZheQ', 'eeRhnURn8ebE84S1xymc7A', '9VedeYzgqzq7MXXOWhbQcg', '5T8SdBwiH0jdwatUJ7KFUg', 'OM
B8Y_DOaB21vCXa-cruwg', 'r6usm4GcaZbuB1iIbkJzSg', 'inEHrQz45dQ_bRCW7HxgQw', 'faoGu7ngJ2t4I9DO5cxG_w', 'eTeQp28_mMoImh9c4l2xxA', '1Gfn4nlqP
EIvRX4cnctJlg', 'PlQSyQNujQkd-ewRLZye_Q', 'aERWRgfZJGSuPoCdhCRUoQ', 'Q2e6q3_5DbF10ZKJCPEuOg', 'tgxgeEvYUA_Gq8Ekc5X1FA', 'ZbWP0L2Ni_icKOJb
oC4SbA', 'j22WZq4GjIhdp1cyGNhJrw', 'kygBLEEZ7lqLi6CnM7gLyw', 'E1zOJdKblkxFEfBGtX-hUw', 'Z7nr-LLKaPG7bwKuk1PWVQ', 'uePTLkka-CXvuoIk1d1Uc
w', 'oPD_t6BVpfhvu1-TS4FNzA', 'BbWSGazo6_fUCKxNLJ4Z2Q', 'FHaG51Fyfj8xq4CNsixWwQ', 'gn3Y0mWduWrvT1hvu7fv4A', 'pzHB2lJov4XgmfGgkh6N3w', 'cA
VLTaT4C5Zdz9lzTVYULQ', 'Bq-fS1NXfBJiaBfLB2vwEA', 'aPQuGzqs4D8my3K0vXjb8Q', 'AGHesuIAE2hJ4yPGZTzO0g', 'q6hiZCOlvZ-JWY5alt4aiw', 'NUeLSLuki
PPCSK6EVdUo_Q', 'rtrh46NJfksxR4TgljYg9w', '52hYzmoL4yUFyc9Ct1Zukg', '94N07GeXPUyC8baIz0l3UA', '2Qo8EXnxxfTtUWIqPPih6g', 'DrKQzBFAvxhyjLgb
PSW2Qw', 'v6giEKBrh7uLR3npuqda9Q', 'DH3X4Qj14nIsdeHMFf4xSw', 'wZtfgvBJ7VihuRyq9Nqepw', 'SwqhvXTAVUrGMxQBlyn8vQ', '8phGuOZ0NIfeJiTX-ZDZL
A', 'dAcYRfvKOVw94_U4gyz8cw', 'TiftBBMFaQIMK-vJaRbZgw', 'XBHJCDzNh--mBhJkzyM_iA', '_YNKf7vrI08r2eI0tM6zqw', 'xUhW65cmXuAjIaVVpZHpTw', 'HH
ij7LRNfzWru9TKEIsWgw', '3Fvga8TuSEOdsN_xYQZR3g', '8L1p_zlmAB_uWyaW0QTaCg', 'g0LFQoGn9TNDpKlFjmFWwQ', 'u3xXsUp-01Tw6IF5eqHbbA', 'TFD4dqa9B
XvDbtbDx8JwuQ', 'E1FiD1I3K7YEVRd00EkPsw', 'BSI42r_Q2NmDeYU-rf12nA', 'DzTMG-LBSfMeFsVdkzoKVg', 'KMGF6KAnPRoOTFVXBy9jtQ', 'PFtbOIt89VdJB6OL
LNaEGw', 'UbLmrkrIga-o2v1IS8mLHA', 'M35ddM0p0VtAb-n3DZtoQQ', 'm6PdLs4FnQn4MHFd7i_wsg', 'rp82QG3ewXv0XCeXqT2ojA', 'Q9n39MgkWbvf6ohzQCDO7

Q', '77NzO7uhl59ZK-EYoi7n9g', 'at9rwt_tidIDDHfFeLFF8w', 'htatikeBPPYOmbiPP0lXbw', 'xQWD-K9GJKt25oFV0VXcSQ', 'A69Q7QPRXs4rvbbpuDsm6g', 'O_
mvwBwSz8KzZkozRPywdA', 'pMlK6piV8k45lhMTRomV5g', 'gOsppOXgtB8S7Cmsqc8Skg', 'i5FE5KHUWxsTgaOkmnyBnw', 'vyfsQo-estP8EfiIFMsL6g', 'NOBR1-mzf
rvDJJEMRxc8KA', 'n7vjwXPlfG9kmC89IhFiWw', 'AufULLEHyH35-1Px95f97Q', 'ErsBXW-E-mWNjMc62TPQNw', 'JM71F4zQwVDDSzqSkiVUhA', 'RiHqU26mvzK1kRhg
5KfzWw', 'ctfrZmE6YZ9veDGcQSbfuw', 'p55Kb76HHeATutpPN68sdg', 'xbSPOPOq57wxCxzg3eJG7A', 'KLOs8RAHj14iuM70dMfUGQ', '6uM5lx0DCX4ctQlebtxra
w', 'itXz3aLcWHzw5ccZXpXHjw', 'Hc6agWZ9jGCxsPAqQwlf2w', 'TK_fTr_VHwS2Ec2ZxUUKLQ', '1-DcXw8_o_oibpBlnXgdEg', 'z39ulrgtvEMB2oeokx0-mw', 'xp
MHUEbvHet50XDg98fJOQ', 'YiuRV_Xy8lpLndMOWi0uAQ', 'lL0oNTVtsFhkGwILqmiYoQ', 'wLt28qh-46LFhKOeByWZ0w', 'R9FFFqYOUmvWjaHqODfzyw', 'p4Jg_rO_A
sCWZGMd_w8xjw', 'yyvwMoMifs1p1cp9B68XIQ', 'MTVsyoL3dajP7VS4iuLP4Q', 'pJppmQyW74ejj8xCa2SHuA', '5MvIg39L6FmBrGZG5C2jjA', 'hzaqnSxfRYv_Wbp6
CMKtQw', 'VFTd1QaNvVVucagkq8ZJAw', '0QwV8PJk9PJDFaMNw1Fcsw', 'lGrDfsaWl9IDUcEeUWyn6A', 'yTAOBwi29nTPnhu9wH75Cw', '6pC5j-hfP4xLZB2W0Wi1v
g', 'JcJwC-NxaVJ6xUTYpC9AMw', '8nAj60xsBjcHXLLSs7cPqw', '7WotaQG7M4JJqaqqDraaMQ', 'wiE8owjhxXYpOxCcYKhbiA', 'wNTZMYPQRZAx645BuhdTyQ', 'sA
A1n6_M6qllKlT-XuWkQA', '7u4J3l42DPIK6ghUu7fs5A', 'qnnieCU8EXZyojWuKg1iVQ', '5h_P5sVbGxtw9ldH4sWZkw', 'HsKPpfcATrQ5Bb0ab9VQNg', '3uPesra2s
EBRrO8kDwO1qA', 'Hm6495EXdwjKqhzvhLgsyg', '5Szwd1CSepRXwAFH12nPzQ', 'GoTkP8VOSk6rqGXS3uQ-bQ', '8S-UrKLKRsJhv6mfpCV7lQ', 'WMA-mhjrNJVZSkMo
BD5ZWA', '4ulkvNf-T_UVM-e4dd0lbw', '-T4sYhtArnj2XNj7xoEeiQ', 'sU9pCk9IvfZtMd0AmrRhvg', 'gcCs-oFd0veNsb10nX_fmA', 'gt03nuYv2BJxuq20d7OGk
w', 'ZH6rqgDyfVEb5yZxRD63-A', 'ZR232ZiiDJ3j3dCyMygA2w', 'JReQtKy2yfrwPXl8efFBKA', 'd54DWwCsmiZLQ3rXfTQldw', 'uBrI2sD8oEHegtiz6g3k2A', 'KW
BbXYiNvwGjEFrhitRwPw', 'sl44rRFiL-my03C0oywKmA', 'AonCa5s2nOfNfZYecxgIDA', 'HBFtzjRarXxDH8xZRSNsPA', 'DVANCbItlJLt3Eqi7CaciQ', '-EAx_2wa-
h4YyNzYyWFfVg', 'nb2wZQ-DL7Bn72N_h9NMXw', 'qjBt7pr7CcQsX7040RYYsA', 'n4QTIJq0uXVcEls_No9DyQ', '__AzEBHrVp19B5RQUUg0vQ', 'zch7L6wKTPUQI2sV
MxUqcg', 'kXuA2ffm0Uw46U3O0mcmtQ', 'zi0kxtr5gq0Flg7jYdIlMA', 'UkYIuhD8kw2y-h1BHNo44A', 'oWEdmTYl13VHh1A_Mkhh2g', 'Y-9yQMPlnRWgD40xAHl1K
A', 'BJUXMLNyQKjQq57BDWy1Ow', 'qFHOh1DMqJ2Zb_pvqR1Muw', 'PO8XNxGUizzzKGHmn24fQQ', 'LdjjwHSHCeqSvm7a73wmXA', '46ASXV3uv_x2QAWe6ocplQ', 'Dz
VKlwlnqwSSiGGzLmAp_g', 'ztpTP8QJIR3wqj-xCw6Mow', '3Np7xOb7SFhgAUbdtvxOyA', 'mxDC5k-GFhdqZLttPM1dZg', 'AzV-8kuqyXJqRc-MNOmOlw', 'Zs5TpdcmC
AFs3SKsftnX6Q', 'KTO2pnfnbqclrNm0HqoyOQ', 'N9yGtVv_uvWmxt9XY7zPUg', 'DB0OfzpdQQw-Q-8udINCSQ', 'gzvltlt2n97tZMnEXihbBg', 'HKHAr2pPrN8rVzUd
YXxVZA', 'XpD6HNw4LoGDBCGkMqxrnw', 'WgnmE0dd3lB71lRP_OdyBw', 'DDxueGylgwBGa7-hR2toZA', 'szLodDvllHAwYqkUVopqHA', 'YYz9sM0_98ksQJUeHs-SJ
w', 'ac-LWMXfGE0ZMChLhYqnnA', 'tXxlJ-tDbFn-52sT8WOdSA', 'Tj8X3qpdjiJpFAEqWILPEg', 'JTRxzA1AZ8Zfwj7FpH2CnA', 'AixLPxra_4HirfchjPkiWA', '_h
yY8DGw9kNpdXLVSn8OeA', 'xOK3i0wgmkpGu54BuIcwQA', 'qm24FjqzuhQqAWGm7VRSYA', 'LCEAg894v5-08hx6av0p4A', '2YHVTrs6EY7Me9NwxiQVpQ', 'BpmYi6K_0
jXj0gO4_GmVHw', 'fPHLPrymsyb6WSFFKoMrTQ', 'F9T6m1YdRFreyKDufcyoOQ', 'eib_gPmmdNMZ5LLZs18p8w', 'kGgAARL2UmvCcTRfiscjug', 'LmsezECOd8IqQUYj
TzWNSg', 'DVZf6SX19FVnzPyJeZ9-eg', 'fd68FgmkzCi_Tud41ex2dQ', 'rX_t4oJNwMTLMleZ7s4ztA', 'wJuqi0qRDVOjSkyWhf4PIg', '1UJbOl-jKNIUjroeC4xWx
g', 'J_uRQSKIxN0MCItOnJzfBA', 'ACLuEsNHV7lfq1O7CHuFaA', '63rmu8axehcaFxQ3TyIFkQ', 'zhV2H02tC04MGbgch5sqNw', 'DJEnhCkQlQ2E2z--aO9j0A', 'Rh
b81Ivo0XQOXuE5-5JPQw', '0ypTVVwcyusnxx4Zbws_bg', 'UyOtYjnavsr-Dtjo6Xy91g', 'EFg-ew3x8j2XPJn4e5t45g', 'IGkoathSlJmIOBDGo9vUZg', '4cI8eBbkY
J5vBaDSZ00RFA', 'FRBXU5mAa9MCAXLCjBTi4g', 'J3wHuqConRziCBkqCxY4UA', 'Lw2ix3qoY7V15se33597Zg', 'k0jwdPwmeR3QXHICqSVL-g', 'HHbe1svajOb3FOXd
oAoGeA', 'eNrtdrGazNbdiQRmOr5EXw', 'n9g51fYo-4mjkvu6wXLrcQ', '3AIZkvXQaThvVODrL8jEeA', 'J5ptmoQJiHzm-XqyNW5d-A', 'gLkQlGtokKIyWyNG040Jv
A', 'US2OXYlR4EvFQyTSKSNMnQ', 'ECGa0INRH7Tr3qGDU8I60A', 'ygronrTGPlQHxUn8NCmaIw', 'd8cenZMDAAUzEtfbIyfwrw', 'wWSRhqRAez5TF6ZWR4od6g', 'C8
ZTiwa7qWoPSMIivTeSfw', 'NgS8s99qQIiHOszKd-m87g', '71vvbcu0wgwu8VcJazSZZA', '9LZP7Ll7N9zkuWfA5DUAYw', 'B6jSXFsQkKyoh9E2u2ynfg', 'vEhOUPzqO
4KJL2dF32I8Qg', 'SNBJb9xiaRJ0ohHhugwn_g', 'x73VgCg2HVjo_SUDBQ4plg', '8W6N_Vj1trKUvtPya9h1xw', 'SbM1m5nejHDD1pJePsar9g', 'N50csIgIytyVZ8VU
sHxZug', 'u6sai1T3fSXsyF-fnMToPQ', 'zQ_FpDX-GDPuai4GOldCJQ', 'bLjKUuAvdx7wbp6arixUfg', 'wF7_-zf7R8Rri7Kq-a0tlw', 'LkP6rtLtC26GrKtgR-COh
Q', '_1mBBkphbaKO3EQHnpBQJA', 'nN6rr9Z6DRYg2J4mXagWZA', 'PNcKplCkU7150DmX0aDylA', 'vrU_kYfmp0Mmh2S5-FY-sg', '2iqG5g2iP83GvtBiqMRQoA', '_2
TQOwIQMaWPjjY0bZxg8g', '_HcZLN9uoS8urIDGyzew8g', 'pXhkHt7mAqhrKLebB6Um8g', 'mAuBHENS_DeFCQqrJC_UxA', 'qbiZyhbuhTMENzs4XH-gzg', '0ZY6rM7Fm
9AbHfG40HCLfw', 'IPcbHkP9eB20ur2oKsLxlw', '-Ll2lk5MuVsJIBSBBnM3Tg', 'TF0wvOmYBOCRYInIlDnulg', 'jlpbR7IurKHCnSrGxELVlg', 'h2j2B7Jif_0lW0H6
FgqGow', 'CLz9JgzYmyGNP5p5EERRdg', 'uCWHw0AcKwTjvxDl8RpHKQ', 'WYWjFF01WzwQNL_whGEkKQ', 'IrmsjAqxw7UgZfA9imDfEA', '3P9Wnn0Z9TSe08bA8IIh_
A', 'N5JPIH_0fUge9suPoyp-9A', 'OhqXsFeqCVVnxbWuZlnGvg', 'oPG5VgtbV0DaDqYjBIgyXw', 'J3NDKW-yjdDXkboig0hAHw', 'WmAyExqSWoiYZ5XEqpk_Uw', 'm7
9wRpkxwlxqCwd5e37vWw', 'RpC1jwOttGJcM5TIgscMTQ', 'mhFF84q9CZbmb5PqKzfGww', 'JqaFfOsOFM_nx-dpKnFp8A', '_iXng1V68InwgFaiy-50eg', 'tlSSQwfHY
Jany7wPoTH46A', 'FBheGN7IBrFkGSNdv3cbGQ', '7m4-lgi0etB6KvIQ2RBBCw', '9BApDAgV8POMQ6VffTz3aA', 'y9NvtHUzizvWFYSZ5FV5_g', '2VfhMrTJ9bZ0O4PZ
mRmxKA', '5pAk-cfZQy2CuhdZrqYCEg', 'TMsX0mrf_wQaxcZXKgIh5g', 'yueiAo4E5vtAqWaqSyVo4g', 'C9tCsaYVHdMcQeACECXDCA', 'bxEeVkcFEkaUn4cfgDGS-
A', 'hQvmGmxWOhoVB7hbDRQb0g', '-2pjspSYwR_0mtJmQcxzQw', 'kCfwfPSOiC59qMu5EkOggA', 'Wnpi3iUqULVG8GG3fdXWMQ', 'u2mMLkTFO4qEEL9QFwUjRg', 'YD
2trEiFhj_ehUNCpFtfsA', 'GoqZDWwCvsgwiJRZJ27CiQ', '-heP_e4m3mEiI75VozrGwA', 'eFuJA-G5a_j4j3qB9p2UCQ', 'mFOZOsPQOacWIMVSyXbEbg', 'wgVQ-y7Q7
ymQ25VVIgZEmw', 'qK6_sysqplVrEkq-MxkpTw', 'A4DM2Piz-hXs-o9X2dSQZw', 'Gf-2KwbHuIzp2YdZXvC3sQ', 'YcmbgKpGDhMu4w2AWuIgWA', '2XCsVpAMsGUykrM-
BBFEXA', 'qVePixO2kuK0Dr03upCWFg', '-ANkfLbDf8aiBQ7vywIL6w', 'u2lanPBcORxbwJzsEbAExg', 'HQJ_2pzqz9DuPHNpTOmrWQ', 'D9CIW04wrcQzZNdCEoASA
g', 'LNvw5VDA9b2PXkD-zn3-OQ', 'Nu0oythh2v7FSfJb30d12g', 'Aq_fbQ4aMVvTCiskOz876Q', 'A0puRx8MGQsnk8uXtpw4hg', 'TLj3XaclA7V4ldJ5yNP-9Q', '0X
RFyY8-dxmrDheFPpe8Sg', '8E0DGec8LNn6oDmPHmj-mg', 'LQWlBt00QZrvn7McdS_Z_g', 'y8ZCNq8HSGDSx7Vm-NYEOw', '270isWtFFYFtlBR5RZUjtw', 'NGaa4JBPX
l8GzdDWWbxb7Q', 'fMzU7cCSgv44IKN5Crs1mQ', '3utEqyNzFcxH8E8pm7ByPA', '7SVxpEnu760QstKGa3x2Mw', 'zRiz9k2brNAW-AgM_TLk6Q', 'LRBeqLo1FhG_9-Wt
m1irGw', 'B4KTa9To2jrqr-NK7c7kwA', 'fczQCSmaWF78toLEmb0Zsw', 'ZNpAVHTehFYp0rObGT4teQ', '_VtoKtLvDAjWttQ9wVD5kw', 'US2UWum3MaQLtFGiONNIv
A', '1F8RulRCjaQTN61TX9Hd7A', '6hRKotVFPYL7Ci0hipN9Eg', '1krwpAcRTPy6lrjAGQQ1lg', 'rvY6h_FqTuB7YCu_TrWRkQ', 'UUfnmyEBi60Qw9tjuQ-1sQ', 'Nh
XOaROBjlU-IuQrmWrQYQ', 'WeX6UjZSTeMfFBQGDfu1jw', 'yPy13LY2idzQ_VpoavbKuw', 'RTwuP9OtmJvMm91wuzxGSQ', 'AGOmxrO-YUnltljd0SI08g', 'q9XgOylNs
SbqZqF_SO3-OQ', 'XKilChOBIMq0hp2kF8KwHw', 'aYpO4xX9yWRGfF09Co4SXw', 'bnVjfyPIsh1dQ8iC1zXNkw', '44R1syuo0et8sLtRnCojkw', '6o0NNvPxXHvNSs3N

T0WkJQ', '8N6y30aGVwtIYFraswbJeA', 'LdrGYCwF8-mc272RdBQSaw', 'D1HMFu5IMz2n49N8ga-IVw', 'o_LCYay4uo5N4eq3U5pbrQ', '1cR5-LZcSYcACxhGe3DEV
A', 'yoQAZWDdBW3kQH7geFEoWA', 'BYFWt4p_KLgINKvSqpEojg', 'y2zwsT-arc0H2rR5yKR2XQ', '6wveyznYgxA0sPKEfe4wOQ', 'dwrLwOylx8KvPers5hAahg', 'Ml
cL_KuqYyvyo3U00MAVwA', '4NX8crn5WLSVjWH1fhLS3A', 'nWry09tpTwD9peA3ERhMJQ', '0Ypfll3kJiL5OrxBBat_Qw', 'nFO4MFFynxIiO1mOY9W8sQ', 'sjG0UdiJa
Ewin_pvWuyxVg', 'BSTOKq5iEIuwzbe-qNzbkQ', 'VLWiX2gzkfDwNAR3fwjtlA', 'h2lZY6mojN2ENQdU9qazGA', 'h_IuHfAeOzE8KT5FT04PVA', 'ZWi5gzaMROxG3oAK
ikHFEg', 'AX5N-yKV0w-lVNTHyb9AXg', 'fGMaltkx1S55k3GpeLpTyQ', '08vW8jwsZcNXc9k1_H7H4Q', '2z9yZ6yWiRlZfF_Lshc6Zg', 'zunSYtFT1h20xnQH_mTsZ
w', 'CJCvh6WSLd4Rx-VnQGsSuw', 'pSOpnpuqHaT0gM9aizQsmw', '5YySKNsPmTeG69nT-dS6Hg', '0dTkjUHgc6OEsSk8JnRzsw', 'S8ZY1HEDoECqKrIUY34JOg', 'eu
adoc7gbju3zFjB37x3Cw', 'kJ44AlQeQIaqbtoGHAQeNQ', 'mTmHkMOdv-VQdE8VCJ8GlQ', 'nQVJ_9f7xVtlozrFu4xmqg', '0bNXP9quoJEgyVZu9ipGgQ', '3JtdBvnY6
Wzp22gt5zFwDg', '1VyqxZ4LawfHiRghQryVWw', 'FEDX7jt7NmAusbDUTXdArQ', '3WsDt1vFkLyCQqb_qUtFVw', 'i2u6i8HWiVvMjcG5nwTAEA', '470dZswdTHz5aaGP
vIYNpg', '19lLsmz8dx97X1APkVNlMQ', 'JT5kM6wRwh7cNt8IAuX_sw', 'cp9eULl_Nn8Mo5hC9obSOQ', 'cbc2PFuBzBAg7TXRPgXwtg', 'xRf8sUczJqHgfgA_Dns9g
Q', '25uBXnNZgHDaovC1VTENrQ', 'Z55xeJuUqA20PL2oY4nMcA', '_E9-6-AYmN0V-l6HsQk7hQ', '0qIsBt4EzBDCKrIviV55Ew', 'lLZfboZB3B0LquJNR6GmkA', 'ca
_Q9OGfyjwOnfvXqmhdIg', 'GrIpU3rvhW1Ydi5DlKQQjQ', 'd42T37_3iMNRrIbi1gq2_Q', 'f5gM-S0atZaG7tsW1pi2NA', 'v1se4CVd74gBqTc0iMcK0A', 'TQ5eVGOFr
_qB5_BbEd3Sow', '_pV81WTJRyppkqj9B7GF2w', 'r7V9Nhes-Eg-lH0cqfpFFw', '8ma1-FGZD63wjmrxyeajUg', 'whkzCRr6zbjjA3O0tVEKow', '8N481iGgsUYA1-Tt
ESOntw', '2nM6phE0Emudtw-feBLWtg', 'enVSF2CIt_vMmsq665LSSQ', 'sdH7B1S-PSngGY5Hi44OqQ', 'DPwh-CgEIleGiZIe0Wqo6g', 'ZTC-ln1GjUB44g7wtfJVH
w', 'ciIdgFaC_gSBHtkU2m7ETQ', '97rWQ31yB_JnjEDIxUI3Og', 'cAsYfm9UkDFDjPRLR5zfWg', 'uF4lwuRTFvh-qrbIDVW32g', 'PMNbT3Y8KDt7E979AJMyng', 'Bp
IXi537n43BFg8Q64a-3A', 'x6HA_v9WhLKMAsRm1n12sA', 'LJ3nfy4DLLfLy6wA8-64Qg', 'Ig7yd58fPKZYFhH2bDYDQQ', 'x8Xx1-_hbjkuH8xXV-BrwA', '7TRprrHXk
VpjhE5WFggkWg', '4qe2MoR7r_UyaDrGe-20AQ', '2m3lE8p8T2nFRDxIUaRXew', 'gxoqD1Y9lRs2CFZMUvxbxw', 'CEfUkE9nqGsJeIpFgAv1yw', 'DzY5lpsa9tBcJtoR
bzCEPQ', 'AaBwzccT3c7vAZUQS4qZeA', '0qfftO54idymAc-6LPVTbA', 'VkOKJOX1gqarNssLxg9LOQ', 'J92aUGPKStpBIwx__yncfA', '-MZOtQxxYX9CljlnHbqWp
A', 'bZFRqP7s0Vszxeu8_IwYow', 'snd45oedwXxF9tsHUX_6dg', '8jEK8gxHYg9w6sINujO0MQ', '3AWY6ilx4acBR08xNJnWew', '8JC-Yb3UDUv2FUl5ym1nVg', '8U
BBkdxYDCnSsOVc1YOEcg', 'qu3uqXJvrQJSqz1XHcN0zw', 'hMMVN5kSxReODk2ofxic7Q', 'CReqkya7EKcp5G_PIBgQrg', 'HW-cL93tvU7qng1sZueKWw', 'Dg560aU9y
mlloG-TOPx0Vg', 'ZmSKVX7q0GVXxmVFUcqGDQ', 'kYi1-z_eErTLozHG9gBYGw', 'ehfZRCVBEFYQzAYx7NLOHw', 'bvQw4Bp68zR7r_-aLmgq9g', 'Ow2FFWLAHDIM3Lp3
q2vwBQ', '2hK0GUNB76HeH3nOUUMesw', 'ZVyQudI56A_5hg-2B_wjbQ', 'uE7jv7qItkIMCCItQRSIbQ', 'yhByTmJlswtYvSq9qKrskQ', 'LTWY2Ee-SYu8ItVOebD3C
Q', 'f0_QSzyQUvuuWpmJCVv_WA', '6nK8JYOfMJBZ2lpN2S0-cQ', 'SeUINpGxGUr9pd7gWACU4A', 'frLdxQKCtofva9b0dA8Eqg', 'qiEqWWU2CMpOHr1tCIdIVg', 'uH
q9T4xEvdldiRI5k1WidQ', '_A_2A-bjADJTKBoWd_B5qg', 'pkXq4ZbKLRlGlMiEJzm0Sg', 'k0iLX341x3mm9golh9wejQ', 'u7cUruVV6_B7bisu-SYIKQ', '7zpDhrRZR
TGCkAh3SHbEww', '50ofEHGBhs1HGQYvCRnFOQ', 'xBvqbRdQlIJcw1DT5Hz_kQ', 'GO11d7wyai9DVtHGV52OcQ', 'qbVAB3Xxxtc_OzsDNCmUBg', 'ItqL2488GCmuFcKx
HDF7Nw', 'DFr0LYHHTokLFCSniznohQ', 'uQgfF-fHN4vhH5sx2W4jaQ', 'NSg7k2UFt_HqM2mvsw-n0w', 'hMhGD_Z6AkG9jC4URiHByA', '5yBkoRI9DSKH9aMV3Vadp
w', 'li148FKNnjlwmC5Mq-qFXg', 'O2Sepw53kNHuf1VqJ85cLg', '1cuKOGuLDfD240LInAlbwA', 'DFlMzHxJOQ2yggfYqnORoA', 'dS02pxg9FMgcJoQEohsMCA', 'Ec
FydWvO9D9jCBUgLSMzKA', 'egdj4-wXAzYzT5bmTAHYqQ', 'VESaDn1MpEIAC7C-SvGO2w', '7z6uuJ3fuaurG4dmdGidJw', 'MNOW-LkZx_Ns7AdTyFIYYA', 'nUHHk3cmr
4ia7u36YMZTtw', 'Am0M-nmJZ9fIU-NBFCzSSQ', 'Hv-U6M-gVXf8jqzgWbYLXA', 'PmLa7Pu9Ba32fhvCOLOr5A', 'C6IOtaaYdLIT5fWd7ZYIuA', 'ToNSCfbIbW0zKKZU
l4cpHA', 'AXHAksv-Y84Ww5uVw8RdHA', '4J4J72Xr67GJYxxGt8rHwA', 'eg0AARmcCCsnyS1WtokWkg', 'EgXJoFdQlcokmIiqgMXd1w', '4CJGQajGdc8JwFneADetc
Q', '6UUlGUc_VjcFslmRle1vPg', '-egXQfAzhRvq3BAgug10fA', 'BqdywQzVmBdjymq7BfdnmA', 'RPn0Avnq-1syMwEYz7kPrQ', 'GTxtOC-bNmfnBrCfRUC9Mg', 'GM
SSg-Ts1XfWvy5kGHTZlw', '6cZuyvWcccCrJeqWF5WPoA', '3x175sXzgadiYMlLErUMoA', 'i1iaLhsO26OefrP-6Y41WA', 'Hdb6HUDha7XG9X5NnmitgQ', 'qYGB7R1vc
O2rDJOeJNDfXQ', 'qaBXZe4O96SDjnxyl6xmuw', '2t9wSXQ3Luadi_eW7O6Tag', 'S3DqpJzGe0iZtDVdgrLRlw', '7h7IlYzn2j2b7AH-801EtQ', 'ec5NsXiWB0C7yeox
VlhxKQ', 'KQonJ09FfP7SA-ChNDgXdg', 'yfQwWLYtqwHKiky1hiTBew', 'lmDxv2cNKO_bKrD25-jAGg', 'xm_DzmDnhtNyNnnxhSDncw', 'mDqWHJG-NQq9ycwV07wjk
g', '3OxcNCuHxHUSEToYXo4LIA', 'AKqC8waFcFS7iYhL-02pgw', 'uo5LWFJTOsla9j4a5fMiig', 'VSRmWUjNkSSeV7PLBOuh1A', 'ltYDFHSVSc_ztIiJrndmeQ', 'gG
FthU9GCGkcSLNo1UyNDQ', 'pqlRaKvhnyrNoxT8yeadWg', '3gBhA4bNoPWQYeONL5NPQQ', '6SxnRYFR3GUzGREKSttxJg', 'Tl7FatOzF33ciAcA1GdTcQ', 'Pponzfdq5
fYm5Za7Oj1BAw', 'xt8iR1kTQKxbBc6tvYZxqg', 'ru8p3RTlk8f9LB_3zLXURQ', 'WZgoM8UuzxjZf1AaWx3qZQ', '086zTSYPSTZs0220Teh5vA', 'GUrPmYze-oF6NFDJ
usiSNw', 'eqi3jBRRvCGQ6zpXkzlJYw', 'q2M1kDqxwSlQxKMyPX7Ttw', 'Prsk7SiPZNfgcPyygOdGHg', '49r9640dX3NPnq3bQiah2A', 'XBdkBQD5GkyCFwLnEIhEJ
A', 'dPcoYTCV53X9Wy-n2lxWhg', 's74Gi6-mS5J5WnwGHKORfg', 'T0vVdZebDAsYN-UttJKJow', 'mRncQJkKWVALsflOVvO2kg', 'ChN9DVAAoNHUY6JljBcPqg', 'eB
UE5o2msLFOjJvFLm17Ig', 'fk9sG-0ayVV7dGsar3vy0g', 'iDs0mr1LAb7m3K2L-KcAAA', '0U9fwlPYRV5cy9G9mb6Q6A', 'zI0j9SN1aYoQSPOlL4E4rA', 'QO3DkQtPK
gZu6G4MXPmnYQ', 'f7Ez84QWvsJPXwVpG9-AjQ', 'FEkrEE6gKYudG4WC2gXPVw', 'eDv4o8Pg3dNLZwX5prO1aA', 'saiZmxzBIowJbMgPBW1Now', 'cJmDizVwKXChEXRr
NaCUDQ', 'VQn4csaGtX0mhr-VaZzREg', 'CQVYCo4OP_TbnZ3DFMi2Kw', 'olT2J66OB2HFlRymvqkxZA', 'Q8KFcuPwUh2v6BMTyFFyBw', 'GiMKnA1NjkMgLYipd-qNS
Q', 'DfxlZ3t6CxbVyKMA-JUSmA', 'IWBD0ipfmAZVKYaUUArIoA', 'RnrVGMWolNXYWzgHg1hyVg', 'aYsjSXX3m6UQUPcVk0yiBw', 'UTGqubcmDv1cNe5-k8-OeA', '10
nYcaEv8zxiPumJECLsDg', 'gBID8jIwsQIE5W_Zr0Vhjg', '2W2kRe_ewEBz_jCGmfdMIw', 'yj4lNvmxAt42NvBCxB3WTw', 'E-E50dUzNkeCGNLqXQ4PpQ', 'LbgQK5B_5
IkN77FgRJHhrg', '_3f44aWnibPGTpmiiNLxQw', '9zW8DTJuCNAN-Y2XqG-xZA', 'vhxh6yKNWA5IJntxjREqMg', 'it23EnutJ6fOjeYQBDKJug', 'd1w5vqi4Dz-Gc-vF
efpOTQ', 'FaX9JLeacAteTlNKgeGNRw', 'cYJyOG1ea1gYhsR_xWUNvg', 'Uf_PrKEpYGWBsL81lY6NSA', 'KqaH0TMidJb6AMAeLXmu4A', 'j5f7ERREO65Qvcrs7KQTw
Q', '0wUw3vODMYJ9IM58ND5gAA', 'l68a-NyKoVJeoMVGk0PMTw', '4mtCpNizYNdXuJk5IpP5JA', 'r7KlsQ0TYCYCBH1s-b2wnQ', 'g3dUh9lpWTVIKxzNmkFp4w', 'UZ
ysWMlFjuMWGTC2qYmIZw', '0IAOkW3KD1Dsx2hnwb0CSA', 'u00EOkApqyRP9YLlI90TKA', 'i2KwDZV-Bs_BrZjGaBn4YA', 'jVQoCoLm8JHqbQI5jrB0LA', 'k5zGEHC3C
7AC6gxIGOdSmw', 'xHAlPHFluxyPWnH95KSJLA', 'Doa1UjBxGVTmcvTxjEDHUQ', 'kl0zgp60RM09pySxA5NubA', '1XWzXgFvcu5pU2wj1ONQ_Q', 'w7YkAgJ2t5Q9zkfq
58tpJg', 'AP6udBIStvWLY0T5rS_24A', 'wNqwKWaRjClmcKsoJJRFow', 'q7Zn9s5wXAjhhSJMf7GiVg', 'V0WOxlZrIhBjcvlpszAgXg', '-UP82TPPpG_NpcsbDbBlS
w', 'G_PCzMHcOLoMyyEsxW1DFQ', '1I3vCdFcsSW3KAppt2sXoQ', 'ZuQ87LQYWRw3AHj-jEiQ1Q', 'j95HloXgQiHWDdkkHwytlQ', '558AjR8esTKOKNAUvwlhxw', '1t
QZrkIweF1iv2RrU_dxJg', 'mKBqGkfk7NqwUoA9P77Qig', '687b_5uUBlitPUac9T_u2g', '5-ZF673BNm9aY6VQQtc97g', 'tQV5mTcDx5r-3JS4BpByaA', 'IHUXwBK3p

```
jsCxzuS_g9jCw', 'ZZ43etAB2n_T53YBYtf8Dw', 'iJvrW5HPprsTlsABAdYeGQ', 'XPGCWhBzJM0bQT-f6wzfRQ', 'z9JL8bX2OkDrigPqXpyERg', 'wwdRyRFoB-zBjXmJ
5ClKrg', '50EcfFlkuvc_T6kf9KbZFA', 'VDHUtEnPh54OJiYow91d3A', 'lsYq1eKzU0X77LN6PfH1Ng', 'HGWIasjq8h6GGNPRXhn3cw', 'sygEVJoT1zWGgv_OjTnGb
A', '-B32HlBjDRafJpD_eW5pxw', 'Ig-WArvZ39LcRvH0kidHNg', 'r8xfTpifZ7gP5Cd5CSQaIQ', 'VVvmWzmjmkasz0AqwQHSEg', 'xAKoiSjFt_k9tWh9RIsj3g', 'aT
i0NVrcPJWbN6jAsJVcAw', 'RoSIhgbHiQFzsgavFFaaeQ', 'ZnOdDbndR0b5p6eF-014vA', 'hU0Vgl_hkJaWM5uUq_9W-g', 'h2xSlFE8yfPn4DMD-3hbxg', '7_Tx5bj6Y
lf90j328qH44A', 'QNlNjDHxj1uonlkxuaJVOg', 'ToJ8ZZITRnU2NoBTP80nHw', 'QmXXmqX7Fdszdd3qDq3-Jw', 'd1B7lZdPgF1M61E1zROjqQ', 'weYT-xJwz8o7mLNW
IhD6HA', 'fwsJGulnozT2U6FefsLiFw', 'KHxnsmfpt_Oo_0iqfToZuA', 'r0wUI4JVDBw9WyRimmyVMg', 'agBpuKvBxrgTUTOydISZRg', 'WvCV79VfjM3rLWqH-k_Lf
g', 'aXwa1hVyCgYn3PUlrkadyA', 'NkNs-f6BzQ2f2m2hJAAc0A', 'fIRFQ3cYna9UK3CwFmYKfw', 'EN7M-RJPfzbUxP7dtKvA-Q', 'pjgE1d3csesAkgn6rQI_RQ', 'd1
iH2EPxYMQLbKHN03Y7XQ', 'gSXgEuv2wf4QBwzu8ySbig', 'IENeTl3e9W4diDMwW72gfA', 'a56VaJVGI00zaJVG2bF60w', 'AZuSSh8xMwLtoapGa02WZQ', '6AUX1beuH
GiaT-EvD_iL7Q', 'g9Q7pHMQJEctwBCi3umr8w', 'WP8A4MKs_LdRZ3TG45QC1A', 'kDZUGqP3j8eiLKIExBWI8Q', 'rMC4XjcyxQtpx4ZejzPibA', 'WxHQX5O24QWJkV3_
5EdFtg', 'wXQvpgxHM-13dulOMO43Pg', 'RjJGTNH1LAL1pJJhqJnvtg', 'D8EE4gbAiXS3DF_HNRJfAw', 'ToY2qQZYBIGXqKGqgx4QkQ', 'EOffNH1xPOh6WMaPQrVF7
Q', 'h_TlJAgBfSXG4Yw7rkC1Kg', '_wY1dHBXKN4NjB5IsIk4eQ', 'Qp1pHOQUssuX2MH_M8CXng', 'Milv8k4zZsoCaNFZugKtJw', 'cD5_lX613zUyG13MT_wQnw', 'LA
SSKMzO-XpNvKDlfTcCIQ', 'U2zN6K22bXa9EgfAmGIEbw', 'rg6MxcrOUNqsvUFl9RWPUA', 'VysCH2SU35qN5WlWJxujsw', 'DORWRp1AfwJwdu0Nfzuz3Q', 'o88vNyL8q
w2q9M1rGgRSzQ', 'ped2oFDSX-0JZOOxDJQEoA', 'RT6qe2ZAANWsBBZLhbTxKw', 'fR3LvglfLdiTXPrqsGaJLQ', 'GALRNo-AbcAHxEdmqKl--w', '1noVqERUUdFh1iHV
xPnllA', '8JDeA__M765evfPv-xO2pw', 'p7PjqH8_0KFe0ZKHOlwz1w', '81XpyQp_ObJMCQ85KFMRKQ', 'Fm1kLWF2t8DdvIMs0zW2aA', 'iFHk1IeIeMxL30QrH14Et
Q', 'xl4yl_IFXmU_9bFL0zB--g', '9VSegbNRzNlGKfBUv23vNQ', 'CRywkebZtGq0lKcbRlOPqQ', 'XOe-I25-W_FnbYfQyafylw', '_46Sn7OmqF-odwJgfyPESw', 'bA
4OlnXLTPTZ6GN0QRmw8A', 'pb0MADLnmfv5SzCVciVBGQ', 'UmHvuFL7DiEA7JkyskbFjQ', 'Op77M7U4GdksPz12FikeQ', 'TEoY60poY_5k82q_w9zt5A', 'tgRwVGIhq
N3TrF6clvRf0g', 'LF8U2She0pRYl5A479rPPA', '7vEqbLlIaBCvcoJUz67yzw', 'sY3vAX43yUeZVk4d99cjYQ', 't-27_YZKsk9tMbAiwQKCwQ', 'Sv5kY5jq0Qulj5fy
dJY-3g', 'H23KoR8A_iQ1RgDZhVqwCw', 'JM26JogHgfsBEFuN41YGhw', 'ifFTbGgwjXALO1PgMHzrTQ', '2dskMMZLzaTjXmJ4ZVi4wQ', 'jRteemWmPzYuTxFe0L0xJ
Q', 'Aaafbip-U2aeTXbvinBLiA', 'R2Pt2kZTrONyedI0-65FWg', 'nRNhtY7ow_mXte9wmrtH4g', 'bt6PwCE8Lc3YBFdx_qITzA', 'RU4VlU9LgbAEDMU3GfoldQ', 'SB
9L3xDwYybGrYNhQPRC9Q', 'p2UKiLd0jVx17b4nVNOeKg', 'kuP0AMnSGgkLfNSt0wOAgA', '2Vv78ycxr9m2ZktiwTbkWQ', 'Mx-vxv_V-SQCe76w4RmUfA', '5QIRfDmbY
hVv7xDeiw4p3w', 'PEw095uxUWPYB-vsugwynQ', 'trhLX2fxdqjuv_hCVm4N0A', 'sLMUvV5osYtLYU3JVBiLgA', 'DD3fOuL8bo6jrEsSeQHx9A', 'IXPibgPATsJo1uVa
evcH6g', '1I3Hd2TlG_BX3xIPXsttyg', 'AR9wLh3lEzFxitOaDFWUiQ', 'wq7PyqXT3QZ1UsOP92ipng', 'cTSSA0PFK6_xJAlj_uhN-A', '6NXn9bd7TEbmiS7e8yf5Z
w', 'socNNfwbqsKoTvfSDA7uVw', 'uh-AusMLL3_83q0ywp_2XQ', '-gAQMwp5b4x0XBqCtwNVtA', 'akjaV3M3qumD5HYkjU9dtw', '_f_6B-oiWSXklz9F4C0Khw', 'bC
LFdVfkAiZ7t1xDskb66Q', 'JGJ4FgFihTApMrjNn9kfDA', 'HIilP51OIzfmgS23vvFEfg', 'pUUmwguzGEAFXDrIkvzuag', 'r8nd1J5aXG1bL5ctt8ucCQ', 'CInFos81G
f4dkBJQL-1jGg', 'LGg1MNLAYWmHfXzp6hC4ng', 'lG08m1dziucV47k4olWjEA', 'dfkRRR7vHoF_0ApWjHsOHA', 'oDO8TUHqP5b1Dq60xITLyA', 'VIOt1BAZ_9Xig0Bf
1EtIEA', '0mVScF9nfPVIkS2OlbvpNw', 'KYOwoMnBtNhwBRFbw0fn5g', '39KYyHuggTbIceHwpujX1g', 'ZDKi3qsO8lSSogrIdbLJKg', '14v766ba7jx8JV6BC1lJx
Q', 'TqkBwRmoEgECElxeY0OgKA', 'RPuCnMVKNaKsSJ_INclzaA', 'NfHxV49mtVRq8enjdq1CeQ', 'tM9nH90xuXsucJ8qXcuUTg', '7rSnmi6siHu6KJU3990oAQ', 'Op
jQft3XugK05MtWegHFGg', 'C-DyhE2OMYRsahCVQFIvxg', 'LXg0qZbRViO02GezPRe67Q', 'f2l2Trs6c0DbWGMMllmJHg', 'ab5LMDx_uraOLlIiwfKI2Q', 'QpVbA2JJv
XzEW7S6wDHV7Q', 'CUUpi7gtqx1zojEGJ6Z5ew', '6NfOfSxsYkWSgYB-oWRoZQ', 'hhlABhXouwdWF5YJ6-tXOw', 'JnEpFNLnj0TWOO9ZZ_Q4-w', 'r9YT-wfk8DlMyq_q
4a_saQ', 'ORRz8KqeiLFYYQuEYWdmUg', 'hf8wMckM9DMg_gp5Vj6PrA', 'PEmmEIagYJ5EfA8xfjWbOQ', '7mqhV16NyehMAJlG6kX_WQ', 'LkCoTvna2lV6MPG4Xzoc8
w', 'CBqhQFyUA2fVMF1NmEEgtQ', '_BWhSAjVNmVSNl7pTG4w2g', 'BCU34iLQ9kHqslWU9fFYzA', '_RN13rQ1c77hAi9LU7nwnA', 'IyEdBo884XBPmNLhIp8dig', 'JR
v7TrR012humEezHENUjQ', 'qM5w7JwlEQGYlGtrOlEDRQ', 'z_umk5Ue-tUcnmcwejPCmA', 'mdqDcnfQfTIcxM_zaXWaig', 'l8q-Mj_CuEKgvUgnG7IkPw', '3NEZvGl12
UVuJcUzAfhAnw', 'vDHtHxgzZqL-qebSa53e9A', 'dFC0MPQQNu89a9LpfL4PhA', 'iK4GPz61Yj_cZCMKC2C9NA', 'GzMh3V8oVBoII38x1GpajQ', 'T463_lixRwQCRjS0
TEe1Zw', '_cdM8vgagBmo3S2PcAycYQ', 'w1y7rcJAokqFyySEhd4r_A', 'HQJmocHzUhcfnfAKKI82qQ', 'bM2OTIopnFoaQGLxK2PxPg', 'g98c3p2XC1ePPDMUODT15
Q', '2uk4R74YwMeBQFFnWj5HQQ', 'AAX8oeoIq2UvMDLghGBclQ', 'OCkge7lzOw33bqMs1244Yw', 'rhxTEDhN6D-uSW2D7weUAg', 'vEdlJ7dtXRQgjxTdjf-j4w', 'rn
uiIVuM6-wT7SJt_z80pg', 'FE3lnTyP4j54XlmHmxNHlg', 'JvylLkohihV4I6b24UCL_Q', '9tELOBy_iGPNpNjXzkekHw', 'nBFR96CJqRtm5pfQ3xMn0g', 'PWXJpyVyA
KtVehw4suOimg', '1mzDvYWjU9lFKzOMHZFlVw', 'D8wdpaUA7cK73VEZ-Y6omA', 'ysB_jaJVwGzgTa3s_KyHnQ', 'wTz9pYPDgg64PFM2v9gScw', 'k8Y5dMaWp8N71OO8
jsl5AA', '19ZqbAxrTdEJbvwjJSFsRg', 'zs_GJibScPz5Cbz1hoNanQ', 'GEUTx3re-HmI4fhc47klAw', 'jbd8Uv0r16Kwj3Ayd9vvuA', 'AHz5oFLWuWvDNWcnooM5k
w', 'dk5zOa6PtED2eAMExoRB3Q', 'T_wjLgPOPXry7Bea4MzoVQ', 'IgMhM28GtVdkg7qwM-auLQ', 'Ke9WsSOUOdQaTRwaYibYpQ', 'bLmp1BzhbHJ8adLQ86loHQ', 'B5
LOrNT4RdtJ-kFEXs7qSg', 'HKMcpGwNg1zi2ghARQdykg', 'GBPEA07x_XnTi366qTrjjw', 'rJW1NZGbDbqtI4BcygndAA', 'o8CJ14Iz7g9A5N8CNfqWiQ', 'POmVbGJ5n
d9aZTxTJ_FVfQ', 'IlQkSpwXtYygzHXRCoveHQ', 'Y9dC6RSXUtPC5o90ZnC3Kg', 'rLtl8ZkDX5vH5nAx9C3q5Q', 'F4Eigu2W0lpF4oEwa6lxDw', '8tbXmjYGsYFZXk6p
puwRWQ', 'dU4aXFeWOBoZvgmBSDdqKw', 'hBwbBaBqofCgvkJ3PatuVw', 'WNfna4-XsIZbCrpCF0ZfPw', 'MdCZyRN4HfrfnmJlvTugEg', '7b134wSLX8gN7G2sbLPZo
Q', 'SIjLiu746VXQ2eI_mQnhWQ', 'o5Uot_QoTnTJB8CTKnPltA', 'epTMlecM8SZNjDhBsd64Jg', 'VFtBm5d3lZLNSnkZCSZgGw', 'zUZIhg3UPaLsWGex5EWdOg', 'QP
Sq3SwbTZbTB1dx6Ms2Lw', 'uWak0g1TXVr8axztY8Bdkw', 'ObAgEif9mNN4VodOi7rEIA', 'DOJM58OkGSsIdk2qCUZnLQ', 'rlTc-qZ5qERiXFRUd4XbMw', 'w9EJaYDPK
YBg8dkHxwNgHg', '6R80N0O5E3FJ42f94O5RPg', 'SYoux78T6d-AVxa7FhD9LA', 'g3nLX8WMsmwH6gSJiPU9Hg', 'PUMROYfbVgKypcfJNxs1mw', 'WI7NLrNZ90cj1vPp
FOOHUg', 'Qd8hqPpawzkezbOZ0kXJGw', 'N9n6XgHfiaaP1axKw0cyyA', 'yaBTkNRszgPLez96qo3lew', 'l-1Ti_hHb8QIJFG0CbINIw', 'LuQD-Pm0zqorZDv1V4VSW
w', 'D_-iaWZwRj5X6Fl6Oca-8A', '82ZqgHOfUMV1xtpG2KfisA', 'u7fkBhl2yGR1ZGZMue2B5w', 'pkW2moNGANM2W50yqwaaOg', 'VhI6xyylcAxi0wOy2HOX3w', 'n7
1jnaDu0mVNW_UJgGKPFg', 'BFZD0XQ4C3n9Uc7TmppaDg', 'TpH2dPNn5ehrluF1190BJA', 'KVaCXKlrKSyqoLcB8WF_2w', 'GBgY4j4mZjB-eoRb359Nnw', 'o1IKKMxnl
5lDR3-z4SucgA', 'ia-Bk0oHTcutM7_Kq6Jrlw', 'D8GD--YyjXVtlYjDGZX1yA', 'xTZwuLbQtNtKl2rd5woIQQ', 'kbXND7Y3udL7sZA2786nCQ', 'EuWhvhW90cg3HLIH
CnXzvQ', 'ghPh5QV6PV7J3Jl4MH8euA', 'LiuKdou7xBsV-29MTlb97w', 'UHOFZkgIvb8sn3RSPMXMSQ', '8OZ3TKist9Ekuw5T1LKRkQ', 'QCKH_57b2Hi146J85ns6O
g', 'uxEe0lQ4XqncfDMMhZGQBQ', 'gVR0LJ5Dbfa3_bGOt-_6VA', 'q3kcZKVht4z6lLOaxRgn3g', 'qJQtWo10SSxifcXw-5RGWg', 'jykemU4-CaDr-8Lnei-_ZA', 'QY
```

ES_c-F13OhsFGB4twmgg', 'wZougLAnwunlZ92tHbyZpg', '3DmA6rp1NHqrq938cnRCmw', 'uklc05zRRZuYzB2X015Vgg', 'bxciP2Aw-DZr9fVkMJtgOA', 'HhE4bvGlK
B5nuN1odc9GjA', 'Bmk46CeiMrga526ZX0eXsQ', '7H82rkP108uQ_wXf_uMeRQ', 'IN2HcBun5PpyfQK8OC68iw', 'T3-UHagV42o4NpgqKJ2K6g', 'zmPzZYdZqP9MUfzh
n8QTfw', '9tngnaDXfVeKtXP84pXSjw', '5k3g8CGc4bXSN3mcjFzWIw', '_9Sj8L_XbVgSmExcvDLHiQ', 'yxHnHENAwM1wlwd5SdS-aA', 'CNw6OzOjme50i7CF8Bwnt
w', 'Wb_kZ07Zsj8PLFcsiMeixw', 'lXhOAP6ncg9ENgA9ln_ijA', 'Les7_9h8xQDZuZbHhjVYkw', 'fm5-O3_otnzaAFHYH4pYRw', 'IkkXHuygD4y1tquSLhnpyQ', 'yt
DfLMUaaHsGbeJMIzWKsQ', 'NZUB8rNpZmAuGtiFfIac9Q', 'k3esMpsFS6lG2oRI6rfnLw', 'blimeRPMCJggra7NOCpErA', 'mGDjwSebGqtbBRREgLaE8g', 'rGXK7xWyv
JLjAG9pHH4brw', '9UtNYyrWnpMbpshMrZSRGQ', 'I5ab7grI_zRFieGVvDUPEA', 'LdEq_hOTltUD_PX_9_qlRg', 'CatKFlIPqxChN1Tu0LkYmw', '9NMPC1bfwcJ2IZu2
k0OAhA', 'glRXVWWD6x1EZKfjJawTOg', 'h-EjNgw2t9V6tiGd_Qxf3w', 'nVQa8qgHbdosO-ZDY9-d_g', 'VqSCQCva71Q-ZVcMsYfQuw', 'BZRGP49I5pK2YxZokJWhH
w', 'yw7_Ba6yRFkK92_srFj97w', 'Nd-bx8nCwoA5c14ugyVI_A', 'V2ARgOj_0v92gSxlgfx4Jw', '6nxt7bPWPRJHwUKI2Leabg', 'xofFo7WFfHOkzbtoG6SgPg', 'DR
6qPrRgeSwmBKAap41Xfw', '56i3ZvUITU5MmP9tjhRCmA', 'W1NZ33Kh4BEZrjHqP1FD-w', 'BVJinahzVxJFLMA9bZY8Bg', 'tx54uB3v9-WNoKxWoJFgGA', 'le5mkF5Be
vJ49WnumKYZ3Q', 'OrVtmaML9UI99bsYrV6IQg', 'bcUjJrpo6OeQxWK9hY-gMQ', '1hBqG6Fk5OCU8IMxCqN3Vw', 'yQZL8MTulNh3d3de-N0oug', 'spJUPXI7QaIctU0F
O5c42w', 'wHfvDGojhXsB5mmAu8ulKQ', 'zCzIbD_gs9hoAkg0_R_Q_Q', 'IMZhBBEG-7b0EaW84Sttmw', 'E_KAdo61uMDvIvqFcOX1Gw', '809QEhbAKPf2drYxrbjM_
g', 'M_Nhgosc63jD-nu3XgpETg', 'nZFcWvgPU8T4JCjOVvJU8Q', 'KOIWgIoBEEIuAnUJm3HXOQ', 'KMpj7TQf2gGFwpL-EK5c7g', '_Fxrn9P_nvvIY4JCvD5tOw', 'JZ
vaa_v0__hzdrwvMb1Rfg', 'xv_XC8Ey8c1Agc7La1pwAg', '8Ggm4j6gnRSKo7k0sBmdyg', 'p3ZoNUziIevbpqVMoN-fDQ', 'TTP7l5g4QUVCpQZ9NTAWqw', '-MyqAKY4n
NPM9PDv2wrxRw', '1Qs8pJAnTEdHv0mTcKny5Q', 'HqXxZd-lapvR0bvVTlKnNg', '98f92DVyKhoQZ94DbTLVBQ', '0MRlubaLu5v3sI4vu_XqDg', 'pXHSbmRWtYjXF21j
qol2ug', 'oWKBHNGxTyp7B0Yr1dRxpw', 'jmv7eDUgx3cn1y-Qw8G-Og', 'NN17WrE5W9oN5UnvSNT8xw', 'LNKdYODHuGqah_nUaDJx_g', 'PHrPDWUCtCmkwx_gzTMGC
Q', 'Bfo9NUm-uYdNzLMyNfy_pA', 'V0WEZ2PQtJGX0sWNe89A8g', 'v5k-__wHo2xPI25WO-dcQA', 'z1_SJGzJhhXxR4ftohOK6g', '2biTJfc25MuCOXD6XCo-zA', 'oz
98YFoHRrJ9u-BIUFvwKw', 'JZ7N_ugLCrooDzkiDYXLdw', 'lhPHOrHafkiAtUeTDee9dA', 'lWcklENPjAi78Miv9EG87A', 'b8vpDCrjDwjXSr92JVp-PQ', 'BKo0QW3LG
BzAZug4doJrgA', 'Au3Qs-AAZEWu2_4gIMwRgw', 'GX0dUsYmVzLgb0aiA73-rw', 'Sptc2IOKAfM3CDd5QTCAAw', '9v8BtiGLKWdVcCN5WK4s3g', 'RQalwji2z28KIPq4
DwLXmw', 'xLbSUpVLC6AQHRZTwJvxkA', '1LfUXEdm0O3qhEqkKZjztA', 'LP5MjcVLtG6TmXRykg3Tzg', '2EDBVBqeQ0fuDvt9m4P8Gg', 'FDPjfyyHNfy9tqyxo27Qe
Q', '640yIMkALhRVE3JYu13OsQ', 'cD9d9XFoC_bETPzjpnRj9g', 'b-Girgjp4FnLdfwfLDIHxw', 'D4KvR-cqveAMH8tLOIks1w', 'rxGMXiiZyxwnYUrWP1OJTQ', '-X
LQKak-GsDt_oTw5MutEQ', 'oXxV7lleT2lF2AwGZM6QWg', 'NzOrlc4LJJLRtfzoxkJFsw', '7V6RYBSmpoHsVkohGD_yGg', 'WpAMN4ebd28Il4yamLv0zg', 'jIuc3nM9R
0KWX3v-8OyLsQ', 'TT5WOWluwF0elhzJPrNnvQ', 'Cs6JRqxOADfeqli20F5Yhg', 'k49tKSAka0GyvbL4HFYDzg', 'F1oGgK3NtTYnyRMxupiiuA', '_3js68YksSedX6bG
n4-2Tg', 'y6M7TDkbEID5urBb_2oQ2A', '4NY9iOBSwI_AKZi1cHPMUw', 'ts5QpC5ymz1m1EeumIm7Hg', '7yWgjGRb02JcUtphd4te8w', 'c45AFDoQz5OJcZISlZ1y0
A', 'OyGx_yDm6XoVDs_MzYjftQ', 'ZPolhetd6Od5_VhXPbFIxw', 'w4dgIzMwwZvKsHaC47Ox3A', 'utcN2FtmIymOprcfFS-Tfg', '8p4at4zdzCpueAmSBaorZA', 'Vo
7oomW3yovftNleHAX9Dg', 'ZW7oLADt3iMhjXloUHJ7mA', '8ncc2JviL8dOl9MLPpYMvQ', 'nh3mYZ0cIXNVXlQAoOt-uQ', 'mtfkAgyunxat9fcCvDLVRw', '5wnW-qphV
h5fa6qfMBzDpw', '5SLqeubMzxW7ifBwmqhH1A', 'ewix80iggo6HsqG1zQ2ZdQ', 'GtV0RPZLK2fLBbGQsm-rGQ', '8Xnd_YqFUIPg0Nflq5wH_Q', '6VCzwT57Vgr3JA4T
IeF_MA', '790JhUvCFexsA5KFReMhig', 'FOfUxCjxD5xpKM_7mphLJg', 'MoVzMz9uqrDQeQeazh3eJQ', 'y_iOYwgsEQ4RYfRQ6tRGwA', '9fTTesCwn9EO-d82ODJAn
w', 'Lb0yp9YlXSh_6miFJCLqZw', 'qhooXf7Uh8NP3mX6YG3Drw', '9UNGbiE9k0fvMpCMJp0Gww', 'nGYaT5sAuqKkd48dqX_2Kw', 'wM-4uxHkuThPWfSRaKXPdQ', 'J1
ATkrYS0IUkV4u-bkltgA', 'ODhV53XhesCBLJVfGsJGNA', 'dLqqNMVnbF6Nngz4R60_Hg', 'dtE9hHw1P2NANIBMCp2cBg', 'VT-4bP-o0aUzRiosIj04dQ', 'nrOCJCQUg
XwdUIwg8QHirw', 'g3z_RHT1oBH8e8vKcAUhTQ', 'CVLN1uZgGUky9Brlr8JJSQ', 'JjKJznvd5f7Io8hFJMGbzA', 'WS9jQRCTMN_TL7W_065G7g', 'dJyY3aXKL34vtCF3
OP3FQw', 'oZWx-Udu1bV_jbR_MuXaAw', 'l5eF4e5dBrB9pXtAJn3hBQ', 'huKYBfeFtYCOWs4I-GESSw', 'SfiNyNLmW7alXejl-m_iYw', 'pwFXtVzH_So9f3iqHCEhH
A', 'u6VZAszfuylriytDU66gxw', '-jgGsPzZ4W2TyGh7KOwl9Q', 'IpjuqXEJFa-tciHOE5NN8Q', 'baQVzpnc1f9JmMpyGAHn4A', 'AODbZttUrVN79H_mzekhYA', 'yd
aYTc4HzEbByz4HhNzC0w', 'mYHfwCjpImZ2hfEn1pkFLQ', '209Pltw3TJ7Zy31RcO8XXQ', '0LoXMgGgnL4c3P8ldx5KVA', 'eAq_2RBtk_zmRy1_tEvnPw', 'fCHvzdXkX
fI2ORPFk1U9-A', 'z5EHulCyTTYqHRDWoyrnKw', 'R6R2WBSbeRRftN-vXETapA', '8zfDUTELhJczCHhq0QuDHA', 'zEggXKgiDxiSTzRo07GFGg', 'NzP9OoNP8vBMRUSs
0UXBNw', 'pZSjYxdRazshCPbSI1Eu6w', 'EpkFuQVu3XBigTU_PnjzLQ', 'M0G3zi2UH1-ESD1rtomXkQ', 'dDdinrAqDdeR-LVhAuiOyg', 'UpPqFMrWGClk9YYfM9Ck5
Q', '_jpghfHskfOsN53gYZtHzw', '3Zi1udoY2iT9s8Q3X6tgZQ', 'UDWNMHV1UqQAVgPzzWvZKA', 'rSbqoimVUWPEbNSGwyjI-Q', 'XEJKtrG2LR_wFjH05s6Qcg', 'Ia
y4jEsj8GHzf1UA40ailg', 'Qelj8uPxf0Ub5CFyB37kgw', '9zZNilcQoiJicpenGM4-Cg', 'lIIoaWG_J96-cQyh0af55w', 's2FbJNdU8vIOEk7n48HJ9w', '1Lu8MAjA8
byyrwHQoYpiVQ', 'WRODj3cwZSRD0QKfapmyiw', 'RoajVinaapQgQxXqCg6KCw', 'XgwpxdJ4dSnKpYYd_07vBA', 'nGvolzdWXhC-U8U0ZLydWQ', '-YVgvwt1Jp4XZ7QF
md-XQw', 'ZTs-go-xYRQOp3PSw_XGDg', 'KVm5lSaNK-fL1OsbukgyuQ', 'C1NCPmIP-sXfYm5Ywhjj9A', 'HCrmPNeV1QzMESHHRBhOkw', 'SpyUjfaUXdhgAaVuhzNVT
w', '26Ow8ovoXuCdduVBwOwX_w', '-K3ZjROK0ml2P-Rk7ttHzA', 'OIT6MBzr-yxw2qtokoSXIQ', 'yC7VOAAvoam0rxEjTUd3UA', 'XJwiv5u7h1giaxgYAOz8jQ', 'h1
BJNxB6jPc6xGMACpqPoA', 'JF1FEhOHxPLgc9dJ_qOXdw', 'GsmmRjg6c059dhxsLf0YDg', 'A4PEFU6xorg3Gr8MssKi-Q', 'akjw1U2vzDrNP7ybg8Rj8A', 'wFweIWhv2
fREZV_dYkz_1g', 'nKaR5Z9Qmqc4RsakLLX_7w', '2fmj5ng3irLWTr7CKrSxxA', 'BDKaQY0N4NQxTUaMdoxKFw', 'pVYLj3qq6-zdEWFUiC2CQw', 'v-FX2WyDYOpA6wqp
zudZ9w', '9TJD_b6WWp2l4NT-qZaH8w', '2ShjrKMCIcLRLCtafbqJmg', 'v4ejNYOneqVyFU-yXO3wEw', 'N18SNshdyaB2vItkP5zG1Q', 'NkaFJUKzzQfMcFekrZMwn
w', 'gIWhLgPNT1TFWBZtShNKnw', 'f3cFc4_r0pdY5Wqb8W9w-Q', 'MOdILSTOFnq0iXHN9l8_zA', 'TWIg1Jx2jGiC8UqaKpNorw', 'sFQsKwx1iTPSAGYOo1yTdw', 'Xo
4tysJw1XdbYF6P44vbWQ', 'hKU9eYeo3gJnduUpWDCCoQ', 'wgcZ6blNmIpIREuJN1vhXw', 'pk2kKnjUhl5-Hv2Gygoctw', 'bKhM7KzqdM0j2O4SOLmSLA', 'tqCKgb6HT
j2Qs9y04p5ZMg', 'QS6NNvy-k132mSImi4kIjA', 'uGWoaSdH2gOtl_Gjo9G40g', 'CvMVd31cnTfzMUsHDXm4zQ', 'bNrr3RJGMO_gOrKpDudP9Q', 'mvRps_pPV9aw18qG
RJ626g', 'NBO1pzogTGH8wejRcdyC0A', 'XlYbPATxPQRJR1FOZ81yog', 'eelWS7zIYMM7ci7dEd7MCg', 'UoS4bUrlIvs1gqmHCoB_Hg', 'm1wgM90q_3Geq8LywXcU6
Q', 'neOOSMNWcVL0o2Xwb0goVg', 'ww2K-1OgVGJXNUd4WtfRGw', 'cTyEgBRwcGhWiq6TZPMFHQ', 'QS5c0pnSNWbv5D9YtuT2vw', 'xwcvsSC8Virgl7CveVvB4w', '17
xLGvnf-fSATC5fuk5BCg', 'VgjgoQDmWTDXrWj3MqJRRw', '3Y27vBOZzbkBhelwqu0rIQ', 'I5tOYH0rnjDxW8ieL3TYyw', 'l0izzNMbD2lL9xcOnnMEag', '5hptLoN75
O1O8_xEgQip9Q', '-XPEaiJ9djZ5FugRVauwsg', '8v9q-mpjdna5X2VYCEniew', 'FbdB9KXyuvT8rC_zmqRSuQ', 'ZFaem6KoZxMy3pgZSet2rg', 'UOppDzNrQokBg--B
r_9wOg', 'W98MlFKogX6oGoO026bTvg', 'gc4rNAgbGydNMAPJ85FZag', 'SRSxFmtaaePlmmebx2QRgw', 'XfkCa6Ma32-KZOpMSvQ0mw', 'UUHbKus9GdbL-bFervUoC

w', 'QB2h8hRFcVokhZuajF7Wiw', '9zlwzi6s2nm-q2fy7hNmtA', 't3rZpAy4CrPi3kuJGfuCTA', 'ngR1Y009kpsuqWRdlKp7hg', 'njQ8Bn6HtqD0iMI5ABX38g', 'JQ
GSDe-kap-IuUSH5q487w', 'EuYX6_YkzKYmazHmQGH_Gg', 'zdZfTTdKbWgTsce5v1HvSQ', 'TQ4vRT9ETlxaL2Bvoj0fMg', 'b-u85eePnSX5XtfEzIz7gg', '3U-MZ5ydj
-s8oOJW7lk-Kg', 'tiPH-oA8Upuls0__lZ6XqQ', 'cduTmD2AKlX6p2pIVsTzpQ', 'ZpYfzg_GZa2CwWflRR2b5A', 'rSBYu1pV5AF4WGJUWIdjnQ', 'V4VW69mBsRzfa3mL
myBqLQ', 'PyeEPahaIMGQo0v3F2tplQ', 'Ikf6HMvFlXMEJLW4HH-tFw', 'gyBs-PcHasLx7IF9B6OBZA', 'TmrLXwR7eTyy0nzxXhdcnQ', 'inkv3_Ja07wnzXvrc_-4e
Q', '7pfs-dhgcye8nE1ktpVJmg', 'jdP_qK6pEfc1XCRnEhlIPA', '2dcPh9j30SrIYO9tB2YXLw', 'vHNtj4-I_H6nTRPUaoU5tQ', 'Ve05sG1S5QLSm5Yuo-96Ew', 'YE
yx8t2Okqy8JDWHcF8C9Q', 'vasHsAZEgLZGJDTlIweUYQ', 'r4i7uL7zx4u0M-WmWdZZgg', '6uRIFyibHWozuRjZ10-jiA', 'sQBMbqZWNvqII8QwW_6RPQ', 'r46fg3yYJ
ganUaF9mCzf-w', '8dbRf1UsWp2ktXHZ6Zv06w', '2VhcOYqEFiZKlFlPrp2VJQ', '_X6TvNAG26owDEKOcPe75Q', 'BzsUOXhZ_i-fjigkudOLcw', 'o1Esx88FqeolMqaj
zjKZpg', 'rWTgooIaB6vmD5bKCJgkLw', 'vY1kebWNJBBLPGZZ9qbl3Q', 'vWZsmtNF2O5W_hcBRFK3lg', 'hLuKgo7G6us4wNQoFb7N8A', '0A2phOr1tuw5xfc5Psc94
w', 'aaton9Cy93Ae-7Kr-OEr9g', '9qUYm17IHOA5AgRfSoKySw', 'URcLQkuZWiAAvVnqAoNfRg', '_W4Pgs4js74DZCMQ4d4jLA', 'rzhjzUy34KZA2cL-gJaDuQ', 'TM
6LLYhkkV1VwTDkHE2tiw', 'yIo81MlQWemhrXOPv-qFdA', 'Lw99LHT54yb7ED0qlgujeg', 'U8l76MT2Ej32cFKfFI24fA', 'Laoc1ZOWoeNjR7CyWGDX-A', '_-W0caBil
ETBPJIlIqbKSQ', 'mzM1y8_2nIdU52BHnCNrXQ', 'UlkTmq_VLqEYwAnh5196Bg', 'BJcN-MemVWH6vPdwbCzE-g', 'LYBDUrVgaeMuD62ADMdqJg', '-ebYDOeqkWCuFCL7
JO5I9A', 'H1r9_Qxbh0zGMii0ax1C_g', '9P4YzvP6My5ktU6s0cA7Yg', 'XY_bQqhZfyyEi_W5n4kQNg', 'XrfkmMwiaPwNN3ww-T_MLA', 'QGRNWxVU-pbd3piNFWU4W
g', 'WBQrry43eaTOIPoJjwczyw', 'HY53AFJzVN8DMHMn6Cd6jA', 'yuVXHT4DzKgWSz7_1RvDWw', 'oPjmenCi0xOw1WnXmSh_MA', 'DdEm24t9KK_C6kP5z1osDw', '6I
Hcb7PC6qiYR9qEsBieVQ', 'lpZtfD-vCqUER1yGKEvUPw', 'zWli0lD5TNKpTSHjXWdHxA', 'Z82HLQBeNdrGKGEzbI_zPw', 'JVnGX-4Rn_jMWB9picw-Vw', 'po_IfliSJ
IPzzvd6UfU-iQ', '-K8xcVpQN7jJLwNDDuj1xA', 'RKDSCRbN17rlE4ayYiyEqw', 'v-b34wAC0rJVWk4dCtgkXw', 'uc_kCbu0N9hztmtSkSHEtg', '2t8z2r3KYbBpZUH_
uq9yFw', 'C5SNlxImCgHrtu3B3faSlQ', '9U1PH64Xqbfjc75mDnXbsA', 'NduGDKc6hLcb1S6OtW62XQ', 'wx12_24dFiL1Pc0H_PygLw', 'd0PHPwf_MZs-ZZ_d0iSap
w', 'aqcxaC1SwdL9XjWuZ4MxoA', '08tQjvwbh5hEoTq_mJjtDw', 'RF74LWUXaoZjNPpt4782pQ', 'Q7SjcvCvxqsbqVnqU7EN1g', 'ugMQjVWUnW_3An7fJEMLbw', 'Nt
qtKtuFNLXFF8gqb9td-g', 'HpjppzvCJpYHzPVtjevACA', 'P5VfQpNis_p_vAh0m8xzsw', 'OXzYyZSHo9Dl6Qd0s_o2Ig', 'DJWC4idmjq_FmcOCbTljLw', 'HPxji3cX5
H460ECCgtnGHw', 'S3HZF5aANmhZoMkFkPMdqQ', 'NBWUrKMUgC6xh5yf10-gzA', '9YVJtEzVfjfNujAfy0ysHg', 'vn3lecMsL3kXVe-7hg3gLg', 'YouW_cDsluvPxV9r
lDYvMA', 'nPDUuuX_2C_GZpjGifWRYg', 't5AWRnd4Eb4cdjtBgBCasg', 'FUId2sKCoanj_P4JRgQn-Q', 'ke7LgKD9XV7tDw9HER-Ggw', 'LdhwY7eudLlx1KFQQe9Gw
A', 'bvJvqYTSnStE8QL5fSJCkw', 'gxOZMidTXSm2z_0AecgbGw', 'BMQdK1kdXCiDTaeaIYxmew', '-OXNFm3dC0UYT-yA4IkrsA', '5HWf00VduXm8olEyR59_Wg', '_A
VSpDL3qhiZD6N2YdhpJA', 'SxHFDNhmPp7xga6ooQ1eBQ', 'lC0KGXmIhyjzghBUlVnkhQ', 'GAeM4pken2drf7CpuKHuLg', 'AQnk7fS3XdXYM2ZWSF_Wqw', 'JnFI2B1H5
DkhVWM5NehsDQ', '4XrpU7kpxbFqMNdDEsj-xQ', 'Ksh-pjTVoRC8EGMR7G-boA', 'JmyFxjiPLygW5ToW775pPQ', '3LaLg5Zg7d3FW7hyq8AQPA', 'OvXnk1S60tKzH5n_
CrwyIQ', 'DLP3XfRNhTVxXlCRJpGLNA', 'ibEK-BLVXBl1cW2lq5FvRA', 'uJVv3nSqscK0_c8cZSlJ_A', 'x-JcwIoL-b3FmJ6I0G8Q7Q', 'deAYF4xLVB9n-Y-fxGyhN
w', 'WpG-g-bcIoi9s0n41gP85g', 'ZNIxRh-iCCynnUk5PFHJnw', '7ACTIN1VopEKbmKXog2DEQ', '1BW2HC851fJKPfJeQxjkTA', 'Sn5VmGYLYkcEMVjvze1k1A', 'M2
R5lDfU72VkRWsd_0uICQ', 'enXVISXPVpRv3InUhexQNg', 'I9mIKgo8cvSwLC4kIe5MMQ', 'ZkuPKOz3tN9iTUZrGzj3nQ', 'Yh6IPYTO2YsunUl9aeyU1g', 'JXOVmy5n0
aANd9TEKvMoCw', 'HRLkhkYHiM-awcj2k0FgJA', '8T7VE9bwNoEUs_0RervT0w', 'dr2Z50uFoCvYkifzxa9IQQ', 'B8pmNy3pUAeJWg0rEtf9dQ', 'l2FAzcmrBNhmuF7G
7XdQpg', 'l10Z3-aGfOokEutif0niag', 'yhDgEDo-u9NlGJynZK8HsQ', '232eTYWlI4pnKRz9nh4nQw', 'IN49cEfDQTbhr_SaL9y1OQ', 'fGBQXv5-9eRBjwhm_EL2j
Q', 'qN6SH4CKMG46LNkl8zL8sg', 'chPrkj4zYNNXig5sI0Udew', 'beFBVr42X5PdzazW6AuthA', 'yUUnTJ4mGs7CxMIWqUSfog', 'C1SiUu-sGXL6lHlRkJxfFg', 'Zc
NOSxKpSw3Xd3ZL4ex0Fg', 'IJFzDrRikDtYXUMqvHGhXw', '1pF_YG6foh3aJqjVZn8Qmw', 'AHgV8ONL8do0Dmlx9-tM0Q', '4dkA654ss5dMR9wG_IrVYw', 'UpdAuxn_6
oxYpSsMowmgbA', 'pQZ-yK9zEnvtg5NRqTowsw', 'C5YT9IID8c-x20EXsxHFXQ', 'ylGYTVXYss5dk4SRqQsu3w', 'uV5fELkeuJflf8slZMhd5A', '3ZQsNZRW-OhGl29f
f2QGjA', 'UCOssThZufcqehMMe6OA0g', 'OXL9-5XdktfJQ3WdEOPSVg', 'ao0KOJbU8qoKz72TO5QRNA', 'ZwtRT0gc4WZ4Q563ifJQQg', 'u_wXnVPQ4d7yAvz7xaoqs
g', 'Ju09FUAava_EWx_6szpnug', 'j68jx3DUkW2_6txDzqTY-g', 'k9NXn2KCQ2k8wteCqH5cSw', 'Ed-2su2WyQRShWrLFdY_Xg', 'Y5B2-04eA_F5zGyHUb7OpQ', 'tR
y3UZHBPT926UgAE9O8_w', '5Q4Xz6QEoVGLlEN_STTtHQ', 'Fky6k88rN4_XN-wicz_9Vg', 'W2qM9vaFHiAonmogmX58AQ', 'fpRXn7pFleWb0sgOHn1npQ', 'ZP2PRJ8DA
xDamxe82CuOCQ', 'BT3AQK9b5oIAdVve3_Sh1Q', '3ceE2d7-nRLASFIALsnorg', '_tdK3xZ-xf1Y_7rL4V5xMQ', 'HIlrU4ShA10pNwwhc4xSaA', 'RenOpB9KRRx3_MFW
IzI3UQ', '5z7yBm-fIyfCF3CQJkuarA', '7EJJUCl3lHM5mYlFX0bLEQ', '1-CrNyLusSNYdOhWCKHnlA', 'EmFOHQzNewGtdLWlvvosFA', 'PCny1cwjCCu19VudKeCan
A', 'CDTNFwOXygHDxxaiIyzzBA', 'FwzXiPL8nY3PmmHjRP1h_g', 'uO8iD4P9kHpzkSXLLK27bA', 'qOVcYwNIOMtPnve8a16tGQ', 'CZAw9nxnBWr-AQHdRLvU6w', 'JF
05ffi0qdheB9Ti1kACOg', 'wKV3xCxJcnBnQ03eGp2bPg', 'Sj4ruod2J8tUc2PKkc0IoQ', 'ml8-oQjWKT5YBCkT9p-yhg', 'en_lBIEoyYdByLHNGdWF7Q', 'dGfMkK_fd
j6nfiSeEmtzrw', 'GlAL3I-vsN5dbnOP_vQV_g', '6rs7VCPIFNsM1bx60qKqvw', 'TFtyj0Cx_gpBi1yFDKccmA', 'aLnH0qJhmI-KVA_rfR-8QA', 'Bxypvn3rP1fwamC8
cU4PYg', 'MTQ7_0ZDOd_vS4SjczQcbw', 'T8N53fAhYESlrJUMEsBenw', 'jcoXg7IlOR1LBCgtMLdUsQ', '-Do-a2vqE9EoXtnV-TZEBg', 'VRsMYNzt0FQ__gUAOuRqc
w', 'ESNUEr2Ex3eU3diQvQWILQ', '2bAbL28lhrnyOlYiZ8yMdg', 'zFW9Mp34CBuAWGYZOt1o5g', 'nG_zliKHG2PUttJa8z2APQ', '4jz83Df1emVCgifPOfuoFQ', '9G
JHHkT8x6NW3x_nXyvPMA', 'gsdSJTKtaLtjynSD6hgQVw', 'QWkXgWM7jjIruQ4Q4wzZWg', 'ZJDoH1d6mPtsdMFDFV7wsw', '49kzkutc5_UFa2hjLEFYGQ', '85uSaCejF
tK-8oVyqN9HHQ', '9OZH1Ecw-qUkCW5MS0NefA', 'o-B090oRMU1Wej-o3KGT9Q', '5-zDa3uehYmGqrzc7esK6A', 'zSsucoCEGxnw4vxmSQXOCg', '4BWGVJEqqOnihPxn
ZbuEzQ', 'dYWzKUPzpuCEJJHzOeVqxQ', 'mEgD2AGWGKD_gs-Jf3Ypew', 'YzKzF-cYKJnnZtIj7fe3_g', '9BALHxACEqnLtLHLY1cNlQ', 'mUznWCN0tScMnMcMFUqmp
g', 'yl3YZddzKKGdm0a71su3Xg', 'IX743aJcFbmqcmne_ov_2A', 'kNtuMa19HAvQzrAFjXc0RQ', '9FDMUyJStI86B07bKQleow', 'fjGY-tlAkyPbIqVWmgBsEw', 'sZ
Uwd8g_crDamsZNvw2E_Q', 'aGpFWlu6vOsV1GRUmpWllA', '34gJ_KlP3RM6jotNT6TcDQ', 'KQP4Xl3kKNALIQse1BeCfA', 'NJB8VnfJZdEnIQxawIh5BA', 'Sx2OM_pi-
NsqNiZk1fMNXw', 'OWwNgrFIzXnUcqPpTx_53g', 'LMJ-eXo_GEKUJwuwOEHzwQ', 'bRLPoRvnbJbh8SiLjQRBlQ', '60lHfhyzSwACS1TRErqUjg', 'prt4Pvu69WGv5pc8
7RuLMQ', 'BkxbkiTa9_9A7j-0BRGEhQ', '8muABd0tSq9xrjCa9TltZQ', 'Q3oLto-17WmWK5-iazpZXQ', 'dbzRMMi4ZWiH37rt7RNizg', 'sQBk81qRtXTx28SHGqoX6
A', '1D5xuAHi9TXN-G5h9tPntg', '_AJekjxvLJlt7i9jXGvumA', 'Pce6723ATKGKuI4SG8JFYw', 'Q1fpW3qiL3oOe3b0si-0nA', 'd_KgMQIH2ekU789WghwKvg', '9L
wyHtKt5iRqlcFH-897aQ', 'We0BLw5csPSuodlGXdTXfA', 'CWqDJ99YEXmqrbXNy7NHLQ', 'PkPyd4PZvlZ-3O2ARq8Oqw', 'R5yR3y6Nlvm0TsvnHXkDSA', 'V0if1DQ0M
aTY-shWddXCsA', 'jqlPckTumnpCSvQ9jkiXwQ', 'AeucYo8J-rZjcq09Wuqsjw', 'LhcGcnw4eHaxzg58y2YYUA', 'Pr2bD8IxmJPOt3MPcwJbJA', 'sPEFql7fLp0oouFx

nooGpQ', 'PUh7fqdsIcLwA5LxD-WCSg', 'rzHZ3iYVQe_8h2e42DbUsw', 'lT5B7K8p57jQ9KlkwMkPmw', 'jdyV9MtV4_94VDP9aYqC0Q', 'yA77gvXdxFViK0mZcSive
A', 'K6MrbZeqKG35ZiucFfucvw', 'UD7Y1CqfY6mDmRwIuCf6nA', 'zo_32oay-tjJ6ckos-vVCw', 'xzAHL7X8LDiSUdFNtB_APA', 'Rf4ldVXX_zHwRauT6ICk2A', '5O
OuiG_basiJFfAln8UtAA', 'voqhxAies6n_95l8usebsg', 'uRhxqIdyu_jjoDKm8lSEDg', 'AiuxEpAxjPN7ciIrNupyWA', 'CxqXgJ672n08peXURbyUAw', '1DB-S0AKd
jkUhUBzSuwB6A', '6VFHoeOaYCwZMgosAf08XA', 'r3wE-UMb-ooPsn0805v-3w', 'eVTcstm1qKf0U7oppUZQFA', 'SgDWDjBr8fV7id1UnFaAFg', 'GJrGPKF2xxB06Es6
aH1VWg', 'Z2plM53IU6v6rVgBSgkopg', 'mqyH3CqjMufsm5FccDrjLA', 'C0F_DP9yO18mLF73IKhcGg', 'NrgIE-qEt-qnGWdZJP_BuA', '74oc3KCvVhvmo73Gk2gv3
Q', 'VrpOQV5Q5K_oIX-pqW6_Og', 'lmiDCrmas8TxRsbIGZX9Pg', 'gFQDzGzufke0NWzb_AJ4oQ', 'yY1MgFvfDVDhotp3xWJLlQ', 'Nf_8RYY4dTnvwaz96KMq-w', 'jK
n-ehjxG6l3o8gy1p3ZOQ', 'QiSSo2Kg-znygvxJz_67sA', 'h0UyMagnO4COVH5KDH3hZA', 'q6QBo2RC67GIHhDLG7ssDA', 'DHZ-kZ9sPDMmpnyFGXQ4hg', 'JXLIHoA6N
GmOl2LznYH3PA', 'hjvsUcztfGB2u130R9WCug', 'llzUppvv0t4noKubpIWCqw', 'DblIKwFdkkTNAeft7J4Taw', 'h4tfHGODErFCFrg6KEZ3xw', 'rnMXl9whqnQSZtBF
tlNy0g', '80c4xyemPV3XaMwlFtzVvw', 'lbg82t_mmEvlku3uVXn5TQ', 'PTqGpKNODTChfyGTRskFzg', 'ja7jzAqT7purefjl6wBxRQ', 'uvoJwcQxHfMfqDIX3a4xK
g', 'mRVLpO33EE6qHJqCfB48MA', 'TPracyTuh0oJikeyTA3w6A', 'wPjdB0viLu-i7YKGG3LJqA', 'jvhVRQ--8liQcdJvqDXwbw', 'hNqEqYaRD_wLgShfn0Kv4Q', 'Ya
Doz22xUMKYKgrtK5cZ5Q', 'VaBcoKQHmRnwtVkUypsvwA', 'G-jipHtQxKH5CLa47mn51A', '23ih9Fdu4v2KpBE1JDoVJw', 'WapiDiv7IjzfpIeczYwMIw', 'HcRkWWnGJ
_7iYJs30zXrxg', 'IT0sp0lM1ETIPQO1Z0c4oQ', '0CC4mLhyxpiq6O4nQpnQqg', 'v4IuHLQsLuJ5vVve6atgzQ', 'cRyNICH0mhjxagvSyVr60Q', 'Ti86eeLrIhpNHDpj
bla_fA', 'Ze4YIsiTXq2yrNM87Ue31A', 'N1EMQkSiTLcWHyG-VweCXg', '8n4SarsuI9Km6lEtydiE0w', 'HKTWp-uHABEbNIFK3i4LCw', 'amr0FHCs4EBYmy8_D2UGG
w', 'CdrOG7bBwAgHfkskPlA8xA', 'V4ND65sPYjWqUbLvpy7C-Q', 'NjOYSgr2LNMoSPi4e140Ig', 'ekfSera3-_wJrAiRaoKJ1A', 'hp6eRTTck1IQ1PwHwJx4cQ', 'z3
Hf0snMcvVTR--nC4uOLg', 'zZlHqWiCrCj0WKSNI1Nxlw', 'h5c9zka_FuMAIpDZ2X2Wew', 'I37ejvLDijHdqv6YOoVmIg', 'rKMJaCa-hUjeZDXEa6rKMg', '1sDu-W8uX
NZZLuWGcohy2g', 'U9j_xFnRVTXJGe2hNUZX7g', 'WmhvywSbr4B-JRcdoHwwVw', 'kV5vDL9e0HxN1FuxxSCBIQ', 'WTEcPmeEf5Pqv7diNm4wuA', 'y87DXWU4qy_OKtPJ
gnQFgg', 'L0Xrx4zdjPtPDcLcE0dJMg', 'NObfIhH5Lrdrw6PALek9LQ', 'lClvoNr2cNYM6hC_rG7yEg', 'taQoRWH9dr0jYPMIJC8e8w', '9hPoAujvzKe-pxeCaHchg
A', 'LsWb7sIq5Ywi01GIYOZzmQ', 'vLFk6nslFe35NKvlU2Tz9Q', 'nkhJYqK24Plr9OHciX5FUw', 'gRxcP_zpGnZ5PFeGgFM3UQ', 'e6f6ETqM6c-SF369w0wqDQ', 'AV
MN_xKLGaUW99lYK4XSHQ', 'bIzXegnjDpfWJazSh6Fu8g', 'BRydUrmdK4pZsbUvZ8tBAw', '4q3czSXkHBPkW21RdzpJ2A', 'Z-TW_TppnxT4TkFeavNaQw', '_L1NbjxnL
RJMmmjgVMdX7A', 'DqQf_U39vTcxd4nwNSbZ1w', 'c1J3fJozDVHeICaYc3t2ew', '4FJKLPkl-mvUD0iV9Q2p4Q', 'r3diS8VPXKbRCsrbPtXFkA', '_VeWBpintcoqjIcJ
dJ0_AQ', 'KyIPG6zBQj9Jqgl69tHtNQ', '4JMLe_6FHVUD2ZRR1r3vlQ', 'gqkKPS92L_qG03gHaEtIsQ', '7s9BkBonSqGHdVkpC_qdbw', '85ZDaEjICAF0-6OnvbeFa
g', 'E4HbTIHd9PVjUnEKpysaLw', 'ANetBXp0CdITDGhp17huGQ', '7lAr0ypf1jEA4qJGCWYaTg', '8qBnLmI1xd5ZrHeV7fBJZw', 'bAyT4kIMky9Gn8eFeZlSnw', 'D7
iY_stJhsdLxbiOR3gOYw', '6wX7uHlJMl_kaGvhNpNTnA', 'bN-oNETOC8RPVRWdBfyMPA', 'VFL4LY37bEvyDPobGoxvXg', 'fsSnng8XoT0sOKW472RQBA', 'Kx_IT3LiB
ivsuoaKnOdNOQ', 'TdjT7TgOWhBxhd_lgGygCA', 'xBOjmYlcTheqrvH-CveHwQ', 'IHt7toZZ4RlGvvz7aaKI6A', 'sVqdFnlxtZxYB80ZzZH8kw', 'FvqYM_zHngZZ5JFG
1hLrtg', '6Jh0nU3Xey6J0YfpSJLUwg', 'Z4Kn84VryzeJK-cGOfm6Pg', 'AbO18pf8bXG1qF9bb9L6wQ', 'QFj64b_eNwQaeJTvgkcvQA', 'fFmLNPsvu-mT0ewk-sG37
w', 'zanDVZZ6u4k3DaGbOa_pKA', 'DODrRvjHBklw-W8ueb1CtA', 'FTSmBX-WvzxFpH_vOiRIkg', 'x7twQq2PxhdPuKTOOhWo7Q', 'hG7tKFN-REC22Zm48IP-0A', 'md
JZA5p391bCOZnvAxHV3w', 'IDHrwv_RCildFvmfWTkj5Q', 'y_jn16HpJkfW4NfnKjfXLQ', 'x3GYZy8dMMw4j7eq-JN7Bg', '3DJLAIE0soSxYdvF50L5jQ', 'dinZ3pxeW
AG9ouMutcH63g', 'M1EQRRC91pofvyUucOEmsw', 'IyKosl07zfd4M1bMGNpxJA', 'e3RgGXBrk9BZ4utixycubA', 'HU4k7g73kIZHAtp6Ajv2-Q', 'XAAs2Y77fUqV26C2
ec5-7g', 'iZkGpLxDooIvqj96CS9BNg', '43WQN_FXtzMDYHUqMeaoTA', 'p7xa33mWZkhV5HECw0Kk-w', 'uuy_NVUeF1Kfc1E6R68MfA', 'mtlNpVemf0u2lIIMgqV8g
w', 'x1E4U9wXGElDDXIBzp4b3w', 'VYVxxvbc1g8aH7kFCP16Bg', 'RTnNGiTz2_EtTUtHGw7IYg', '1UXOwFKujaGNCNCijrdbsA', 'fwkidAyUXAHyXWtMxH0Wnw', 'xc
mFjlhZj1Wq-oeYEd8gfQ', 'WhAYw3hYWFA647Lrf2OJNg', 'n-0JFhD3V0FfN7sVmQs5lg', '7wSi6CwaxhR5rbphALIg5g', 'utIPtVx2gryj5cWJzb1iYg', 'FFluq9Bpo
G4ADhi9gI_LzQ', '7E5cTuA0Zt3xFbKNKaU9-Q', 'x3G-ZnbYzcFK3n2-8havGw', 'CK_Hg-YIpS_WSkN6iSWu5g', '-3wX7RLYheuW65F-IOF8KA', 'pKh4Pc96YbOg-6hB
OOajAA', 'sKtmrRvPMn5GRTnHYd41CA', '3nKW07v5Ot1WkDESYHOgJA', 'L5cvv2zsyP6AOZ1GjCP_dw', 'QGFLP49noPFZtFePsCcT4Q', 'yvATWMLzso1M0MQaqxq5R
w', 'oOayyTrVO-PbVeXXsh_JEQ', 'aBs5NRDe8q3Di7Bkreg92A', 'So32N7bSbUd1RwhFtI6jTQ', 'bjw9ZpD3nwlvt7osANXLFg', 'BiZvuNKi0bs_YwttHhhq4w', 'Dj
uPUzqoUTYi4ecFxXHbZA', 'v2bkiiHE_hJbKcyvlCk0_Q', 'f5udyzZy4SX-vphL_gUcBA', 'j1ndTaipSt1ATuY3kynR0A', 'DDPwyrdmtcQVkscBtn83Ow', 'pfITP0e6V
RrZiJphzxUjug', 'f3r3W8YPUcgRB41J8JE5EA', 'yvws3aMGz76FiQKIaatqqA', 'fwEPqYiyZCaB7QKY6a7h5g', 'VeBhqVUGbFbjgD08mEewyQ', 'pnDIFiXeU9tm3KtI
ISe64Q', 'DvpUCL8h6QVU-Cb258cCeQ', 'U6OFbI210w6QZlsCbXZZtA', 'LLZu1gmhfEgCgffnF1kZZQ', 'GpPKhPQ3mMX1_NJRLYcVrQ', '7kZM6JmGp0K-kfMWI-Cu_
w', '3AgV08454BMMO08vqBUJRw', 'tPTQ7dOR63nSkG4bza-lEA', 'SntjekOZ1a2PoTPyMnYWTw', 'v6rb-4YhADpuD99f3RKmgg', 'k5yMFDnQrBbc82qxRr_a8g', 'M5
jekfwRaw52RWpKNOkbqg', 'MfhoYW96OfXcjF4TgvxawQ', 'DZm20Z2tH25wKfyVkpmF4g', 'B_2NDDzcZELoxFbAPXwJLQ', 'fy8u9MsHxUk1h78r4AlmGg', 'kRLZKani2
Te8eSNqQaz5rg', 'dHVdhgc-9Wf4246uCkuAmg', 'N7wNlo3_EaHUWHiFDyGh5g', 'e0XJXCxhA8zTHG3_dAxpeQ', 'cSwEein5F83c1VP7n69C4A', '9WRfZHW7jYoj1JyY
ebA0_Q', '9H2kQNvpuHpK3YK4-r2R0Q', 'U4wiRY7GgB_SqREcXpewoQ', 'BGLDcowP44WPtsNbrP5wtQ', 'VjZZ_TKemzg1Z4yaJM6J7w', '2ELMxD4wdjzy9HdjwG0Jz
Q', 'RsUNADZTrJ5xFXVMXJC1MQ', '6urFO5AiJCIwmJVATmWCyA', 'OA9I8dKT9Uims5rp7YFKzA', 'zHJelsCN3Iknvc-0dRoktA', 'UnGPr6ZnGe9YOSZRopnS5g', 'Wz
aaorVCmUTQvu4mScunNg', 'CdXusNhIEfwM2nJaNxGqNw', 'bRFynDPNsgFTEOLU2Rz7Nw', 'yuB1BSvEuPjWmBG7PIGsOA', 'Qew7qNGE8ksnrZmwrOvAQg', 'UzsKmJRX6
x8E1PX6ClSw1Q', 'ZilAKKl3F4DqWSZPSnZfhw', 'RYkCobCYYZ2HqL6TdEqoqQ', 'yC1giQ-YlFaF2m0pgLRcow', 'ZJAYnSbP6-38rKCxcNZIGg', 'w8pZYayZQY5PmywZ
Q83KRw', 'ygv557H4_4VEqlDdPFr9Sw', 'rpJXWxl83G-b3yQ3u9rCTQ', '5ltY6j-HCw9MVer8puq65Q', 'absZ2-rmeH4NKo2VtWWDUQ', 'JLYqE7ITnkTupHgyOM6gv
g', 'ADhJUB2UfJ88aRNAIuzXlA', 'ihUHxlE5erKlMaxl52fTDw', 'L-cexm4-3KDqtL6kGfjzBA', '1K50PvXa6l5PBXZXhzkB-g', 'd2HaQ8x-3kF6DEg_tkOI7Q', 'Gd
XDLrMruYKSsEcb9Lvg6Q', 'icf8Tr75xv-QOnI7wZX9Xw', 'B9UvDrXUpZTNhjrV1BO0jA', 'ClioP8ShELrgCyxoD4wK2w', 'PX3bqC0n1syKPulnpe9NFw', '4qRz_FoKP
wHNvPdESPa8xA', '03vp3tenQ0aQOr0AE5j8Uw', '5mEtUgrWR1epdj2VFzwMaA', '-gg-WvyzPXVjmbqMIVBP4w', '2mBin9oV_a3QnbjidmbVaA', 'PR-EPp7Wru_Igb-i
r4Yh4Q', 'ZHLEWpQkQ35bY4QnTCT51Q', '4YfoQcMx1CFOZghBnzS_6g', 'ofEx4LotDyaxIjWB9zbQiQ', 's4iBNiw0iy3YZU3R7Os-pw', 'dkmeYJYEGhAV0kijSHpef
A', '2CjqjRENwrrcEPtM5fAujQ', '46QnJ8IGQBxL0sQ9cq2f9Q', 'NgnwzZ9YLaKj0HnRKhUH6g', 'h-3HtyPSh_1270egSeak7Q', 'XZ_484uR0Q_YHf61D5RO_A'], 'f
ans': 1012, 'average_stars': 3.64, 'type': 'user', 'compliments': {'profile': 110, 'cute': 209, 'funny': 561, 'plain': 921, 'writer': 29

```
0, 'list': 37, 'note': 589, 'photos': 287, 'hot': 1032, 'cool': 1521, 'more': 129}, 'elite': [2005, 2006, 2007, 2008, 2009, 2010, 2011, 2
012, 2013, 2014, 2015]}]
```

The third dataset we are going to load is information about business checkins reported by users on Yelp. Each checkin's information will be stored as a Python dictionary within an RDD. The dictionary consists of the following fields:

- "checkin_info":["an array of check ins with the format day-hour:number of check ins from hour to hour+1"]
- "business_id":"encrypted business id"
- "type":"checkin"

In [5]:
```
# load the data about business checkins reported by users on Yelp in an RDD
# each RDD element is a Python dictionary parsed from JSON using json.loads()
checkins_rdd = get_rdd_from_path('/ocean/projects/cis220071p/shared/data/yelp_academic_dataset_checkin.json')
print (checkins_rdd.count())
print (checkins_rdd.take(2))
```

```
45166
[{'checkin_info': {'9-5': 1, '7-5': 1, '13-3': 1, '17-6': 1, '13-0': 1, '17-3': 1, '10-0': 1, '18-4': 1, '14-6': 1}, 'type': 'checkin',
'business_id': 'cE27W9VPgO88Qxe4ol6y_g'}, {'checkin_info': {'22-5': 1, '9-5': 1, '15-1': 1, '15-4': 1, '16-2': 1, '21-4': 1, '13-1': 1,
'14-4': 1, '12-5': 1, '12-1': 1}, 'type': 'checkin', 'business_id': 'mVHrayjG3uZ_RLHkLj-AMg'}]
```

The fourth dataset we are going to load is information about business reviews written by users on Yelp. Each review's data will be stored as a Python dictionary within an RDD. The dictionary consists of the following fields:

- "review_id":"encrypted review id"
- "user_id":"encrypted user id"
- "business_id":"encrypted business id"
- "stars":star rating rounded to half-stars
- "date":"date formatted like 2009-12-19"
- "text":"review text"
- "useful":number of useful votes received
- "funny":number of funny votes received
- "cool": number of cool review votes received
- "type": "review"

In [6]:
```
# load the data about business reviews written by users on Yelp in an RDD, limited to businesses in Pittsburgh due to DataBricks computat
# each RDD element is a Python dictionary parsed from JSON using json.loads()
reviews_rdd = get_rdd_from_path('/ocean/projects/cis220071p/shared/data/yelp_academic_dataset_review_pittsburgh.json')
print (reviews_rdd.count())
print (reviews_rdd.take(2))
```

```
62608
[{'votes': {'funny': 3, 'useful': 7, 'cool': 7}, 'user_id': 'JbAeIYc89Sk8SWmrBCJs9g', 'review_id': 'fBQ69-NU9ZyTjjS7Tb5tww', 'stars': 5,
'date': '2013-06-10', 'text': "THANK YOU ROB! i truly appreciated all the help i received from this agent today who was able to removed t
he extra charges on my bill that the Pasadena Verizon Store on Lake was charging me on my bill for upgrading my phone.   When i went in i
was having problems with my Blacberry and had to switch to the Iphone last week. Rob from the Pennsylvania store who i was connected toda
y was able to look at my bill and all the notes and correct the problem immediately. Great Customer Service! He even set up a FOLLOW UP P
hone Call with me On July 5th to make sure the credit goes through on my bill...I can't thank him enough!!!!", 'type': 'review', 'busines
s_id': 'HZdLhv6COCleJMo7nPl-RA'}, {'votes': {'funny': 1, 'useful': 1, 'cool': 1}, 'user_id': 'l_szjd-ken3ma6oHDkTYXg', 'review_id': 'CFiL
h7WvH7dM3qVZvNiacQ', 'stars': 2, 'date': '2013-12-23', 'text': "After waiting for almost 30 minutes to trade in an old phone part of the
buy back program, our customer service rep incorrectly processed the transaction. This led to us waiting another 30 minutes for him to co
rrect it. Don't visit this store if you want pleasant or good service.", 'type': 'review', 'business_id': 'HZdLhv6COCleJMo7nPl-RA'}]
```

Finally, we will load two lists. The first list consists of male names, and the second list consists of female names, to map Yelp user names to gender.

In [7]:
```
# helper function to load a list of names from a publicly accessible url
def get_names_from_path(path):
    file_reader = open(path, 'r')
    str_contents = file_reader.readlines()
    str_contents = [x.strip() for x in str_contents]
    result = str_contents[6:]
    return result

male_names = get_names_from_path('/ocean/projects/cis220071p/shared/data/male.txt')
print('First five male names: ', male_names[:5])
print('Number of male names: ', len(male_names))

female_names = get_names_from_path('/ocean/projects/cis220071p/shared/data/female.txt')
print('First five female names: ', female_names[:5])
print('Number of female names: ', len(female_names))
```

```
First five male names:  ['Aamir', 'Aaron', 'Abbey', 'Abbie', 'Abbot']
Number of male names:  2943
First five female names:  ['Abagael', 'Abagail', 'Abbe', 'Abbey', 'Abbi']
Number of female names:  5001
```

Part 1: Exploratory Data Analysis

Performing some exploratory analysis is a great step toward understanding the data before building any statistical machine learning models on it.

Please replace `<FILL IN>` with your solution. This is the general form that exercises will take. Exercises will include an explanation of what is expected, followed by code cells where one cell will have one or more `<FILL IN>` sections. The cell that needs to be modified will have `# TODO: Replace <FILL IN> with appropriate code` on its first line.

In [8]:
```
print ('Number of businesses: ', businesses_rdd.count())
print ('Number of users: ', users_rdd.count())
print ('Number of checkins: ', checkins_rdd.count())
print ('Number of reviews: ', reviews_rdd.count())
```

```
Number of businesses:  61184
Number of users:  366715
Number of checkins:  45166
Number of reviews:  62608
```

Question1: Print the top 5 business categories by frequency and the number of times they appear in the businesses data.

In [9]:
```python
from collections import Counter

# Extract the categories from the RDD
categories_rdd = businesses_rdd.flatMap(lambda business: business["categories"])

# Count the occurrences of each category
category_count = categories_rdd.countByValue()

# Find the top 5 business categories by frequency using Counter
top_categories = Counter(category_count).most_common(5)

# Print the top 5 business categories and their frequencies
for category, count in top_categories:
    print(f"{category}: {count}")
```

```
Restaurants: 21892
Shopping: 8919
Food: 7862
Beauty & Spas: 4738
Nightlife: 4340
```

Question2: Print the top 5 cities by frequency and the number of times they appear in the businesses data.

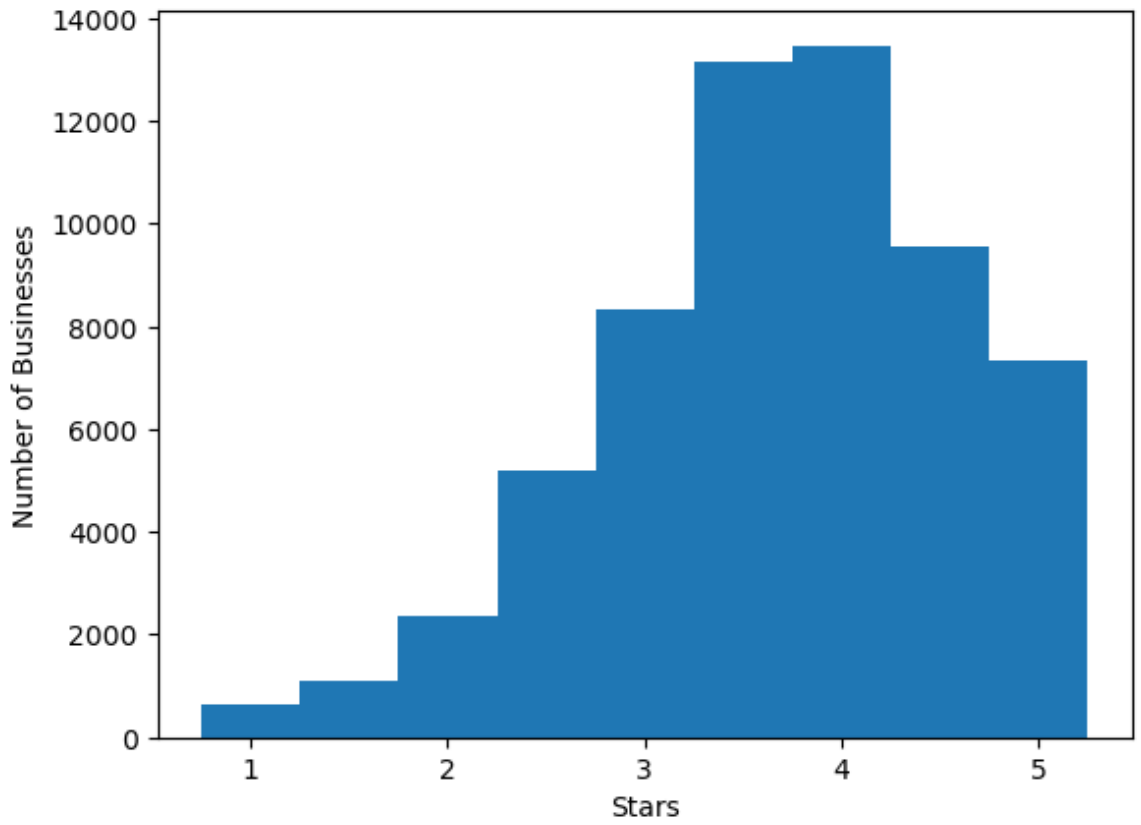
In [10]:
```python
# TODO: Replace <FILL IN>
cities_rdd = businesses_rdd.map(lambda business: business["city"])

# Count the occurrences of each category
city_count = cities_rdd.countByValue()

# Find the top 5 business cities by frequency using Counter
top_cities = Counter(city_count).most_common(5)

# Print the top 5 cities and their frequencies
for city, count in top_cities:
    print(f"{city}: {count}")
```

```
Las Vegas: 13601
Phoenix: 8410
Charlotte: 4224
Scottsdale: 4039
Edinburgh: 3031
```

Question3: Plot the histogram of stars received by businesses.

In [11]:
```
# TODO: Replace <FILL IN>

businesses_stars_counts = businesses_rdd.map(lambda business: business["stars"]).collect()
plt.hist(businesses_stars_counts, bins=[x/2-0.25 for x in range(2, 12)])
plt.xlabel('Stars')
plt.ylabel('Number of Businesses')
```

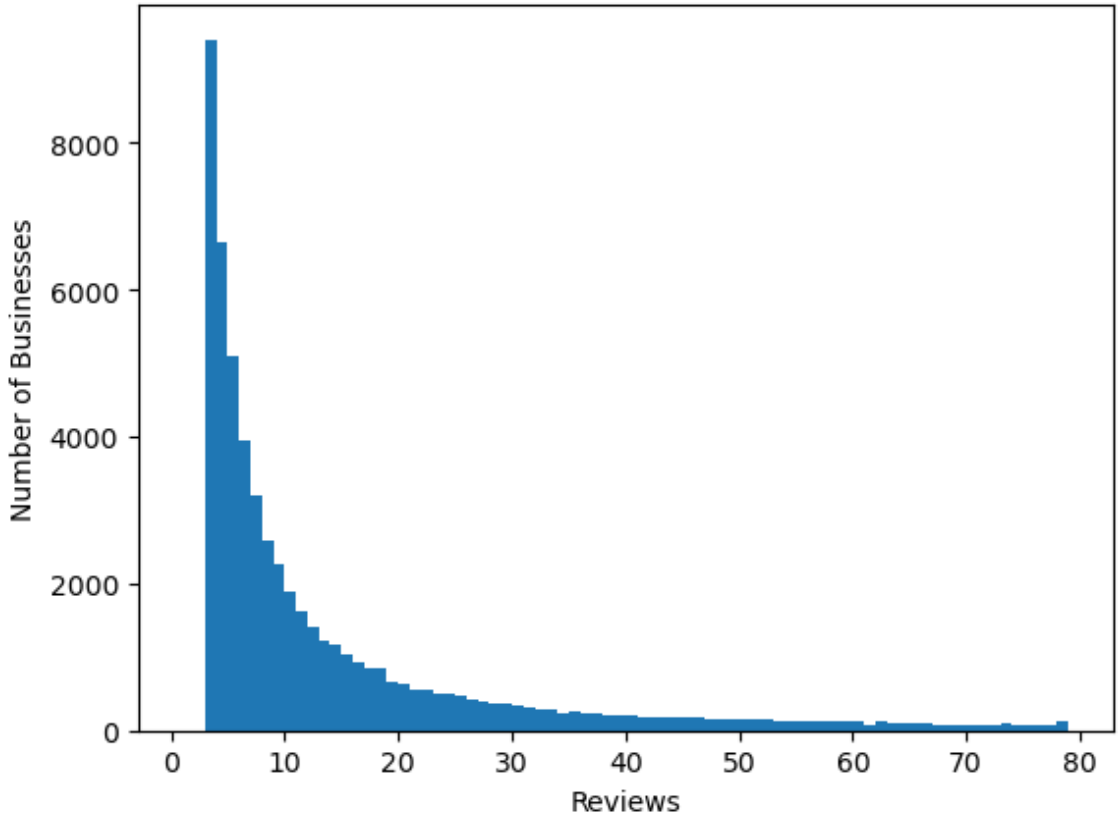
Out[11]:
```
Text(0, 0.5, 'Number of Businesses')
```



Question4: Plot the histogram of number of reviews received by businesses.

In [12]:
```
# TODO: Replace <FILL IN>

businesses_review_counts = businesses_rdd.map(lambda business: business["review_count"]).collect()

plt.hist(businesses_review_counts, bins=range(1,80))
plt.xlabel('Reviews')
plt.ylabel('Number of Businesses')
```

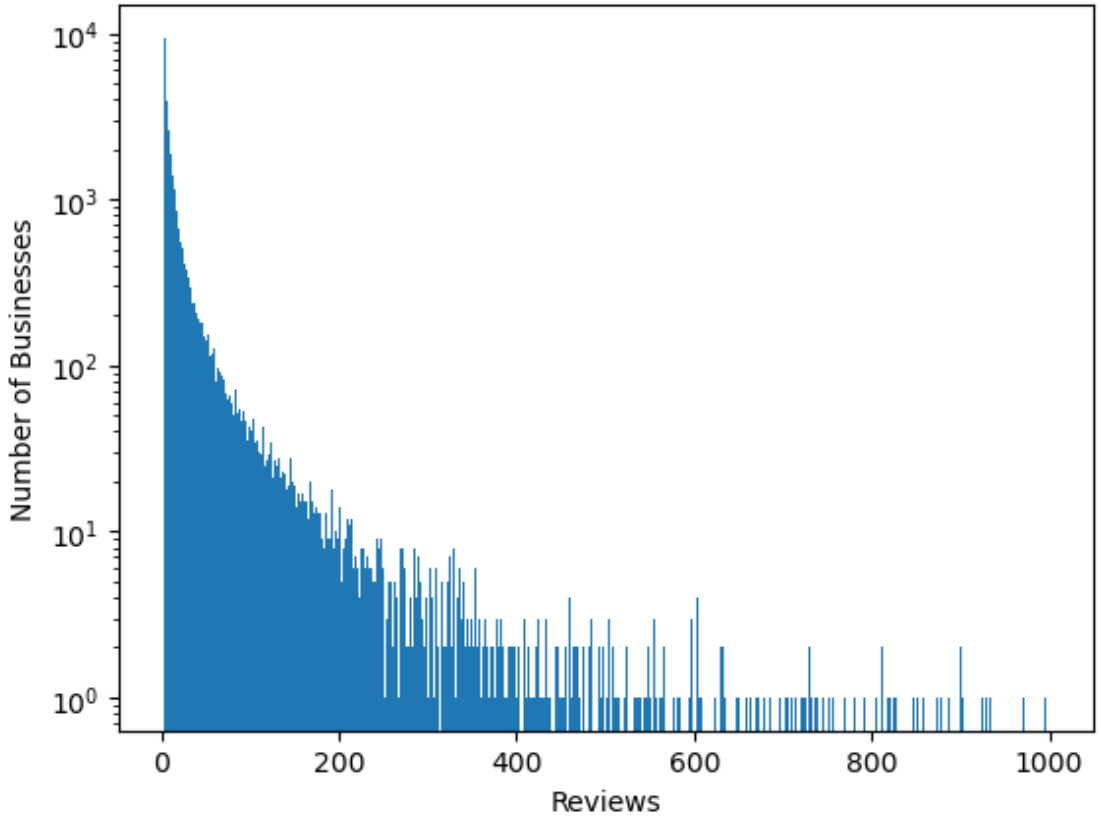
Out[12]:
```
Text(0, 0.5, 'Number of Businesses')
```



Question5: Plot the above histogram but now on a log-log scale using `bins=range(1,1000)`. Do you see a Power Law relationship in the plot? Explain your answer.

Answer: Yes. We can see a straight line in the histogram plotted between log reviews and log businesses. This explains an exponential relationship between the two features. As we can see, there a large number of businesses that received log of reviews in the range of 10 and 100. But there are very few businesses that have got a large number of reviews.

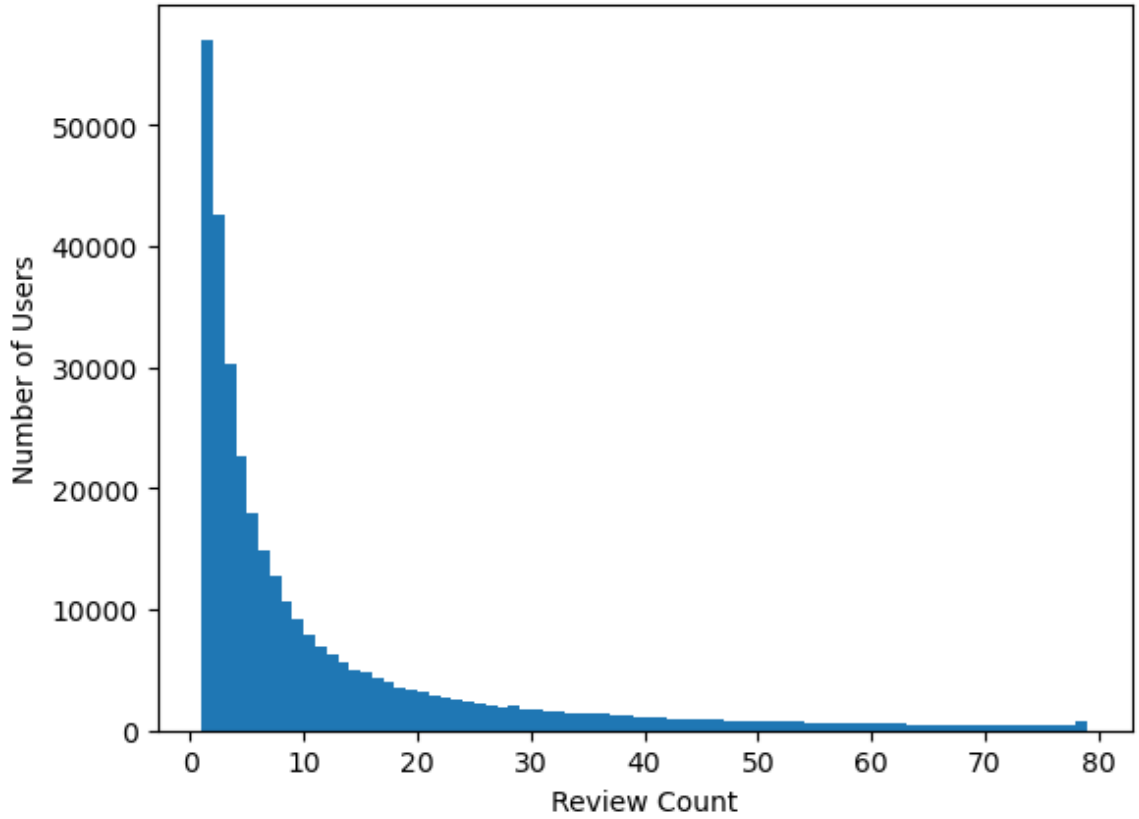
In [13]:
```
# TODO: Replace <FILL IN>
# Plot the histogram on a log-log scale
fig, ax = plt.subplots()
hist, bins, _ = ax.hist(businesses_review_counts, bins=range(1, 1000), log=True)
plt.xlabel('Reviews')
plt.ylabel('Number of Businesses')
plt.show()
```



Question6: Plot the histogram of number of reviews written by users.

In [14]:
```
# TODO: Replace <FILL IN>

users_review_counts = users_rdd.map(lambda user: user["review_count"]).collect()

plt.hist(users_review_counts, bins=range(1,80))
plt.xlabel('Review Count')
plt.ylabel('Number of Users')
```

Out[14]:
```
Text(0, 0.5, 'Number of Users')
```

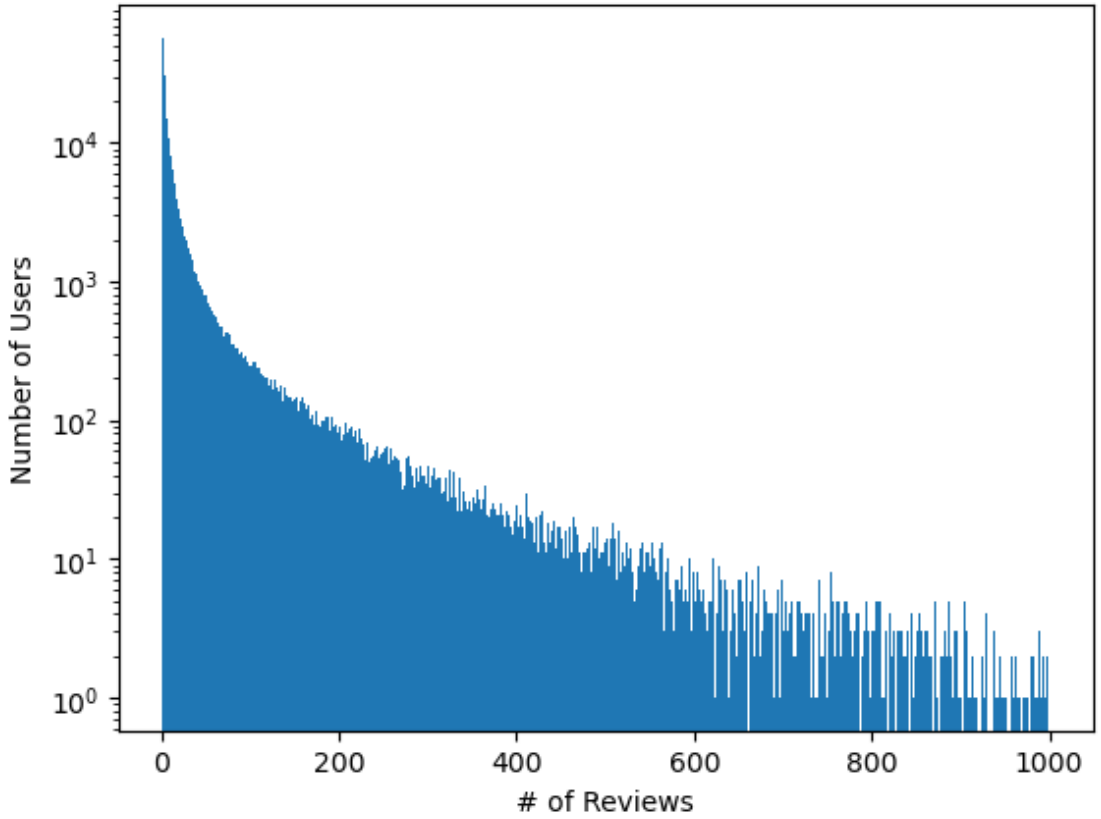


Question7: Plot the above histogram but now on a log-log scale using `bins=range(1,1000)`. Do you see a Power Law relationship in the plot? Explain your answer.

Answer: Yes. We can see a straight line in the histogram plotted between log reviews and log users. This explains an exponential relationship between the two features. As we can see, there a large number of users that wrote few reviews. But there are very few large # of reviews that have been given by users.

In [15]:
```
# TODO: Replace <FILL IN>
fig, ax = plt.subplots()
hist, bins, _ = ax.hist(users_review_counts, bins=range(1, 1000), log=True)
plt.xlabel('# of Reviews')
plt.ylabel('Number of Users')
```

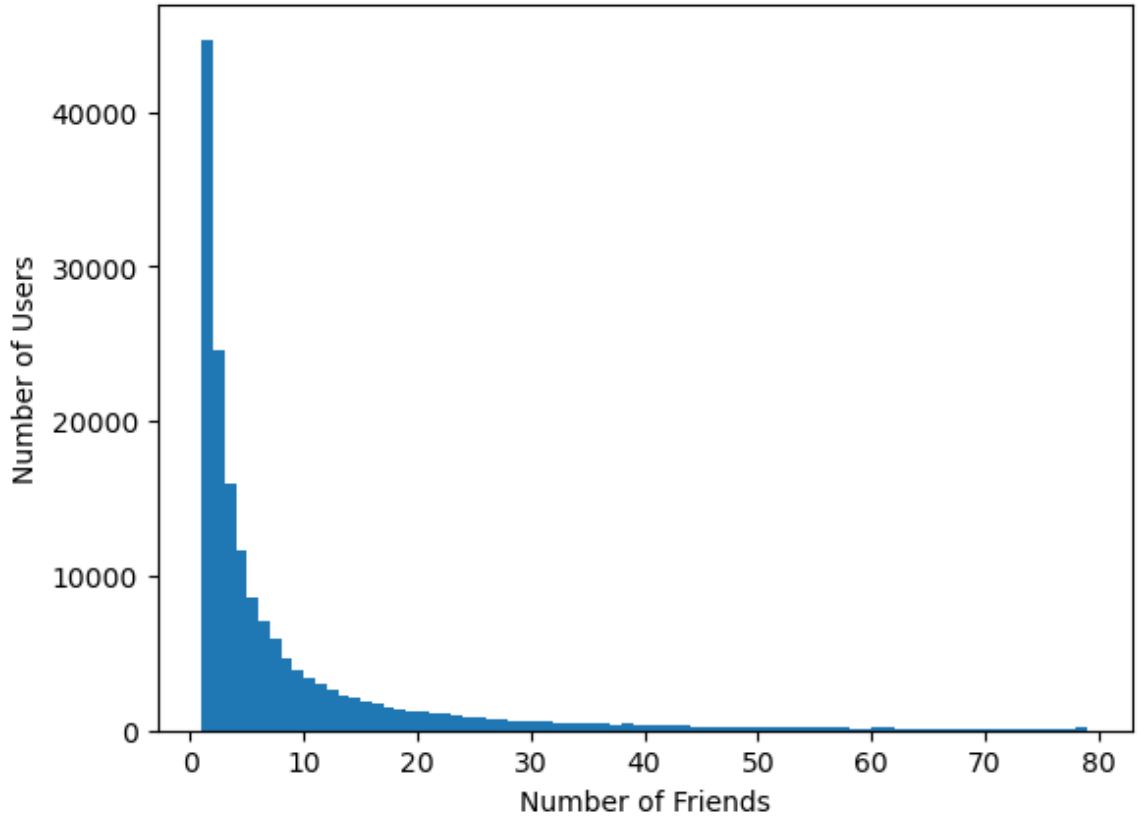
Out[15]:
```
Text(0, 0.5, 'Number of Users')
```



Question8: Plot the histogram of number of friends a Yelp user has.

In [16]:
```
# TODO: Replace <FILL IN>

user_friend_counts = users_rdd.map(lambda user: len(user["friends"])).collect()

plt.hist(user_friend_counts, bins=range(1,80))
plt.xlabel('Number of Friends')
plt.ylabel('Number of Users')
```

Out[16]:
```
Text(0, 0.5, 'Number of Users')
```

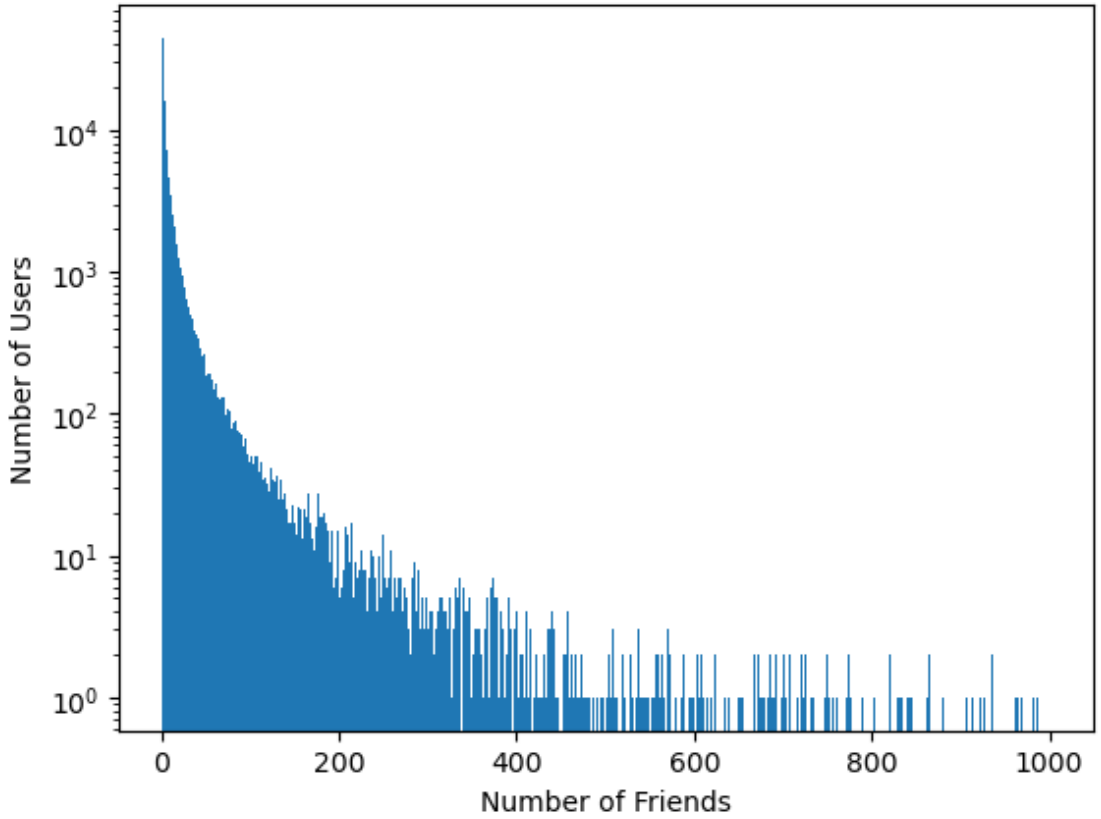


Question9: Plot the above histogram but now on a log-log scale. Do you see a Power Law relationship in the plot? Explain your answer.

Answer: Yes. We can see a straight line in the histogram plotted between log users and log friends. This explains an exponential relationship between the two features. As we can see, there a large number of users that have few friends. But there are very few users that have lot of friends.

In [17]:
```
# TODO: Replace <FILL IN>
fig, ax = plt.subplots()
hist, bins, _ = ax.hist(user_friend_counts, bins=range(1, 1000), log=True)
plt.xlabel('Number of Friends')
plt.ylabel('Number of Users')
```

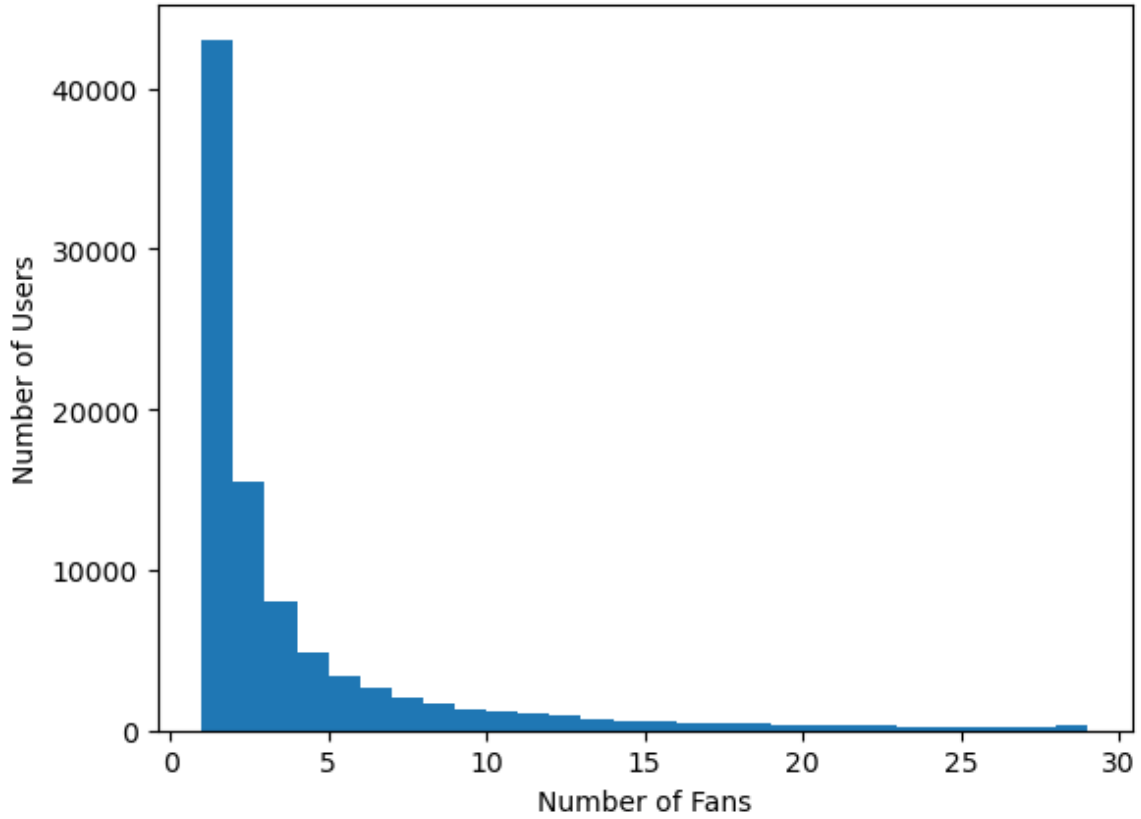
Out[17]:
```
Text(0, 0.5, 'Number of Users')
```



Question10: Plot the histogram of number of fans a Yelp user has.

In [18]:
```
# TODO: Replace <FILL IN>

users_fan_counts = users_rdd.map(lambda user: user["fans"]).collect()

plt.hist(users_fan_counts, bins=range(1,30))
plt.xlabel('Number of Fans')
plt.ylabel('Number of Users')
```

Out[18]:
```
Text(0, 0.5, 'Number of Users')
```

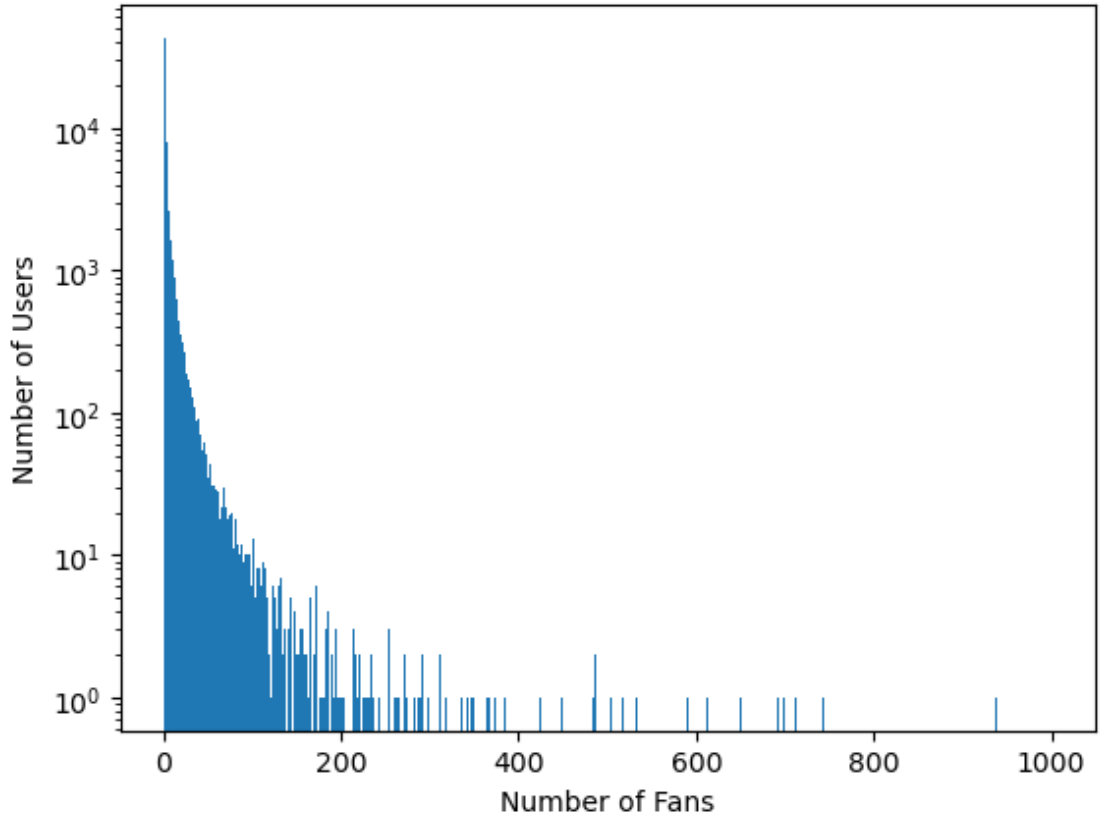


Question11: Plot the above histogram but now on a log-log scale. Do you see a Power Law relationship in the plot? Explain your answer.

Answer: Yes. We can see a straight line in the histogram plotted between log users and log fans. This explains an exponential relationship between the two features. As we can see, there a large number of users that have few fans. But there are very few users that have lot of fans.

In [19]:
```
# TODO: Replace <FILL IN>
fig, ax = plt.subplots()
hist, bins, _ = ax.hist(users_fan_counts, bins=range(1, 1000), log=True)
plt.xlabel('Number of Fans')
plt.ylabel('Number of Users')
```

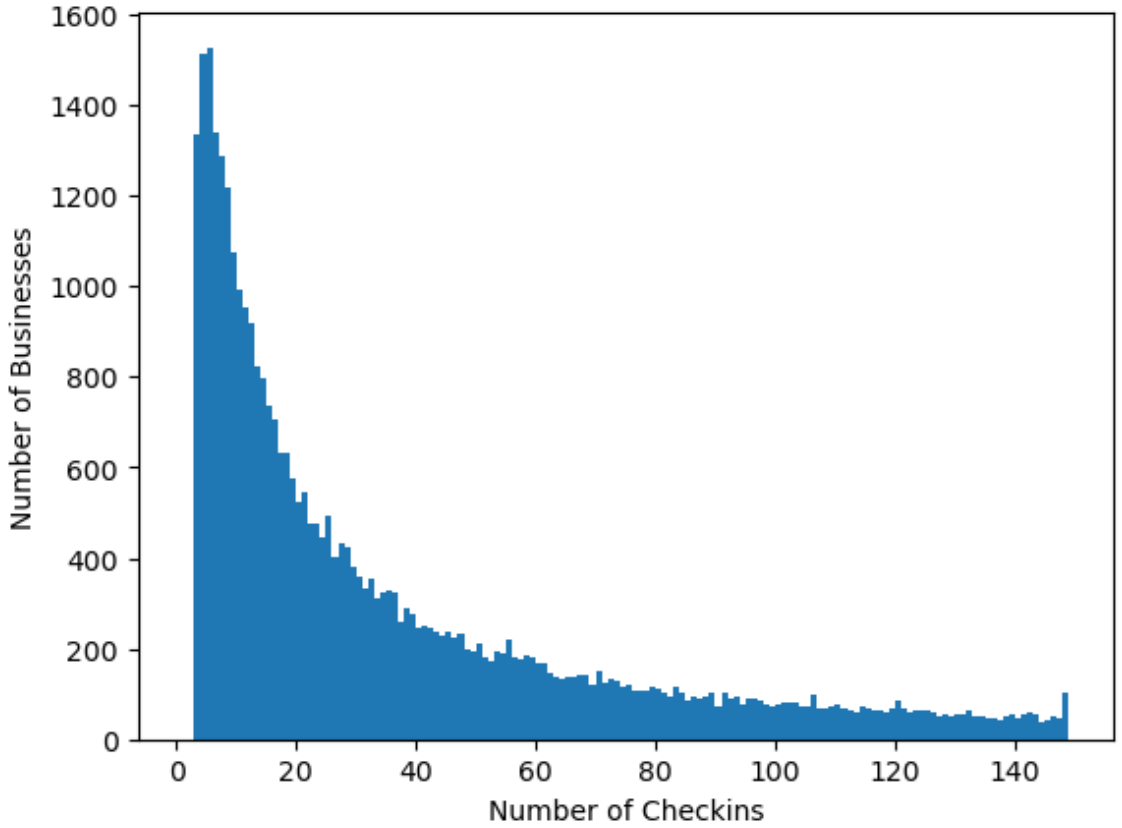
Out[19]:
```
Text(0, 0.5, 'Number of Users')
```



Question12: Plot the histogram of number of checkins per Yelp business.

In [20]:
```
# TODO: Replace <FILL IN>

business_checkin_counts = checkins_rdd.map(lambda checkin: sum(checkin["checkin_info"].values())).collect()

plt.hist(business_checkin_counts, bins=range(1,150))
plt.xlabel('Number of Checkins')
plt.ylabel('Number of Businesses')
```

Out[20]:
```
Text(0, 0.5, 'Number of Businesses')
```

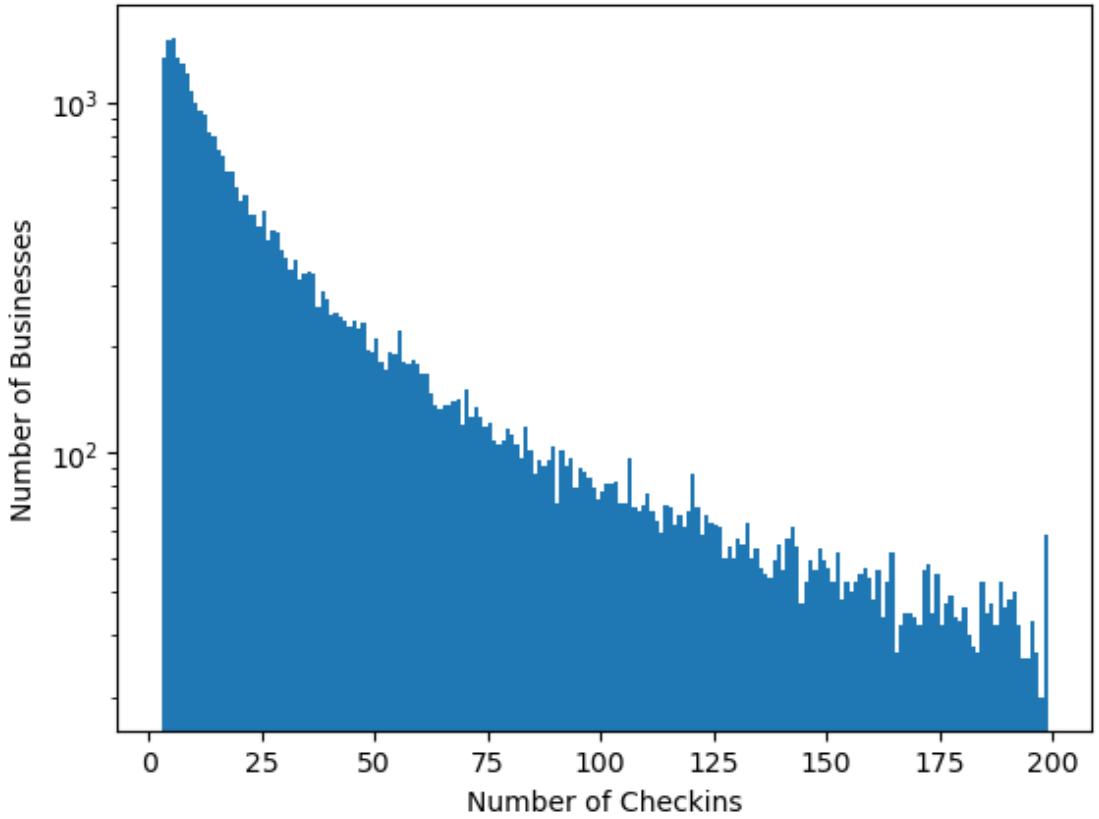


Question13: Plot the above histogram but now on a log-log scale using `bins=range(3,200)`. Do you see a Power Law relationship in the plot? Explain your answer.

Answer: No. Here, power law is violated. As we can see, there a large number of businesses that have few checkins. But there are also many businesses that have lot of checkins.

In [21]:
```
# TODO: Replace <FILL IN>
fig, ax = plt.subplots()
hist, bins, _ = ax.hist(business_checkin_counts, bins=range(3, 200), log=True)
plt.xlabel('Number of Checkins')
plt.ylabel('Number of Businesses')
```

Out[21]:
```
Text(0, 0.5, 'Number of Businesses')
```



Question14: Find the maximum value of checkins per business. Filter to obtain business IDs of businesses that had these maximum number of checkins. Fill in the code required to carry out these steps.

In [22]:
```
# TODO: Replace <FILL IN>
max_checkin_count = checkins_rdd.map(lambda checkin: len(checkin["checkin_info"])).max()
business_ids_with_max_checkins = checkins_rdd \
    .filter(lambda checkin: len(checkin["checkin_info"]) == max_checkin_count) \
    .map(lambda checkin: checkin["business_id"]).collect()
len(business_ids_with_max_checkins)
```

Out[22]:
```
41
```

In [23]:
```
# TODO: Replace <FILL IN>
business_names_with_max_checkins = businesses_rdd \
    .filter(lambda business: business["business_id"] in business_ids_with_max_checkins) \
    .map(lambda x: (x['name'], x['city'])).collect()
business_names_with_max_checkins
```

Out[23]:
```
[('Charlotte Douglas International Airport', 'Charlotte'),
 ('Phoenix Sky Harbor International Airport', 'Phoenix'),
 ('Pho Kim Long', 'Las Vegas'),
 ('McCarran International Airport', 'Las Vegas'),
 ('The California Hotel & Casino', 'Las Vegas'),
 ('Golden Nugget Hotel & Casino', 'Las Vegas'),
 ('Rio All Suites Hotel & Casino', 'Las Vegas'),
 ('24 Hour Fitness', 'Las Vegas'),
 ('Excalibur Hotel', 'Las Vegas'),
 ('Ellis Island Casino & Brewery', 'Las Vegas'),
 ('Las Vegas Athletic Club', 'Las Vegas'),
 ('Orleans Hotel & Casino', 'Las Vegas'),
 ('Palms Casino Resort', 'Las Vegas'),
 ('Bellagio Hotel', 'Las Vegas'),
 ('New York - New York', 'Las Vegas'),
 ('Paris Las Vegas Hotel & Casino', 'Las Vegas'),
 ('Monte Carlo Hotel And Casino', 'Las Vegas'),
 ('The Venetian Resort Hotel Casino', 'Las Vegas'),
 ('The Mirage', 'Las Vegas'),
 ('MGM Grand Hotel', 'Las Vegas'),
 ('Treasure Island, LLC', 'Las Vegas'),
 ('The Peppermill Restaurant & Fireside Lounge', 'Las Vegas'),
 ('Flamingo Las Vegas Hotel & Casino', 'Las Vegas'),
 ('Luxor Hotel And Casino Las Vegas', 'Las Vegas'),
 ('Mandalay Bay Resort & Casino', 'Las Vegas'),
 ('Grand Lux Cafe', 'Las Vegas'),
 ('Market Street Cafe', 'Las Vegas'),
 ('Caesars Palace Las Vegas Hotel & Casino', 'Las Vegas'),
 ('Las Vegas Athletic Club', 'Las Vegas'),
 ('South Point Hotel, Casino and Spa', 'Las Vegas'),
 ('Starbucks', 'Las Vegas'),
 ('Planet Hollywood Las Vegas Resort & Casino', 'Las Vegas'),
 ('Earl of Sandwich', 'Las Vegas'),
 ('InterContinental Alliance Resorts THE PALAZZO', 'Las Vegas'),
 ('Las Vegas Athletic Club', 'Las Vegas'),
 ('Encore', 'Las Vegas'),
 ('ARIA Hotel & Casino', 'Las Vegas'),
 ('Vdara Hotel', 'Las Vegas'),
 ('The Cosmopolitan of Las Vegas', 'Las Vegas'),
 ("Bally's Las Vegas Hotel & Casino", 'Las Vegas'),
 ('Hard Rock Hotel & Casino', 'Las Vegas')]
```

Question15: Why do you think the above list sees much higher checkins than other businesses in the dataset?

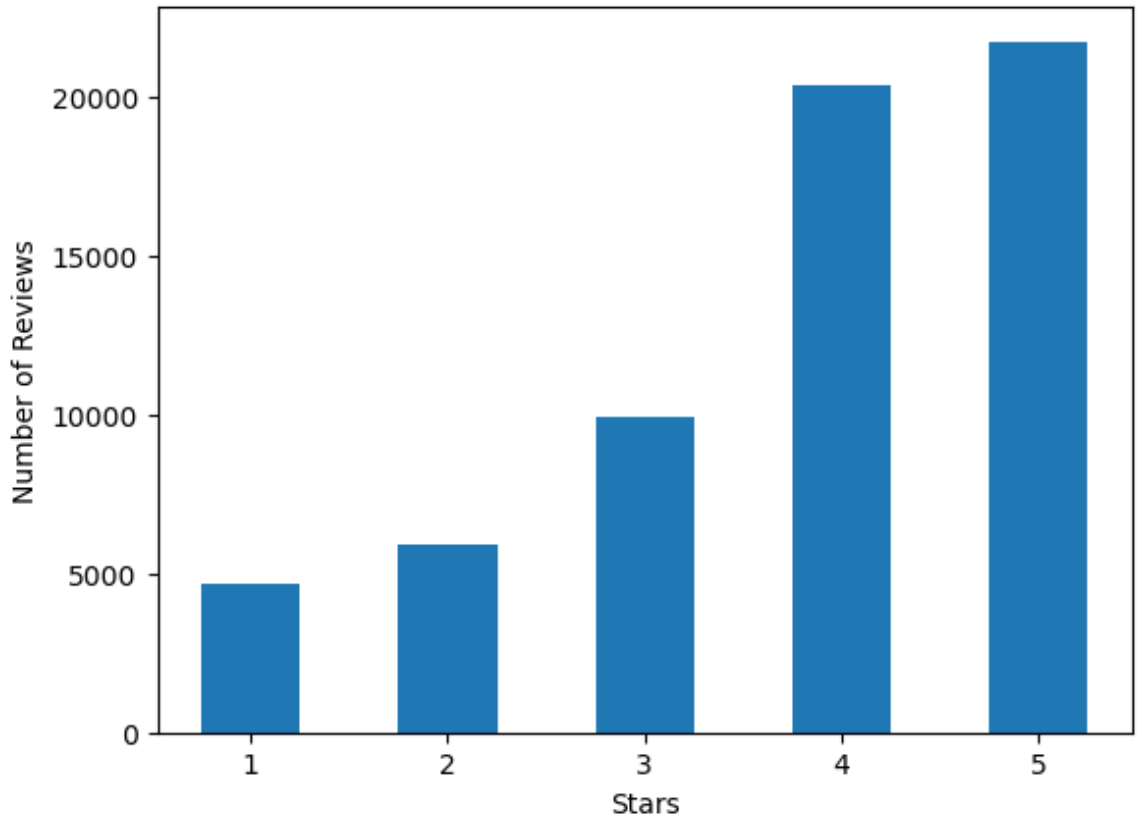
Answer: Some possible reasons are the following:

1. High-quality services: This attracts more customers, leading to more check-ins.
2. Convenient location: Located in prime locations near airport, making them more accessible to potential customers.
3. Social media influence: More reviews on platforms like yelp, google reviews attracting more customers

Question16: Plot a histogram of the stars associated with business reviews.

In [24]:
```
# TODO: Replace <FILL IN>

review_stars_counts = reviews_rdd.map(lambda review: review["stars"]).collect()
plt.hist(review_stars_counts, bins=[x/2-0.25 for x in range(2, 12)])
plt.xlabel('Stars')
plt.ylabel('Number of Reviews')
```

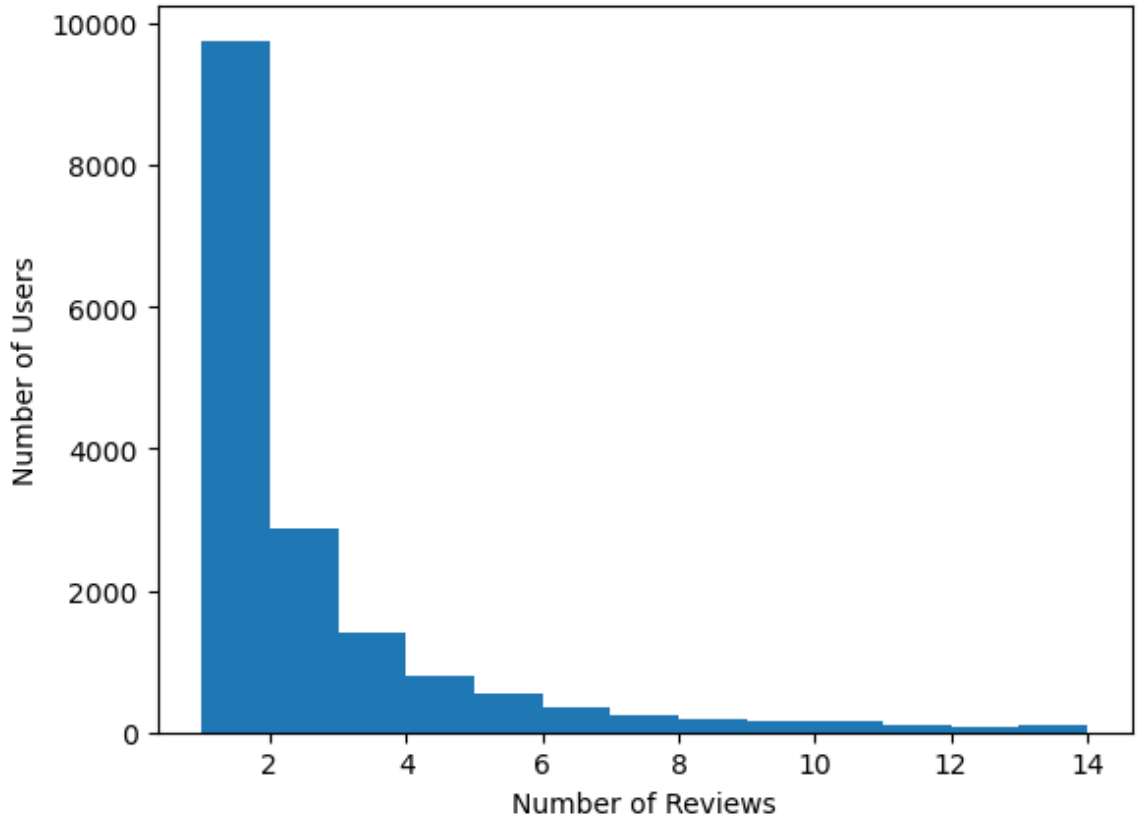
Out[24]:
```
Text(0, 0.5, 'Number of Reviews')
```



Question17: Plot a histogram of the number of reviews written per Yelp user.

In [25]:
```
# TODO: Replace <FILL IN>

user_review_counts = list(reviews_rdd.map(lambda review: review["user_id"]).countByValue().values())

plt.hist(user_review_counts, bins=[x for x in range(1, 15)])
plt.xlabel('Number of Reviews')
plt.ylabel('Number of Users')
```

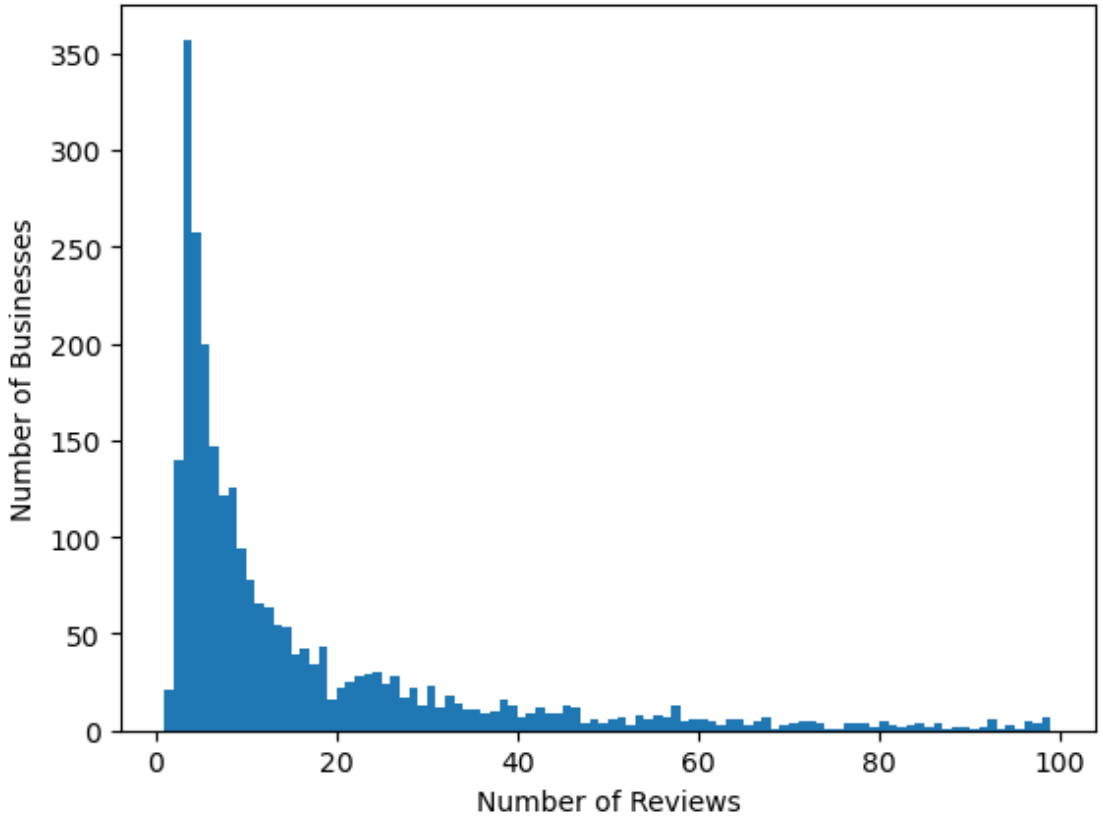
Out[25]: Text(0, 0.5, 'Number of Users')



Question18: Plot a histogram of the number of reviews written per Yelp business.

In [26]:
```
# TODO: Replace <FILL IN>

business_review_counts = list(reviews_rdd.map(lambda review: review["business_id"]).countByValue().values())

plt.hist(business_review_counts, bins=[x for x in range(1, 100)])
plt.xlabel('Number of Reviews')
plt.ylabel('Number of Businesses')
```

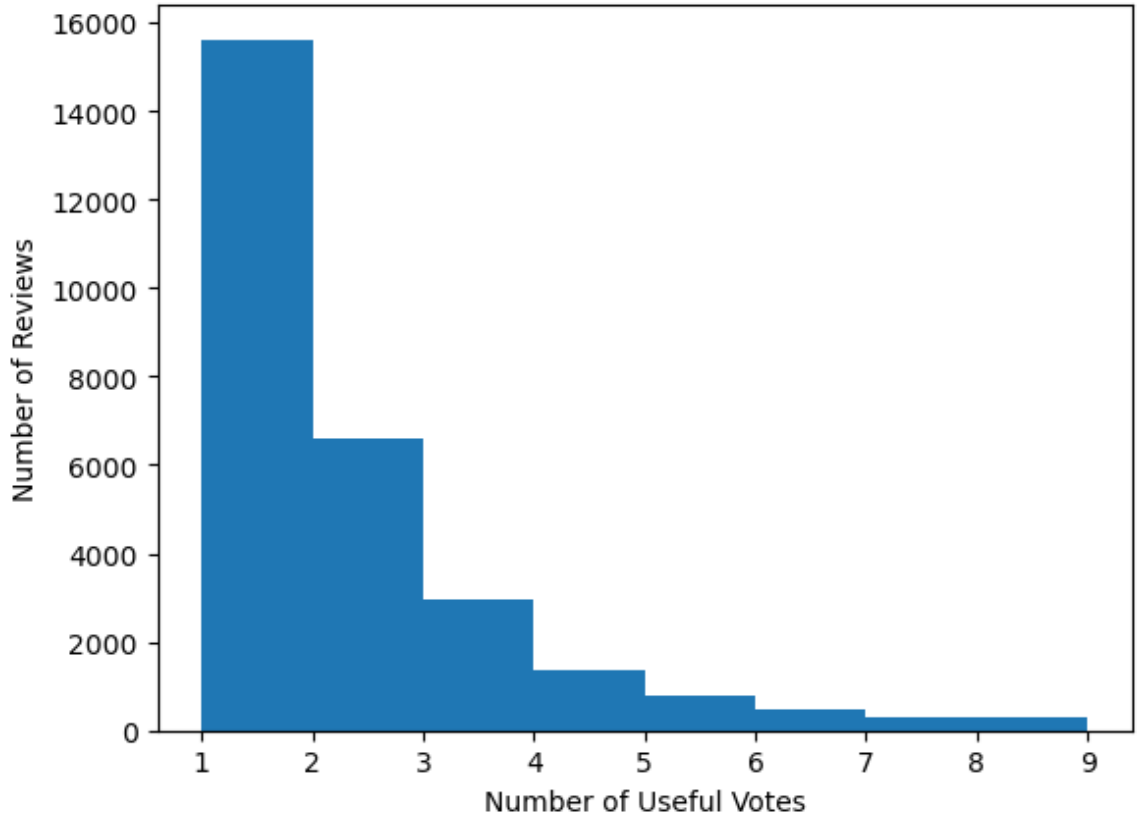
Out[26]: Text(0, 0.5, 'Number of Businesses')



Question19: Plot a histogram of the number of useful votes received by Yelp reviews.

In [27]:
```
# TODO: Replace <FILL IN>

review_useful_counts = reviews_rdd.map(lambda review: review["votes"]["useful"]).collect()

plt.hist(review_useful_counts, bins=[x for x in range(1, 10)])
plt.xlabel('Number of Useful Votes')
plt.ylabel('Number of Reviews')
```

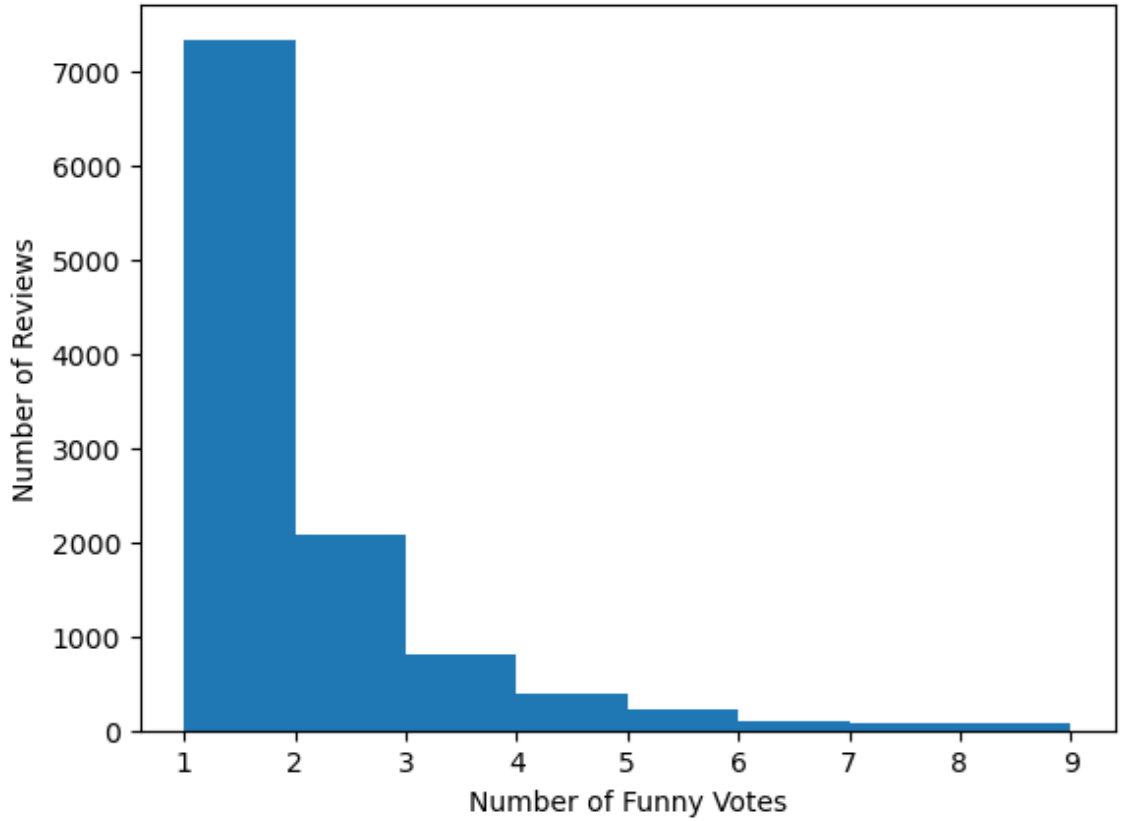
Out[27]:
```
Text(0, 0.5, 'Number of Reviews')
```



Question20: Plot a histogram of the number of funny votes received by Yelp reviews.

In [28]:
```
# TODO: Replace <FILL IN>

review_funny_counts = reviews_rdd.map(lambda review: review["votes"]["funny"]).collect()

plt.hist(review_funny_counts, bins=[x for x in range(1, 10)])
plt.xlabel('Number of Funny Votes')
plt.ylabel('Number of Reviews')
```

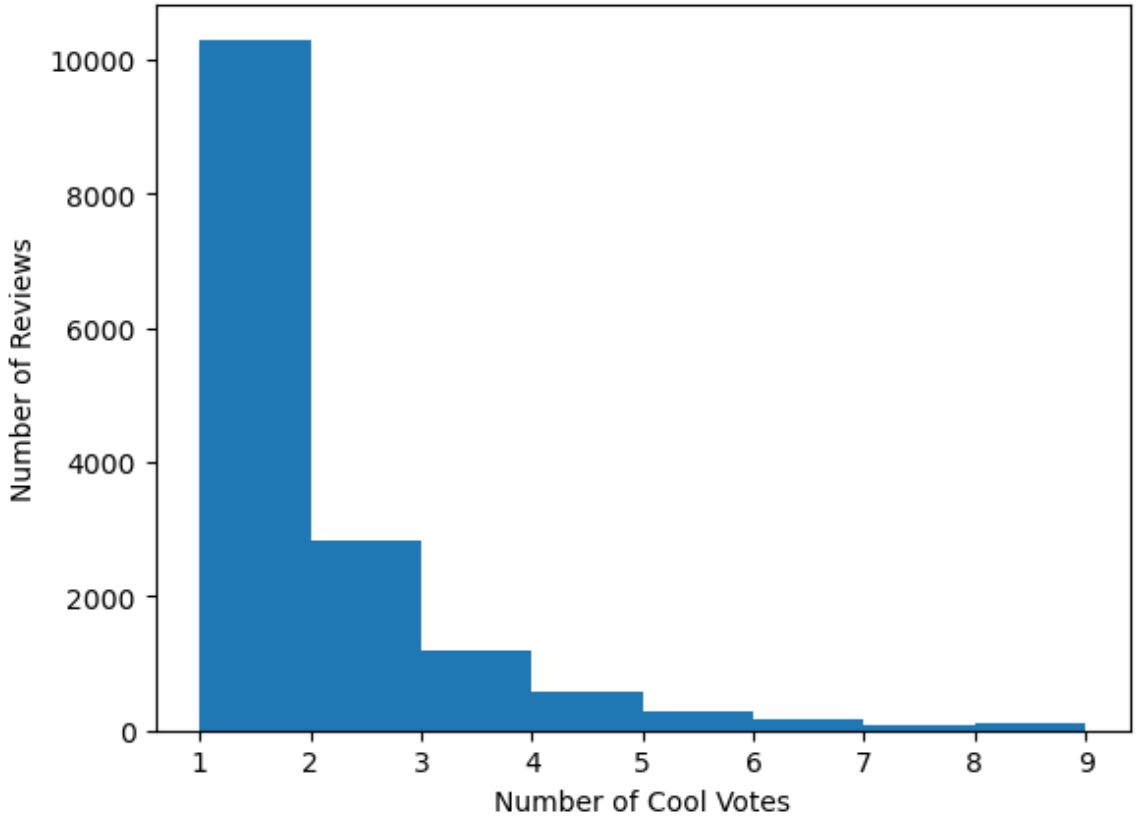
Out[28]:
```
Text(0, 0.5, 'Number of Reviews')
```



Question21: Plot a histogram of the number of cool votes received by Yelp reviews.

In [29]:
```
# TODO: Replace <FILL IN>

review_cool_counts = reviews_rdd.map(lambda review: review["votes"]["cool"]).collect()

plt.hist(review_cool_counts, bins=[x for x in range(1, 10)])
plt.xlabel('Number of Cool Votes')
plt.ylabel('Number of Reviews')
```

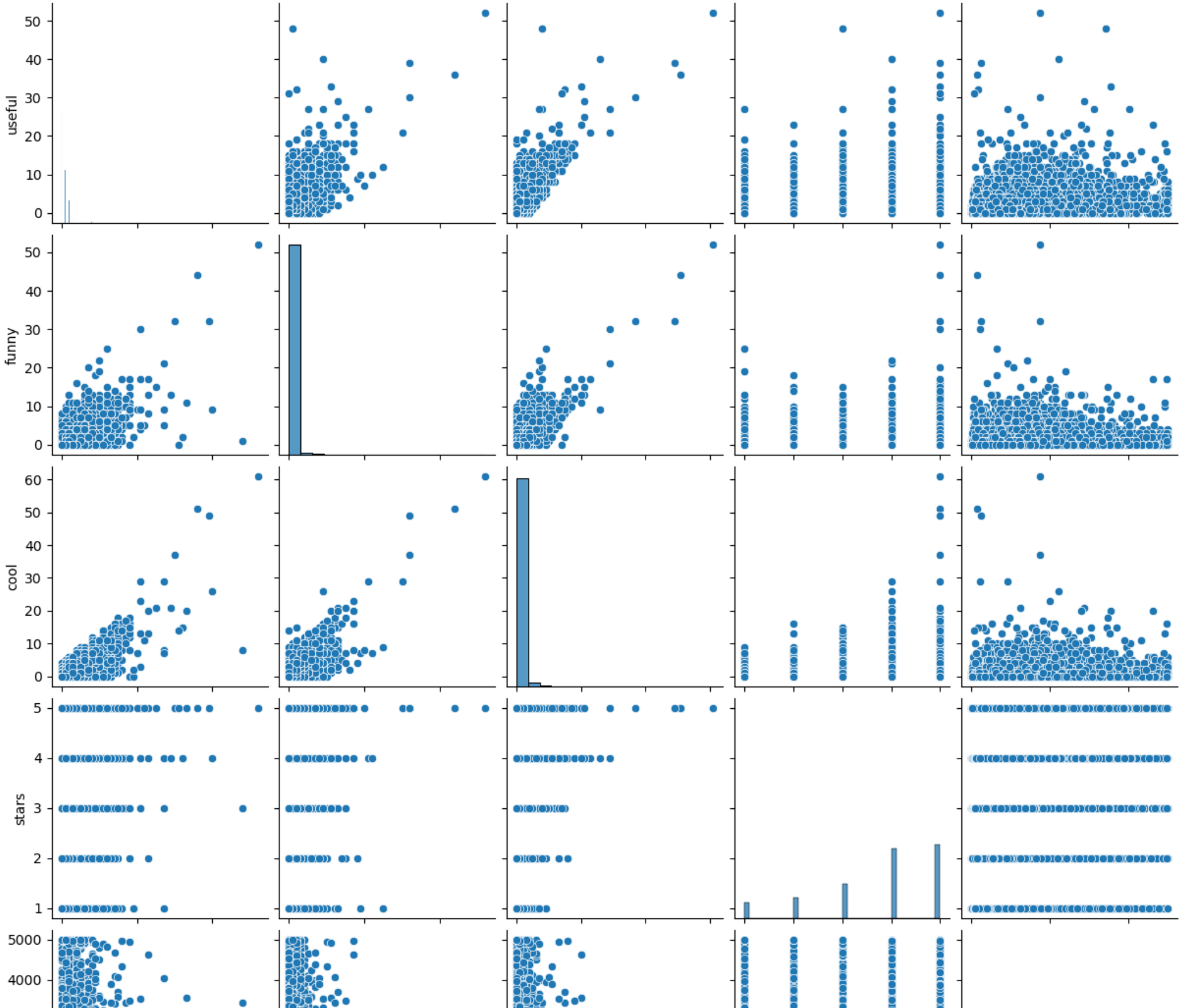
Out[29]:
```
Text(0, 0.5, 'Number of Reviews')
```



Question22: Plot a pair-plot of the number of useful, funny, and cool votes received by Yelp reviews alongwith the stars associated with the review and the length of the review.

In [30]:
```
# TODO: Replace <FILL IN>

review_votes_length = reviews_rdd.map(lambda review: (review["votes"]["useful"],
                                                      review["votes"]["funny"],
                                                      review["votes"]["cool"],
                                                      review["stars"],
                                                      len(review["text"]))).collect()
review_votes_length_df = pd.DataFrame(review_votes_length, columns=['useful', 'funny', 'cool', 'stars', 'length'])
sns.pairplot(review_votes_length_df)
```

Out[30]:
```
<seaborn.axisgrid.PairGrid at 0x1529dad2bca0>
```


useful
funny
cool
stars
0
10
20
30
40
50
60
1
2
3
4
5
4000
5000

Question23: Let us plot the distribution of the number of words used by males and females in their reviews. We will use the lists "male_names" and "female_names" we had created earlier for this purpose. Let's first find the user IDs associated with males and females.

In [31]:
```
# TODO: Replace <FILL IN>

male_users = users_rdd.filter(lambda user: user["name"] in male_names)
female_users = users_rdd.filter(lambda user: user["name"] in female_names)

male_user_ids = male_users.map(lambda user: user["user_id"]).collect()
female_user_ids = female_users.map(lambda user: user["user_id"]).collect()

print (len(male_user_ids))
print (len(female_user_ids))
print (users_rdd.count())
```

```
166682
174869
366715
```

Question24: We can now use the user ID lists to separate the reviews into those by males and females and calculate the length of each review.

In [32]:
```
# TODO: Replace <FILL IN>

male_reviews = reviews_rdd.filter(lambda review: review["user_id"] in male_user_ids).map(lambda x : x['text'])
female_reviews = reviews_rdd.filter(lambda review: review["user_id"] in female_user_ids).map(lambda x : x['text'])

male_word_count = male_reviews.map(lambda text: len(text.split()))
female_word_count = female_reviews.map(lambda text: len(text.split()))

male_avg_review_length = male_word_count.mean()
female_avg_review_length = female_word_count.mean()

print ('Male and female review length averages: ', male_avg_review_length, female_avg_review_length)
```

```
[Stage 34:=============================================>         (413 + 87) / 500]
Male and female review length averages:  127.3470787972135 139.21062276751027
```

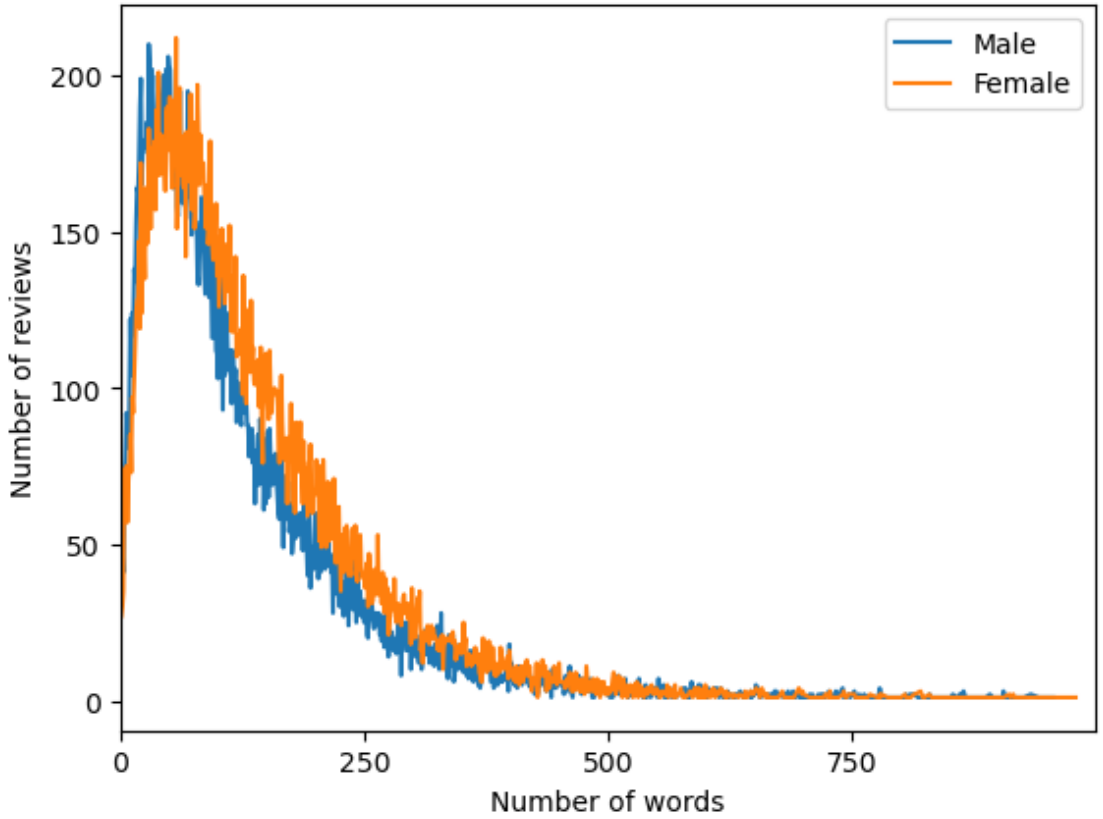
Question25: The code below calculates the distributions of review lengths for males and female reviewers and plots them. Do you see a marked difference between the average review length of male and female reviewers? Are there any major trends or differences between the distributions of review length of male and female reviewers?

Answer: There isnt any significant difference. However, females generally write longer reviews than males. This is seen in the distributions. The shorter and longer reviews are the same among male and females. Only in reviews having # of words between 125 and 475, females' reviews are more than males

In [33]:
```
male_word_distribution = list(male_word_count.map(lambda x : (x,1)).countByKey().items())
female_word_distribution = list(female_word_count.map(lambda x : (x,1)).countByKey().items())

male_word_distribution = sorted(male_word_distribution, key=lambda x: x[0])
female_word_distribution = sorted(female_word_distribution, key=lambda x: x[0])
```

In [34]:
```
fig, ax = plt.subplots()
ax.plot([x[0] for x in male_word_distribution], [x[1] for x in male_word_distribution], label = 'Male')
ax.plot([x[0] for x in female_word_distribution], [x[1] for x in female_word_distribution], label = 'Female')

ax.set_xlim((0, 1000))
ax.set_xticks([0, 250, 500, 750])
ax.set_xticklabels(['0', '250','500','750'])

plt.xlabel('Number of words')
plt.ylabel('Number of reviews')
plt.legend()
```

Out[34]:
```
<matplotlib.legend.Legend at 0x1529d6b00cd0>
```



Part 2: Classification using tree ensemble methods

In this section, we will predict the number of funny votes that a review has earned, indicating how funny readers found the review.

In [35]:
```
from pyspark.mllib.tree import DecisionTree, DecisionTreeModel
from pyspark.mllib.tree import RandomForest, RandomForestModel
from pyspark.mllib.tree import GradientBoostedTrees, GradientBoostedTreesModel
from pyspark.mllib.linalg import Vectors, DenseVector, SparseVector
from pyspark.mllib.regression import LabeledPoint
```

Question1: Fill in the necessary code to calculate word counts from text reviews below.

In [36]:
```
# TODO: Replace <FILL IN>

max_words = 50000

all_reviews = reviews_rdd.map(lambda x : (x['text'], x['votes']['funny']))
word_counts = list(all_reviews.flatMap(lambda x: x[0].lower().split()).map(lambda x: (x,1)).countByKey().items())
word_counts = sorted(word_counts, key=lambda x: -x[1])
```

```
unique_words = [x[0] for x in word_counts[:max_words]]
num_unique_words = len(unique_words)
print('Number of unique words: ', num_unique_words)
```

```
Number of unique words:  50000
```

Question2: We will now construct two dictionaries - one which maps from each word to a unique integer index and the second one which maps back from the index to the word. Write the code required to do this.

In [37]:
```
# TODO: Replace <FILL IN>

word_to_index_dict = {unique_words[i]: i for i in range(len(unique_words))}
index_to_word_dict = {i: unique_words[i] for i in range(len(unique_words))}
```

Question3: Fill in the required code below to obtain a LabeledPoint RDD that can be used to train an mllib classifier/regressor.

In [38]:
```
# TODO: Replace <FILL IN>
doc_vectors = all_reviews.map(lambda x: (x[1], x[0].split())).map(lambda x: (x[0], [word_to_index_dict[w] for w in x[1] if w in word_to_i
doc_vectors = doc_vectors.map(lambda x: LabeledPoint(x[0], SparseVector(len(word_to_index_dict), sorted(set(x[1])), [1.0]*len(set(x[1]))))
print(doc_vectors.count())
print(doc_vectors.take(2))
```

```
62608
[LabeledPoint(3.0, (50000,[0,1,2,3,4,6,8,9,11,12,14,15,16,22,25,38,44,52,58,79,99,102,116,144,162,182,289,307,334,352,362,456,470,656,67
2,711,740,888,928,931,984,1008,1114,1311,2209,2315,2805,3249,3738,3965,4148,4166,4530,5305,5391,5700,14456,15371],[1.0,1.0,1.0,1.0,1.0,1.
0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.
0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0])), LabeledPoint(1.0, (50000,[0,4,5,8,9,15,17,30,36,42,45,47,7
3,74,98,100,127,172,229,292,300,304,352,363,364,425,456,564,628,906,908,973,2805,3231,5799,6281,9776,13656,16459,18524],[1.0,1.0,1.0,1.0,
1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,
1.0,1.0]))]
```

Question4: Randomly split the doc_vectors RDD into 80% training and 20% validation data.

In [39]:
```
# TODO: Replace <FILL IN>
doc_vectors_train, doc_vectors_val = doc_vectors.randomSplit([0.8, 0.2], seed=0)
```

Question5: Let us implement the baseline predictor which always outputs the most common value of funny votes. Fill in appropriate code.

In [40]:
```
# TODO: Replace <FILL IN>

most_common_prediction = sorted(doc_vectors_train.map(lambda lp: lp.label).countByValue().items(), key=lambda x: -x[1])[0][0]
labels_and_predictions = doc_vectors_val.map(lambda lp: (lp.label, most_common_prediction))
val_mse = labels_and_predictions.map(lambda lp: (lp[0] - lp[1]) * (lp[0] - lp[1])).sum() /\
    float(doc_vectors_val.count())
print('Validation Root Mean Squared Error (Baseline) = ' + str(val_mse))
print('Learned baseline prediction: ', most_common_prediction)
```

```
Validation Root Mean Squared Error (Baseline) = 1.0672479212077177
Learned baseline prediction:  0.0
```

Question6: Let us now use a Decision Tree to predict the number of funny votes. Set the maximum depth of the tree to 5 and use an appropriate impurity metric for regression.

In [41]:
```
# TODO: Replace <FILL IN>
dt_model = DecisionTree.trainRegressor(doc_vectors_train, categoricalFeaturesInfo={}, impurity='variance', maxDepth=5)

predictions = dt_model.predict(doc_vectors_val.map(lambda x: x.features))
labels_and_predictions = doc_vectors_val.map(lambda lp: lp.label).zip(predictions)
val_mse = labels_and_predictions.map(lambda lp: (lp[0] - lp[1]) * (lp[0] - lp[1])).sum() /\
    float(doc_vectors_val.count())
print('Validation Root Mean Squared Error (Decision Tree) = ' + str(val_mse))
print('Learned regression tree model:')
print(dt_model.toDebugString())
```

```
23/04/30 22:28:55 WARN DAGScheduler: Broadcasting large task binary with size 1160.7 KiB
23/04/30 22:29:03 WARN DAGScheduler: Broadcasting large task binary with size 1794.2 KiB
23/04/30 22:29:09 WARN DAGScheduler: Broadcasting large task binary with size 1795.1 KiB
23/04/30 22:29:13 WARN DAGScheduler: Broadcasting large task binary with size 1795.7 KiB
23/04/30 22:29:16 WARN DAGScheduler: Broadcasting large task binary with size 1796.4 KiB
23/04/30 22:29:18 WARN DAGScheduler: Broadcasting large task binary with size 1797.0 KiB
23/04/30 22:29:22 WARN DAGScheduler: Broadcasting large task binary with size 1303.2 KiB
[Stage 59:=============================================>      (442 + 58) / 500]
```

```
Validation Root Mean Squared Error (Decision Tree) = 1.0737230552414534
Learned regression tree model:
DecisionTreeModel regressor of depth 5 with 23 nodes
  If (feature 46056 <= 0.5)
   If (feature 52 <= 0.5)
    If (feature 39307 <= 0.5)
     If (feature 11 <= 0.5)
      If (feature 19910 <= 0.5)
       Predict: 0.1743649965205289
      Else (feature 19910 > 0.5)
       Predict: 11.0
     Else (feature 11 > 0.5)
      If (feature 17363 <= 0.5)
       Predict: 0.3471738231646093
      Else (feature 17363 > 0.5)
       Predict: 7.2
    Else (feature 39307 > 0.5)
     Predict: 30.0
   Else (feature 52 > 0.5)
    If (feature 15465 <= 0.5)
     If (feature 32640 <= 0.5)
      If (feature 31874 <= 0.5)
       Predict: 0.530241151302995
      Else (feature 31874 > 0.5)
       Predict: 12.5
     Else (feature 32640 > 0.5)
      Predict: 19.0
    Else (feature 15465 > 0.5)
     Predict: 20.0
  Else (feature 46056 > 0.5)
   If (feature 12 <= 0.5)
    If (feature 6 <= 0.5)
     Predict: 0.0
    Else (feature 6 > 0.5)
     Predict: 1.0
   Else (feature 12 > 0.5)
    Predict: 52.0
```

Question7: Let us now use a Random Forest ensemble to predict the number of funny votes. Set the maximum depth of the tree to 5 and use an appropriate impurity metric for regression. Build a random forest regressor with 10 trees.

In [42]:

```
# TODO: Replace <FILL IN>

rf_model = RandomForest.trainRegressor(doc_vectors_train, categoricalFeaturesInfo={}, numTrees=10, featureSubsetStrategy="auto", impurity

predictions = rf_model.predict(doc_vectors_val.map(lambda x: x.features))
labels_and_predictions = doc_vectors_val.map(lambda lp: lp.label).zip(predictions)
val_mse = labels_and_predictions.map(lambda lp: (lp[0] - lp[1]) * (lp[0] - lp[1])).sum() /\
```

```
    float(doc_vectors_val.count())
print('Validation Root Mean Squared Error (Random Forest) = ' + str(val_mse))
print('Learned regression RF model:')
print(rf_model.toDebugString())
```

```
23/04/30 22:29:40 WARN DAGScheduler: Broadcasting large task binary with size 1160.7 KiB
23/04/30 22:29:46 WARN DAGScheduler: Broadcasting large task binary with size 2.4 MiB
23/04/30 22:29:58 WARN DAGScheduler: Broadcasting large task binary with size 3.0 MiB
23/04/30 22:30:06 WARN DAGScheduler: Broadcasting large task binary with size 3.4 MiB
23/04/30 22:30:14 WARN DAGScheduler: Broadcasting large task binary with size 3.8 MiB
23/04/30 22:30:24 WARN DAGScheduler: Broadcasting large task binary with size 4.3 MiB
23/04/30 22:30:35 WARN DAGScheduler: Broadcasting large task binary with size 1331.0 KiB
23/04/30 22:30:35 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeSystemBLAS
23/04/30 22:30:35 WARN BLAS: Failed to load implementation from: com.github.fommil.netlib.NativeRefBLAS
```

```
Validation Root Mean Squared Error (Random Forest) = 0.9558091833955712
Learned regression RF model:
TreeEnsembleModel regressor with 10 trees

  Tree 0:
    If (feature 39307 <= 0.5)
     If (feature 150 <= 0.5)
      If (feature 29911 <= 0.5)
       If (feature 35 <= 0.5)
        If (feature 28284 <= 0.5)
         Predict: 0.2179126958839035
        Else (feature 28284 > 0.5)
         Predict: 11.0
       Else (feature 35 > 0.5)
        If (feature 5950 <= 0.5)
         Predict: 0.4108573890564412
        Else (feature 5950 > 0.5)
         Predict: 4.588235294117647
      Else (feature 29911 > 0.5)
       If (feature 11 <= 0.5)
        Predict: 2.0
       Else (feature 11 > 0.5)
        Predict: 15.0
     Else (feature 150 > 0.5)
      If (feature 15747 <= 0.5)
       If (feature 31342 <= 0.5)
        If (feature 22446 <= 0.5)
         Predict: 0.6604127579737336
        Else (feature 22446 > 0.5)
         Predict: 15.0
       Else (feature 31342 > 0.5)
        If (feature 20 <= 0.5)
         Predict: 12.0
        Else (feature 20 > 0.5)
         Predict: 9.0
      Else (feature 15747 > 0.5)
       If (feature 21 <= 0.5)
        If (feature 9 <= 0.5)
         Predict: 8.0
        Else (feature 9 > 0.5)
         Predict: 13.0
       Else (feature 21 > 0.5)
        Predict: 0.0
    Else (feature 39307 > 0.5)
     If (feature 11 <= 0.5)
      Predict: 30.0
     Else (feature 11 > 0.5)
      Predict: 0.0
  Tree 1:
    If (feature 32080 <= 0.5)
     If (feature 33 <= 0.5)
```

```
    If (feature 19870 <= 0.5)
     If (feature 16370 <= 0.5)
      If (feature 21731 <= 0.5)
       Predict: 0.23031760232610154
      Else (feature 21731 > 0.5)
       Predict: 6.166666666666667
     Else (feature 16370 > 0.5)
      If (feature 12 <= 0.5)
       Predict: 0.6666666666666666
      Else (feature 12 > 0.5)
       Predict: 8.0
    Else (feature 19870 > 0.5)
     Predict: 13.0
   Else (feature 33 > 0.5)
    If (feature 28460 <= 0.5)
     If (feature 23505 <= 0.5)
      If (feature 21656 <= 0.5)
       Predict: 0.4684927716677042
      Else (feature 21656 > 0.5)
       Predict: 15.0
     Else (feature 23505 > 0.5)
      Predict: 13.0
    Else (feature 28460 > 0.5)
     Predict: 17.0
  Else (feature 32080 > 0.5)
   If (feature 78 <= 0.5)
    If (feature 12 <= 0.5)
     If (feature 1 <= 0.5)
      Predict: 0.0
     Else (feature 1 > 0.5)
      Predict: 2.0
    Else (feature 12 > 0.5)
     Predict: 5.0
   Else (feature 78 > 0.5)
    Predict: 17.0
Tree 2:
  If (feature 39307 <= 0.5)
   If (feature 11 <= 0.5)
    If (feature 12851 <= 0.5)
     If (feature 7287 <= 0.5)
      If (feature 35242 <= 0.5)
       Predict: 0.19192832160962
      Else (feature 35242 > 0.5)
       Predict: 11.0
     Else (feature 7287 > 0.5)
      If (feature 18 <= 0.5)
       Predict: 11.0
      Else (feature 18 > 0.5)
       Predict: 0.0
    Else (feature 12851 > 0.5)
     If (feature 7 <= 0.5)
```

```
      Predict: 0.0
     Else (feature 7 > 0.5)
      Predict: 11.0
   Else (feature 11 > 0.5)
    If (feature 31874 <= 0.5)
     If (feature 33153 <= 0.5)
      If (feature 16882 <= 0.5)
       Predict: 0.4181386008770132
      Else (feature 16882 > 0.5)
       Predict: 8.166666666666666
     Else (feature 33153 > 0.5)
      If (feature 30 <= 0.5)
       Predict: 17.0
      Else (feature 30 > 0.5)
       Predict: 2.0
    Else (feature 31874 > 0.5)
     If (feature 7 <= 0.5)
      Predict: 25.0
     Else (feature 7 > 0.5)
      Predict: 3.0
  Else (feature 39307 > 0.5)
   If (feature 2 <= 0.5)
    Predict: 30.0
   Else (feature 2 > 0.5)
    Predict: 0.0
Tree 3:
  If (feature 31874 <= 0.5)
   If (feature 11 <= 0.5)
    If (feature 39307 <= 0.5)
     If (feature 19803 <= 0.5)
      If (feature 13789 <= 0.5)
       Predict: 0.1886682469980181
      Else (feature 13789 > 0.5)
       Predict: 6.0
     Else (feature 19803 > 0.5)
      Predict: 13.0
    Else (feature 39307 > 0.5)
     Predict: 30.0
   Else (feature 11 > 0.5)
    If (feature 17885 <= 0.5)
     If (feature 17 <= 0.5)
      If (feature 29911 <= 0.5)
       Predict: 0.30949049304469506
      Else (feature 29911 > 0.5)
       Predict: 15.0
     Else (feature 17 > 0.5)
      If (feature 17363 <= 0.5)
       Predict: 0.5588188475007959
      Else (feature 17363 > 0.5)
       Predict: 16.0
    Else (feature 17885 > 0.5)
```

```
      If (feature 172 <= 0.5)
       If (feature 26 <= 0.5)
        Predict: 0.0
       Else (feature 26 > 0.5)
        Predict: 1.25
      Else (feature 172 > 0.5)
       If (feature 7 <= 0.5)
        Predict: 19.0
       Else (feature 7 > 0.5)
        Predict: 15.0
  Else (feature 31874 > 0.5)
   If (feature 31 <= 0.5)
    Predict: 0.0
   Else (feature 31 > 0.5)
    Predict: 25.0
Tree 4:
  If (feature 25973 <= 0.5)
   If (feature 33 <= 0.5)
    If (feature 31122 <= 0.5)
     If (feature 31736 <= 0.5)
      If (feature 40194 <= 0.5)
       Predict: 0.235006564429174
      Else (feature 40194 > 0.5)
       Predict: 11.0
     Else (feature 31736 > 0.5)
      Predict: 19.0
    Else (feature 31122 > 0.5)
     If (feature 2 <= 0.5)
      Predict: 30.0
     Else (feature 2 > 0.5)
      Predict: 0.0
   Else (feature 33 > 0.5)
    If (feature 28183 <= 0.5)
     If (feature 24552 <= 0.5)
      If (feature 32945 <= 0.5)
       Predict: 0.4654673721340388
      Else (feature 32945 > 0.5)
       Predict: 9.25
     Else (feature 24552 > 0.5)
      If (feature 31 <= 0.5)
       Predict: 15.333333333333334
      Else (feature 31 > 0.5)
       Predict: 3.5
    Else (feature 28183 > 0.5)
     If (feature 18 <= 0.5)
      Predict: 15.0
     Else (feature 18 > 0.5)
      If (feature 15 <= 0.5)
       Predict: 2.0
      Else (feature 15 > 0.5)
       Predict: 4.0
```

```
  Else (feature 25973 > 0.5)
   If (feature 17 <= 0.5)
    Predict: 52.0
   Else (feature 17 > 0.5)
    Predict: 5.0
Tree 5:
  If (feature 52 <= 0.5)
   If (feature 39307 <= 0.5)
    If (feature 35 <= 0.5)
     If (feature 12800 <= 0.5)
      If (feature 33674 <= 0.5)
       Predict: 0.20207889469946413
      Else (feature 33674 > 0.5)
       Predict: 12.0
     Else (feature 12800 > 0.5)
      If (feature 1 <= 0.5)
       Predict: 32.0
      Else (feature 1 > 0.5)
       Predict: 0.0
    Else (feature 35 > 0.5)
     If (feature 29911 <= 0.5)
      If (feature 32141 <= 0.5)
       Predict: 0.394479460986742
      Else (feature 32141 > 0.5)
       Predict: 11.0
     Else (feature 29911 > 0.5)
      If (feature 10 <= 0.5)
       Predict: 2.0
      Else (feature 10 > 0.5)
       Predict: 13.0
   Else (feature 39307 > 0.5)
    Predict: 30.0
  Else (feature 52 > 0.5)
   If (feature 37864 <= 0.5)
    If (feature 13029 <= 0.5)
     If (feature 17415 <= 0.5)
      If (feature 25015 <= 0.5)
       Predict: 0.5291431902078881
      Else (feature 25015 > 0.5)
       Predict: 17.0
     Else (feature 17415 > 0.5)
      Predict: 17.0
    Else (feature 13029 > 0.5)
     If (feature 29 <= 0.5)
      Predict: 52.0
     Else (feature 29 > 0.5)
      If (feature 6 <= 0.5)
       Predict: 0.0
      Else (feature 6 > 0.5)
       Predict: 1.0
   Else (feature 37864 > 0.5)
```

```
    Predict: 20.0
Tree 6:
  If (feature 11 <= 0.5)
   If (feature 44435 <= 0.5)
    If (feature 7287 <= 0.5)
     If (feature 21837 <= 0.5)
      If (feature 43917 <= 0.5)
       Predict: 0.18853483286472977
      Else (feature 43917 > 0.5)
       Predict: 11.0
     Else (feature 21837 > 0.5)
      If (feature 5 <= 0.5)
       Predict: 12.0
      Else (feature 5 > 0.5)
       Predict: 0.0
    Else (feature 7287 > 0.5)
     If (feature 15 <= 0.5)
      Predict: 0.0
     Else (feature 15 > 0.5)
      Predict: 11.0
   Else (feature 44435 > 0.5)
    Predict: 13.0
  Else (feature 11 > 0.5)
   If (feature 35634 <= 0.5)
    If (feature 17363 <= 0.5)
     If (feature 5906 <= 0.5)
      If (feature 32924 <= 0.5)
       Predict: 0.41859321200196753
      Else (feature 32924 > 0.5)
       Predict: 11.333333333333334
     Else (feature 5906 > 0.5)
      If (feature 266 <= 0.5)
       Predict: 2.3333333333333335
      Else (feature 266 > 0.5)
       Predict: 52.0
    Else (feature 17363 > 0.5)
     If (feature 5 <= 0.5)
      Predict: 32.0
     Else (feature 5 > 0.5)
      Predict: 1.0
   Else (feature 35634 > 0.5)
    If (feature 7 <= 0.5)
     Predict: 19.0
    Else (feature 7 > 0.5)
     Predict: 1.0
Tree 7:
  If (feature 18623 <= 0.5)
   If (feature 52 <= 0.5)
    If (feature 26944 <= 0.5)
     If (feature 37159 <= 0.5)
      If (feature 49299 <= 0.5)
```

```
      Predict: 0.2438896517288728
     Else (feature 49299 > 0.5)
      Predict: 12.0
    Else (feature 37159 > 0.5)
     Predict: 11.0
   Else (feature 26944 > 0.5)
    Predict: 15.0
  Else (feature 52 > 0.5)
   If (feature 16882 <= 0.5)
    If (feature 29650 <= 0.5)
     If (feature 32080 <= 0.5)
      Predict: 0.5362459861827381
     Else (feature 32080 > 0.5)
      Predict: 13.0
    Else (feature 29650 > 0.5)
     Predict: 17.0
   Else (feature 16882 > 0.5)
    Predict: 20.0
 Else (feature 18623 > 0.5)
  If (feature 14 <= 0.5)
   If (feature 57 <= 0.5)
    Predict: 0.0
   Else (feature 57 > 0.5)
    Predict: 3.0
  Else (feature 14 > 0.5)
   Predict: 52.0
Tree 8:
 If (feature 17363 <= 0.5)
  If (feature 52 <= 0.5)
   If (feature 39307 <= 0.5)
    If (feature 14285 <= 0.5)
     If (feature 43917 <= 0.5)
      Predict: 0.24617196702002356
     Else (feature 43917 > 0.5)
      Predict: 11.0
    Else (feature 14285 > 0.5)
     If (feature 62 <= 0.5)
      Predict: 0.2
     Else (feature 62 > 0.5)
      Predict: 13.0
   Else (feature 39307 > 0.5)
    Predict: 30.0
  Else (feature 52 > 0.5)
   If (feature 41724 <= 0.5)
    If (feature 9633 <= 0.5)
     If (feature 27539 <= 0.5)
      Predict: 0.5274078363250337
     Else (feature 27539 > 0.5)
      Predict: 12.0
    Else (feature 9633 > 0.5)
     If (feature 26 <= 0.5)
```

```
      Predict: 52.0
     Else (feature 26 > 0.5)
      Predict: 2.5
   Else (feature 41724 > 0.5)
    Predict: 17.0
 Else (feature 17363 > 0.5)
  If (feature 100 <= 0.5)
   If (feature 83 <= 0.5)
    Predict: 0.0
   Else (feature 83 > 0.5)
    Predict: 1.0
  Else (feature 100 > 0.5)
   Predict: 32.0
Tree 9:
 If (feature 46056 <= 0.5)
  If (feature 17 <= 0.5)
   If (feature 24783 <= 0.5)
    If (feature 37031 <= 0.5)
     If (feature 25371 <= 0.5)
      Predict: 0.21219115734720417
     Else (feature 25371 > 0.5)
      Predict: 13.0
    Else (feature 37031 > 0.5)
     Predict: 12.0
   Else (feature 24783 > 0.5)
    Predict: 10.0
  Else (feature 17 > 0.5)
   If (feature 29287 <= 0.5)
    If (feature 41279 <= 0.5)
     If (feature 21656 <= 0.5)
      Predict: 0.4386702918996563
     Else (feature 21656 > 0.5)
      Predict: 9.4
    Else (feature 41279 > 0.5)
     Predict: 17.0
   Else (feature 29287 > 0.5)
    Predict: 17.0
 Else (feature 46056 > 0.5)
  If (feature 18 <= 0.5)
   If (feature 8 <= 0.5)
    Predict: 0.0
   Else (feature 8 > 0.5)
    Predict: 1.0
  Else (feature 18 > 0.5)
   Predict: 52.0
```

Question8: Let us now use a Gradient Boosting Trees (GBT) ensemble to predict the number of funny votes. Set the maximum number of iterations to 10. Does this affect the number of trees in the ensemble? Do we need to set the maximum depth of trees in the ensemble? Why or why not?

Answer: The number of iterations is equal to the number of trees in the ensemble. Yes, it affects the # of trees. We do not need to set maximum depth of trees because it tries to correct the errors of the previous trees in a sequential manner and the trees used here are decision stumps (depth=1) and we can control the complexity of the model through the number of trees used

In [43]:
```
# TODO: Replace <FILL IN>

gb_model = GradientBoostedTrees.trainRegressor(doc_vectors_train, categoricalFeaturesInfo={}, numIterations=10)

predictions = gb_model.predict(doc_vectors_val.map(lambda x: x.features))
labels_and_predictions = doc_vectors_val.map(lambda lp: lp.label).zip(predictions)
val_mse = labels_and_predictions.map(lambda lp: (lp[0] - lp[1]) * (lp[0] - lp[1])).sum() /\
    float(doc_vectors_val.count())
print('Validation Root Mean Squared Error (Gradient Boosting Trees) = ' + str(val_mse))
print('Learned regression GBT model:')
print(gb_model.toDebugString())
```

```
23/04/30 22:30:38 WARN DAGScheduler: Broadcasting large task binary with size 1160.9 KiB
23/04/30 22:30:44 WARN DAGScheduler: Broadcasting large task binary with size 1795.6 KiB
23/04/30 22:30:48 WARN DAGScheduler: Broadcasting large task binary with size 1796.5 KiB
23/04/30 22:30:51 WARN DAGScheduler: Broadcasting large task binary with size 1797.1 KiB
23/04/30 22:30:53 WARN DAGScheduler: Broadcasting large task binary with size 1803.5 KiB
23/04/30 22:30:56 WARN DAGScheduler: Broadcasting large task binary with size 1804.1 KiB
23/04/30 22:30:58 WARN DAGScheduler: Broadcasting large task binary with size 1804.7 KiB
23/04/30 22:31:01 WARN DAGScheduler: Broadcasting large task binary with size 1806.6 KiB
23/04/30 22:31:03 WARN DAGScheduler: Broadcasting large task binary with size 1807.2 KiB
23/04/30 22:31:06 WARN DAGScheduler: Broadcasting large task binary with size 1807.8 KiB
23/04/30 22:31:09 WARN DAGScheduler: Broadcasting large task binary with size 1809.8 KiB
23/04/30 22:31:11 WARN DAGScheduler: Broadcasting large task binary with size 1810.4 KiB
23/04/30 22:31:13 WARN DAGScheduler: Broadcasting large task binary with size 1811.0 KiB
23/04/30 22:31:17 WARN DAGScheduler: Broadcasting large task binary with size 1812.9 KiB
23/04/30 22:31:19 WARN DAGScheduler: Broadcasting large task binary with size 1813.4 KiB
23/04/30 22:31:21 WARN DAGScheduler: Broadcasting large task binary with size 1814.1 KiB
23/04/30 22:31:24 WARN DAGScheduler: Broadcasting large task binary with size 1815.9 KiB
23/04/30 22:31:26 WARN DAGScheduler: Broadcasting large task binary with size 1816.5 KiB
23/04/30 22:31:29 WARN DAGScheduler: Broadcasting large task binary with size 1817.1 KiB
23/04/30 22:31:32 WARN DAGScheduler: Broadcasting large task binary with size 1818.8 KiB
23/04/30 22:31:34 WARN DAGScheduler: Broadcasting large task binary with size 1819.4 KiB
23/04/30 22:31:37 WARN DAGScheduler: Broadcasting large task binary with size 1820.0 KiB
23/04/30 22:31:40 WARN DAGScheduler: Broadcasting large task binary with size 1822.0 KiB
23/04/30 22:31:42 WARN DAGScheduler: Broadcasting large task binary with size 1822.6 KiB
23/04/30 22:31:45 WARN DAGScheduler: Broadcasting large task binary with size 1823.2 KiB
23/04/30 22:31:47 WARN DAGScheduler: Broadcasting large task binary with size 1825.1 KiB
23/04/30 22:31:50 WARN DAGScheduler: Broadcasting large task binary with size 1825.7 KiB
23/04/30 22:31:52 WARN DAGScheduler: Broadcasting large task binary with size 1826.3 KiB
23/04/30 22:31:55 WARN DAGScheduler: Broadcasting large task binary with size 1828.1 KiB
23/04/30 22:31:57 WARN DAGScheduler: Broadcasting large task binary with size 1828.7 KiB
23/04/30 22:32:00 WARN DAGScheduler: Broadcasting large task binary with size 1829.3 KiB
23/04/30 22:32:03 WARN DAGScheduler: Broadcasting large task binary with size 1316.0 KiB
[Stage 143:================================================>    (459 + 41) / 500]
```

```
Validation Root Mean Squared Error (Gradient Boosting Trees) = 1.06678101043372
Learned regression GBT model:
TreeEnsembleModel regressor with 10 trees

  Tree 0:
    If (feature 46056 <= 0.5)
     If (feature 52 <= 0.5)
      If (feature 39307 <= 0.5)
       Predict: 0.24879160509905082
      Else (feature 39307 > 0.5)
       Predict: 30.0
     Else (feature 52 > 0.5)
      If (feature 15465 <= 0.5)
       Predict: 0.5343637600855449
      Else (feature 15465 > 0.5)
       Predict: 20.0
    Else (feature 46056 > 0.5)
     If (feature 12 <= 0.5)
      If (feature 6 <= 0.5)
       Predict: 0.0
      Else (feature 6 > 0.5)
       Predict: 1.0
     Else (feature 12 > 0.5)
      Predict: 52.0
  Tree 1:
    If (feature 17 <= 0.5)
     If (feature 22066 <= 0.5)
      If (feature 25371 <= 0.5)
       Predict: -0.15738879910281123
      Else (feature 25371 > 0.5)
       Predict: 25.502416789801828
     Else (feature 22066 > 0.5)
      Predict: 29.502416789801828
    Else (feature 17 > 0.5)
     If (feature 22241 <= 0.5)
      If (feature 35 <= 0.5)
       Predict: 0.06901320784792546
      Else (feature 35 > 0.5)
       Predict: 0.546504850804352
     Else (feature 22241 > 0.5)
      Predict: 32.931272479829204
  Tree 2:
    If (feature 150 <= 0.5)
     If (feature 33 <= 0.5)
      If (feature 32640 <= 0.5)
       Predict: -0.13311335660131662
      Else (feature 32640 > 0.5)
       Predict: 36.91746983825942
     Else (feature 33 > 0.5)
      If (feature 28460 <= 0.5)
       Predict: 0.16016004212947788
```

```
    Else (feature 28460 > 0.5)
     Predict: 32.91746983825942
  Else (feature 150 > 0.5)
   If (feature 22446 <= 0.5)
    If (feature 14064 <= 0.5)
     Predict: 0.543246637428251
    Else (feature 14064 > 0.5)
     Predict: 29.4886141482325
   Else (feature 22446 > 0.5)
    If (feature 31 <= 0.5)
     Predict: 28.82197150966804
    Else (feature 31 > 0.5)
     Predict: 3.5338945496224596
Tree 3:
  If (feature 11 <= 0.5)
   If (feature 16599 <= 0.5)
    If (feature 12986 <= 0.5)
     Predict: -0.14806307164951082
    Else (feature 12986 > 0.5)
     Predict: 10.83194560317986
   Else (feature 16599 > 0.5)
    Predict: 24.930718231223636
  Else (feature 11 > 0.5)
   If (feature 17363 <= 0.5)
    If (feature 31874 <= 0.5)
     Predict: 0.14811578920740145
    Else (feature 31874 > 0.5)
     Predict: 15.846421014294114
   Else (feature 17363 > 0.5)
    If (feature 100 <= 0.5)
     Predict: 1.3393467542230013
    Else (feature 100 > 0.5)
     Predict: 63.515236819552584
Tree 4:
  If (feature 17 <= 0.5)
   If (feature 22066 <= 0.5)
    If (feature 49299 <= 0.5)
     Predict: -0.11202027781346013
    Else (feature 49299 > 0.5)
     Predict: 23.472239383354918
   Else (feature 22066 > 0.5)
    Predict: 23.598932945320485
  Else (feature 17 > 0.5)
   If (feature 21656 <= 0.5)
    If (feature 37658 <= 0.5)
     Predict: 0.16865460791383574
    Else (feature 37658 > 0.5)
     Predict: 15.770242567949936
   Else (feature 21656 > 0.5)
    If (feature 29 <= 0.5)
     Predict: 26.379209817669295
```

```
    Else (feature 29 > 0.5)
     Predict: 1.4861947541363207
Tree 5:
  If (feature 58 <= 0.5)
   If (feature 34173 <= 0.5)
    If (feature 16835 <= 0.5)
     Predict: -0.06917198199496928
    Else (feature 16835 > 0.5)
     Predict: 6.819544170470737
   Else (feature 34173 > 0.5)
    Predict: 22.785240101563886
  Else (feature 58 > 0.5)
   If (feature 33153 <= 0.5)
    If (feature 36791 <= 0.5)
     Predict: 0.2507586820228091
    Else (feature 36791 > 0.5)
     Predict: 29.47062179118302
   Else (feature 33153 > 0.5)
    Predict: 32.785240101563886
Tree 6:
  If (feature 150 <= 0.5)
   If (feature 1802 <= 0.5)
    If (feature 43225 <= 0.5)
     Predict: -0.046380724611116565
    Else (feature 43225 > 0.5)
     Predict: 21.562382154430907
   Else (feature 1802 > 0.5)
    If (feature 3432 <= 0.5)
     Predict: 1.331579855719083
    Else (feature 3432 > 0.5)
     Predict: 20.74041981821688
  Else (feature 150 > 0.5)
   If (feature 22846 <= 0.5)
    If (feature 12037 <= 0.5)
     Predict: 0.39535092989468634
    Else (feature 12037 > 0.5)
     Predict: 26.1868489733522
   Else (feature 22846 > 0.5)
    If (feature 25 <= 0.5)
     Predict: 8.663802499157136
    Else (feature 25 > 0.5)
     Predict: 28.759300827748426
Tree 7:
  If (feature 51 <= 0.5)
   If (feature 24552 <= 0.5)
    If (feature 43917 <= 0.5)
     Predict: -0.0653545837737193
    Else (feature 43917 > 0.5)
     Predict: 16.022881073213057
   Else (feature 24552 > 0.5)
    If (feature 151 <= 0.5)
```

```
      Predict: 2.3776332853564677
     Else (feature 151 > 0.5)
      Predict: 24.505696110002972
   Else (feature 51 > 0.5)
    If (feature 31874 <= 0.5)
     If (feature 30325 <= 0.5)
      Predict: 0.2172628935107402
     Else (feature 30325 > 0.5)
      Predict: 26.749695963138947
    Else (feature 31874 > 0.5)
     Predict: 44.487860074501896
Tree 8:
  If (feature 35 <= 0.5)
   If (feature 45973 <= 0.5)
    If (feature 36791 <= 0.5)
     Predict: -0.07041112979928053
    Else (feature 36791 > 0.5)
     Predict: 23.542320999166805
   Else (feature 45973 > 0.5)
    Predict: 25.1439362402366
  Else (feature 35 > 0.5)
   If (feature 45736 <= 0.5)
    If (feature 15747 <= 0.5)
     Predict: 0.18101379072332596
    Else (feature 15747 > 0.5)
     Predict: 8.582265167707249
   Else (feature 45736 > 0.5)
    If (feature 7 <= 0.5)
     Predict: 22.477293601672496
    Else (feature 7 > 0.5)
     Predict: 9.874674921512524
Tree 9:
  If (feature 64 <= 0.5)
   If (feature 28460 <= 0.5)
    If (feature 19619 <= 0.5)
     Predict: -0.05204325524385449
    Else (feature 19619 > 0.5)
     Predict: 8.652200873179677
   Else (feature 28460 > 0.5)
    If (feature 18 <= 0.5)
     Predict: 21.40667533646367
    Else (feature 18 > 0.5)
     Predict: 14.441090843527832
  Else (feature 64 > 0.5)
   If (feature 14507 <= 0.5)
    If (feature 29479 <= 0.5)
     Predict: 0.23176738759814336
    Else (feature 29479 > 0.5)
     Predict: 25.2411849362652
   Else (feature 14507 > 0.5)
    If (feature 31 <= 0.5)
```

```
 Predict: 21.905017903781804
Else (feature 31 > 0.5)
 Predict: 6.751837806184554
```

Question9: Which of the four methods we tried gave the best validation RMSE results?

Answer: Random Forest has the lowest RMSE

Part 3: Collaborative filtering for recommendation

In this section, we will tackle a collaborative filtering task which can be used to recommend businesses to users based on the ratings they have already assigned to some businesses they have visited.

In [44]:
```
from pyspark.mllib.recommendation import ALS, MatrixFactorizationModel, Rating
```

Question1: Let us first determine the unique user and business IDs that appear in the reviews data. This will help us build dictionaries for mapping the user/business IDs to unique integer indices. Fill in the required code to build these dictionaries below.

In [45]:
```
# TODO: Replace <FILL IN>

review_user_ids = reviews_rdd.map(lambda review: review['user_id']).distinct().collect()
review_business_ids = reviews_rdd.map(lambda review: review['business_id']).distinct().collect()

user_to_index_dict = {review_user_ids[i]: i for i in range(len(review_user_ids))}
business_to_index_dict = {review_business_ids[i]: i for i in range(len(review_business_ids))}
```

Question2: Next, transform each review into a rating. The Rating object takes a unique user index, a unique business index, and float-valued rating.

In [46]:
```
# TODO: Replace <FILL IN>

ratings_rdd = reviews_rdd.map(lambda x: Rating(user_to_index_dict[x['user_id']],
                                               business_to_index_dict[x['business_id']],
                                               float(x['stars'])))
print(ratings_rdd.take(2))
```

```
[Rating(user=14449, product=1855, rating=5.0), Rating(user=5866, product=1855, rating=2.0)]
```

Question3: Let us randomly split data into 80% train and 20% validation set.

In [47]:
```
# TODO: Replace <FILL IN>
```

```
ratings_rdd_train, ratings_rdd_val = ratings_rdd.randomSplit([0.8, 0.2])
```

Question4: For a succession of ranks, we will now build an collaborative filtering algorithm using ALS (Alternating Least Squares). We will use the model to obtain train as well as validation RMSE for each rank. In the cell below, you can fill in the code to carry out the model-building, prediction, and RMSE calculation.

In [48]:
```
# TODO: Replace <FILL IN>

numIterations=10
ranks = list(range(1,20)) + list(range(20, 201, 20))
train_rmses = []
val_rmses = []

for rank in ranks:
    cf_model = ALS.train(ratings_rdd_train, rank=rank, iterations=numIterations)

    train_data = ratings_rdd_train.map(lambda p: (p[0], p[1]))
    predictions = cf_model.predictAll(train_data).map(lambda r: ((r[0], r[1]), r[2]))
    rates_and_preds = ratings_rdd_train.map(lambda r: ((r[0], r[1]), r[2])).join(predictions)
    train_rmse = np.sqrt(rates_and_preds.map(lambda r: (r[1][0] - r[1][1])**2).mean())
    train_rmses.append(train_rmse)

    val_data = ratings_rdd_val.map(lambda p: (p[0], p[1]))
    predictions = cf_model.predictAll(val_data).map(lambda r: ((r[0], r[1]), r[2]))
    rates_and_preds = ratings_rdd_val.map(lambda r: ((r[0], r[1]), r[2])).join(predictions)
    val_rmse = np.sqrt(rates_and_preds.map(lambda r: (r[1][0] - r[1][1])**2).mean())
    val_rmses.append(val_rmse)

    print("Root Mean Squared Error (rank={}) = Train {}, Validation {}".format(rank, train_rmse, val_rmse))
```

```
23/04/30 22:32:36 WARN LAPACK: Failed to load implementation from: com.github.fommil.netlib.NativeSystemLAPACK
23/04/30 22:32:36 WARN LAPACK: Failed to load implementation from: com.github.fommil.netlib.NativeRefLAPACK
23/04/30 22:32:48 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:32:52 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=1) = Train 2.2510278011448706, Validation 8.15444275112275
23/04/30 22:33:12 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:33:16 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=2) = Train 0.7488240671922028, Validation 2.6313136254853418
23/04/30 22:33:36 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:33:39 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=3) = Train 0.6815517042749296, Validation 2.7373622440217846
23/04/30 22:33:59 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:34:02 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=4) = Train 0.5395510911272018, Validation 2.501727749262729
```

```
23/04/30 22:34:22 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:34:25 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=5) = Train 0.47525822087177294, Validation 2.215550098738347
23/04/30 22:34:46 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:34:50 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=6) = Train 0.4430414183336026, Validation 2.291467511508185
23/04/30 22:35:09 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:35:13 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=7) = Train 0.39932826115442976, Validation 2.155409276528969
23/04/30 22:35:34 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:35:37 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=8) = Train 0.36776830681869216, Validation 2.097009541931733
23/04/30 22:35:56 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:36:00 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=9) = Train 0.34899048960365714, Validation 2.134740462091387
23/04/30 22:36:20 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:36:23 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=10) = Train 0.3275627747809437, Validation 2.18902041243086
23/04/30 22:36:46 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:36:50 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=11) = Train 0.30872223667172216, Validation 2.1759335475153767
23/04/30 22:37:12 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:37:15 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=12) = Train 0.2886932541512754, Validation 2.1261414875563163
23/04/30 22:37:37 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:37:42 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=13) = Train 0.2740880161520304, Validation 2.089543044465042
23/04/30 22:38:04 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:38:07 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=14) = Train 0.26175105219413897, Validation 2.0632212829221683
23/04/30 22:38:29 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:38:32 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB

Root Mean Squared Error (rank=15) = Train 0.25128447916838953, Validation 2.0710746918042755
23/04/30 22:38:55 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:38:59 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
```

```
Root Mean Squared Error (rank=16) = Train 0.2430549135691912, Validation 2.082420803077882
23/04/30 22:39:21 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:39:24 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=17) = Train 0.2332660056547698, Validation 1.95841470919879
23/04/30 22:39:46 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:39:49 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=18) = Train 0.22655350515972236, Validation 2.0116898221909634
23/04/30 22:40:12 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:40:16 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=19) = Train 0.21922569811921103, Validation 1.9980019679114482
23/04/30 22:40:37 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:40:41 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=20) = Train 0.21333365263130435, Validation 1.9636691290685484
23/04/30 22:41:04 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:41:07 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=40) = Train 0.18728325235185767, Validation 1.74614593100712
23/04/30 22:41:29 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:41:32 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=60) = Train 0.1855175755259221, Validation 1.628357614584002
23/04/30 22:41:54 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:41:58 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=80) = Train 0.185071117028088, Validation 1.6054316927422752
23/04/30 22:42:21 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:42:24 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=100) = Train 0.18483818045054354, Validation 1.601459841843747
23/04/30 22:42:47 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
23/04/30 22:42:50 WARN DAGScheduler: Broadcasting large task binary with size 1151.8 KiB
Root Mean Squared Error (rank=120) = Train 0.18487581629386884, Validation 1.5736282926258598
23/04/30 22:43:13 WARN DAGScheduler: Broadcasting large task binary with size 1151.9 KiB
23/04/30 22:43:18 WARN DAGScheduler: Broadcasting large task binary with size 1151.9 KiB
Root Mean Squared Error (rank=140) = Train 0.18474106766641266, Validation 1.5679474536528637
23/04/30 22:43:41 WARN DAGScheduler: Broadcasting large task binary with size 1151.9 KiB
23/04/30 22:43:44 WARN DAGScheduler: Broadcasting large task binary with size 1151.9 KiB
Root Mean Squared Error (rank=160) = Train 0.18477419331550296, Validation 1.5546719700281915
```

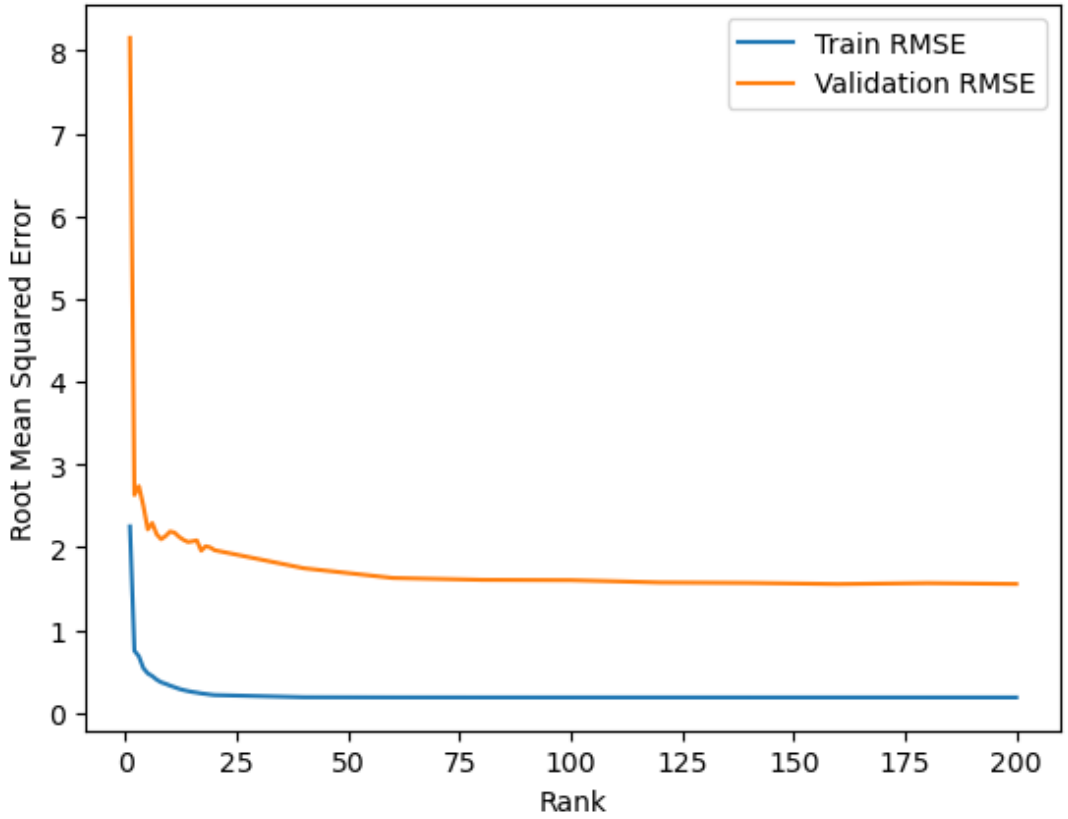
```
23/04/30 22:44:07 WARN DAGScheduler: Broadcasting large task binary with size 1151.9 KiB
23/04/30 22:44:10 WARN DAGScheduler: Broadcasting large task binary with size 1151.9 KiB

Root Mean Squared Error (rank=180) = Train 0.18473281834053376, Validation 1.565783827267214
23/04/30 22:44:34 WARN DAGScheduler: Broadcasting large task binary with size 1151.9 KiB
23/04/30 22:44:38 WARN DAGScheduler: Broadcasting large task binary with size 1151.9 KiB
[Stage 6179:=============================================>      (647 + 103) / 750]
Root Mean Squared Error (rank=200) = Train 0.18473319450122808, Validation 1.555880541516929
```

Question5: Let us plot the train and validation RMSE versus the rank. The code below does this for you. Based on this plot, what would your choice of the rank hyperparameter be? Is this choice conclusive or do we need to conduct a more extensive hyperparameter search at larger ranks than the ones we have evaluated?

Answer: Based on the plots, I choose rank=25 as for this value both train and validation RMSE are low. And I do not recommend doing hyperparameter tuning with more rank values because the RMSE for both train and validation became constant after rank 25. So it does not make sense to try after higher ranks

In [49]:
```
fig, ax = plt.subplots()
ax.plot(ranks, train_rmses, label='Train RMSE')
ax.plot(ranks, val_rmses, label='Validation RMSE')

plt.xlabel('Rank')
plt.ylabel('Root Mean Squared Error')
plt.legend()
```

Out[49]:
```
<matplotlib.legend.Legend at 0x1529d54d4970>
```



Part 4: Topic modeling for text reviews

In this section, we will build and examine a Bayesian topic model named Latent Dirichlet Allocation (LDA). The goal of textual topic modeling is to discover latent topics whose mixtures generate textual documents through a stylized probabilistic generatve model. The topics often have semantic meaning. They may be associated with various aspects discussed in the text corpus such as politics, health, education, etc. Topic models are unsupervised machine learning algorithms. Hence, the nature of discovered topics is entirely dependent of the context of your dataset.

In [50]:
```
from pyspark.mllib.linalg import Vectors, DenseVector, SparseVector
from pyspark.mllib.clustering import LDA, LDAModel
```

Question1: Let us create a new RDD of just textual reviews from reviews_rdd, obtain word counts, and build a list of unique words that do not include stop words. Use num_stop_words as a measure of how many of the most frequent words to filter out.

In [51]:
```
# TODO: Replace <FILL IN>

# number of stopwords i.e. most frequent words to remove
# removal of stopwords such as a, the, from, etc. that occur across a vast majority of documents improves topic models
num_stop_words = 1000
```

```
all_reviews = reviews_rdd.map(lambda x: x['text'])
word_counts = list(all_reviews.flatMap(lambda x: x.lower().split()).map(lambda x: (x,1)).countByKey().items())
# sort words in descending order of frequency
word_counts = sorted(word_counts, key=lambda x: -x[1])

# remove stopwords
unique_words = [x[0] for x in word_counts[num_stop_words:]]
num_unique_words = len(unique_words)
print('Number of unique words: ', num_unique_words)
```

```
Number of unique words:  192742
```

Question2: We will now construct two dictionaries - one which maps from each word to a unique integer index and the second one which maps back from the index to the word. Write the code required to do this.

In [52]:
```
# TODO: Replace <FILL IN>

word_to_index_dict = {unique_words[i]: i for i in range(len(unique_words))}
index_to_word_dict = {i: unique_words[i] for i in range(len(unique_words))}
```

Question3: Construct an RDD of SparseVectors. Each SparseVector is built using the word counts of a review. Hence, the RDD of SparseVectors should be obtained as a map from the RDD of document word counts.

In [53]:
```
# TODO: Replace <FILL IN>

doc_vectors = all_reviews.map(lambda x: x.lower().split()).map(lambda x: [word_to_index_dict[w] for w in x if w in word_to_index_dict])
doc_vectors = doc_vectors.map(lambda x: SparseVector(num_unique_words, [(i, x.count(i)) for i in set(x)]))
# zipWithIndex result needs a minor transform to be acceptable to the LDA training procedure
doc_vectors = doc_vectors.zipWithIndex().map(lambda x: [x[1], x[0]])
print(doc_vectors.count())
print(doc_vectors.take(2))
```

```
62608
[[0, SparseVector(192742, {8: 2.0, 114: 1.0, 311: 1.0, 1209: 1.0, 1315: 1.0, 1444: 1.0, 1805: 1.0, 1987: 1.0, 2249: 1.0, 2738: 1.0, 2965:
1.0, 3148: 1.0, 3166: 1.0, 3530: 1.0, 3614: 1.0, 3977: 1.0, 4305: 1.0, 4391: 1.0, 4700: 1.0, 6041: 1.0, 8947: 1.0, 11064: 1.0, 13456: 1.
0, 14371: 1.0, 23207: 1.0, 54812: 1.0, 77509: 1.0, 77510: 1.0, 77511: 1.0, 77512: 1.0})], [1, SparseVector(192742, {1805: 1.0, 2231: 1.0,
4799: 1.0, 5281: 1.0, 8776: 1.0, 12656: 1.0, 15459: 1.0, 17524: 1.0})]]
```

Question4: Train an LDA model with a 100 topics and the random seed set to 42.

In [54]:
```
# TODO: Replace <FILL IN>
lda_model = LDA.train(doc_vectors, k=100, seed=42)
```

```
                                                                                500]
```

Question5: Display the LDA model vocabulary size.

In [55]:
```
# TODO: Replace <FILL IN>

print('Model vocabulary size: ', lda_model.vocabSize())
```

```
Model vocabulary size:  192742
```

Question6: Display 5 learned topics and the top 100 terms that appear in each of these topics. Assign a semantic label/meaning to each of them (e.g. food, ambience, drinks, service, etc.) You can access the topic matrix using the function topicsMatrix on the model. Do the topics learned from Yelp reviews look representative of the corpus?

Answer Yes since we are considering Yelp dataset that has details/reviews about restaurants, hotels - the topics learned are also closely associated with these

In [56]:
```
# TODO: Replace <FILL IN>
topics = lda_model.topicsMatrix()
num_top_words = 100
for topic_idx in range(5):
    print("Topic #{}:".format(topic_idx))
    top_words = [index_to_word_dict[i] for i in topics[topic_idx,:].argsort()[-num_top_words:]]
    print(" ".join(top_words))
    print("Semantic label/meaning:")
    if topic_idx == 0:
        print("Food")
    elif topic_idx == 1:
        print("Rooms")
    elif topic_idx == 2:
        print("Service")
    elif topic_idx == 3:
        print("Ambience")
    elif topic_idx == 4:
        print("Location")
    print()
```

Topic #0:
cook life apple special. bottom wait. chili drink. means greasy choices true noodle hostess supposed perfect. stayed side, burrito tasty, room. hand pancakes fresh. picked orders thank great! thinking mine weeks atmosphere, thick location. else. inside. small, amazing! oil covered generally normally total bread. town. except they've sign you, la minutes. minute you'd times. visit. penn best. pricey longer terrible we're thin share nearly creamy meal, honestly up, bean somewhere okay, please charge selection. recommended area, pizza, paying whatever dressing wall rooms finished avoid seriously pepper ... forget personal card clearly general limited employees , is, and, bars recent sour
Semantic label/meaning:
Food

Topic #1:
okay, rooms personal pricey hostess thick side, thinking times. life dressing recommended sign perfect. they've selection. minute supposed stayed bottom else. ... employees bars up, visit. sour greasy except small, share choices wait. room. town. is, general you, please clearly picked area, drink. chili means great! orders honestly meal, creamy covered limited recent pizza, weeks charge seriously special. burrito wall fresh. avoid nearly noodle inside. pepper total oil pancakes paying , location. bean longer whatever apple thank true terrible amazing! bread. tasty, we're minutes. normally and, card forget hand mine best. generally you'd penn thin finished la cook somewhere atmosphere,
Semantic label/meaning:
Rooms

Topic #2:
cook bread. total greasy forget you'd card pricey somewhere sour okay, ... chili clearly rooms visit. perfect. atmosphere, hand nearly noodle charge area, pizza, stayed creamy pepper finished whatever inside. else. mine sign minutes. bars means thick is, fresh. thinking picked terrible recommended honestly pancakes amazing! thank please special. la bean generally up, penn side, normally selection. meal, wall oil dressing true life times. limited employees they've weeks seriously small, burrito and, orders drink. tasty, best. supposed you, choices hostess covered recent we're apple room. longer great! , thin bottom town. paying share except wait. avoid location. minute personal general
Semantic label/meaning:
Service

Topic #3:
pancakes else. perfect. you'd meal, card share sign we're rooms cook seriously inside. drink. picked charge selection. avoid recommended you, forget side, great! town. longer pizza, pepper mine somewhere personal room. normally okay, true visit. choices thank sour limited means terrible stayed apple paying and, whatever nearly except generally up, is, please they've honestly la minute fresh. small, times. orders bean tasty, recent life bars thinking oil clearly supposed thick , hostess creamy wall penn special. general covered dressing ... noodle burrito employees location. total area, wait. chili minutes. greasy pricey atmosphere, amazing! bottom hand weeks bread. best. thin finished
Semantic label/meaning:
Ambience

Topic #4:
you'd atmosphere, choices oil supposed we're hostess recommended meal, wall else. card total seriously pricey paying small, forget creamy up, nearly somewhere wait. terrible special. you, town. charge covered thick bottom ... drink. recent fresh. , generally apple stayed honestly please is, side, weeks personal selection. visit. burrito mine area, greasy minutes. cook rooms bars thin sign amazing! means hand picked thinking normally la dressing inside. noodle minute whatever bean share and, true thank chili pepper okay, they've location. employees sour except general clearly pancakes perfect. orders great! tasty, limited longer room. best. life times. avoid bread. pizza, penn finished
Semantic label/meaning:
Location

Part 5: Word2Vec for text reviews

In this section, we will fit a Word2Vec model to the Yelp reviews text. Word2Vec is a popular model for embedding words in Euclidean space so they can be analyzed similar to real-valued vectors. Contrary to popular belief, Word2Vec models are not deep neural models. Inspite of being shallow neural networks, they capture word associations and analogies remarkably well.

In [57]:
```
from pyspark.mllib.feature import Word2Vec
import re
pattern = re.compile('[\W_]+')
```

In [58]:
```
review_docs = reviews_rdd.map(lambda x : x['text'].lower().split())
review_docs = review_docs.map(lambda x : [pattern.sub('', w) for w in x])
print(review_docs.take(2))
```

```
[['thank', 'you', 'rob', 'i', 'truly', 'appreciated', 'all', 'the', 'help', 'i', 'received', 'from', 'this', 'agent', 'today', 'who', 'wa
s', 'able', 'to', 'removed', 'the', 'extra', 'charges', 'on', 'my', 'bill', 'that', 'the', 'pasadena', 'verizon', 'store', 'on', 'lake',
'was', 'charging', 'me', 'on', 'my', 'bill', 'for', 'upgrading', 'my', 'phone', 'when', 'i', 'went', 'in', 'i', 'was', 'having', 'problem
s', 'with', 'my', 'blacberry', 'and', 'had', 'to', 'switch', 'to', 'the', 'iphone', 'last', 'week', 'rob', 'from', 'the', 'pennsylvania',
'store', 'who', 'i', 'was', 'connected', 'today', 'was', 'able', 'to', 'look', 'at', 'my', 'bill', 'and', 'all', 'the', 'notes', 'and',
'correct', 'the', 'problem', 'immediately', 'great', 'customer', 'service', 'he', 'even', 'set', 'up', 'a', 'follow', 'up', 'phone', 'cal
l', 'with', 'me', 'on', 'july', '5th', 'to', 'make', 'sure', 'the', 'credit', 'goes', 'through', 'on', 'my', 'billi', 'cant', 'thank', 'h
im', 'enough'], ['after', 'waiting', 'for', 'almost', '30', 'minutes', 'to', 'trade', 'in', 'an', 'old', 'phone', 'part', 'of', 'the', 'b
uy', 'back', 'program', 'our', 'customer', 'service', 'rep', 'incorrectly', 'processed', 'the', 'transaction', 'this', 'led', 'to', 'us',
'waiting', 'another', '30', 'minutes', 'for', 'him', 'to', 'correct', 'it', 'dont', 'visit', 'this', 'store', 'if', 'you', 'want', 'pleas
ant', 'or', 'good', 'service']]
```

Question1: Fit a Word2Vec model to the review_docs RDD. Set the size of embedding vectors to 10, the random seed to 42, and the number of iterations to 10.

In [59]:
```
# TODO: Replace <FILL IN>

word2vec_model = Word2Vec().setVectorSize(10).setSeed(42).setNumIterations(10).fit(review_docs)
```

Let's us examine what words are closely associated with some example words. Run the cell below to see word associations. Feel free to add any additional words whose results you find interesting, but do not delete any of the words already in the list.

In [60]:
```
for word in ['salt', 'pepper', 'restaurant', 'italian', 'indian', 'chinese', 'direction', 'pittsburgh', 'burgh', 'city', 'location', 'cmu
    syms = word2vec_model.findSynonyms(word, 5)
    print('Words most similar to ', word, ' : ', [s[0] for s in syms])
```

```
Words most similar to  salt  :  ['cholula', 'fluff', 'milkshake', 'concoction', 'starch']
Words most similar to  pepper  :  ['celery', 'chilies', 'garlic', 'chutney', 'anchovy']
Words most similar to  restaurant  :  ['resturant', 'restaraunt', 'establishment', 'location', 'particular']
Words most similar to  italian  :  ['style', 'greek', 'shacks', 'homestyle', 'traditional']
Words most similar to  indian  :  ['chinese', 'mexican', 'chinesetaiwanese', 'cuisine', 'japanese']
Words most similar to  chinese  :  ['indian', 'mexican', 'chinesetaiwanese', 'japanese', 'cuisine']
Words most similar to  direction  :  ['driveway', 'journey', 'anger', 'upmc', 'greyhound']
Words most similar to  pittsburgh  :  ['pgh', 'connecticut', 'bloomfield', 'burgh', 'dc']
Words most similar to  burgh  :  ['city', 'berlin', 'squirrell', 'los', 'robinson']
Words most similar to  city  :  ['robinson', 'burg', 'bloomfield', 'eastside', 'burgh']
Words most similar to  location  :  ['neighborhood', 'scene', 'toonseum', 'pburgh', 'conveniently']
Words most similar to  cmu  :  ['campus', 'pitt', 'college', 'graduate', 'jitters']
Words most similar to  pizza  :  ['sub', 'barbecue', 'mediterranean', 'bbq', 'hunan']
```

Question2: What "synonyms" in the result above give rise to perfect analogies? Are there words in the result that are spurious and not good substitutes for the originally supplied word?

Answer: Spurious: Milkshake, cholula for salt ; direction -> anger location -> toonseum

Synonyms that are perfect:

chinese -> chinesetaiwanese indian -> cuisine pittsburgh -> burgh

Part 6: Frequent pattern mining using FP-Growth algorithm

In this section, we will mine frequent subsets of items that appear together in datapoints. This type of analysis is also known as frequent itemset mining or market basket analysis. Since the tags associated with Yelp businesses are sets, we can use them to carry out the frequent item set mining by employing the FP-Growth algorithm available in Spark.

In [61]:
```
from pyspark.mllib.fpm import FPGrowth
```

Question1: Fill in the required code to perform itemset mining on business categories represented as an RDD of sets. Train the FP-Growth algorithm with a minimum support parameter of 0.01 and 10 partitions.

In [62]:
```
# TODO: Replace <FILL IN>

business_categories = businesses_rdd.map(lambda business: business['categories'])

fpgrowth_model = FPGrowth.train(business_categories, minSupport=0.01, numPartitions=10)
result = sorted(fpgrowth_model.freqItemsets().collect(), key=lambda x: -x[1])
for fi in result:
    if len(fi[0]) > 1:
        print(fi)
```

```
23/04/30 22:53:53 WARN FPGrowth: Input data is not cached.
[Stage 7023:>                                                       (0 + 10) / 10]
```

```
FreqItemset(items=['Bars', 'Nightlife'], freq=3628)
FreqItemset(items=['Fashion', 'Shopping'], freq=2566)
FreqItemset(items=['Fast Food', 'Restaurants'], freq=2383)
FreqItemset(items=['Pizza', 'Restaurants'], freq=2223)
FreqItemset(items=['Mexican', 'Restaurants'], freq=2208)
FreqItemset(items=['American (Traditional)', 'Restaurants'], freq=2113)
FreqItemset(items=['Nightlife', 'Restaurants'], freq=2045)
FreqItemset(items=['Sandwiches', 'Restaurants'], freq=1981)
FreqItemset(items=['Bars', 'Nightlife', 'Restaurants'], freq=1934)
FreqItemset(items=['Bars', 'Restaurants'], freq=1934)
FreqItemset(items=['Coffee & Tea', 'Food'], freq=1890)
FreqItemset(items=['Food', 'Restaurants'], freq=1807)
FreqItemset(items=['Italian', 'Restaurants'], freq=1633)
FreqItemset(items=['Chinese', 'Restaurants'], freq=1496)
FreqItemset(items=['American (New)', 'Restaurants'], freq=1494)
FreqItemset(items=['Burgers', 'Restaurants'], freq=1481)
FreqItemset(items=['Hair Salons', 'Beauty & Spas'], freq=1388)
FreqItemset(items=['Hotels & Travel', 'Event Planning & Services'], freq=1339)
FreqItemset(items=['Hotels', 'Event Planning & Services'], freq=1307)
FreqItemset(items=['Hotels', 'Hotels & Travel'], freq=1307)
FreqItemset(items=['Hotels', 'Hotels & Travel', 'Event Planning & Services'], freq=1307)
FreqItemset(items=['Nail Salons', 'Beauty & Spas'], freq=1256)
FreqItemset(items=['Grocery', 'Food'], freq=1233)
FreqItemset(items=['Auto Repair', 'Automotive'], freq=1220)
FreqItemset(items=['Home & Garden', 'Shopping'], freq=1173)
FreqItemset(items=['Breakfast & Brunch', 'Restaurants'], freq=1116)
FreqItemset(items=['Doctors', 'Health & Medical'], freq=1077)
FreqItemset(items=['Fitness & Instruction', 'Active Life'], freq=1068)
FreqItemset(items=['Specialty Food', 'Food'], freq=1001)
FreqItemset(items=['Bakeries', 'Food'], freq=941)
FreqItemset(items=["Women's Clothing", 'Shopping'], freq=916)
FreqItemset(items=["Women's Clothing", 'Fashion'], freq=916)
FreqItemset(items=["Women's Clothing", 'Fashion', 'Shopping'], freq=916)
FreqItemset(items=['Ice Cream & Frozen Yogurt', 'Food'], freq=867)
FreqItemset(items=['Real Estate', 'Home Services'], freq=850)
FreqItemset(items=['Pubs', 'Bars'], freq=784)
FreqItemset(items=['Pubs', 'Bars', 'Nightlife'], freq=784)
FreqItemset(items=['Pubs', 'Nightlife'], freq=784)
FreqItemset(items=['Cafes', 'Restaurants'], freq=776)
FreqItemset(items=['Dentists', 'Health & Medical'], freq=752)
FreqItemset(items=['Japanese', 'Restaurants'], freq=746)
FreqItemset(items=['Sports Bars', 'Bars'], freq=713)
FreqItemset(items=['Sports Bars', 'Bars', 'Nightlife'], freq=713)
FreqItemset(items=['Sports Bars', 'Nightlife'], freq=713)
FreqItemset(items=['Sushi Bars', 'Restaurants'], freq=671)
FreqItemset(items=['Burgers', 'Fast Food'], freq=654)
FreqItemset(items=['Burgers', 'Fast Food', 'Restaurants'], freq=654)
FreqItemset(items=['Delis', 'Restaurants'], freq=649)
FreqItemset(items=['Italian', 'Pizza'], freq=641)
FreqItemset(items=['Italian', 'Pizza', 'Restaurants'], freq=641)
FreqItemset(items=['Pet Services', 'Pets'], freq=634)
```

```
FreqItemset(items=['American (Traditional)', 'Nightlife'], freq=617)
FreqItemset(items=['American (Traditional)', 'Nightlife', 'Restaurants'], freq=617)
FreqItemset(items=['American (Traditional)', 'Bars'], freq=612)
FreqItemset(items=['American (Traditional)', 'Bars', 'Nightlife'], freq=612)
FreqItemset(items=['American (Traditional)', 'Bars', 'Nightlife', 'Restaurants'], freq=612)
FreqItemset(items=['American (Traditional)', 'Bars', 'Restaurants'], freq=612)
```

Question2: Fill in the required code to perform itemset mining on business categories represented as an RDD of sets. Train the FP-Growth algorithm with a minimum support parameter of 0.001 and 10 partitions.

In [63]:
```
# TODO: Replace <FILL IN>

fpgrowth_model = FPGrowth.train(business_categories, minSupport=0.001, numPartitions=10)
result = sorted(fpgrowth_model.freqItemsets().collect(), key=lambda x: -x[1])
for fi in result:
    if len(fi[0]) > 1:
        print(fi)
```

```
23/04/30 22:53:56 WARN FPGrowth: Input data is not cached.
[Stage 7028:>                                                       (0 + 10) / 10]
```

```
FreqItemset(items=['Bars', 'Nightlife'], freq=3628)
FreqItemset(items=['Fashion', 'Shopping'], freq=2566)
FreqItemset(items=['Fast Food', 'Restaurants'], freq=2383)
FreqItemset(items=['Pizza', 'Restaurants'], freq=2223)
FreqItemset(items=['Mexican', 'Restaurants'], freq=2208)
FreqItemset(items=['American (Traditional)', 'Restaurants'], freq=2113)
FreqItemset(items=['Nightlife', 'Restaurants'], freq=2045)
FreqItemset(items=['Sandwiches', 'Restaurants'], freq=1981)
FreqItemset(items=['Bars', 'Nightlife', 'Restaurants'], freq=1934)
FreqItemset(items=['Bars', 'Restaurants'], freq=1934)
FreqItemset(items=['Coffee & Tea', 'Food'], freq=1890)
FreqItemset(items=['Food', 'Restaurants'], freq=1807)
FreqItemset(items=['Italian', 'Restaurants'], freq=1633)
FreqItemset(items=['Chinese', 'Restaurants'], freq=1496)
FreqItemset(items=['American (New)', 'Restaurants'], freq=1494)
FreqItemset(items=['Burgers', 'Restaurants'], freq=1481)
FreqItemset(items=['Hair Salons', 'Beauty & Spas'], freq=1388)
FreqItemset(items=['Hotels & Travel', 'Event Planning & Services'], freq=1339)
FreqItemset(items=['Hotels', 'Event Planning & Services'], freq=1307)
FreqItemset(items=['Hotels', 'Hotels & Travel'], freq=1307)
FreqItemset(items=['Hotels', 'Hotels & Travel', 'Event Planning & Services'], freq=1307)
FreqItemset(items=['Nail Salons', 'Beauty & Spas'], freq=1256)
FreqItemset(items=['Grocery', 'Food'], freq=1233)
FreqItemset(items=['Auto Repair', 'Automotive'], freq=1220)
FreqItemset(items=['Home & Garden', 'Shopping'], freq=1173)
FreqItemset(items=['Breakfast & Brunch', 'Restaurants'], freq=1116)
FreqItemset(items=['Doctors', 'Health & Medical'], freq=1077)
FreqItemset(items=['Fitness & Instruction', 'Active Life'], freq=1068)
FreqItemset(items=['Specialty Food', 'Food'], freq=1001)
FreqItemset(items=['Bakeries', 'Food'], freq=941)
FreqItemset(items=["Women's Clothing", 'Shopping'], freq=916)
FreqItemset(items=["Women's Clothing", 'Fashion'], freq=916)
FreqItemset(items=["Women's Clothing", 'Fashion', 'Shopping'], freq=916)
FreqItemset(items=['Ice Cream & Frozen Yogurt', 'Food'], freq=867)
FreqItemset(items=['Real Estate', 'Home Services'], freq=850)
FreqItemset(items=['Pubs', 'Bars'], freq=784)
FreqItemset(items=['Pubs', 'Bars', 'Nightlife'], freq=784)
FreqItemset(items=['Pubs', 'Nightlife'], freq=784)
FreqItemset(items=['Cafes', 'Restaurants'], freq=776)
FreqItemset(items=['Dentists', 'Health & Medical'], freq=752)
FreqItemset(items=['Japanese', 'Restaurants'], freq=746)
FreqItemset(items=['Sports Bars', 'Bars'], freq=713)
FreqItemset(items=['Sports Bars', 'Bars', 'Nightlife'], freq=713)
FreqItemset(items=['Sports Bars', 'Nightlife'], freq=713)
FreqItemset(items=['Sushi Bars', 'Restaurants'], freq=671)
FreqItemset(items=['Burgers', 'Fast Food'], freq=654)
FreqItemset(items=['Burgers', 'Fast Food', 'Restaurants'], freq=654)
FreqItemset(items=['Delis', 'Restaurants'], freq=649)
FreqItemset(items=['Italian', 'Pizza'], freq=641)
FreqItemset(items=['Italian', 'Pizza', 'Restaurants'], freq=641)
FreqItemset(items=['Pet Services', 'Pets'], freq=634)
```

```
FreqItemset(items=['American (Traditional)', 'Nightlife'], freq=617)
FreqItemset(items=['American (Traditional)', 'Nightlife', 'Restaurants'], freq=617)
FreqItemset(items=['American (Traditional)', 'Bars'], freq=612)
FreqItemset(items=['American (Traditional)', 'Bars', 'Nightlife'], freq=612)
FreqItemset(items=['American (Traditional)', 'Bars', 'Nightlife', 'Restaurants'], freq=612)
FreqItemset(items=['American (Traditional)', 'Bars', 'Restaurants'], freq=612)
FreqItemset(items=['Food', 'Shopping'], freq=604)
FreqItemset(items=['Beauty & Spas', 'Shopping'], freq=595)
FreqItemset(items=['Sporting Goods', 'Shopping'], freq=585)
FreqItemset(items=['Convenience Stores', 'Food'], freq=578)
FreqItemset(items=['Desserts', 'Food'], freq=576)
FreqItemset(items=['Department Stores', 'Shopping'], freq=573)
FreqItemset(items=['Department Stores', 'Fashion'], freq=573)
FreqItemset(items=['Department Stores', 'Fashion', 'Shopping'], freq=573)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Shopping'], freq=563)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Beauty & Spas'], freq=563)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Beauty & Spas', 'Shopping'], freq=563)
FreqItemset(items=['Drugstores', 'Shopping'], freq=561)
FreqItemset(items=['Gyms', 'Fitness & Instruction'], freq=560)
FreqItemset(items=['Gyms', 'Fitness & Instruction', 'Active Life'], freq=560)
FreqItemset(items=['Gyms', 'Active Life'], freq=560)
FreqItemset(items=['Steakhouses', 'Restaurants'], freq=554)
FreqItemset(items=['Seafood', 'Restaurants'], freq=554)
FreqItemset(items=['Lounges', 'Bars'], freq=536)
FreqItemset(items=['Lounges', 'Bars', 'Nightlife'], freq=536)
FreqItemset(items=['Lounges', 'Nightlife'], freq=536)
FreqItemset(items=['Day Spas', 'Beauty & Spas'], freq=519)
FreqItemset(items=['Chicken Wings', 'Restaurants'], freq=516)
FreqItemset(items=['Coffee & Tea', 'Food', 'Restaurants'], freq=513)
FreqItemset(items=['Coffee & Tea', 'Restaurants'], freq=513)
FreqItemset(items=['Sports Bars', 'Bars', 'Nightlife', 'Restaurants'], freq=512)
FreqItemset(items=['Sports Bars', 'Bars', 'Restaurants'], freq=512)
FreqItemset(items=['Sports Bars', 'Nightlife', 'Restaurants'], freq=512)
FreqItemset(items=['Sports Bars', 'Restaurants'], freq=512)
FreqItemset(items=['Accessories', 'Shopping'], freq=510)
FreqItemset(items=['Accessories', 'Fashion'], freq=510)
FreqItemset(items=['Accessories', 'Fashion', 'Shopping'], freq=510)
FreqItemset(items=['Books, Mags, Music & Video', 'Shopping'], freq=505)
FreqItemset(items=['General Dentistry', 'Dentists'], freq=496)
FreqItemset(items=['General Dentistry', 'Dentists', 'Health & Medical'], freq=496)
FreqItemset(items=['General Dentistry', 'Health & Medical'], freq=496)
FreqItemset(items=["Men's Clothing", 'Shopping'], freq=494)
FreqItemset(items=["Men's Clothing", 'Fashion'], freq=494)
FreqItemset(items=["Men's Clothing", 'Fashion', 'Shopping'], freq=494)
FreqItemset(items=['Flowers & Gifts', 'Shopping'], freq=492)
FreqItemset(items=['Arts & Crafts', 'Shopping'], freq=492)
FreqItemset(items=['Mediterranean', 'Restaurants'], freq=490)
FreqItemset(items=['Massage', 'Beauty & Spas'], freq=489)
FreqItemset(items=['Tires', 'Automotive'], freq=487)
FreqItemset(items=['Arts & Entertainment', 'Nightlife'], freq=479)
FreqItemset(items=['Barbeque', 'Restaurants'], freq=477)
```

```
FreqItemset(items=['Skin Care', 'Beauty & Spas'], freq=475)
FreqItemset(items=['Thai', 'Restaurants'], freq=472)
FreqItemset(items=['Apartments', 'Home Services'], freq=469)
FreqItemset(items=['Apartments', 'Real Estate'], freq=469)
FreqItemset(items=['Apartments', 'Real Estate', 'Home Services'], freq=469)
FreqItemset(items=['Beer, Wine & Spirits', 'Food'], freq=465)
FreqItemset(items=['Jewelry', 'Shopping'], freq=454)
FreqItemset(items=['American (New)', 'Nightlife'], freq=446)
FreqItemset(items=['American (New)', 'Nightlife', 'Restaurants'], freq=446)
FreqItemset(items=['Asian Fusion', 'Restaurants'], freq=446)
FreqItemset(items=['Hair Removal', 'Beauty & Spas'], freq=439)
FreqItemset(items=['French', 'Restaurants'], freq=433)
FreqItemset(items=['Oil Change Stations', 'Automotive'], freq=433)
FreqItemset(items=['Sandwiches', 'Fast Food'], freq=428)
FreqItemset(items=['Sandwiches', 'Fast Food', 'Restaurants'], freq=428)
FreqItemset(items=['Dry Cleaning & Laundry', 'Local Services'], freq=425)
FreqItemset(items=['American (New)', 'Bars'], freq=425)
FreqItemset(items=['American (New)', 'Bars', 'Nightlife'], freq=425)
FreqItemset(items=['American (New)', 'Bars', 'Nightlife', 'Restaurants'], freq=425)
FreqItemset(items=['American (New)', 'Bars', 'Restaurants'], freq=425)
FreqItemset(items=['Buffets', 'Restaurants'], freq=420)
FreqItemset(items=['Gas & Service Stations', 'Automotive'], freq=412)
FreqItemset(items=['Veterinarians', 'Pets'], freq=401)
FreqItemset(items=['Performing Arts', 'Arts & Entertainment'], freq=401)
FreqItemset(items=['Venues & Event Spaces', 'Event Planning & Services'], freq=397)
FreqItemset(items=['Furniture Stores', 'Shopping'], freq=397)
FreqItemset(items=['Furniture Stores', 'Home & Garden'], freq=397)
FreqItemset(items=['Furniture Stores', 'Home & Garden', 'Shopping'], freq=397)
FreqItemset(items=['Barbers', 'Beauty & Spas'], freq=379)
FreqItemset(items=['Sandwiches', 'Food'], freq=379)
FreqItemset(items=['Sandwiches', 'Food', 'Restaurants'], freq=379)
FreqItemset(items=['Pubs', 'Bars', 'Nightlife', 'Restaurants'], freq=376)
FreqItemset(items=['Pubs', 'Bars', 'Restaurants'], freq=376)
FreqItemset(items=['Pubs', 'Nightlife', 'Restaurants'], freq=376)
FreqItemset(items=['Pubs', 'Restaurants'], freq=376)
FreqItemset(items=['Indian', 'Restaurants'], freq=376)
FreqItemset(items=['Pet Groomers', 'Pets'], freq=373)
FreqItemset(items=['Pet Groomers', 'Pet Services'], freq=373)
FreqItemset(items=['Pet Groomers', 'Pet Services', 'Pets'], freq=373)
FreqItemset(items=['Mexican', 'Fast Food'], freq=366)
FreqItemset(items=['Mexican', 'Fast Food', 'Restaurants'], freq=366)
FreqItemset(items=['Greek', 'Restaurants'], freq=363)
FreqItemset(items=['Banks & Credit Unions', 'Financial Services'], freq=355)
FreqItemset(items=['Trainers', 'Fitness & Instruction'], freq=353)
FreqItemset(items=['Trainers', 'Fitness & Instruction', 'Active Life'], freq=353)
FreqItemset(items=['Trainers', 'Active Life'], freq=353)
FreqItemset(items=['Parks', 'Active Life'], freq=352)
FreqItemset(items=['Sushi Bars', 'Japanese'], freq=345)
FreqItemset(items=['Sushi Bars', 'Japanese', 'Restaurants'], freq=345)
FreqItemset(items=["Men's Clothing", "Women's Clothing"], freq=343)
FreqItemset(items=["Men's Clothing", "Women's Clothing", 'Shopping'], freq=343)
```

```
FreqItemset(items=["Men's Clothing", "Women's Clothing", 'Fashion'], freq=343)
FreqItemset(items=["Men's Clothing", "Women's Clothing", 'Fashion', 'Shopping'], freq=343)
FreqItemset(items=['Music Venues', 'Nightlife'], freq=343)
FreqItemset(items=['Music Venues', 'Arts & Entertainment'], freq=343)
FreqItemset(items=['Music Venues', 'Arts & Entertainment', 'Nightlife'], freq=343)
FreqItemset(items=['Auto Parts & Supplies', 'Automotive'], freq=334)
FreqItemset(items=['Shoe Stores', 'Shopping'], freq=331)
FreqItemset(items=['Shoe Stores', 'Fashion'], freq=331)
FreqItemset(items=['Shoe Stores', 'Fashion', 'Shopping'], freq=331)
FreqItemset(items=['Dance Clubs', 'Nightlife'], freq=330)
FreqItemset(items=['Juice Bars & Smoothies', 'Food'], freq=325)
FreqItemset(items=['Optometrists', 'Health & Medical'], freq=324)
FreqItemset(items=['Accessories', "Women's Clothing"], freq=323)
FreqItemset(items=['Accessories', "Women's Clothing", 'Shopping'], freq=323)
FreqItemset(items=['Accessories', "Women's Clothing", 'Fashion'], freq=323)
FreqItemset(items=['Accessories', "Women's Clothing", 'Fashion', 'Shopping'], freq=323)
FreqItemset(items=['Car Wash', 'Automotive'], freq=323)
FreqItemset(items=['Oil Change Stations', 'Auto Repair'], freq=322)
FreqItemset(items=['Oil Change Stations', 'Auto Repair', 'Automotive'], freq=322)
FreqItemset(items=['Pet Stores', 'Pets'], freq=319)
FreqItemset(items=['Drugstores', 'Food'], freq=317)
FreqItemset(items=['Drugstores', 'Food', 'Shopping'], freq=317)
FreqItemset(items=['Home Decor', 'Shopping'], freq=304)
FreqItemset(items=['Home Decor', 'Home & Garden'], freq=304)
FreqItemset(items=['Home Decor', 'Home & Garden', 'Shopping'], freq=304)
FreqItemset(items=['Delis', 'Sandwiches'], freq=304)
FreqItemset(items=['Delis', 'Sandwiches', 'Restaurants'], freq=304)
FreqItemset(items=['Tires', 'Auto Repair'], freq=303)
FreqItemset(items=['Tires', 'Auto Repair', 'Automotive'], freq=303)
FreqItemset(items=['Diners', 'Restaurants'], freq=302)
FreqItemset(items=['Printing Services', 'Local Services'], freq=299)
FreqItemset(items=['Mobile Phones', 'Shopping'], freq=287)
FreqItemset(items=['Bakeries', 'Food', 'Restaurants'], freq=284)
FreqItemset(items=['Bakeries', 'Restaurants'], freq=284)
FreqItemset(items=['Grocery', 'Food', 'Shopping'], freq=284)
FreqItemset(items=['Grocery', 'Shopping'], freq=284)
FreqItemset(items=['Sports Bars', 'American (Traditional)'], freq=283)
FreqItemset(items=['Sports Bars', 'American (Traditional)', 'Bars'], freq=283)
FreqItemset(items=['Sports Bars', 'American (Traditional)', 'Bars', 'Nightlife'], freq=283)
FreqItemset(items=['Sports Bars', 'American (Traditional)', 'Bars', 'Nightlife', 'Restaurants'], freq=283)
FreqItemset(items=['Sports Bars', 'American (Traditional)', 'Bars', 'Restaurants'], freq=283)
FreqItemset(items=['Sports Bars', 'American (Traditional)', 'Nightlife'], freq=283)
FreqItemset(items=['Sports Bars', 'American (Traditional)', 'Nightlife', 'Restaurants'], freq=283)
FreqItemset(items=['Sports Bars', 'American (Traditional)', 'Restaurants'], freq=283)
FreqItemset(items=['Vietnamese', 'Restaurants'], freq=283)
FreqItemset(items=['Health & Medical', 'Beauty & Spas'], freq=282)
FreqItemset(items=['Cafes', 'Food'], freq=281)
FreqItemset(items=['Cafes', 'Food', 'Restaurants'], freq=281)
FreqItemset(items=['Transportation', 'Hotels & Travel'], freq=271)
FreqItemset(items=['Electronics', 'Shopping'], freq=271)
FreqItemset(items=['Cosmetic Dentists', 'Dentists'], freq=265)
```

```
FreqItemset(items=['Cosmetic Dentists', 'Dentists', 'Health & Medical'], freq=265)
FreqItemset(items=['Cosmetic Dentists', 'Health & Medical'], freq=265)
FreqItemset(items=['Home Services', 'Shopping'], freq=257)
FreqItemset(items=['Pet Boarding/Pet Sitting', 'Pets'], freq=255)
FreqItemset(items=['Pet Boarding/Pet Sitting', 'Pet Services'], freq=255)
FreqItemset(items=['Pet Boarding/Pet Sitting', 'Pet Services', 'Pets'], freq=255)
FreqItemset(items=['Trainers', 'Gyms'], freq=251)
FreqItemset(items=['Trainers', 'Gyms', 'Fitness & Instruction'], freq=251)
FreqItemset(items=['Trainers', 'Gyms', 'Fitness & Instruction', 'Active Life'], freq=251)
FreqItemset(items=['Trainers', 'Gyms', 'Active Life'], freq=251)
FreqItemset(items=['Tex-Mex', 'Restaurants'], freq=250)
FreqItemset(items=['Local Services', 'Shopping'], freq=250)
FreqItemset(items=['Chiropractors', 'Health & Medical'], freq=249)
FreqItemset(items=['Breakfast & Brunch', 'Food'], freq=248)
FreqItemset(items=['Breakfast & Brunch', 'Food', 'Restaurants'], freq=248)
FreqItemset(items=['Arts & Entertainment', 'Shopping'], freq=245)
FreqItemset(items=['Vegetarian', 'Restaurants'], freq=244)
FreqItemset(items=['Donuts', 'Food'], freq=243)
FreqItemset(items=['Thrift Stores', 'Shopping'], freq=243)
FreqItemset(items=['Eyewear & Opticians', 'Shopping'], freq=241)
FreqItemset(items=['Wine Bars', 'Bars'], freq=236)
FreqItemset(items=['Wine Bars', 'Bars', 'Nightlife'], freq=236)
FreqItemset(items=['Wine Bars', 'Nightlife'], freq=236)
FreqItemset(items=['Burgers', 'American (Traditional)'], freq=236)
FreqItemset(items=['Burgers', 'American (Traditional)', 'Restaurants'], freq=236)
FreqItemset(items=['Heating & Air Conditioning/HVAC', 'Home Services'], freq=234)
FreqItemset(items=['Bookstores', 'Books, Mags, Music & Video'], freq=233)
FreqItemset(items=['Bookstores', 'Books, Mags, Music & Video', 'Shopping'], freq=233)
FreqItemset(items=['Bookstores', 'Shopping'], freq=233)
FreqItemset(items=['Florists', 'Flowers & Gifts'], freq=233)
FreqItemset(items=['Florists', 'Flowers & Gifts', 'Shopping'], freq=233)
FreqItemset(items=['Florists', 'Shopping'], freq=233)
FreqItemset(items=['Cosmetic Dentists', 'General Dentistry'], freq=230)
FreqItemset(items=['Cosmetic Dentists', 'General Dentistry', 'Dentists'], freq=230)
FreqItemset(items=['Cosmetic Dentists', 'General Dentistry', 'Dentists', 'Health & Medical'], freq=230)
FreqItemset(items=['Cosmetic Dentists', 'General Dentistry', 'Health & Medical'], freq=230)
FreqItemset(items=['Car Dealers', 'Automotive'], freq=230)
FreqItemset(items=['Caterers', 'Event Planning & Services'], freq=229)
FreqItemset(items=['Salad', 'Restaurants'], freq=228)
FreqItemset(items=['Medical Centers', 'Health & Medical'], freq=227)
FreqItemset(items=['Sandwiches', 'Pizza'], freq=226)
FreqItemset(items=['Sandwiches', 'Pizza', 'Restaurants'], freq=226)
FreqItemset(items=['Casinos', 'Arts & Entertainment'], freq=225)
FreqItemset(items=['Hot Dogs', 'Restaurants'], freq=223)
FreqItemset(items=['Used, Vintage & Consignment', 'Shopping'], freq=219)
FreqItemset(items=['Used, Vintage & Consignment', 'Fashion'], freq=219)
FreqItemset(items=['Used, Vintage & Consignment', 'Fashion', 'Shopping'], freq=219)
FreqItemset(items=['Shipping Centers', 'Local Services'], freq=216)
FreqItemset(items=['Golf', 'Active Life'], freq=215)
FreqItemset(items=['Specialty Schools', 'Education'], freq=212)
FreqItemset(items=['Hardware Stores', 'Shopping'], freq=211)
```

```
FreqItemset(items=['Hardware Stores', 'Home & Garden'], freq=211)
FreqItemset(items=['Hardware Stores', 'Home & Garden', 'Shopping'], freq=211)
FreqItemset(items=['Sporting Goods', 'Fashion'], freq=211)
FreqItemset(items=['Sporting Goods', 'Fashion', 'Shopping'], freq=211)
FreqItemset(items=['Tanning', 'Beauty & Spas'], freq=210)
FreqItemset(items=['Party & Event Planning', 'Event Planning & Services'], freq=210)
FreqItemset(items=['Middle Eastern', 'Restaurants'], freq=209)
FreqItemset(items=['Event Planning & Services', 'Shopping'], freq=208)
FreqItemset(items=['Health & Medical', 'Shopping'], freq=208)
FreqItemset(items=['Hair Stylists', 'Hair Salons'], freq=207)
FreqItemset(items=['Hair Stylists', 'Hair Salons', 'Beauty & Spas'], freq=207)
FreqItemset(items=['Hair Stylists', 'Beauty & Spas'], freq=207)
FreqItemset(items=['Yoga', 'Fitness & Instruction'], freq=206)
FreqItemset(items=['Yoga', 'Fitness & Instruction', 'Active Life'], freq=206)
FreqItemset(items=['Yoga', 'Active Life'], freq=206)
FreqItemset(items=['Event Planning & Services', 'Restaurants'], freq=205)
FreqItemset(items=['Tex-Mex', 'Mexican'], freq=205)
FreqItemset(items=['Tex-Mex', 'Mexican', 'Restaurants'], freq=205)
FreqItemset(items=['Art Galleries', 'Shopping'], freq=204)
FreqItemset(items=['Art Galleries', 'Arts & Entertainment'], freq=204)
FreqItemset(items=['Art Galleries', 'Arts & Entertainment', 'Shopping'], freq=204)
FreqItemset(items=['Dive Bars', 'Bars'], freq=203)
FreqItemset(items=['Dive Bars', 'Bars', 'Nightlife'], freq=203)
FreqItemset(items=['Dive Bars', 'Nightlife'], freq=203)
FreqItemset(items=['Contractors', 'Home Services'], freq=202)
FreqItemset(items=['Hobby Shops', 'Shopping'], freq=197)
FreqItemset(items=['Tobacco Shops', 'Shopping'], freq=195)
FreqItemset(items=['Convenience Stores', 'Food', 'Shopping'], freq=194)
FreqItemset(items=['Convenience Stores', 'Shopping'], freq=194)
FreqItemset(items=['Specialty Food', 'Food', 'Restaurants'], freq=194)
FreqItemset(items=['Specialty Food', 'Restaurants'], freq=194)
FreqItemset(items=['Delis', 'Food'], freq=192)
FreqItemset(items=['Delis', 'Food', 'Restaurants'], freq=192)
FreqItemset(items=['Chicken Wings', 'Pizza'], freq=191)
FreqItemset(items=['Chicken Wings', 'Pizza', 'Restaurants'], freq=191)
FreqItemset(items=['Ethnic Food', 'Specialty Food'], freq=190)
FreqItemset(items=['Ethnic Food', 'Specialty Food', 'Food'], freq=190)
FreqItemset(items=['Ethnic Food', 'Food'], freq=190)
FreqItemset(items=['Eyelash Service', 'Beauty & Spas'], freq=190)
FreqItemset(items=['Lounges', 'Bars', 'Nightlife', 'Restaurants'], freq=190)
FreqItemset(items=['Lounges', 'Bars', 'Restaurants'], freq=190)
FreqItemset(items=['Lounges', 'Nightlife', 'Restaurants'], freq=190)
FreqItemset(items=['Lounges', 'Restaurants'], freq=190)
FreqItemset(items=['Arts & Entertainment', 'Bars'], freq=190)
FreqItemset(items=['Arts & Entertainment', 'Bars', 'Nightlife'], freq=190)
FreqItemset(items=['Plumbing', 'Home Services'], freq=189)
FreqItemset(items=['Sports Wear', 'Sporting Goods'], freq=188)
FreqItemset(items=['Sports Wear', 'Sporting Goods', 'Shopping'], freq=188)
FreqItemset(items=['Sports Wear', 'Sporting Goods', 'Fashion'], freq=188)
FreqItemset(items=['Sports Wear', 'Sporting Goods', 'Fashion', 'Shopping'], freq=188)
FreqItemset(items=['Sports Wear', 'Shopping'], freq=188)
```

```
FreqItemset(items=['Sports Wear', 'Fashion'], freq=188)
FreqItemset(items=['Sports Wear', 'Fashion', 'Shopping'], freq=188)
FreqItemset(items=['Korean', 'Restaurants'], freq=186)
FreqItemset(items=['Bagels', 'Food'], freq=185)
FreqItemset(items=['Arts & Entertainment', 'Event Planning & Services'], freq=184)
FreqItemset(items=['Oil Change Stations', 'Tires'], freq=183)
FreqItemset(items=['Oil Change Stations', 'Tires', 'Automotive'], freq=183)
FreqItemset(items=['Canadian (New)', 'Restaurants'], freq=183)
FreqItemset(items=['Shipping Centers', 'Printing Services'], freq=181)
FreqItemset(items=['Shipping Centers', 'Printing Services', 'Local Services'], freq=181)
FreqItemset(items=['Oil Change Stations', 'Tires', 'Auto Repair'], freq=181)
FreqItemset(items=['Oil Change Stations', 'Tires', 'Auto Repair', 'Automotive'], freq=181)
FreqItemset(items=['Body Shops', 'Automotive'], freq=181)
FreqItemset(items=['Greek', 'Mediterranean'], freq=181)
FreqItemset(items=['Greek', 'Mediterranean', 'Restaurants'], freq=181)
FreqItemset(items=['Bikes', 'Sporting Goods'], freq=180)
FreqItemset(items=['Bikes', 'Sporting Goods', 'Shopping'], freq=180)
FreqItemset(items=['Bikes', 'Shopping'], freq=180)
FreqItemset(items=['Tours', 'Hotels & Travel'], freq=179)
FreqItemset(items=['Sewing & Alterations', 'Local Services'], freq=178)
FreqItemset(items=['Food Trucks', 'Food'], freq=178)
FreqItemset(items=['Cinema', 'Arts & Entertainment'], freq=178)
FreqItemset(items=['Nightlife', 'Food'], freq=177)
FreqItemset(items=['Drugstores', 'Convenience Stores'], freq=177)
FreqItemset(items=['Drugstores', 'Convenience Stores', 'Food'], freq=177)
FreqItemset(items=['Drugstores', 'Convenience Stores', 'Food', 'Shopping'], freq=177)
FreqItemset(items=['Drugstores', 'Convenience Stores', 'Shopping'], freq=177)
FreqItemset(items=['Arts & Entertainment', 'Restaurants'], freq=176)
FreqItemset(items=['Car Rental', 'Hotels & Travel'], freq=176)
FreqItemset(items=['Automotive', 'Food'], freq=175)
FreqItemset(items=['IT Services & Computer Repair', 'Local Services'], freq=174)
FreqItemset(items=['Breakfast & Brunch', 'American (Traditional)'], freq=174)
FreqItemset(items=['Breakfast & Brunch', 'American (Traditional)', 'Restaurants'], freq=174)
FreqItemset(items=['Eyewear & Opticians', 'Health & Medical'], freq=173)
FreqItemset(items=['Eyewear & Opticians', 'Health & Medical', 'Shopping'], freq=173)
FreqItemset(items=['Optometrists', 'Shopping'], freq=171)
FreqItemset(items=['Optometrists', 'Health & Medical', 'Shopping'], freq=171)
FreqItemset(items=['Drugstores', 'Beauty & Spas'], freq=170)
FreqItemset(items=['Drugstores', 'Beauty & Spas', 'Shopping'], freq=170)
FreqItemset(items=['Beauty & Spas', 'Food'], freq=169)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply'], freq=169)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Shopping'], freq=169)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Beauty & Spas'], freq=169)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Beauty & Spas', 'Shopping'], freq=169)
FreqItemset(items=['Wine Bars', 'Bars', 'Nightlife', 'Restaurants'], freq=167)
FreqItemset(items=['Wine Bars', 'Bars', 'Restaurants'], freq=167)
FreqItemset(items=['Wine Bars', 'Nightlife', 'Restaurants'], freq=167)
FreqItemset(items=['Wine Bars', 'Restaurants'], freq=167)
FreqItemset(items=['Eyewear & Opticians', 'Optometrists'], freq=167)
FreqItemset(items=['Eyewear & Opticians', 'Optometrists', 'Shopping'], freq=167)
FreqItemset(items=['Eyewear & Opticians', 'Optometrists', 'Health & Medical'], freq=167)
```

```
FreqItemset(items=['Eyewear & Opticians', 'Optometrists', 'Health & Medical', 'Shopping'], freq=167)
FreqItemset(items=['Gas & Service Stations', 'Food'], freq=166)
FreqItemset(items=['Gas & Service Stations', 'Automotive', 'Food'], freq=166)
FreqItemset(items=['Photographers', 'Event Planning & Services'], freq=165)
FreqItemset(items=['Discount Store', 'Shopping'], freq=164)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Food'], freq=163)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Food', 'Shopping'], freq=163)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Beauty & Spas', 'Food'], freq=163)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Beauty & Spas', 'Food', 'Shopping'], freq=163)
FreqItemset(items=['Beauty & Spas', 'Food', 'Shopping'], freq=163)
FreqItemset(items=['Antiques', 'Shopping'], freq=162)
FreqItemset(items=['Convenience Stores', 'Automotive'], freq=161)
FreqItemset(items=['Convenience Stores', 'Automotive', 'Food'], freq=161)
FreqItemset(items=['Home & Garden', 'Home Services'], freq=161)
FreqItemset(items=['Home & Garden', 'Home Services', 'Shopping'], freq=161)
FreqItemset(items=['Makeup Artists', 'Beauty & Spas'], freq=161)
FreqItemset(items=['Convenience Stores', 'Beauty & Spas'], freq=160)
FreqItemset(items=['Convenience Stores', 'Beauty & Spas', 'Food'], freq=160)
FreqItemset(items=['Convenience Stores', 'Beauty & Spas', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Convenience Stores', 'Beauty & Spas', 'Shopping'], freq=160)
FreqItemset(items=['Cafes', 'Coffee & Tea'], freq=160)
FreqItemset(items=['Cafes', 'Coffee & Tea', 'Food'], freq=160)
FreqItemset(items=['Cafes', 'Coffee & Tea', 'Food', 'Restaurants'], freq=160)
FreqItemset(items=['Cafes', 'Coffee & Tea', 'Restaurants'], freq=160)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Convenience Stores'], freq=160)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Convenience Stores', 'Food'], freq=160)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Convenience Stores', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Convenience Stores', 'Shopping'], freq=160)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Convenience Stores', 'Beauty & Spas'], freq=160)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Convenience Stores', 'Beauty & Spas', 'Food'], freq=160)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Convenience Stores', 'Beauty & Spas', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Cosmetics & Beauty Supply', 'Convenience Stores', 'Beauty & Spas', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Convenience Stores', 'Beauty & Spas'], freq=160)
FreqItemset(items=['Drugstores', 'Convenience Stores', 'Beauty & Spas', 'Food'], freq=160)
FreqItemset(items=['Drugstores', 'Convenience Stores', 'Beauty & Spas', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Convenience Stores', 'Beauty & Spas', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Convenience Stores'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Convenience Stores', 'Food'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Convenience Stores', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Convenience Stores', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Convenience Stores', 'Beauty & Spas'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Convenience Stores', 'Beauty & Spas', 'Food'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Convenience Stores', 'Beauty & Spas', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Convenience Stores', 'Beauty & Spas', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Food'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Beauty & Spas', 'Food'], freq=160)
FreqItemset(items=['Drugstores', 'Cosmetics & Beauty Supply', 'Beauty & Spas', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Drugstores', 'Beauty & Spas', 'Food'], freq=160)
FreqItemset(items=['Drugstores', 'Beauty & Spas', 'Food', 'Shopping'], freq=160)
FreqItemset(items=['Health Markets', 'Specialty Food'], freq=160)
```

```
FreqItemset(items=['Health Markets', 'Specialty Food', 'Food'], freq=160)
FreqItemset(items=['Health Markets', 'Food'], freq=160)
FreqItemset(items=['Festivals', 'Arts & Entertainment'], freq=159)
FreqItemset(items=['Desserts', 'Bakeries'], freq=158)
FreqItemset(items=['Desserts', 'Bakeries', 'Food'], freq=158)
FreqItemset(items=['Gas & Service Stations', 'Convenience Stores'], freq=157)
FreqItemset(items=['Gas & Service Stations', 'Convenience Stores', 'Food'], freq=157)
FreqItemset(items=['Gas & Service Stations', 'Convenience Stores', 'Automotive'], freq=157)
FreqItemset(items=['Gas & Service Stations', 'Convenience Stores', 'Automotive', 'Food'], freq=157)
FreqItemset(items=['Museums', 'Arts & Entertainment'], freq=157)
FreqItemset(items=['Nurseries & Gardening', 'Shopping'], freq=156)
FreqItemset(items=['Nurseries & Gardening', 'Home & Garden'], freq=156)
FreqItemset(items=['Nurseries & Gardening', 'Home & Garden', 'Shopping'], freq=156)
FreqItemset(items=['Gluten-Free', 'Restaurants'], freq=156)
FreqItemset(items=['Auto Glass Services', 'Automotive'], freq=155)
FreqItemset(items=['Home Cleaning', 'Home Services'], freq=155)
FreqItemset(items=['Shopping Centers', 'Shopping'], freq=154)
FreqItemset(items=['Pet Stores', 'Pet Services'], freq=153)
FreqItemset(items=['Pet Stores', 'Pet Services', 'Pets'], freq=153)
FreqItemset(items=['Massage Therapy', 'Health & Medical'], freq=152)
FreqItemset(items=['Family Practice', 'Doctors'], freq=152)
FreqItemset(items=['Family Practice', 'Doctors', 'Health & Medical'], freq=152)
FreqItemset(items=['Family Practice', 'Health & Medical'], freq=152)
FreqItemset(items=['Local Services', 'Home Services'], freq=152)
FreqItemset(items=['Latin American', 'Restaurants'], freq=151)
FreqItemset(items=['Professional Services', 'Home Services'], freq=151)
FreqItemset(items=['Toy Stores', 'Shopping'], freq=151)
FreqItemset(items=['Tattoo', 'Beauty & Spas'], freq=151)
FreqItemset(items=['Appliances', 'Shopping'], freq=150)
FreqItemset(items=['Appliances', 'Home & Garden'], freq=150)
FreqItemset(items=['Appliances', 'Home & Garden', 'Shopping'], freq=150)
FreqItemset(items=['Dance Clubs', 'Bars'], freq=149)
FreqItemset(items=['Dance Clubs', 'Bars', 'Nightlife'], freq=149)
FreqItemset(items=['Massage', 'Day Spas'], freq=149)
FreqItemset(items=['Massage', 'Day Spas', 'Beauty & Spas'], freq=149)
FreqItemset(items=['Tex-Mex', 'Fast Food'], freq=148)
FreqItemset(items=['Tex-Mex', 'Fast Food', 'Restaurants'], freq=148)
FreqItemset(items=['Tex-Mex', 'Mexican', 'Fast Food'], freq=146)
FreqItemset(items=['Tex-Mex', 'Mexican', 'Fast Food', 'Restaurants'], freq=146)
FreqItemset(items=['Nail Salons', 'Hair Salons'], freq=146)
FreqItemset(items=['Nail Salons', 'Hair Salons', 'Beauty & Spas'], freq=146)
FreqItemset(items=['Carpet Cleaning', 'Local Services'], freq=145)
FreqItemset(items=['Breakfast & Brunch', 'Sandwiches'], freq=145)
FreqItemset(items=['Breakfast & Brunch', 'Sandwiches', 'Restaurants'], freq=145)
FreqItemset(items=['Bars', 'Food'], freq=145)
FreqItemset(items=['Bars', 'Nightlife', 'Food'], freq=145)
FreqItemset(items=['British', 'Restaurants'], freq=143)
FreqItemset(items=['Professional Services', 'Local Services'], freq=141)
FreqItemset(items=['Landscaping', 'Home Services'], freq=140)
FreqItemset(items=['Urgent Care', 'Health & Medical'], freq=140)
FreqItemset(items=['Gastropubs', 'Restaurants'], freq=139)
```

```
FreqItemset(items=['Auto Detailing', 'Automotive'], freq=139)
FreqItemset(items=['Coffee & Tea', 'Sandwiches'], freq=138)
FreqItemset(items=['Coffee & Tea', 'Sandwiches', 'Food'], freq=138)
FreqItemset(items=['Coffee & Tea', 'Sandwiches', 'Food', 'Restaurants'], freq=138)
FreqItemset(items=['Coffee & Tea', 'Sandwiches', 'Restaurants'], freq=138)
FreqItemset(items=['Real Estate Services', 'Home Services'], freq=138)
FreqItemset(items=['Real Estate Services', 'Real Estate'], freq=138)
FreqItemset(items=['Real Estate Services', 'Real Estate', 'Home Services'], freq=138)
FreqItemset(items=['Drugstores', 'Grocery'], freq=137)
FreqItemset(items=['Drugstores', 'Grocery', 'Food'], freq=137)
FreqItemset(items=['Drugstores', 'Grocery', 'Food', 'Shopping'], freq=137)
FreqItemset(items=['Drugstores', 'Grocery', 'Shopping'], freq=137)
FreqItemset(items=['Hair Removal', 'Skin Care'], freq=137)
FreqItemset(items=['Hair Removal', 'Skin Care', 'Beauty & Spas'], freq=137)
FreqItemset(items=['Chocolatiers & Shops', 'Specialty Food'], freq=136)
FreqItemset(items=['Chocolatiers & Shops', 'Specialty Food', 'Food'], freq=136)
FreqItemset(items=['Chocolatiers & Shops', 'Food'], freq=136)
FreqItemset(items=['Ice Cream & Frozen Yogurt', 'Food', 'Restaurants'], freq=136)
FreqItemset(items=['Ice Cream & Frozen Yogurt', 'Restaurants'], freq=136)
FreqItemset(items=['Arts & Crafts', 'Event Planning & Services'], freq=135)
FreqItemset(items=['Arts & Crafts', 'Event Planning & Services', 'Shopping'], freq=135)
FreqItemset(items=['Southern', 'Restaurants'], freq=134)
FreqItemset(items=['Donuts', 'Coffee & Tea'], freq=133)
FreqItemset(items=['Donuts', 'Coffee & Tea', 'Food'], freq=133)
FreqItemset(items=['Breweries', 'Food'], freq=132)
FreqItemset(items=['Notaries', 'Local Services'], freq=132)
FreqItemset(items=['Sports Bars', 'American (New)'], freq=132)
FreqItemset(items=['Sports Bars', 'American (New)', 'Bars'], freq=132)
FreqItemset(items=['Sports Bars', 'American (New)', 'Bars', 'Nightlife'], freq=132)
FreqItemset(items=['Sports Bars', 'American (New)', 'Bars', 'Nightlife', 'Restaurants'], freq=132)
FreqItemset(items=['Sports Bars', 'American (New)', 'Bars', 'Restaurants'], freq=132)
FreqItemset(items=['Sports Bars', 'American (New)', 'Nightlife'], freq=132)
FreqItemset(items=['Sports Bars', 'American (New)', 'Nightlife', 'Restaurants'], freq=132)
FreqItemset(items=['Sports Bars', 'American (New)', 'Restaurants'], freq=132)
FreqItemset(items=['Specialty Food', 'Grocery'], freq=132)
FreqItemset(items=['Specialty Food', 'Grocery', 'Food'], freq=132)
FreqItemset(items=['Medical Spas', 'Beauty & Spas'], freq=132)
FreqItemset(items=['Medical Spas', 'Health & Medical'], freq=132)
FreqItemset(items=['Medical Spas', 'Health & Medical', 'Beauty & Spas'], freq=132)
FreqItemset(items=['Movers', 'Home Services'], freq=131)
FreqItemset(items=['Farmers Market', 'Food'], freq=130)
FreqItemset(items=['Burgers', 'Nightlife'], freq=130)
FreqItemset(items=['Burgers', 'Nightlife', 'Restaurants'], freq=130)
FreqItemset(items=['Day Spas', 'Hair Salons'], freq=129)
FreqItemset(items=['Day Spas', 'Hair Salons', 'Beauty & Spas'], freq=129)
FreqItemset(items=['Pet Stores', 'Pet Groomers'], freq=129)
FreqItemset(items=['Pet Stores', 'Pet Groomers', 'Pets'], freq=129)
FreqItemset(items=['Pet Stores', 'Pet Groomers', 'Pet Services'], freq=129)
FreqItemset(items=['Pet Stores', 'Pet Groomers', 'Pet Services', 'Pets'], freq=129)
FreqItemset(items=['Obstetricians & Gynecologists', 'Doctors'], freq=129)
FreqItemset(items=['Obstetricians & Gynecologists', 'Doctors', 'Health & Medical'], freq=129)
```

```
FreqItemset(items=['Obstetricians & Gynecologists', 'Health & Medical'], freq=129)
FreqItemset(items=['Music Venues', 'Bars'], freq=129)
FreqItemset(items=['Music Venues', 'Bars', 'Nightlife'], freq=129)
FreqItemset(items=['Music Venues', 'Arts & Entertainment', 'Bars'], freq=129)
FreqItemset(items=['Music Venues', 'Arts & Entertainment', 'Bars', 'Nightlife'], freq=129)
FreqItemset(items=['Active Life', 'Shopping'], freq=128)
FreqItemset(items=['Self Storage', 'Local Services'], freq=128)
FreqItemset(items=['Pet Training', 'Pets'], freq=128)
FreqItemset(items=['Pet Training', 'Pet Services'], freq=128)
FreqItemset(items=['Pet Training', 'Pet Services', 'Pets'], freq=128)
FreqItemset(items=['Flowers & Gifts', 'Event Planning & Services'], freq=128)
FreqItemset(items=['Flowers & Gifts', 'Event Planning & Services', 'Shopping'], freq=128)
FreqItemset(items=['Asian Fusion', 'Chinese'], freq=128)
FreqItemset(items=['Asian Fusion', 'Chinese', 'Restaurants'], freq=128)
FreqItemset(items=['Desserts', 'Ice Cream & Frozen Yogurt'], freq=127)
FreqItemset(items=['Desserts', 'Ice Cream & Frozen Yogurt', 'Food'], freq=127)
FreqItemset(items=['Auto Parts & Supplies', 'Auto Repair'], freq=127)
FreqItemset(items=['Auto Parts & Supplies', 'Auto Repair', 'Automotive'], freq=127)
FreqItemset(items=['Vape Shops', 'Shopping'], freq=126)
FreqItemset(items=['Vegan', 'Restaurants'], freq=126)
FreqItemset(items=['Cocktail Bars', 'Bars'], freq=126)
FreqItemset(items=['Cocktail Bars', 'Bars', 'Nightlife'], freq=126)
FreqItemset(items=['Cocktail Bars', 'Nightlife'], freq=126)
FreqItemset(items=['Burgers', 'Bars'], freq=125)
FreqItemset(items=['Burgers', 'Bars', 'Nightlife'], freq=125)
FreqItemset(items=['Burgers', 'Bars', 'Nightlife', 'Restaurants'], freq=125)
FreqItemset(items=['Burgers', 'Bars', 'Restaurants'], freq=125)
FreqItemset(items=['Desserts', 'Food', 'Restaurants'], freq=124)
FreqItemset(items=['Desserts', 'Restaurants'], freq=124)
FreqItemset(items=['Churches', 'Religious Organizations'], freq=123)
FreqItemset(items=['Hawaiian', 'Restaurants'], freq=122)
FreqItemset(items=["Children's Clothing", 'Shopping'], freq=122)
FreqItemset(items=["Children's Clothing", 'Fashion'], freq=122)
FreqItemset(items=["Children's Clothing", 'Fashion', 'Shopping'], freq=122)
FreqItemset(items=['Car Dealers', 'Auto Repair'], freq=121)
FreqItemset(items=['Car Dealers', 'Auto Repair', 'Automotive'], freq=121)
FreqItemset(items=['Salad', 'Sandwiches'], freq=121)
FreqItemset(items=['Salad', 'Sandwiches', 'Restaurants'], freq=121)
FreqItemset(items=['Notaries', 'Shipping Centers'], freq=120)
FreqItemset(items=['Notaries', 'Shipping Centers', 'Local Services'], freq=120)
FreqItemset(items=['German', 'Restaurants'], freq=120)
FreqItemset(items=["Men's Clothing", 'Accessories'], freq=120)
FreqItemset(items=["Men's Clothing", 'Accessories', 'Shopping'], freq=120)
FreqItemset(items=["Men's Clothing", 'Accessories', 'Fashion'], freq=120)
FreqItemset(items=["Men's Clothing", 'Accessories', 'Fashion', 'Shopping'], freq=120)
FreqItemset(items=['Kitchen & Bath', 'Shopping'], freq=119)
FreqItemset(items=['Kitchen & Bath', 'Home & Garden'], freq=119)
FreqItemset(items=['Kitchen & Bath', 'Home & Garden', 'Shopping'], freq=119)
FreqItemset(items=['Physical Therapy', 'Health & Medical'], freq=119)
FreqItemset(items=['Steakhouses', 'American (Traditional)'], freq=119)
FreqItemset(items=['Steakhouses', 'American (Traditional)', 'Restaurants'], freq=119)
```

```
FreqItemset(items=['Seafood', 'Steakhouses'], freq=118)
FreqItemset(items=['Seafood', 'Steakhouses', 'Restaurants'], freq=118)
FreqItemset(items=['Pest Control', 'Local Services'], freq=117)
FreqItemset(items=['Office Equipment', 'Shopping'], freq=116)
FreqItemset(items=['Hair Extensions', 'Hair Salons'], freq=116)
FreqItemset(items=['Hair Extensions', 'Hair Salons', 'Beauty & Spas'], freq=116)
FreqItemset(items=['Hair Extensions', 'Beauty & Spas'], freq=116)
FreqItemset(items=['Arts & Entertainment', 'Nightlife', 'Restaurants'], freq=115)
FreqItemset(items=['Bagels', 'Food', 'Restaurants'], freq=114)
FreqItemset(items=['Bagels', 'Restaurants'], freq=114)
FreqItemset(items=['Arts & Crafts', 'Flowers & Gifts'], freq=114)
FreqItemset(items=['Arts & Crafts', 'Flowers & Gifts', 'Shopping'], freq=114)
FreqItemset(items=['Music & DVDs', 'Books, Mags, Music & Video'], freq=114)
FreqItemset(items=['Music & DVDs', 'Books, Mags, Music & Video', 'Shopping'], freq=114)
FreqItemset(items=['Music & DVDs', 'Shopping'], freq=114)
FreqItemset(items=['Orthodontists', 'Dentists'], freq=113)
FreqItemset(items=['Orthodontists', 'Dentists', 'Health & Medical'], freq=113)
FreqItemset(items=['Orthodontists', 'Health & Medical'], freq=113)
FreqItemset(items=['Notaries', 'Printing Services'], freq=113)
FreqItemset(items=['Notaries', 'Printing Services', 'Local Services'], freq=113)
FreqItemset(items=['American (Traditional)', 'Fast Food'], freq=113)
FreqItemset(items=['American (Traditional)', 'Fast Food', 'Restaurants'], freq=113)
FreqItemset(items=['Venues & Event Spaces', 'Hotels & Travel'], freq=113)
FreqItemset(items=['Venues & Event Spaces', 'Hotels & Travel', 'Event Planning & Services'], freq=113)
FreqItemset(items=['Hair Removal', 'Nail Salons'], freq=113)
FreqItemset(items=['Hair Removal', 'Nail Salons', 'Beauty & Spas'], freq=113)
FreqItemset(items=['Notaries', 'Shipping Centers', 'Printing Services'], freq=112)
FreqItemset(items=['Notaries', 'Shipping Centers', 'Printing Services', 'Local Services'], freq=112)
FreqItemset(items=['Internet Service Providers', 'Home Services'], freq=112)
FreqItemset(items=['Internet Service Providers', 'Professional Services'], freq=112)
FreqItemset(items=['Internet Service Providers', 'Professional Services', 'Home Services'], freq=112)
FreqItemset(items=['Appliances & Repair', 'Local Services'], freq=111)
FreqItemset(items=['Caterers', 'Event Planning & Services', 'Restaurants'], freq=111)
FreqItemset(items=['Caterers', 'Restaurants'], freq=111)
FreqItemset(items=['Real Estate Agents', 'Home Services'], freq=110)
FreqItemset(items=['Real Estate Agents', 'Real Estate'], freq=110)
FreqItemset(items=['Real Estate Agents', 'Real Estate', 'Home Services'], freq=110)
FreqItemset(items=['Printing Services', 'Professional Services'], freq=110)
FreqItemset(items=['Printing Services', 'Professional Services', 'Local Services'], freq=110)
FreqItemset(items=['Cards & Stationery', 'Arts & Crafts'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Arts & Crafts', 'Event Planning & Services'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Arts & Crafts', 'Event Planning & Services', 'Shopping'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Arts & Crafts', 'Flowers & Gifts'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Arts & Crafts', 'Flowers & Gifts', 'Event Planning & Services'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Arts & Crafts', 'Flowers & Gifts', 'Event Planning & Services', 'Shopping'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Arts & Crafts', 'Flowers & Gifts', 'Shopping'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Arts & Crafts', 'Shopping'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Event Planning & Services'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Event Planning & Services', 'Shopping'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Flowers & Gifts'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Flowers & Gifts', 'Event Planning & Services'], freq=109)
```

```
FreqItemset(items=['Cards & Stationery', 'Flowers & Gifts', 'Event Planning & Services', 'Shopping'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Flowers & Gifts', 'Shopping'], freq=109)
FreqItemset(items=['Cards & Stationery', 'Shopping'], freq=109)
FreqItemset(items=["Men's Clothing", 'Accessories', "Women's Clothing"], freq=109)
FreqItemset(items=["Men's Clothing", 'Accessories', "Women's Clothing", 'Shopping'], freq=109)
FreqItemset(items=["Men's Clothing", 'Accessories', "Women's Clothing", 'Fashion'], freq=109)
FreqItemset(items=["Men's Clothing", 'Accessories', "Women's Clothing", 'Fashion', 'Shopping'], freq=109)
FreqItemset(items=['Arts & Crafts', 'Flowers & Gifts', 'Event Planning & Services'], freq=109)
FreqItemset(items=['Arts & Crafts', 'Flowers & Gifts', 'Event Planning & Services', 'Shopping'], freq=109)
FreqItemset(items=['Skin Care', 'Day Spas'], freq=108)
FreqItemset(items=['Skin Care', 'Day Spas', 'Beauty & Spas'], freq=108)
FreqItemset(items=['Mortgage Brokers', 'Home Services'], freq=108)
FreqItemset(items=['Mortgage Brokers', 'Real Estate'], freq=108)
FreqItemset(items=['Mortgage Brokers', 'Real Estate', 'Home Services'], freq=108)
FreqItemset(items=['Skin Care', 'Massage'], freq=107)
FreqItemset(items=['Skin Care', 'Massage', 'Beauty & Spas'], freq=107)
FreqItemset(items=['Swimming Pools', 'Active Life'], freq=107)
FreqItemset(items=['Insurance', 'Financial Services'], freq=107)
FreqItemset(items=['Community Service/Non-Profit', 'Local Services'], freq=107)
FreqItemset(items=['Post Offices', 'Public Services & Government'], freq=107)
FreqItemset(items=['Fashion', 'Food'], freq=107)
FreqItemset(items=['Fashion', 'Food', 'Shopping'], freq=107)
FreqItemset(items=['Wedding Planning', 'Event Planning & Services'], freq=106)
FreqItemset(items=['Food Delivery Services', 'Food'], freq=106)
FreqItemset(items=['Buffets', 'Chinese'], freq=106)
FreqItemset(items=['Buffets', 'Chinese', 'Restaurants'], freq=106)
FreqItemset(items=['Arts & Entertainment', 'Hotels & Travel'], freq=106)
FreqItemset(items=['Property Management', 'Home Services'], freq=104)
FreqItemset(items=['Property Management', 'Real Estate'], freq=104)
FreqItemset(items=['Property Management', 'Real Estate', 'Home Services'], freq=104)
FreqItemset(items=['Meat Shops', 'Specialty Food'], freq=104)
FreqItemset(items=['Meat Shops', 'Specialty Food', 'Food'], freq=104)
FreqItemset(items=['Meat Shops', 'Food'], freq=104)
FreqItemset(items=['Adult Entertainment', 'Nightlife'], freq=104)
FreqItemset(items=['Venues & Event Spaces', 'Hotels'], freq=104)
FreqItemset(items=['Venues & Event Spaces', 'Hotels', 'Event Planning & Services'], freq=104)
FreqItemset(items=['Venues & Event Spaces', 'Hotels', 'Hotels & Travel'], freq=104)
FreqItemset(items=['Venues & Event Spaces', 'Hotels', 'Hotels & Travel', 'Event Planning & Services'], freq=104)
FreqItemset(items=['Massage', 'Health & Medical'], freq=103)
FreqItemset(items=['Massage', 'Health & Medical', 'Beauty & Spas'], freq=103)
FreqItemset(items=['Chicken Wings', 'Sandwiches'], freq=103)
FreqItemset(items=['Chicken Wings', 'Sandwiches', 'Restaurants'], freq=103)
FreqItemset(items=['Breakfast & Brunch', 'Coffee & Tea'], freq=103)
FreqItemset(items=['Breakfast & Brunch', 'Coffee & Tea', 'Food'], freq=103)
FreqItemset(items=['Breakfast & Brunch', 'Coffee & Tea', 'Food', 'Restaurants'], freq=103)
FreqItemset(items=['Breakfast & Brunch', 'Coffee & Tea', 'Restaurants'], freq=103)
FreqItemset(items=['Candy Stores', 'Specialty Food'], freq=103)
FreqItemset(items=['Candy Stores', 'Specialty Food', 'Food'], freq=103)
FreqItemset(items=['Candy Stores', 'Food'], freq=103)
FreqItemset(items=['Department Stores', 'Food'], freq=102)
FreqItemset(items=['Department Stores', 'Food', 'Shopping'], freq=102)
```

```
FreqItemset(items=['Department Stores', 'Fashion', 'Food'], freq=102)
FreqItemset(items=['Department Stores', 'Fashion', 'Food', 'Shopping'], freq=102)
FreqItemset(items=['Bakeries', 'Coffee & Tea'], freq=102)
FreqItemset(items=['Bakeries', 'Coffee & Tea', 'Food'], freq=102)
FreqItemset(items=['Sporting Goods', 'Active Life'], freq=102)
FreqItemset(items=['Sporting Goods', 'Active Life', 'Shopping'], freq=102)
FreqItemset(items=['Arts & Entertainment', 'Active Life'], freq=102)
FreqItemset(items=['Juice Bars & Smoothies', 'Food', 'Restaurants'], freq=101)
FreqItemset(items=['Juice Bars & Smoothies', 'Restaurants'], freq=101)
FreqItemset(items=['Department Stores', 'Grocery'], freq=101)
FreqItemset(items=['Department Stores', 'Grocery', 'Food'], freq=101)
FreqItemset(items=['Department Stores', 'Grocery', 'Food', 'Shopping'], freq=101)
FreqItemset(items=['Department Stores', 'Grocery', 'Shopping'], freq=101)
FreqItemset(items=['Department Stores', 'Grocery', 'Fashion'], freq=101)
FreqItemset(items=['Department Stores', 'Grocery', 'Fashion', 'Food'], freq=101)
FreqItemset(items=['Department Stores', 'Grocery', 'Fashion', 'Food', 'Shopping'], freq=101)
FreqItemset(items=['Department Stores', 'Grocery', 'Fashion', 'Shopping'], freq=101)
FreqItemset(items=['Fast Food', 'Food'], freq=101)
FreqItemset(items=['Fast Food', 'Food', 'Restaurants'], freq=101)
FreqItemset(items=['Lawyers', 'Professional Services'], freq=101)
FreqItemset(items=['Libraries', 'Public Services & Government'], freq=101)
FreqItemset(items=['Grocery', 'Fashion'], freq=101)
FreqItemset(items=['Grocery', 'Fashion', 'Food'], freq=101)
FreqItemset(items=['Grocery', 'Fashion', 'Food', 'Shopping'], freq=101)
FreqItemset(items=['Grocery', 'Fashion', 'Shopping'], freq=101)
FreqItemset(items=['Day Spas', 'Nail Salons'], freq=100)
FreqItemset(items=['Day Spas', 'Nail Salons', 'Beauty & Spas'], freq=100)
FreqItemset(items=['Breakfast & Brunch', 'American (New)'], freq=100)
FreqItemset(items=['Breakfast & Brunch', 'American (New)', 'Restaurants'], freq=100)
FreqItemset(items=['Hotels', 'Arts & Entertainment'], freq=100)
FreqItemset(items=['Hotels', 'Arts & Entertainment', 'Event Planning & Services'], freq=100)
FreqItemset(items=['Hotels', 'Arts & Entertainment', 'Hotels & Travel'], freq=100)
FreqItemset(items=['Hotels', 'Arts & Entertainment', 'Hotels & Travel', 'Event Planning & Services'], freq=100)
FreqItemset(items=['Karaoke', 'Nightlife'], freq=100)
FreqItemset(items=['Arts & Entertainment', 'Hotels & Travel', 'Event Planning & Services'], freq=100)
FreqItemset(items=['Computers', 'Shopping'], freq=99)
FreqItemset(items=['American (Traditional)', 'Food'], freq=99)
FreqItemset(items=['American (Traditional)', 'Food', 'Restaurants'], freq=99)
FreqItemset(items=['Casinos', 'Event Planning & Services'], freq=99)
FreqItemset(items=['Casinos', 'Arts & Entertainment', 'Event Planning & Services'], freq=99)
FreqItemset(items=['Fish & Chips', 'Restaurants'], freq=99)
FreqItemset(items=['Amusement Parks', 'Active Life'], freq=98)
FreqItemset(items=['Colleges & Universities', 'Education'], freq=98)
FreqItemset(items=['Hospitals', 'Health & Medical'], freq=98)
FreqItemset(items=['Martial Arts', 'Fitness & Instruction'], freq=98)
FreqItemset(items=['Martial Arts', 'Fitness & Instruction', 'Active Life'], freq=98)
FreqItemset(items=['Martial Arts', 'Active Life'], freq=98)
FreqItemset(items=['Home Decor', 'Furniture Stores'], freq=98)
FreqItemset(items=['Home Decor', 'Furniture Stores', 'Shopping'], freq=98)
FreqItemset(items=['Home Decor', 'Furniture Stores', 'Home & Garden'], freq=98)
FreqItemset(items=['Home Decor', 'Furniture Stores', 'Home & Garden', 'Shopping'], freq=98)
```

```
FreqItemset(items=['Casinos', 'Hotels & Travel'], freq=98)
FreqItemset(items=['Casinos', 'Arts & Entertainment', 'Hotels & Travel'], freq=98)
FreqItemset(items=['Videos & Video Game Rental', 'Books, Mags, Music & Video'], freq=97)
FreqItemset(items=['Videos & Video Game Rental', 'Books, Mags, Music & Video', 'Shopping'], freq=97)
FreqItemset(items=['Videos & Video Game Rental', 'Shopping'], freq=97)
FreqItemset(items=['Casinos', 'Hotels'], freq=97)
FreqItemset(items=['Casinos', 'Hotels', 'Event Planning & Services'], freq=97)
FreqItemset(items=['Casinos', 'Hotels', 'Hotels & Travel'], freq=97)
FreqItemset(items=['Casinos', 'Hotels', 'Hotels & Travel', 'Event Planning & Services'], freq=97)
FreqItemset(items=['Casinos', 'Hotels', 'Arts & Entertainment'], freq=97)
FreqItemset(items=['Casinos', 'Hotels', 'Arts & Entertainment', 'Event Planning & Services'], freq=97)
FreqItemset(items=['Casinos', 'Hotels', 'Arts & Entertainment', 'Hotels & Travel'], freq=97)
FreqItemset(items=['Casinos', 'Hotels', 'Arts & Entertainment', 'Hotels & Travel', 'Event Planning & Services'], freq=97)
FreqItemset(items=['Casinos', 'Hotels & Travel', 'Event Planning & Services'], freq=97)
FreqItemset(items=['Casinos', 'Arts & Entertainment', 'Hotels & Travel', 'Event Planning & Services'], freq=97)
FreqItemset(items=['Electronics', 'Mobile Phones'], freq=96)
FreqItemset(items=['Electronics', 'Mobile Phones', 'Shopping'], freq=96)
FreqItemset(items=['Bridal', 'Shopping'], freq=96)
FreqItemset(items=['Preschools', 'Education'], freq=96)
FreqItemset(items=['Baby Gear & Furniture', 'Shopping'], freq=95)
FreqItemset(items=['Event Planning & Services', 'Active Life'], freq=95)
FreqItemset(items=['Building Supplies', 'Home Services'], freq=95)
FreqItemset(items=['Delis', 'Fast Food'], freq=95)
FreqItemset(items=['Delis', 'Fast Food', 'Restaurants'], freq=95)
FreqItemset(items=['Jewelry', 'Fashion'], freq=95)
FreqItemset(items=['Jewelry', 'Fashion', 'Shopping'], freq=95)
FreqItemset(items=['Outdoor Gear', 'Sporting Goods'], freq=95)
FreqItemset(items=['Outdoor Gear', 'Sporting Goods', 'Shopping'], freq=95)
FreqItemset(items=['Outdoor Gear', 'Shopping'], freq=95)
FreqItemset(items=['Chicken Wings', 'American (Traditional)'], freq=94)
FreqItemset(items=['Chicken Wings', 'American (Traditional)', 'Restaurants'], freq=94)
FreqItemset(items=['Pediatricians', 'Doctors'], freq=94)
FreqItemset(items=['Pediatricians', 'Doctors', 'Health & Medical'], freq=94)
FreqItemset(items=['Pediatricians', 'Health & Medical'], freq=94)
FreqItemset(items=['American (New)', 'Food'], freq=94)
FreqItemset(items=['American (New)', 'Food', 'Restaurants'], freq=94)
FreqItemset(items=['Pet Boarding/Pet Sitting', 'Pet Groomers'], freq=94)
FreqItemset(items=['Pet Boarding/Pet Sitting', 'Pet Groomers', 'Pets'], freq=94)
FreqItemset(items=['Pet Boarding/Pet Sitting', 'Pet Groomers', 'Pet Services'], freq=94)
FreqItemset(items=['Pet Boarding/Pet Sitting', 'Pet Groomers', 'Pet Services', 'Pets'], freq=94)
FreqItemset(items=['Ethnic Food', 'Specialty Food', 'Food', 'Restaurants'], freq=94)
FreqItemset(items=['Ethnic Food', 'Specialty Food', 'Restaurants'], freq=94)
FreqItemset(items=['Ethnic Food', 'Food', 'Restaurants'], freq=94)
FreqItemset(items=['Ethnic Food', 'Restaurants'], freq=94)
FreqItemset(items=['Tapas Bars', 'Restaurants'], freq=93)
FreqItemset(items=['Landmarks & Historical Buildings', 'Public Services & Government'], freq=93)
FreqItemset(items=['Child Care & Day Care', 'Local Services'], freq=93)
FreqItemset(items=["Men's Hair Salons", 'Hair Salons'], freq=93)
FreqItemset(items=["Men's Hair Salons", 'Hair Salons', 'Beauty & Spas'], freq=93)
FreqItemset(items=["Men's Hair Salons", 'Beauty & Spas'], freq=93)
FreqItemset(items=['Gay Bars', 'Bars'], freq=93)
```

```
FreqItemset(items=['Gay Bars', 'Bars', 'Nightlife'], freq=93)
FreqItemset(items=['Gay Bars', 'Nightlife'], freq=93)
FreqItemset(items=['Pool Cleaners', 'Home Services'], freq=92)
FreqItemset(items=['Delis', 'Sandwiches', 'Fast Food'], freq=92)
FreqItemset(items=['Delis', 'Sandwiches', 'Fast Food', 'Restaurants'], freq=92)
FreqItemset(items=['Musical Instruments & Teachers', 'Shopping'], freq=92)
FreqItemset(items=['Gift Shops', 'Flowers & Gifts'], freq=92)
FreqItemset(items=['Gift Shops', 'Flowers & Gifts', 'Shopping'], freq=92)
FreqItemset(items=['Gift Shops', 'Shopping'], freq=92)
FreqItemset(items=['Dermatologists', 'Doctors'], freq=91)
FreqItemset(items=['Dermatologists', 'Doctors', 'Health & Medical'], freq=91)
FreqItemset(items=['Dermatologists', 'Health & Medical'], freq=91)
FreqItemset(items=['Mattresses', 'Shopping'], freq=91)
FreqItemset(items=['Mattresses', 'Home & Garden'], freq=91)
FreqItemset(items=['Mattresses', 'Home & Garden', 'Shopping'], freq=91)
FreqItemset(items=['Italian', 'Nightlife'], freq=90)
FreqItemset(items=['Italian', 'Nightlife', 'Restaurants'], freq=90)
FreqItemset(items=['Soup', 'Restaurants'], freq=90)
FreqItemset(items=['Auto Detailing', 'Car Wash'], freq=90)
FreqItemset(items=['Auto Detailing', 'Car Wash', 'Automotive'], freq=90)
FreqItemset(items=['Pubs', 'American (Traditional)'], freq=89)
FreqItemset(items=['Pubs', 'American (Traditional)', 'Bars'], freq=89)
FreqItemset(items=['Pubs', 'American (Traditional)', 'Bars', 'Nightlife'], freq=89)
FreqItemset(items=['Pubs', 'American (Traditional)', 'Bars', 'Nightlife', 'Restaurants'], freq=89)
FreqItemset(items=['Pubs', 'American (Traditional)', 'Bars', 'Restaurants'], freq=89)
FreqItemset(items=['Pubs', 'American (Traditional)', 'Nightlife'], freq=89)
FreqItemset(items=['Pubs', 'American (Traditional)', 'Nightlife', 'Restaurants'], freq=89)
FreqItemset(items=['Pubs', 'American (Traditional)', 'Restaurants'], freq=89)
FreqItemset(items=['Chicken Wings', 'Bars'], freq=89)
FreqItemset(items=['Chicken Wings', 'Bars', 'Nightlife'], freq=89)
FreqItemset(items=['Chicken Wings', 'Bars', 'Nightlife', 'Restaurants'], freq=89)
FreqItemset(items=['Chicken Wings', 'Bars', 'Restaurants'], freq=89)
FreqItemset(items=['Chicken Wings', 'Nightlife'], freq=89)
FreqItemset(items=['Chicken Wings', 'Nightlife', 'Restaurants'], freq=89)
FreqItemset(items=['Pediatric Dentists', 'Dentists'], freq=89)
FreqItemset(items=['Pediatric Dentists', 'Dentists', 'Health & Medical'], freq=89)
FreqItemset(items=['Pediatric Dentists', 'Health & Medical'], freq=89)
FreqItemset(items=['Pet Training', 'Pet Groomers'], freq=89)
FreqItemset(items=['Pet Training', 'Pet Groomers', 'Pets'], freq=89)
FreqItemset(items=['Pet Training', 'Pet Groomers', 'Pet Services'], freq=89)
FreqItemset(items=['Pet Training', 'Pet Groomers', 'Pet Services', 'Pets'], freq=89)
FreqItemset(items=['Event Planning & Services', 'Food'], freq=88)
FreqItemset(items=['Burgers', 'Sandwiches'], freq=88)
FreqItemset(items=['Burgers', 'Sandwiches', 'Restaurants'], freq=88)
FreqItemset(items=['Pakistani', 'Restaurants'], freq=88)
FreqItemset(items=['Chicken Wings', 'Sandwiches', 'Pizza'], freq=87)
FreqItemset(items=['Chicken Wings', 'Sandwiches', 'Pizza', 'Restaurants'], freq=87)
FreqItemset(items=['Pizza', 'Nightlife'], freq=87)
FreqItemset(items=['Pizza', 'Nightlife', 'Restaurants'], freq=87)
FreqItemset(items=['Ophthalmologists', 'Doctors'], freq=87)
FreqItemset(items=['Ophthalmologists', 'Doctors', 'Health & Medical'], freq=87)
```

```
FreqItemset(items=['Ophthalmologists', 'Health & Medical'], freq=87)
FreqItemset(items=['Italian', 'Bars'], freq=86)
FreqItemset(items=['Italian', 'Bars', 'Nightlife'], freq=86)
FreqItemset(items=['Italian', 'Bars', 'Nightlife', 'Restaurants'], freq=86)
FreqItemset(items=['Italian', 'Bars', 'Restaurants'], freq=86)
FreqItemset(items=['Pizza', 'Bars'], freq=86)
FreqItemset(items=['Pizza', 'Bars', 'Nightlife'], freq=86)
FreqItemset(items=['Pizza', 'Bars', 'Nightlife', 'Restaurants'], freq=86)
FreqItemset(items=['Pizza', 'Bars', 'Restaurants'], freq=86)
FreqItemset(items=['Tapas/Small Plates', 'Restaurants'], freq=86)
FreqItemset(items=['Auto Parts & Supplies', 'Tires'], freq=86)
FreqItemset(items=['Auto Parts & Supplies', 'Tires', 'Automotive'], freq=86)
FreqItemset(items=['Street Vendors', 'Food'], freq=86)
FreqItemset(items=['Mexican', 'Nightlife'], freq=85)
FreqItemset(items=['Mexican', 'Nightlife', 'Restaurants'], freq=85)
FreqItemset(items=['Art Supplies', 'Arts & Crafts'], freq=85)
FreqItemset(items=['Art Supplies', 'Arts & Crafts', 'Shopping'], freq=85)
FreqItemset(items=['Art Supplies', 'Shopping'], freq=85)
FreqItemset(items=['Pubs', 'American (New)'], freq=84)
FreqItemset(items=['Pubs', 'American (New)', 'Bars'], freq=84)
FreqItemset(items=['Pubs', 'American (New)', 'Bars', 'Nightlife'], freq=84)
FreqItemset(items=['Pubs', 'American (New)', 'Bars', 'Nightlife', 'Restaurants'], freq=84)
FreqItemset(items=['Pubs', 'American (New)', 'Bars', 'Restaurants'], freq=84)
FreqItemset(items=['Pubs', 'American (New)', 'Nightlife'], freq=84)
FreqItemset(items=['Pubs', 'American (New)', 'Nightlife', 'Restaurants'], freq=84)
FreqItemset(items=['Pubs', 'American (New)', 'Restaurants'], freq=84)
FreqItemset(items=['Lingerie', 'Shopping'], freq=84)
FreqItemset(items=['Lingerie', 'Fashion'], freq=84)
FreqItemset(items=['Lingerie', 'Fashion', 'Shopping'], freq=84)
FreqItemset(items=['Waxing', 'Hair Removal'], freq=84)
FreqItemset(items=['Waxing', 'Hair Removal', 'Beauty & Spas'], freq=84)
FreqItemset(items=['Waxing', 'Beauty & Spas'], freq=84)
FreqItemset(items=['Sports Bars', 'Pubs'], freq=84)
FreqItemset(items=['Sports Bars', 'Pubs', 'Bars'], freq=84)
FreqItemset(items=['Sports Bars', 'Pubs', 'Bars', 'Nightlife'], freq=84)
FreqItemset(items=['Sports Bars', 'Pubs', 'Nightlife'], freq=84)
FreqItemset(items=['Arts & Entertainment', 'Food'], freq=84)
FreqItemset(items=['Graphic Design', 'Professional Services'], freq=83)
FreqItemset(items=['Chicken Wings', 'Italian'], freq=82)
FreqItemset(items=['Chicken Wings', 'Italian', 'Pizza'], freq=82)
FreqItemset(items=['Chicken Wings', 'Italian', 'Pizza', 'Restaurants'], freq=82)
FreqItemset(items=['Chicken Wings', 'Italian', 'Restaurants'], freq=82)
FreqItemset(items=['Diners', 'Breakfast & Brunch'], freq=82)
FreqItemset(items=['Diners', 'Breakfast & Brunch', 'Restaurants'], freq=82)
FreqItemset(items=['Convenience Stores', 'Grocery'], freq=81)
FreqItemset(items=['Convenience Stores', 'Grocery', 'Food'], freq=81)
FreqItemset(items=['Desserts', 'Coffee & Tea'], freq=81)
FreqItemset(items=['Desserts', 'Coffee & Tea', 'Food'], freq=81)
FreqItemset(items=['Mexican', 'Bars'], freq=81)
FreqItemset(items=['Mexican', 'Bars', 'Nightlife'], freq=81)
FreqItemset(items=['Mexican', 'Bars', 'Nightlife', 'Restaurants'], freq=81)
```

```
FreqItemset(items=['Mexican', 'Bars', 'Restaurants'], freq=81)
FreqItemset(items=['Food Trucks', 'Food', 'Restaurants'], freq=81)
FreqItemset(items=['Food Trucks', 'Restaurants'], freq=81)
FreqItemset(items=['Sandwiches', 'American (Traditional)'], freq=81)
FreqItemset(items=['Sandwiches', 'American (Traditional)', 'Restaurants'], freq=81)
FreqItemset(items=['Veterinarians', 'Pet Services'], freq=80)
FreqItemset(items=['Veterinarians', 'Pet Services', 'Pets'], freq=80)
FreqItemset(items=['Steakhouses', 'Nightlife'], freq=80)
FreqItemset(items=['Steakhouses', 'Nightlife', 'Restaurants'], freq=80)
FreqItemset(items=['Pet Training', 'Pet Stores'], freq=80)
FreqItemset(items=['Pet Training', 'Pet Stores', 'Pets'], freq=80)
FreqItemset(items=['Pet Training', 'Pet Stores', 'Pet Services'], freq=80)
FreqItemset(items=['Pet Training', 'Pet Stores', 'Pet Services', 'Pets'], freq=80)
FreqItemset(items=['Medical Centers', 'Doctors'], freq=80)
FreqItemset(items=['Medical Centers', 'Doctors', 'Health & Medical'], freq=80)
FreqItemset(items=['Seafood', 'American (Traditional)'], freq=80)
FreqItemset(items=['Seafood', 'American (Traditional)', 'Restaurants'], freq=80)
FreqItemset(items=['Italian', 'Sandwiches'], freq=79)
FreqItemset(items=['Italian', 'Sandwiches', 'Restaurants'], freq=79)
FreqItemset(items=['Shoe Repair', 'Local Services'], freq=79)
FreqItemset(items=['Makeup Artists', 'Hair Salons'], freq=79)
FreqItemset(items=['Makeup Artists', 'Hair Salons', 'Beauty & Spas'], freq=79)
FreqItemset(items=['Nurseries & Gardening', 'Hardware Stores'], freq=78)
FreqItemset(items=['Nurseries & Gardening', 'Hardware Stores', 'Shopping'], freq=78)
FreqItemset(items=['Nurseries & Gardening', 'Hardware Stores', 'Home & Garden'], freq=78)
FreqItemset(items=['Nurseries & Gardening', 'Hardware Stores', 'Home & Garden', 'Shopping'], freq=78)
FreqItemset(items=['Cafes', 'Breakfast & Brunch'], freq=78)
FreqItemset(items=['Cafes', 'Breakfast & Brunch', 'Restaurants'], freq=78)
FreqItemset(items=['Optometrists', 'Doctors'], freq=78)
FreqItemset(items=['Optometrists', 'Doctors', 'Health & Medical'], freq=78)
FreqItemset(items=['Taxis', 'Transportation'], freq=78)
FreqItemset(items=['Taxis', 'Transportation', 'Hotels & Travel'], freq=78)
FreqItemset(items=['Taxis', 'Hotels & Travel'], freq=78)
FreqItemset(items=['Soul Food', 'Restaurants'], freq=78)
FreqItemset(items=['Pakistani', 'Indian'], freq=78)
FreqItemset(items=['Pakistani', 'Indian', 'Restaurants'], freq=78)
FreqItemset(items=['Caribbean', 'Restaurants'], freq=78)
FreqItemset(items=['Weight Loss Centers', 'Health & Medical'], freq=78)
FreqItemset(items=['Asian Fusion', 'Sushi Bars'], freq=78)
FreqItemset(items=['Asian Fusion', 'Sushi Bars', 'Restaurants'], freq=78)
FreqItemset(items=['Motorcycle Dealers', 'Automotive'], freq=77)
FreqItemset(items=['Home & Garden', 'Fashion'], freq=77)
FreqItemset(items=['Home & Garden', 'Fashion', 'Shopping'], freq=77)
FreqItemset(items=['Cafes', 'Bakeries'], freq=77)
FreqItemset(items=['Cafes', 'Bakeries', 'Food'], freq=77)
FreqItemset(items=['Cafes', 'Bakeries', 'Food', 'Restaurants'], freq=77)
FreqItemset(items=['Cafes', 'Bakeries', 'Restaurants'], freq=77)
FreqItemset(items=['Event Planning & Services', 'Nightlife'], freq=77)
FreqItemset(items=['Pet Training', 'Pet Stores', 'Pet Groomers'], freq=77)
FreqItemset(items=['Pet Training', 'Pet Stores', 'Pet Groomers', 'Pets'], freq=77)
FreqItemset(items=['Pet Training', 'Pet Stores', 'Pet Groomers', 'Pet Services'], freq=77)
```

```
FreqItemset(items=['Pet Training', 'Pet Stores', 'Pet Groomers', 'Pet Services', 'Pets'], freq=77)
FreqItemset(items=['Beer, Wine & Spirits', 'Food', 'Restaurants'], freq=77)
FreqItemset(items=['Beer, Wine & Spirits', 'Restaurants'], freq=77)
FreqItemset(items=['Music Venues', 'Nightlife', 'Restaurants'], freq=77)
FreqItemset(items=['Music Venues', 'Arts & Entertainment', 'Nightlife', 'Restaurants'], freq=77)
FreqItemset(items=['Music Venues', 'Arts & Entertainment', 'Restaurants'], freq=77)
FreqItemset(items=['Music Venues', 'Restaurants'], freq=77)
FreqItemset(items=['Irish', 'Restaurants'], freq=77)
FreqItemset(items=['Bowling', 'Active Life'], freq=77)
FreqItemset(items=['Wheel & Rim Repair', 'Automotive'], freq=76)
FreqItemset(items=['Chicken Wings', 'Sports Bars'], freq=76)
FreqItemset(items=['Chicken Wings', 'Sports Bars', 'Bars'], freq=76)
FreqItemset(items=['Chicken Wings', 'Sports Bars', 'Bars', 'Nightlife'], freq=76)
FreqItemset(items=['Chicken Wings', 'Sports Bars', 'Bars', 'Nightlife', 'Restaurants'], freq=76)
FreqItemset(items=['Chicken Wings', 'Sports Bars', 'Bars', 'Restaurants'], freq=76)
FreqItemset(items=['Chicken Wings', 'Sports Bars', 'Nightlife'], freq=76)
FreqItemset(items=['Chicken Wings', 'Sports Bars', 'Nightlife', 'Restaurants'], freq=76)
FreqItemset(items=['Chicken Wings', 'Sports Bars', 'Restaurants'], freq=76)
FreqItemset(items=['Hiking', 'Active Life'], freq=76)
FreqItemset(items=['Airport Shuttles', 'Transportation'], freq=76)
FreqItemset(items=['Airport Shuttles', 'Transportation', 'Hotels & Travel'], freq=76)
FreqItemset(items=['Airport Shuttles', 'Hotels & Travel'], freq=76)
FreqItemset(items=['Body Shops', 'Auto Repair'], freq=76)
FreqItemset(items=['Body Shops', 'Auto Repair', 'Automotive'], freq=76)
FreqItemset(items=['Food Delivery Services', 'Food', 'Restaurants'], freq=75)
FreqItemset(items=['Food Delivery Services', 'Restaurants'], freq=75)
FreqItemset(items=['Watches', 'Shopping'], freq=75)
FreqItemset(items=['Flooring', 'Home Services'], freq=75)
FreqItemset(items=['Steakhouses', 'Bars'], freq=75)
FreqItemset(items=['Steakhouses', 'Bars', 'Nightlife'], freq=75)
FreqItemset(items=['Steakhouses', 'Bars', 'Nightlife', 'Restaurants'], freq=75)
FreqItemset(items=['Steakhouses', 'Bars', 'Restaurants'], freq=75)
FreqItemset(items=['Ophthalmologists', 'Optometrists'], freq=75)
FreqItemset(items=['Ophthalmologists', 'Optometrists', 'Doctors'], freq=75)
FreqItemset(items=['Ophthalmologists', 'Optometrists', 'Doctors', 'Health & Medical'], freq=75)
FreqItemset(items=['Ophthalmologists', 'Optometrists', 'Health & Medical'], freq=75)
FreqItemset(items=['Department Stores', "Women's Clothing"], freq=74)
FreqItemset(items=['Department Stores', "Women's Clothing", 'Shopping'], freq=74)
FreqItemset(items=['Department Stores', "Women's Clothing", 'Fashion'], freq=74)
FreqItemset(items=['Department Stores', "Women's Clothing", 'Fashion', 'Shopping'], freq=74)
FreqItemset(items=['Nightlife', 'Food', 'Restaurants'], freq=74)
FreqItemset(items=['Food Stands', 'Restaurants'], freq=74)
FreqItemset(items=['Professional Services', 'Shopping'], freq=74)
FreqItemset(items=['Tea Rooms', 'Food'], freq=74)
FreqItemset(items=['Grocery', 'Food', 'Restaurants'], freq=74)
FreqItemset(items=['Grocery', 'Restaurants'], freq=74)
FreqItemset(items=['Stadiums & Arenas', 'Arts & Entertainment'], freq=73)
FreqItemset(items=['Bagels', 'Breakfast & Brunch'], freq=73)
FreqItemset(items=['Bagels', 'Breakfast & Brunch', 'Food'], freq=73)
FreqItemset(items=['Bagels', 'Breakfast & Brunch', 'Food', 'Restaurants'], freq=73)
FreqItemset(items=['Bagels', 'Breakfast & Brunch', 'Restaurants'], freq=73)
```

```
FreqItemset(items=['Financial Services', 'Home Services'], freq=73)
FreqItemset(items=['Financial Services', 'Real Estate'], freq=73)
FreqItemset(items=['Financial Services', 'Real Estate', 'Home Services'], freq=73)
FreqItemset(items=['Diagnostic Services', 'Health & Medical'], freq=73)
FreqItemset(items=['Hookah Bars', 'Bars'], freq=73)
FreqItemset(items=['Hookah Bars', 'Bars', 'Nightlife'], freq=73)
FreqItemset(items=['Hookah Bars', 'Nightlife'], freq=73)
FreqItemset(items=['Active Life', 'Health & Medical'], freq=72)
FreqItemset(items=['Wheel & Rim Repair', 'Tires'], freq=72)
FreqItemset(items=['Wheel & Rim Repair', 'Tires', 'Automotive'], freq=72)
FreqItemset(items=['Comfort Food', 'Restaurants'], freq=72)
FreqItemset(items=['Electronics', 'Home Services'], freq=72)
FreqItemset(items=['Electronics', 'Home Services', 'Shopping'], freq=72)
FreqItemset(items=['Playgrounds', 'Active Life'], freq=72)
FreqItemset(items=['Cajun/Creole', 'Restaurants'], freq=72)
FreqItemset(items=['Pizza', 'Food'], freq=72)
FreqItemset(items=['Pizza', 'Food', 'Restaurants'], freq=72)
FreqItemset(items=['Eyelash Service', 'Hair Removal'], freq=72)
FreqItemset(items=['Eyelash Service', 'Hair Removal', 'Beauty & Spas'], freq=72)
FreqItemset(items=['Hair Removal', 'Hair Salons'], freq=72)
FreqItemset(items=['Hair Removal', 'Hair Salons', 'Beauty & Spas'], freq=72)
FreqItemset(items=['Outlet Stores', 'Shopping'], freq=71)
FreqItemset(items=['Mortgage Brokers', 'Financial Services'], freq=71)
FreqItemset(items=['Mortgage Brokers', 'Financial Services', 'Home Services'], freq=71)
FreqItemset(items=['Mortgage Brokers', 'Financial Services', 'Real Estate'], freq=71)
FreqItemset(items=['Mortgage Brokers', 'Financial Services', 'Real Estate', 'Home Services'], freq=71)
FreqItemset(items=['IT Services & Computer Repair', 'Shopping'], freq=70)
FreqItemset(items=['IT Services & Computer Repair', 'Local Services', 'Shopping'], freq=70)
FreqItemset(items=['Dance Studios', 'Fitness & Instruction'], freq=70)
FreqItemset(items=['Dance Studios', 'Fitness & Instruction', 'Active Life'], freq=70)
FreqItemset(items=['Dance Studios', 'Active Life'], freq=70)
FreqItemset(items=['Keys & Locksmiths', 'Home Services'], freq=70)
FreqItemset(items=['Interior Design', 'Home Services'], freq=70)
FreqItemset(items=['Limos', 'Transportation'], freq=70)
FreqItemset(items=['Limos', 'Transportation', 'Hotels & Travel'], freq=70)
FreqItemset(items=['Limos', 'Hotels & Travel'], freq=70)
FreqItemset(items=['Hot Dogs', 'Burgers'], freq=70)
FreqItemset(items=['Hot Dogs', 'Burgers', 'Restaurants'], freq=70)
FreqItemset(items=['Wholesale Stores', 'Shopping'], freq=69)
FreqItemset(items=['Elementary Schools', 'Education'], freq=69)
FreqItemset(items=['Spanish', 'Restaurants'], freq=69)
FreqItemset(items=['Fitness & Instruction', 'Active Life', 'Health & Medical'], freq=69)
FreqItemset(items=['Fitness & Instruction', 'Health & Medical'], freq=69)
FreqItemset(items=['Middle Eastern', 'Mediterranean'], freq=69)
FreqItemset(items=['Middle Eastern', 'Mediterranean', 'Restaurants'], freq=69)
FreqItemset(items=['Flowers & Gifts', 'Food'], freq=69)
FreqItemset(items=['Flowers & Gifts', 'Food', 'Shopping'], freq=69)
FreqItemset(items=['Electricians', 'Home Services'], freq=69)
FreqItemset(items=['American (New)', 'American (Traditional)'], freq=68)
FreqItemset(items=['American (New)', 'American (Traditional)', 'Restaurants'], freq=68)
FreqItemset(items=['Guns & Ammo', 'Shopping'], freq=68)
```

```
FreqItemset(items=['Gastropubs', 'Nightlife'], freq=68)
FreqItemset(items=['Gastropubs', 'Nightlife', 'Restaurants'], freq=68)
FreqItemset(items=['Smog Check Stations', 'Automotive'], freq=68)
FreqItemset(items=['Performing Arts', 'Nightlife'], freq=68)
FreqItemset(items=['Performing Arts', 'Arts & Entertainment', 'Nightlife'], freq=68)
FreqItemset(items=['Karaoke', 'Bars'], freq=68)
FreqItemset(items=['Karaoke', 'Bars', 'Nightlife'], freq=68)
FreqItemset(items=['Carpet Cleaning', 'Home Services'], freq=67)
FreqItemset(items=['Carpet Cleaning', 'Local Services', 'Home Services'], freq=67)
FreqItemset(items=['Chicken Wings', 'Fast Food'], freq=67)
FreqItemset(items=['Chicken Wings', 'Fast Food', 'Restaurants'], freq=67)
FreqItemset(items=['Skin Care', 'Nail Salons'], freq=67)
FreqItemset(items=['Skin Care', 'Nail Salons', 'Beauty & Spas'], freq=67)
FreqItemset(items=['Sports Bars', 'Pubs', 'Bars', 'Nightlife', 'Restaurants'], freq=67)
FreqItemset(items=['Sports Bars', 'Pubs', 'Bars', 'Restaurants'], freq=67)
FreqItemset(items=['Sports Bars', 'Pubs', 'Nightlife', 'Restaurants'], freq=67)
FreqItemset(items=['Sports Bars', 'Pubs', 'Restaurants'], freq=67)
FreqItemset(items=['Pilates', 'Fitness & Instruction'], freq=67)
FreqItemset(items=['Pilates', 'Fitness & Instruction', 'Active Life'], freq=67)
FreqItemset(items=['Pilates', 'Active Life'], freq=67)
FreqItemset(items=['Fabric Stores', 'Arts & Crafts'], freq=67)
FreqItemset(items=['Fabric Stores', 'Arts & Crafts', 'Shopping'], freq=67)
FreqItemset(items=['Fabric Stores', 'Shopping'], freq=67)
FreqItemset(items=['Electronics Repair', 'Local Services'], freq=67)
FreqItemset(items=['Acupuncture', 'Health & Medical'], freq=67)
FreqItemset(items=['Oral Surgeons', 'Dentists'], freq=66)
FreqItemset(items=['Oral Surgeons', 'Dentists', 'Health & Medical'], freq=66)
FreqItemset(items=['Oral Surgeons', 'Health & Medical'], freq=66)
FreqItemset(items=['Skin Care', 'Hair Salons'], freq=66)
FreqItemset(items=['Skin Care', 'Hair Salons', 'Beauty & Spas'], freq=66)
FreqItemset(items=['Bars', 'Food', 'Restaurants'], freq=66)
FreqItemset(items=['Bars', 'Nightlife', 'Food', 'Restaurants'], freq=66)
FreqItemset(items=['Diners', 'American (Traditional)'], freq=66)
FreqItemset(items=['Diners', 'American (Traditional)', 'Restaurants'], freq=66)
FreqItemset(items=['Shaved Ice', 'Food'], freq=66)
FreqItemset(items=['Resorts', 'Hotels & Travel'], freq=65)
FreqItemset(items=['Towing', 'Automotive'], freq=65)
FreqItemset(items=['Soup', 'Sandwiches'], freq=65)
FreqItemset(items=['Soup', 'Sandwiches', 'Restaurants'], freq=65)
FreqItemset(items=['Doctors', 'Beauty & Spas'], freq=65)
FreqItemset(items=['Doctors', 'Health & Medical', 'Beauty & Spas'], freq=65)
FreqItemset(items=['Sports Bars', 'Burgers'], freq=65)
FreqItemset(items=['Sports Bars', 'Burgers', 'Bars'], freq=65)
FreqItemset(items=['Sports Bars', 'Burgers', 'Bars', 'Nightlife'], freq=65)
FreqItemset(items=['Sports Bars', 'Burgers', 'Bars', 'Nightlife', 'Restaurants'], freq=65)
FreqItemset(items=['Sports Bars', 'Burgers', 'Bars', 'Restaurants'], freq=65)
FreqItemset(items=['Sports Bars', 'Burgers', 'Nightlife'], freq=65)
FreqItemset(items=['Sports Bars', 'Burgers', 'Nightlife', 'Restaurants'], freq=65)
FreqItemset(items=['Sports Bars', 'Burgers', 'Restaurants'], freq=65)
FreqItemset(items=['Gastropubs', 'Bars'], freq=65)
FreqItemset(items=['Gastropubs', 'Bars', 'Nightlife'], freq=65)
```

```
FreqItemset(items=['Gastropubs', 'Bars', 'Nightlife', 'Restaurants'], freq=65)
FreqItemset(items=['Gastropubs', 'Bars', 'Restaurants'], freq=65)
FreqItemset(items=['Banks & Credit Unions', 'Home Services'], freq=65)
FreqItemset(items=['Banks & Credit Unions', 'Financial Services', 'Home Services'], freq=65)
FreqItemset(items=['Banks & Credit Unions', 'Financial Services', 'Real Estate'], freq=65)
FreqItemset(items=['Banks & Credit Unions', 'Financial Services', 'Real Estate', 'Home Services'], freq=65)
FreqItemset(items=['Banks & Credit Unions', 'Real Estate'], freq=65)
FreqItemset(items=['Banks & Credit Unions', 'Real Estate', 'Home Services'], freq=65)
FreqItemset(items=['Caterers', 'Event Planning & Services', 'Food'], freq=65)
FreqItemset(items=['Caterers', 'Food'], freq=65)
FreqItemset(items=['Mortgage Brokers', 'Banks & Credit Unions'], freq=65)
FreqItemset(items=['Mortgage Brokers', 'Banks & Credit Unions', 'Home Services'], freq=65)
FreqItemset(items=['Mortgage Brokers', 'Banks & Credit Unions', 'Financial Services'], freq=65)
FreqItemset(items=['Mortgage Brokers', 'Banks & Credit Unions', 'Financial Services', 'Home Services'], freq=65)
FreqItemset(items=['Mortgage Brokers', 'Banks & Credit Unions', 'Financial Services', 'Real Estate'], freq=65)
FreqItemset(items=['Mortgage Brokers', 'Banks & Credit Unions', 'Financial Services', 'Real Estate', 'Home Services'], freq=65)
FreqItemset(items=['Mortgage Brokers', 'Banks & Credit Unions', 'Real Estate'], freq=65)
FreqItemset(items=['Mortgage Brokers', 'Banks & Credit Unions', 'Real Estate', 'Home Services'], freq=65)
FreqItemset(items=['Beer, Wine & Spirits', 'Nightlife'], freq=65)
FreqItemset(items=['Beer, Wine & Spirits', 'Nightlife', 'Food'], freq=65)
FreqItemset(items=['Internet Service Providers', 'Home Services', 'Shopping'], freq=64)
FreqItemset(items=['Internet Service Providers', 'Professional Services', 'Home Services', 'Shopping'], freq=64)
FreqItemset(items=['Internet Service Providers', 'Professional Services', 'Shopping'], freq=64)
FreqItemset(items=['Internet Service Providers', 'Shopping'], freq=64)
FreqItemset(items=['Skin Care', 'Health & Medical'], freq=64)
FreqItemset(items=['Skin Care', 'Health & Medical', 'Beauty & Spas'], freq=64)
FreqItemset(items=['Laser Hair Removal', 'Hair Removal'], freq=64)
FreqItemset(items=['Laser Hair Removal', 'Hair Removal', 'Beauty & Spas'], freq=64)
FreqItemset(items=['Laser Hair Removal', 'Beauty & Spas'], freq=64)
FreqItemset(items=['Comic Books', 'Books, Mags, Music & Video'], freq=64)
FreqItemset(items=['Comic Books', 'Books, Mags, Music & Video', 'Shopping'], freq=64)
FreqItemset(items=['Comic Books', 'Shopping'], freq=64)
FreqItemset(items=['Hobby Shops', 'Arts & Crafts'], freq=64)
FreqItemset(items=['Hobby Shops', 'Arts & Crafts', 'Shopping'], freq=64)
FreqItemset(items=['Windshield Installation & Repair', 'Automotive'], freq=64)
FreqItemset(items=['Turkish', 'Restaurants'], freq=64)
FreqItemset(items=['Professional Services', 'Home Services', 'Shopping'], freq=64)
FreqItemset(items=['Irish', 'Bars'], freq=64)
FreqItemset(items=['Irish', 'Bars', 'Nightlife'], freq=64)
FreqItemset(items=['Irish', 'Bars', 'Nightlife', 'Restaurants'], freq=64)
FreqItemset(items=['Irish', 'Bars', 'Restaurants'], freq=64)
FreqItemset(items=['Irish', 'Nightlife'], freq=64)
FreqItemset(items=['Irish', 'Nightlife', 'Restaurants'], freq=64)
FreqItemset(items=['Breakfast & Brunch', 'Sandwiches', 'Food'], freq=63)
FreqItemset(items=['Breakfast & Brunch', 'Sandwiches', 'Food', 'Restaurants'], freq=63)
FreqItemset(items=['Brasseries', 'Restaurants'], freq=63)
FreqItemset(items=['Bike Rentals', 'Active Life'], freq=63)
FreqItemset(items=['Session Photography', 'Event Planning & Services'], freq=63)
FreqItemset(items=['Session Photography', 'Photographers'], freq=63)
FreqItemset(items=['Session Photography', 'Photographers', 'Event Planning & Services'], freq=63)
FreqItemset(items=['Arts & Entertainment', 'Bars', 'Nightlife', 'Restaurants'], freq=63)
```

```
FreqItemset(items=['Arts & Entertainment', 'Bars', 'Restaurants'], freq=63)
FreqItemset(items=['Signmaking', 'Professional Services'], freq=62)
FreqItemset(items=['Internet Service Providers', 'Electronics'], freq=62)
FreqItemset(items=['Internet Service Providers', 'Electronics', 'Home Services'], freq=62)
FreqItemset(items=['Internet Service Providers', 'Electronics', 'Home Services', 'Shopping'], freq=62)
FreqItemset(items=['Internet Service Providers', 'Electronics', 'Professional Services'], freq=62)
FreqItemset(items=['Internet Service Providers', 'Electronics', 'Professional Services', 'Home Services'], freq=62)
FreqItemset(items=['Internet Service Providers', 'Electronics', 'Professional Services', 'Home Services', 'Shopping'], freq=62)
FreqItemset(items=['Internet Service Providers', 'Electronics', 'Professional Services', 'Shopping'], freq=62)
FreqItemset(items=['Internet Service Providers', 'Electronics', 'Shopping'], freq=62)
FreqItemset(items=['Italian', 'Sandwiches', 'Pizza'], freq=62)
FreqItemset(items=['Italian', 'Sandwiches', 'Pizza', 'Restaurants'], freq=62)
FreqItemset(items=['Electronics', 'Professional Services'], freq=62)
FreqItemset(items=['Electronics', 'Professional Services', 'Home Services'], freq=62)
FreqItemset(items=['Electronics', 'Professional Services', 'Home Services', 'Shopping'], freq=62)
FreqItemset(items=['Electronics', 'Professional Services', 'Shopping'], freq=62)
FreqItemset(items=['Hair Salons', 'Shopping'], freq=62)
FreqItemset(items=['Hair Salons', 'Beauty & Spas', 'Shopping'], freq=62)
FreqItemset(items=['Asian Fusion', 'Japanese'], freq=62)
FreqItemset(items=['Asian Fusion', 'Japanese', 'Restaurants'], freq=62)
```

Question3: Are all the itemsets obtained by setting minimum support 0.01 included in the itemsets obtained when we set the minimum support to 0.001?

Answer: Yes all the items in support threshold 0.01 are included in 0.001 because the former is just a subset of the latter with the latter having additional related words due to lower support values

Part 7: Bonus Analysis (if any)

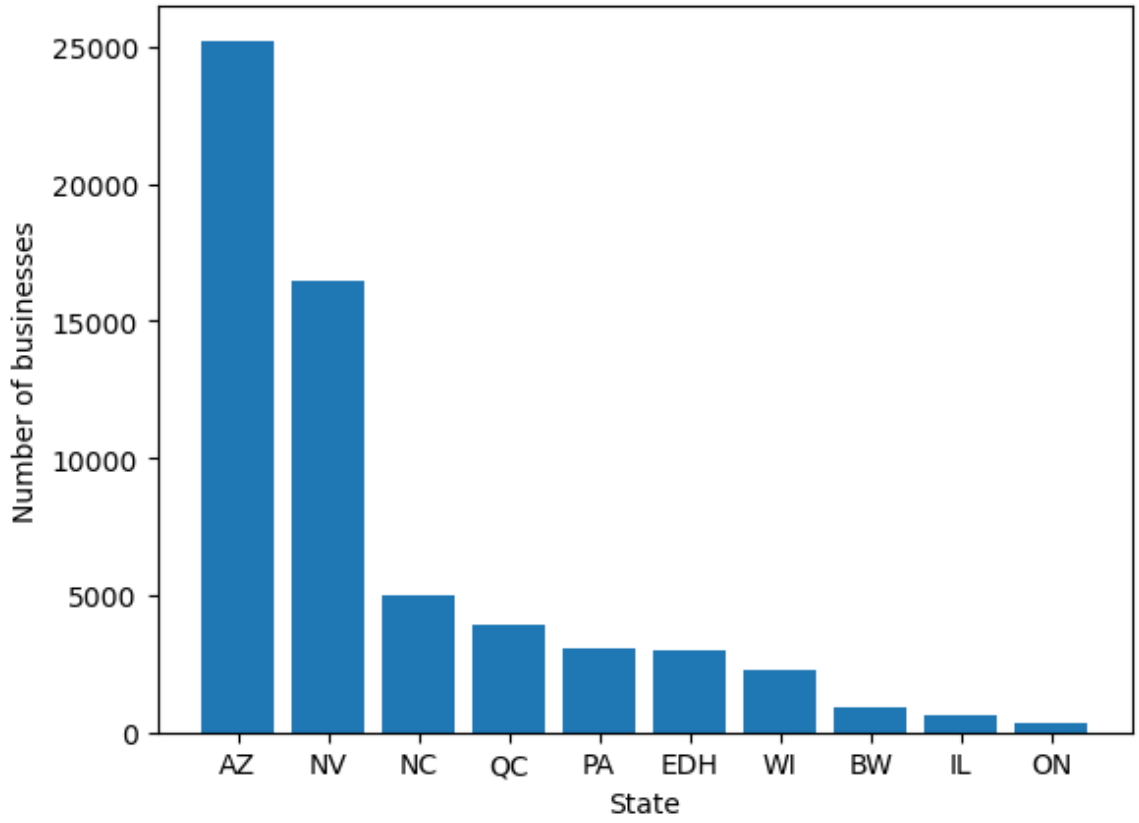
Here, you can include any additional and insightful exploratory data analysis or machine learning tasks you have carried out in addition to the guided exploration of the dataset above. Feel free to add code/markdown cells here to present your analysis.

In [64]:
```
# Count the number of users by year they joined Yelp
user_by_year = users_rdd.map(lambda x: (x['yelping_since'][:4], 1)) \
                        .reduceByKey(lambda x, y: x + y) \
                        .sortBy(lambda x: x[0])
print("Number of users by year they joined Yelp:")
print(user_by_year.collect())
```

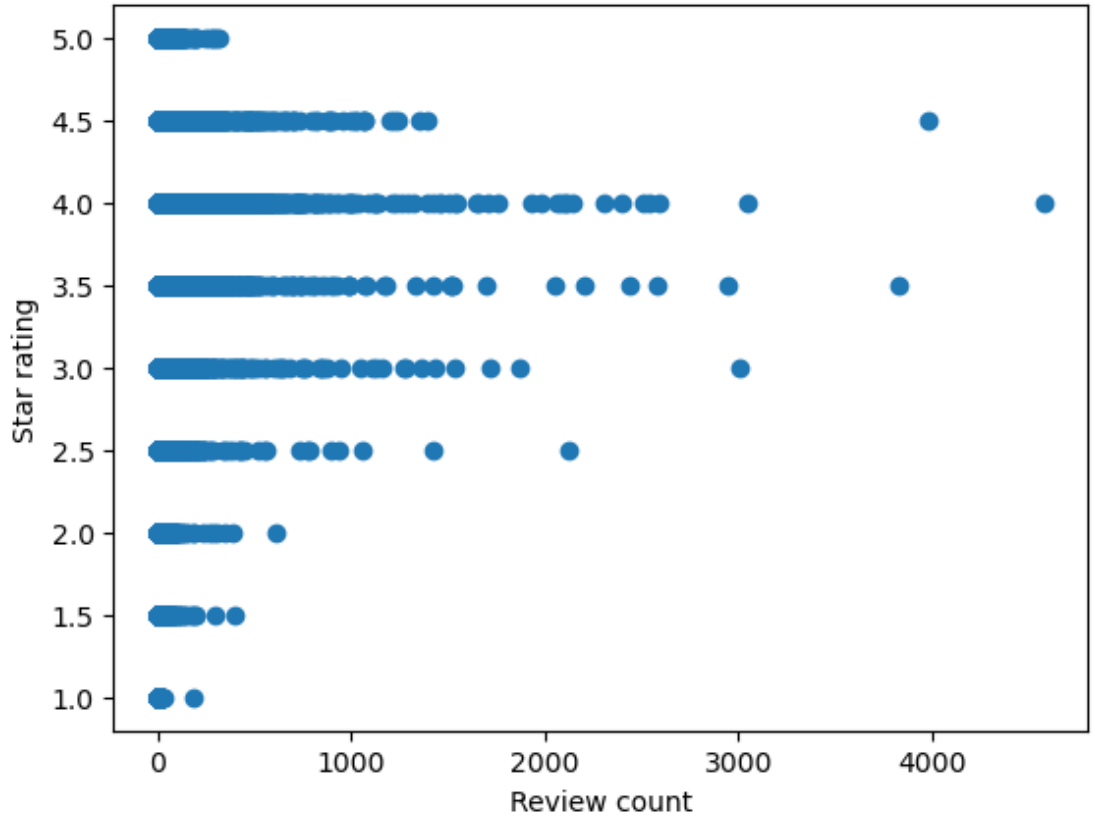
```
Number of users by year they joined Yelp:
[('2004', 51), ('2005', 691), ('2006', 3974), ('2007', 10676), ('2008', 19390), ('2009', 32968), ('2010', 50722), ('2011', 69210), ('201
2', 63897), ('2013', 63483), ('2014', 50505), ('2015', 1148)]
```

In [65]:
```
# Create a bar chart of the number of businesses by state
business_by_state = businesses_rdd.map(lambda x: (x['state'], 1)) \
                                  .reduceByKey(lambda x, y: x + y) \
                                  .sortBy(lambda x: -x[1])
```

```
states = [x[0] for x in business_by_state.take(10)]
counts = [x[1] for x in business_by_state.take(10)]
plt.bar(states, counts)
plt.xlabel('State')
plt.ylabel('Number of businesses')
plt.show()
```

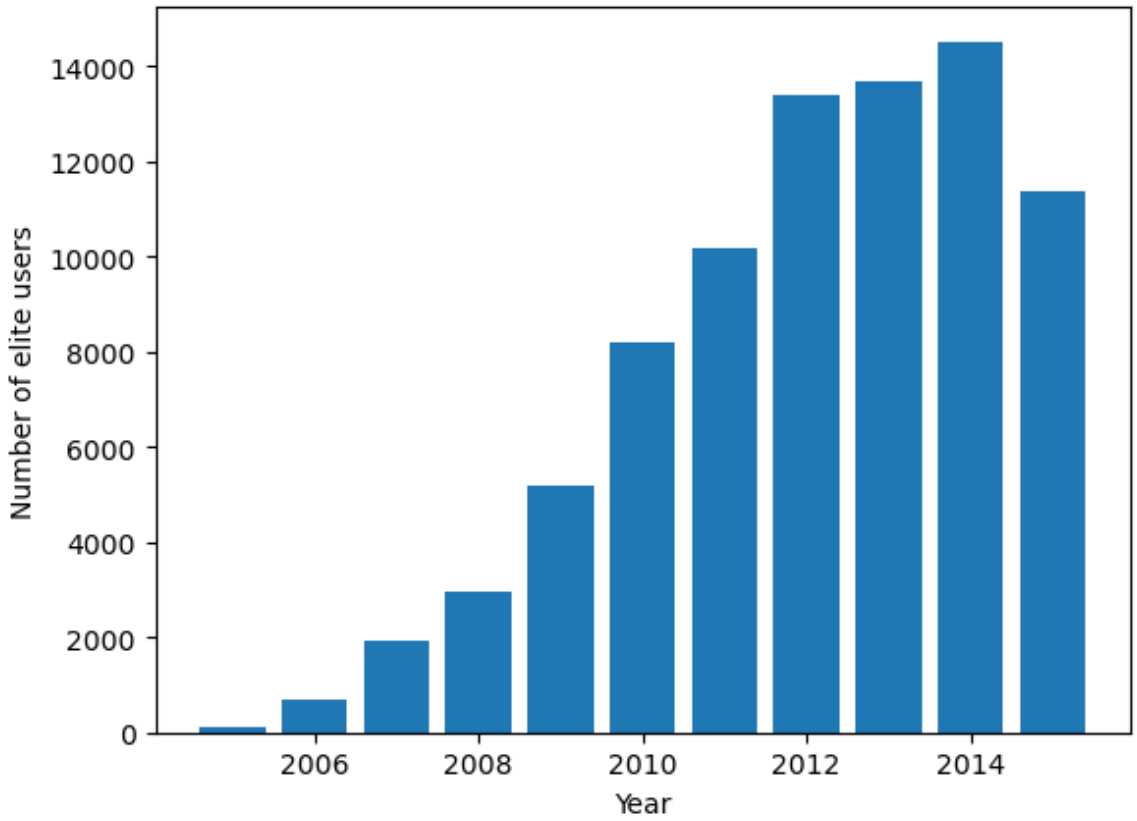


In [66]:
```
# Create a scatter plot of the relationship between review count and star rating
review_count = businesses_rdd.map(lambda x: x['review_count']).collect()
star_rating = businesses_rdd.map(lambda x: x['stars']).collect()
plt.scatter(review_count, star_rating)
plt.xlabel('Review count')
plt.ylabel('Star rating')
plt.show()
```



In [67]:

```
# Create a bar chart of the number of elite users by year
user_by_year = users_rdd.flatMap(lambda x: x['elite']) \
                        .map(lambda x: (x, 1)) \
                        .reduceByKey(lambda x, y: x + y) \
                        .sortBy(lambda x: x[0])
years = [x[0] for x in user_by_year.collect()]
counts = [x[1] for x in user_by_year.collect()]
plt.bar(years, counts)
plt.xlabel('Year')
plt.ylabel('Number of elite users')
plt.show()
```



In [68]:
```
# Calculate the average number of reviews per user by elite customer status
reviews_by_elite = users_rdd.map(lambda x: (len(x['elite']) > 0, x['review_count'])) \
                            .groupByKey() \
                            .mapValues(lambda x: sum(x) / len(x)) \
                            .sortBy(lambda x: x[0], ascending=False)

# Print the average number of reviews per user by elite status
print('Average number of reviews per user by elite status:')
for elite, count in reviews_by_elite.collect():
    status = 'Elite' if elite else 'Non-elite'
    print(f"{status}: {count:.2f}")
```

```
Average number of reviews per user by elite status:
Elite: 245.09
Non-elite: 16.44
```

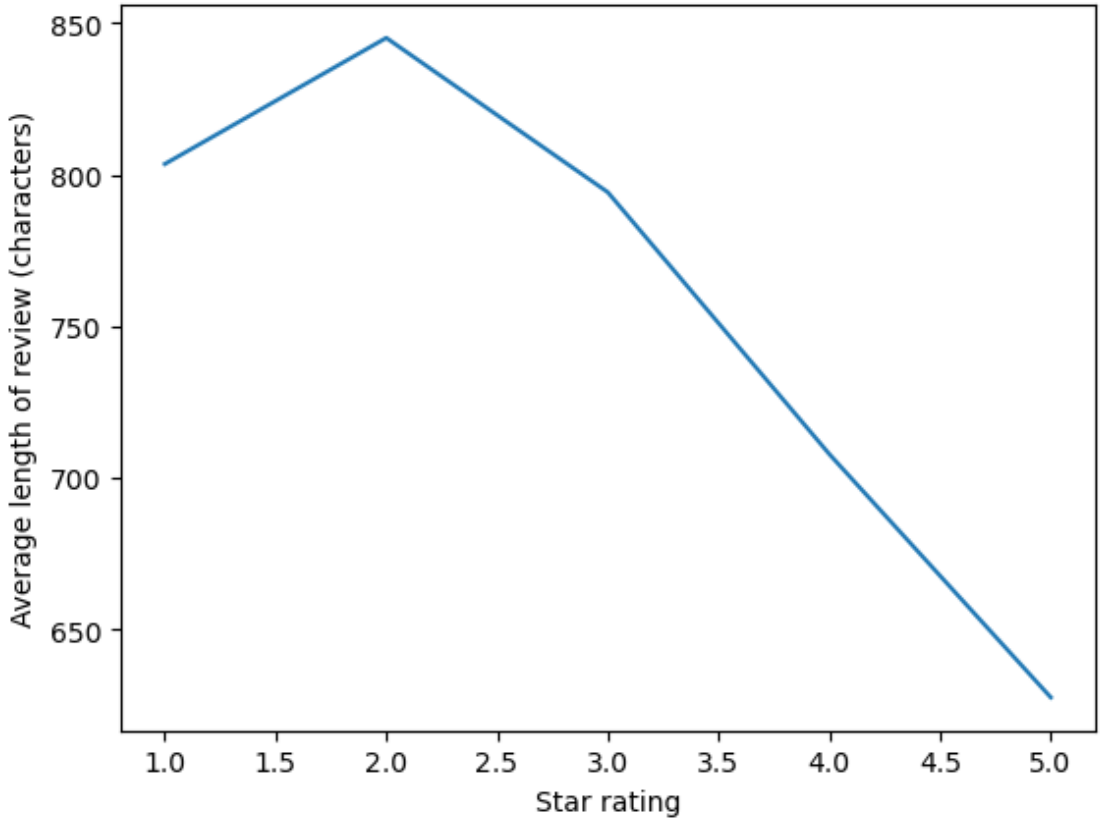
In [69]:
```
# Calculate the average star rating by state
stars_by_state = businesses_rdd.map(lambda x: (x['state'], x['stars'])) \
                               .groupByKey() \
                               .mapValues(lambda x: sum(x) / len(x)) \
                               .sortBy(lambda x: -x[1])

# Print the top 10 states with the highest average star rating
```

```
print('Top 10 states with the highest average star rating:')
for state, stars in stars_by_state.take(10):
    print(f"{state}: {stars:.2f}")
```

```
Top 10 states with the highest average star rating:
MN: 5.00
MA: 5.00
WA: 5.00
NW: 5.00
HAM: 4.50
RP: 4.23
SCB: 4.00
MLN: 3.83
EDH: 3.79
BW: 3.77
```

In [70]:

```
# Calculate the average length of reviews by star rating
reviews_by_length = reviews_rdd.map(lambda x: (x['stars'], len(x['text']))) \
                               .groupByKey() \
                               .mapValues(lambda x: sum(x) / len(x)) \
                               .sortBy(lambda x: x[0])

# Plot a line chart of the average length of reviews by star rating
stars = [x[0] for x in reviews_by_length.collect()]
lengths = [x[1] for x in reviews_by_length.collect()]
plt.plot(stars, lengths)
plt.xlabel('Star rating')
plt.ylabel('Average length of review (characters)')
plt.show()
```


850
800
750
700
650
Average length of review (characters)
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
Star rating

In [ ]:

In [ ]: