00a_S3_Setup_Final

April 14, 2023

1 ADS-508-01-SP23 Team 8: Final Project

2 Setup Database and Athena Tables

Much of the code is modified from `Fregly, C., & Barth, A. (2021). Data science on AWS: Implementing end-to-end, continuous AI and machine learning pipelines. O'Reilly.`

2.1 Install missing dependencies

PyAthena is a Python DB API 2.0 (PEP 249) compliant client for Amazon Athena.

[2]:
```
!pip install --disable-pip-version-check -q PyAthena==2.1.0
```

```
WARNING: The directory '/root/.cache/pip' or its parent directory is not
owned or is not writable by the current user. The cache has been disabled. Check
the permissions and owner of that directory. If executing pip with sudo, you
should use sudo's -H flag.
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
```

2.2 Globally import libraries

[3]:
```
import boto3
from botocore.client import ClientError
import sagemaker
import pandas as pd
from pyathena import connect
from IPython.core.display import display, HTML

%matplotlib inline
```

2.3 Instantiate AWS SageMaker session

[4]:
```
session = boto3.session.Session()
region = session.region_name
sagemaker_session = sagemaker.Session()
def_bucket = sagemaker_session.default_bucket()
bucket = 'sagemaker-us-east-ads508-sp23-t8'

s3 = boto3.Session().client(service_name="s3",
                            region_name=region)

role = sagemaker.get_execution_role()
account_id = boto3.client("sts").get_caller_identity().get("Account")

sm = boto3.Session().client(service_name="sagemaker",
                            region_name=region)
```

[5]:
```
setup_s3_bucket_passed = False
ingest_create_athena_db_passed = False
ingest_create_athena_table_tsv_passed = False
```

[6]:
```
print(f"Default bucket: {def_bucket}")
print(f"Public T8 bucket: {bucket}")
```

```
Default bucket: sagemaker-us-east-1-657724983756
Public T8 bucket: sagemaker-us-east-ads508-sp23-t8
```

2.4 Verify S3 Bucket Creation

[7]:
```
%%bash

aws s3 ls s3://${bucket}/
```

```
2023-03-16 17:05:02 aws-athena-query-results-657724983756-us-east-1
2023-03-02 16:56:48 sagemaker-studio-657724983756-5nh7ydsouq7
2023-03-02 17:25:41 sagemaker-studio-657724983756-7yc8bp8xk0b
2023-03-02 17:01:51 sagemaker-us-east-1-657724983756
2023-03-17 05:19:31 sagemaker-us-east-ads508-sp23-t8
```

[8]:
```
response = None

try:
    response = s3.head_bucket(Bucket=bucket)
    print(response)
    setup_s3_bucket_passed = True
except ClientError as e:
    print(f"[ERROR] Cannot find bucket {bucket} in {response} due to {e}.")
```

```
{'ResponseMetadata': {'RequestId': 'SVMXN6X37MEBHKWD', 'HostId':
```

```
'RXgRFVHFnPcOr1MLGT3ZVZzuS5ZyT09xU8/1usp6nCDCivyey/7QG2Q/A5Z9fyNsL0et/C5+S2g=',
'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-2':
'RXgRFVHFnPcOr1MLGT3ZVZzuS5ZyT09xU8/1usp6nCDCivyey/7QG2Q/A5Z9fyNsL0et/C5+S2g=',
'x-amz-request-id': 'SVMXN6X37MEBHKWD', 'date': 'Thu, 13 Apr 2023 16:48:46 GMT',
'x-amz-bucket-region': 'us-east-1', 'x-amz-access-point-alias': 'false',
'content-type': 'application/xml', 'server': 'AmazonS3'}, 'RetryAttempts': 0}}
```

[9]:
```
%store setup_s3_bucket_passed
```

```
Stored 'setup_s3_bucket_passed' (bool)
```

3 Set S3 Source Location (Public S3 Bucket)

[10]:
```
s3_public_path_tsv = f"s3://{bucket}"
```

[11]:
```
%store s3_public_path_tsv
```

```
Stored 's3_public_path_tsv' (str)
```

4 Set S3 Destination Location (Our Private S3 Bucket)

[12]:
```
s3_private_path_tsv = f"s3://{def_bucket}/team_8_data"
print(s3_private_path_tsv)
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data
```

[13]:
```
%store s3_private_path_tsv
```

```
Stored 's3_private_path_tsv' (str)
```

5 Copy Data From the Public S3 Bucket to our Private S3 Bucket in this Account

[14]:
```
!aws s3 cp --recursive $s3_public_path_tsv/ $s3_private_path_tsv/
```

```
copy: s3://sagemaker-us-east-
ads508-sp23-t8/raw_data/census_block/census_block_loc.csv to s3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/census_block/census_block_loc.csv
copy: s3://sagemaker-us-east-
ads508-sp23-t8/raw_data/census/nyc_census_tracts.csv to s3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/census/nyc_census_tracts.csv
copy: s3://sagemaker-us-east-
ads508-sp23-t8/raw_data/grad_outcomes/2005-2010_Graduation_Outcomes_-
_School_Level.tsv to s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/
grad_outcomes/2005-2010_Graduation_Outcomes_-_School_Level.tsv
copy: s3://sagemaker-us-east-
ads508-sp23-t8/raw_data/hs_dir/2014_-_2015_DOE_High_School_Directory.tsv to
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/hs_dir/2014_-_2015_DO
```

```
E_High_School_Directory.tsv
copy: s3://sagemaker-us-east-ads508-sp23-t8/raw_data/jobs/NYC_Jobs.tsv to
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/jobs/NYC_Jobs.tsv
copy: s3://sagemaker-us-east-ads508-sp23-t8/raw_data/evictions/Evictions.tsv to
s3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/evictions/Evictions.tsv
copy: s3://sagemaker-us-east-
ads508-sp23-t8/raw_data/crime/NYPD_Complaint_Data_Historic (1).tsv.gz to
s3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/crime/NYPD_Complaint_Data_Historic
(1).tsv.gz
```

6 List Files in our Private S3 Bucket in this Account

[15]:
```
print(s3_private_path_tsv)
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data
```

[16]:
```
!aws s3 ls $s3_private_path_tsv/
```

```
                           PRE raw_data/
```

[17]:
```
from IPython.core.display import display, HTML

display(
    HTML(
        f'<b>Review <a target="blank" href="https://s3.console.aws.amazon.com/
 ↪s3/buckets/sagemaker-{region}-{account_id}/amazon-reviews-pds/?
 ↪region={region}&tab=overview">S3 Bucket</a></b>'
    )
)
```

```
<IPython.core.display.HTML object>
```

6.1 Create Athena Database and Tables

[18]:
```
database_name = "ads508_t8"
```

[19]:
```
# Set S3 staging directory -- this is a temporary directory used for Athena␣
 ↪queries
s3_staging_dir = f"s3://{def_bucket}/team_8_data/athena/staging"
print(s3_staging_dir)
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/athena/staging
```

[20]:
```
conn = connect(region_name=region,
               s3_staging_dir=s3_staging_dir)
```

[21]:
```
create_db_stmnt = f"CREATE DATABASE IF NOT EXISTS {database_name}"
print(create_db_stmnt)
```

```
CREATE DATABASE IF NOT EXISTS ads508_t8
```

[22]:
```
pd.read_sql(create_db_stmnt,
            conn)
```

[22]:
```
Empty DataFrame
Columns: []
Index: []
```

6.1.1 Verify The Database Has Been Created Succesfully

[23]:
```
show_db_stmnt = "SHOW DATABASES"

df_show = pd.read_sql(show_db_stmnt,
                      conn)

df_show.head(17)
```

[23]:

	database_name
0	ads508_t8
1	default
2	dsoaws
3	sagemaker_featurestore

[24]:
```
if database_name in df_show.values:
    ingest_create_athena_db_passed = True
```

[25]:
```
%store ingest_create_athena_db_passed
```

```
Stored 'ingest_create_athena_db_passed' (bool)
```

6.2 Define custom function to create tables in existing database

[26]:
```
def create_athena_tbl_tsv(conn=None,
                          db=None,
                          tbl_name=None,
                          fields='',
                          s3_path=None,
                          delim=',',
                          ret='',
                          comp='',
                          skip=''):
    # Set Athena parameters

    # SQL statement to execute
```

```
    drop_tsv_tbl_stmnt = f"""DROP TABLE IF EXISTS {db}.{tbl_name}"""

    create_tsv_tbl_stmnt = f"""
        CREATE EXTERNAL TABLE IF NOT EXISTS {db}.{tbl_name}({fields})
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY '{delim}'
            LINES
                TERMINATED BY '{ret}\\n'
        LOCATION '{s3_path}'
        TBLPROPERTIES ({comp}{skip})
        """

    print(f'Create table statement:\n{create_tsv_tbl_stmnt}')

    pd.read_sql(drop_tsv_tbl_stmnt,
                conn)

    pd.read_sql(create_tsv_tbl_stmnt,
                conn)

    # Verify The Table Has Been Created Succesfully
    show_tsv_tbl_stmnt = f"SHOW TABLES IN {db}"

    df_show = pd.read_sql(show_tsv_tbl_stmnt,
                          conn)
    display(df_show.head(17))

    if tbl_name in df_show.values:
        ingest_create_athena_table_tsv_passed = True

    print(f'\nDataframe contains records:␣
↪{ingest_create_athena_table_tsv_passed}')
```

6.3 Create Athena Table from Local TSV File - 2005-2010_Graduation_Outcomes_-_School_Level.tsv

[27]:
```
grd_tsv_tbl_name = 'grad_outcomes'
grd_tsv_field_list = """
demographic string,
dbn string,
school_name string,
cohort string,
total_cohort string,
total_grads_n string,
total_grads_perc_cohort string,
total_regents_n string,
```

```
total_regents_perc_cohort string,
total_regents_perc_grads string,
advanced_regents_n string,
advanced_regents_perc_cohort string,
advanced_regents_perc_grads string,
regents_wo_advanced_n string,
regents_wo_advanced_perc_cohort string,
regents_wo_advanced_perc_grads string,
local_n string,
local_perc_cohort string,
local_perc_grads string,
still_enrolled_n string,
still_enrolled_perc_cohort string,
dropped_out_n string,
dropped_out_perc_cohort string
"""
grd_tsv_s3_raw_data_path = f"s3://{def_bucket}/team_8_data/raw_data/
 ↪grad_outcomes"
print(grd_tsv_s3_raw_data_path)

create_athena_tbl_tsv(conn=conn,
                      db=database_name,
                      tbl_name=grd_tsv_tbl_name,
                      fields=grd_tsv_field_list,
                      s3_path=grd_tsv_s3_raw_data_path,
                      delim='\\t',
                      comp='',
                      skip="'skip.header.line.count'='1'")
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/grad_outcomes
Create table statement:

        CREATE EXTERNAL TABLE IF NOT EXISTS ads508_t8.grad_outcomes(
demographic string,
dbn string,
school_name string,
cohort string,
total_cohort string,
total_grads_n string,
total_grads_perc_cohort string,
total_regents_n string,
total_regents_perc_cohort string,
total_regents_perc_grads string,
advanced_regents_n string,
advanced_regents_perc_cohort string,
advanced_regents_perc_grads string,
regents_wo_advanced_n string,
regents_wo_advanced_perc_cohort string,
```

```
regents_wo_advanced_perc_grads string,
local_n string,
local_perc_cohort string,
local_perc_grads string,
still_enrolled_n string,
still_enrolled_perc_cohort string,
dropped_out_n string,
dropped_out_perc_cohort string
)
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY '\t'
            LINES
                TERMINATED BY '\n'
        LOCATION 's3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/grad_outcomes'
        TBLPROPERTIES ('skip.header.line.count'='1')


        tab_name
0         census
1   census_block
2          crime
3      crime_pqt
4      evictions
5  grad_outcomes
6        hs_info
7           jobs


Dataframe contains records: True
```

6.3.1 Run A Sample Query

[28]:
```
grd_dbn_id01 = "01M448"

grd_select_dbn_stmnt = f"""
SELECT * FROM {database_name}.{grd_tsv_tbl_name}
WHERE dbn = '{grd_dbn_id01}'
LIMIT 17
"""

print(grd_select_dbn_stmnt)

grd_df01_s01 = pd.read_sql(grd_select_dbn_stmnt,
                           conn)

grd_df01_s01.head(17)
```

```
SELECT * FROM ads508_t8.grad_outcomes
WHERE dbn = '01M448'
LIMIT 17
```

[28]:

	demographic	dbn	school_name	\
0	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
1	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
2	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
3	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
4	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
5	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
6	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
7	English Language Learners	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
8	English Language Learners	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
9	English Language Learners	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
10	English Language Learners	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
11	English Language Learners	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
12	English Language Learners	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
13	English Language Learners	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
14	English Proficient Students	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
15	English Proficient Students	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	
16	English Proficient Students	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	

	cohort	total_cohort	total_grads_n	total_grads_perc_cohort	\
0	2001	64	46	71.900000000000006	
1	2002	52	33	63.5	
2	2003	87	67	77	
3	2004	112	75	67	
4	2005	121	64	52.9	
5	2006	124	53	42.7	
6	2006 Aug	124	60	48.4	
7	2002	1	s		
8	2001	5	s		
9	2003	1	s		
10	2004	9	s		
11	2005	7	s		
12	2006	3	s		
13	2006 Aug	3	s		
14	2001	59	44	74.599999999999994	
15	2002	51	32	62.7	
16	2003	86	67	77.900000000000006	

	total_regents_n	total_regents_perc_cohort	total_regents_perc_grads	…	\
0	32	50	69.599999999999994	…	
1	19	36.5	57.6	…	

```
2                39                    44.8                    58.2  …
3                36                    32.1                      48  …
4                35                    28.9                    54.7  …
5                42                    33.9                    79.2  …
6                42                    33.9                      70  …
7                 s                                                    …
8                 s                                                    …
9                 s                                                    …
10                s                                                    …
11                s                                                    …
12                s                                                    …
13                s                                                    …
14               31                    52.5                    70.5  …
15               19       37.299999999999997                    59.4  …
16               39                    45.3                    58.2  …

   regents_wo_advanced_n regents_wo_advanced_perc_cohort  \
0                     25                            39.1
1                     11                            21.2
2                     28          32.200000000000003
3                     30                            26.8
4                     31                            25.6
5                     34                            27.4
6                     34                            27.4
7                      s
8                      s
9                      s
10                     s
11                     s
12                     s
13                     s
14                    24          40.700000000000003
15                    11                            21.6
16                    28                            32.6

   regents_wo_advanced_perc_grads local_n   local_perc_cohort  \
0                            54.3      14                21.9
1               33.299999999999997      14                26.9
2                            41.8      28  32.200000000000003
3                              40      39  34.799999999999997
4                            48.4      29                  24
5                            64.2      11                 8.9
6                            56.7      18                14.5
7                                       s
8                                       s
9                                       s
10                                      s
```

```
11                                    s
12                                    s
13                                    s
14                          54.5      13                  22
15                          34.4      13                25.5
16                          41.8      28                32.6

   local_perc_grads still_enrolled_n still_enrolled_perc_cohort dropped_out_n  \
0              30.4               10                       15.6             6
1              42.4               16                       30.8             1
2              41.8                9                       10.3            11
3                52               33                       29.5             4
4              45.3               41                       33.9            11
5              20.8               46                       37.1            20
6                30               39                       31.5            20
7                                    s                                        s
8                                    s                                        s
9                                    s                                        s
10                                   s                                        s
11                                   s                                        s
12                                   s                                        s
13                                   s                                        s
14             29.5                8                       13.6             5
15             40.6               16                       31.4             1
16             41.8                9                       10.5            10

   dropped_out_perc_cohort
0                      9.4
1                      1.9
2                     12.6
3                      3.6
4                      9.1
5       16.100000000000001
6       16.100000000000001
7
8
9
10
11
12
13
14                     8.5
15                       2
16                    11.6

[17 rows x 23 columns]
```

[29]:
```
if not grd_df01_s01.empty:
    print("[OK]")
else:
    print("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOUR DATA HAS NOT BEEN REGISTERED WITH ATHENA. LOOK IN␣
 ↪PREVIOUS CELLS TO FIND THE ISSUE.")
    print("+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

```
[OK]
```

6.4 Create Athena Table from Local TSV File - 2014_-_2015_DOE_High_School_Directory.tsv

[30]:
```
hsi_tsv_tbl_name = 'hs_info'
hsi_tsv_field_list = """
dbn string,
school_name string,
borough string,
building_code string,
phone_number string,
fax_number string,
grade_span_min string,
grade_span_max string,
expgrade_span_min string,
expgrade_span_max string,
bus string,
subway string,
primary_address_line_1 string,
city string,
state_code string,
postcode string,
website string,
total_students string,
campus_name string,
school_type string,
overview_paragraph string,
program_highlights string,
language_classes string,
advancedplacement_courses string,
online_ap_courses string,
online_language_courses string,
extracurricular_activities string,
psal_sports_boys string,
psal_sports_girls string,
psal_sports_coed string,
school_sports string,
partner_cbo string,
```

```
partner_hospital string,
partner_highered string,
partner_cultural string,
partner_nonprofit string,
partner_corporate string,
partner_financial string,
partner_other string,
addtl_info1 string,
addtl_info2 string,
start_time string,
end_time string,
se_services string,
ell_programs string,
school_accessibility_description string,
number_programs string,
priority01 string,
priority02 string,
priority03 string,
priority04 string,
priority05 string,
priority06 string,
priority07 string,
priority08 string,
priority09 string,
priority10 string,
location_1 string,
community_board string,
council_district string,
census_tract string,
bin string,
bbl string,
nta string
"""
hsi_tsv_s3_raw_data_path = f"s3://{def_bucket}/team_8_data/raw_data/hs_dir"
print(hsi_tsv_s3_raw_data_path)

create_athena_tbl_tsv(conn=conn,
                      db=database_name,
                      tbl_name=hsi_tsv_tbl_name,
                      fields=hsi_tsv_field_list,
                      s3_path=hsi_tsv_s3_raw_data_path,
                      delim='\\t',
                      comp='',
                      skip="'skip.header.line.count'='1'")
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/hs_dir
Create table statement:
```

```
        CREATE EXTERNAL TABLE IF NOT EXISTS ads508_t8.hs_info(
dbn string,
school_name string,
borough string,
building_code string,
phone_number string,
fax_number string,
grade_span_min string,
grade_span_max string,
expgrade_span_min string,
expgrade_span_max string,
bus string,
subway string,
primary_address_line_1 string,
city string,
state_code string,
postcode string,
website string,
total_students string,
campus_name string,
school_type string,
overview_paragraph string,
program_highlights string,
language_classes string,
advancedplacement_courses string,
online_ap_courses string,
online_language_courses string,
extracurricular_activities string,
psal_sports_boys string,
psal_sports_girls string,
psal_sports_coed string,
school_sports string,
partner_cbo string,
partner_hospital string,
partner_highered string,
partner_cultural string,
partner_nonprofit string,
partner_corporate string,
partner_financial string,
partner_other string,
addtl_info1 string,
addtl_info2 string,
start_time string,
end_time string,
se_services string,
ell_programs string,
school_accessibility_description string,
number_programs string,
```

```
priority01 string,
priority02 string,
priority03 string,
priority04 string,
priority05 string,
priority06 string,
priority07 string,
priority08 string,
priority09 string,
priority10 string,
location_1 string,
community_board string,
council_district string,
census_tract string,
bin string,
bbl string,
nta string
)
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY '\t'
            LINES
                TERMINATED BY '\n'
        LOCATION 's3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/hs_dir'
        TBLPROPERTIES ('skip.header.line.count'='1')


        tab_name
0         census
1   census_block
2          crime
3      crime_pqt
4      evictions
5  grad_outcomes
6        hs_info
7           jobs


Dataframe contains records: True
```

6.4.1 Run A Sample Query

[31]:
```
hsi_dbn_id01 = "01M448"

hsi_select_dbn_stmnt = f"""
SELECT * FROM {database_name}.{hsi_tsv_tbl_name}
WHERE dbn = '{hsi_dbn_id01}'
```

```
LIMIT 17
"""

print(hsi_select_dbn_stmnt)

hsi_df01_s01 = pd.read_sql(hsi_select_dbn_stmnt,
                           conn)

hsi_df01_s01.head(17)
```

```
SELECT * FROM ads508_t8.hs_info
WHERE dbn = '01M448'
LIMIT 17
```

[31]:
```
       dbn                          school_name    borough building_code  \
0  01M448  University Neighborhood High School  Manhattan          M446

   phone_number    fax_number grade_span_min grade_span_max expgrade_span_min  \
0  212-962-4341  212-267-5611              9             12

  expgrade_span_max  … priority08 priority09 priority10          location_1  \
0                    …                                  "200 Monroe Street

  community_board council_district census_tract   bin   bbl   nta
0            None             None         None  None  None  None

[1 rows x 64 columns]
```

[32]:
```
if not hsi_df01_s01.empty:
    print("[OK]")
else:
    print("+++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOUR DATA HAS NOT BEEN REGISTERED WITH ATHENA. LOOK IN␣
 ↪PREVIOUS CELLS TO FIND THE ISSUE.")
    print("+++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

```
[OK]
```

6.5 Create Athena Table from Local CSV File - `nyc_census_tracts.csv`

[33]:
```
cen_tsv_tbl_name = 'census'
cen_tsv_field_list = """
censustract string,
county string,
borough string,
totalpop int,
```

```
men int,
women int,
hispanic double,
white double,
black double,
native double,
asian double,
citizen int,
income int,
incomeerr int,
incomepercap int,
incomepercaperr int,
poverty double,
childpoverty double,
professional double,
service double,
office double,
construction double,
production double,
drive double,
carpool double,
transit double,
walk double,
othertransp double,
workathome double,
meancommute double,
employed int,
privatework double,
publicwork double,
selfemployed double,
familywork double,
unemployment double
"""
cen_tsv_s3_raw_data_path = f"s3://{def_bucket}/team_8_data/raw_data/census"
print(cen_tsv_s3_raw_data_path)

create_athena_tbl_tsv(conn=conn,
                      db=database_name,
                      tbl_name=cen_tsv_tbl_name,
                      fields=cen_tsv_field_list,
                      s3_path=cen_tsv_s3_raw_data_path,
                      comp='',
                      skip="'skip.header.line.count'='1'")
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/census
Create table statement:

        CREATE EXTERNAL TABLE IF NOT EXISTS ads508_t8.census(
```

```
censustract string,
county string,
borough string,
totalpop int,
men int,
women int,
hispanic double,
white double,
black double,
native double,
asian double,
citizen int,
income int,
incomeerr int,
incomepercap int,
incomepercaperr int,
poverty double,
childpoverty double,
professional double,
service double,
office double,
construction double,
production double,
drive double,
carpool double,
transit double,
walk double,
othertransp double,
workathome double,
meancommute double,
employed int,
privatework double,
publicwork double,
selfemployed double,
familywork double,
unemployment double
)
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY ','
            LINES
                TERMINATED BY '\n'
        LOCATION 's3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/census'
        TBLPROPERTIES ('skip.header.line.count'='1')


        tab_name
```

```
0         census
1   census_block
2          crime
3      crime_pqt
4      evictions
5  grad_outcomes
6        hs_info
7           jobs
```

```
Dataframe contains records: True
```

6.5.1 Run A Sample Query

[34]:
```
cen_bourough_id01 = "Bronx"

cen_select_dbn_stmnt = f"""
SELECT * FROM {database_name}.{cen_tsv_tbl_name}
WHERE borough = '{cen_bourough_id01}'
LIMIT 17
"""

print(cen_select_dbn_stmnt)

cen_df01_s01 = pd.read_sql(cen_select_dbn_stmnt,
                           conn)

cen_df01_s01.head(17)
```

```
SELECT * FROM ads508_t8.census
WHERE borough = 'Bronx'
LIMIT 17
```

[34]:

	censustract	county	borough	totalpop	men	women	hispanic	white	black	\
0	36005000100	Bronx	Bronx	7703	7133	570	29.9	6.1	60.9	
1	36005000200	Bronx	Bronx	5403	2659	2744	75.8	2.3	16.0	
2	36005000400	Bronx	Bronx	5915	2896	3019	62.7	3.6	30.7	
3	36005001600	Bronx	Bronx	5879	2558	3321	65.1	1.6	32.4	
4	36005001900	Bronx	Bronx	2591	1206	1385	55.4	9.0	29.0	
5	36005002000	Bronx	Bronx	8516	3301	5215	61.1	1.6	31.1	
6	36005002300	Bronx	Bronx	4774	2130	2644	62.3	0.2	36.5	
7	36005002400	Bronx	Bronx	150	109	41	0.0	52.0	48.0	
8	36005002500	Bronx	Bronx	5355	2338	3017	76.5	1.5	18.9	
9	36005002701	Bronx	Bronx	3016	1375	1641	68.0	0.0	31.2	
10	36005002702	Bronx	Bronx	4778	2427	2351	71.3	1.6	26.2	
11	36005002800	Bronx	Bronx	5299	2292	3007	23.0	0.2	71.4	

```
12  36005003100  Bronx   Bronx      1466   769    697      72.3    0.6   24.6
13  36005003300  Bronx   Bronx      3912  1824   2088      65.6    1.0   30.6
14  36005003500  Bronx   Bronx      3948  1921   2027      73.5    0.7   25.9
15  36005003700  Bronx   Bronx       246   128    118      57.7   24.4   17.9
16  36005003800  Bronx   Bronx      1193   542    651      53.1    1.8   42.7

    native  …  walk  othertransp  workathome  meancommute  employed  \
0      0.2  …   NaN          NaN         NaN          NaN         0
1      0.0  …   2.9          0.0         0.0         43.0      2308
2      0.0  …   1.4          0.5         2.1         45.0      2675
3      0.0  …   8.6          1.6         1.7         38.8      2120
4      0.0  …   3.0          2.4         6.2         45.4      1083
5      0.3  …   4.3          1.0         0.0         46.0      2508
6      1.0  …  14.0          1.5         4.1         42.7      1191
7      0.0  …   0.0          0.0         0.0          NaN       113
8      0.0  …  17.7          1.8         2.7         35.5      1691
9      0.0  …  18.0          0.0         1.6         42.8      1102
10     0.0  …   7.1          0.7         0.5         44.0      1559
11     0.0  …   2.0          0.6         2.7         47.3      2394
12     0.0  …  14.6          3.5         0.0         40.1       722
13     1.7  …  13.5          0.8         1.8         42.5      1113
14     0.0  …   7.4          1.0         4.7         41.6      1360
15     0.0  …  12.5          0.0         0.0         33.0        96
16     0.0  …   6.9          1.3         1.1         41.3       476

    privatework  publicwork  selfemployed  familywork  unemployment
0           NaN         NaN           NaN         NaN           NaN
1          80.8        16.2           2.9         0.0           7.7
2          71.7        25.3           2.5         0.6           9.5
3          75.0        21.3           3.8         0.0           8.7
4          76.8        15.5           7.7         0.0          19.2
5          71.0        21.3           7.7         0.0          17.2
6          74.2        16.1           9.7         0.0          18.9
7          62.8        37.2           0.0         0.0           0.0
8          85.1         8.3           6.1         0.5           9.4
9          86.9         8.5           4.5         0.0          15.2
10         75.0        14.0          11.0         0.0          10.6
11         61.9        37.4           0.6         0.0          12.8
12         79.2        10.2          10.5         0.0           6.6
13         77.2        16.9           5.9         0.0          18.5
14         83.2        13.4           3.4         0.0          11.8
15        100.0         0.0           0.0         0.0          11.1
16         71.8        27.1           1.1         0.0          13.1

[17 rows x 36 columns]
```

[35]:
```
if not cen_df01_s01.empty:
    print("[OK]")
else:
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOUR DATA HAS NOT BEEN REGISTERED WITH ATHENA. LOOK IN␣
 ↪PREVIOUS CELLS TO FIND THE ISSUE.")
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

```
[OK]
```

6.6 Create Athena Table from Local TSV File - NYPD_Complaint_Data_Historic (1).csv

[36]:
```
cri_tsv_tbl_name = 'crime'
cri_tsv_field_list = """
cmplnt_num string,
cmplnt_fr_dt string,
cmplnt_fr_tm string,
cmplnt_to_dt string,
cmplnt_to_tm string,
addr_pct_cd string,
rpt_dt string,
ky_cd string,
ofns_desc string,
pd_cd string,
pd_desc string,
crm_atpt_cptd_cd string,
law_cat_cd string,
borough string,
loc_of_occur_desc string,
prem_typ_desc string,
juris_desc string,
jurisdiction_code string,
parks_nm string,
hadevelopt string,
housing_psa string,
x_coord_cd string,
y_coord_cd string,
susp_age_group string,
susp_race string,
susp_sex string,
transit_district string,
latitude string,
longitude string,
lat_lon string,
patrol_boro string,
station_name string,
```

```
vic_age_group string,
vic_race string,
vic_sex string
"""
cri_tsv_s3_raw_data_path = f"s3://{def_bucket}/team_8_data/raw_data/crime"
print(cri_tsv_s3_raw_data_path)

create_athena_tbl_tsv(conn=conn,
                      db=database_name,
                      tbl_name=cri_tsv_tbl_name,
                      fields=cri_tsv_field_list,
                      s3_path=cri_tsv_s3_raw_data_path,
                      delim='\\t',
                      comp="'compressionType'='gzip', ",
                      skip="'skip.header.line.count'='1'")
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/crime
Create table statement:

        CREATE EXTERNAL TABLE IF NOT EXISTS ads508_t8.crime(
cmplnt_num string,
cmplnt_fr_dt string,
cmplnt_fr_tm string,
cmplnt_to_dt string,
cmplnt_to_tm string,
addr_pct_cd string,
rpt_dt string,
ky_cd string,
ofns_desc string,
pd_cd string,
pd_desc string,
crm_atpt_cptd_cd string,
law_cat_cd string,
borough string,
loc_of_occur_desc string,
prem_typ_desc string,
juris_desc string,
jurisdiction_code string,
parks_nm string,
hadevelopt string,
housing_psa string,
x_coord_cd string,
y_coord_cd string,
susp_age_group string,
susp_race string,
susp_sex string,
transit_district string,
latitude string,
```

```
longitude string,
lat_lon string,
patrol_boro string,
station_name string,
vic_age_group string,
vic_race string,
vic_sex string
)
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY '\t'
            LINES
                TERMINATED BY '\n'
        LOCATION 's3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/crime'
        TBLPROPERTIES ('compressionType'='gzip', 'skip.header.line.count'='1')


        tab_name
0         census
1   census_block
2          crime
3      crime_pqt
4      evictions
5  grad_outcomes
6        hs_info
7           jobs


Dataframe contains records: True
```

6.6.1 Run A Sample Query

[37]:
```
cri_law_cat_cd01 = "misdemeanor"
cri_borough01 = "bronx"

cri_select_dbn_stmnt01 = f"""
SELECT * FROM {database_name}.{cri_tsv_tbl_name}
WHERE LOWER(law_cat_cd) = '{cri_law_cat_cd01}'
    AND LOWER(borough) = '{cri_borough01}'
LIMIT 17
"""

print(cri_select_dbn_stmnt01)

cri_df01_s01 = pd.read_sql(cri_select_dbn_stmnt01,
                           conn)
```

```
cri_df01_s01.head(17)
```

```
SELECT * FROM ads508_t8.crime
WHERE LOWER(law_cat_cd) = 'misdemeanor'
    AND LOWER(borough) = 'bronx'
LIMIT 17
```

[37]:

	cmplnt_num	cmplnt_fr_dt	cmplnt_fr_tm	cmplnt_to_dt	cmplnt_to_tm	addr_pct_cd	\
0	629632833	02/06/2018	23:15:00			52	
1	377132404	08/04/2018	22:15:00			44	
2	584276892	02/11/2018	17:30:00	02/12/2018	06:00:00	41	
3	599398393	05/23/2018	23:30:00	05/24/2018	02:00:00	47	
4	955332763	02/23/2018	13:55:00			43	
5	412087799	05/07/2018	15:00:00	05/19/2018	18:00:00	47	
6	692539256	08/30/2018	17:01:00	08/31/2018	17:41:00	52	
7	763109503	05/03/2018	16:55:00			44	
8	472961714	08/01/2018	11:30:00	08/01/2018	11:33:00	49	
9	249426294	06/14/2018	14:50:00	06/14/2018	14:55:00	49	
10	828720525	04/06/2018	17:25:00	04/06/2018	17:25:00	40	
11	783013551	04/28/2018	09:40:00			40	
12	847047309	10/06/2018	08:01:00			45	
13	746565217	07/14/2018	22:00:00			44	
14	700974838	12/09/2018	17:00:00	12/09/2018	18:30:00	43	
15	140793872	10/10/2018	00:20:00	10/10/2018	00:35:00	40	
16	544404081	05/10/2018	11:00:00	05/10/2018	11:50:00	46	

	rpt_dt	ky_cd	ofns_desc	pd_cd	…	susp_sex	\
0	02/07/2018	341	PETIT LARCENY	333	…	F	
1	08/04/2018	344	ASSAULT 3 & RELATED OFFENSES	101	…	M	
2	02/12/2018	351	CRIMINAL MISCHIEF & RELATED OF	254	…	U	
3	05/24/2018	351	CRIMINAL MISCHIEF & RELATED OF	254	…		
4	02/23/2018	351	CRIMINAL MISCHIEF & RELATED OF	259	…	F	
5	05/21/2018	361	OFF. AGNST PUB ORD SENSBLTY &	639	…	U	
6	09/01/2018	341	PETIT LARCENY	313	…		
7	05/03/2018	341	PETIT LARCENY	333	…	M	
8	08/16/2018	361	OFF. AGNST PUB ORD SENSBLTY &	639	…	M	
9	06/14/2018	351	CRIMINAL MISCHIEF & RELATED OF	259	…	M	
10	04/06/2018	235	DANGEROUS DRUGS	567	…	M	
11	04/28/2018	341	PETIT LARCENY	333	…	F	
12	10/06/2018	341	PETIT LARCENY	333	…	M	
13	07/21/2018	344	ASSAULT 3 & RELATED OFFENSES	101	…	M	
14	12/10/2018	341	PETIT LARCENY	321	…	U	
15	10/10/2018	344	ASSAULT 3 & RELATED OFFENSES	101	…	M	
16	05/18/2018	361	OFF. AGNST PUB ORD SENSBLTY &	639	…	M	

	transit_district	latitude	longitude	\
0		40.87367103500002	-73.90801364899994	
1		40.82616961200006	-73.91683070899995	
2		40.827049319000025	-73.89499419099997	
3		40.882615325000074	-73.85194765899996	
4		40.82870937100006	-73.87776995499998	
5		40.881300913000075	-73.85433733899998	
6		40.86840712200007	-73.89260767699994	
7		40.83778161800007	-73.91945797099999	
8		40.846705615000076	-73.86472139499993	
9		40.844996090000045	-73.85167356799997	
10		40.80806113500005	-73.92248131399998	
11		40.80719918600005	-73.91835350199995	
12		40.84306602000004	-73.83703668899994	
13		40.83857962500008	-73.92663072799998	
14		40.82667327000007	-73.88353694199996	
15		40.82142768200004	-73.91436893199995	
16		40.85036729300003	-73.91725589399994	

	lat_lon	patrol_boro	station_name	\
0	(40.873671035, -73.908013649)	PATROL BORO	BRONX	
1	(40.826169612, -73.916830709)	PATROL BORO	BRONX	
2	(40.827049319, -73.894994191)	PATROL BORO	BRONX	
3	(40.882615325, -73.851947659)	PATROL BORO	BRONX	
4	(40.828709371, -73.877769955)	PATROL BORO	BRONX	
5	(40.881300913, -73.854337339)	PATROL BORO	BRONX	
6	(40.868407122, -73.892607677)	PATROL BORO	BRONX	
7	(40.837781618, -73.919457971)	PATROL BORO	BRONX	
8	(40.846705615, -73.864721395)	PATROL BORO	BRONX	
9	(40.84499609, -73.851673568)	PATROL BORO	BRONX	
10	(40.808061135, -73.922481314)	PATROL BORO	BRONX	
11	(40.807199186, -73.918353502)	PATROL BORO	BRONX	
12	(40.84306602, -73.837036689)	PATROL BORO	BRONX	
13	(40.838579625, -73.926630728)	PATROL BORO	BRONX	
14	(40.82667327, -73.883536942)	PATROL BORO	BRONX	
15	(40.821427682, -73.914368932)	PATROL BORO	BRONX	
16	(40.850367293, -73.917255894)	PATROL BORO	BRONX	

	vic_age_group	vic_race	vic_sex
0	UNKNOWN	UNKNOWN	D
1	25-44	WHITE HISPANIC	F
2	45-64	BLACK	F
3	25-44	ASIAN / PACIFIC ISLANDER	F
4	UNKNOWN	UNKNOWN	D
5	<18	WHITE HISPANIC	F
6	65+	BLACK HISPANIC	F
7	UNKNOWN	UNKNOWN	D

```
8          25-44  ASIAN / PACIFIC ISLANDER       M
9          45-64                     WHITE       M
10       UNKNOWN                   UNKNOWN       E
11       UNKNOWN                   UNKNOWN       D
12       UNKNOWN                   UNKNOWN       D
13         45-64                     WHITE       M
14       UNKNOWN                   UNKNOWN       D
15         45-64            BLACK HISPANIC       F
16         18-24            WHITE HISPANIC       F

[17 rows x 35 columns]
```

[38]:
```
if not cri_df01_s01.empty:
    print("[OK]")
else:
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOUR DATA HAS NOT BEEN REGISTERED WITH ATHENA. LOOK IN␣
 ↪PREVIOUS CELLS TO FIND THE ISSUE.")
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

```
[OK]
```

6.7 Create Athena Table from Local TSV File - Evictions.tsv

[39]:
```
evi_tsv_tbl_name = 'evictions'
evi_tsv_field_list = """
court_index_number string,
docket_number string,
eviction_address string,
eviction_apartment_number string,
executed_date string,
marshal_first_name string,
marshal_last_name string,
residential_or_commercial string,
borough string,
eviction_postcode string,
ejectment string,
eviction_or_legal_possession string,
latitude string,
longitude string,
community_board string,
council_district string,
census_tract string,
bin string,
bbl string,
nta string
"""
evi_tsv_s3_raw_data_path = f"s3://{def_bucket}/team_8_data/raw_data/evictions"
```

```
print(evi_tsv_s3_raw_data_path)

create_athena_tbl_tsv(conn=conn,
                      db=database_name,
                      tbl_name=evi_tsv_tbl_name,
                      fields=evi_tsv_field_list,
                      s3_path=evi_tsv_s3_raw_data_path,
                      delim='\\t',
                      comp='',
                      skip="'skip.header.line.count'='1'")
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/evictions
Create table statement:

        CREATE EXTERNAL TABLE IF NOT EXISTS ads508_t8.evictions(
court_index_number string,
docket_number string,
eviction_address string,
eviction_apartment_number string,
executed_date string,
marshal_first_name string,
marshal_last_name string,
residential_or_commercial string,
borough string,
eviction_postcode string,
ejectment string,
eviction_or_legal_possession string,
latitude string,
longitude string,
community_board string,
council_district string,
census_tract string,
bin string,
bbl string,
nta string
)
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY '\t'
            LINES
                TERMINATED BY '\n'
        LOCATION 's3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/evictions'
        TBLPROPERTIES ('skip.header.line.count'='1')


        tab_name
0         census
```

```
1   census_block
2          crime
3     crime_pqt
4     evictions
5  grad_outcomes
6       hs_info
7          jobs
```

```
Dataframe contains records: True
```

6.7.1 Run A Sample Query

[40]:
```
evi_borough01 = "BRONX"

evi_select_dbn_stmnt = f"""
SELECT * FROM {database_name}.{evi_tsv_tbl_name}
WHERE borough = '{evi_borough01}'
LIMIT 17
"""

print(evi_select_dbn_stmnt)

evi_df01_s01 = pd.read_sql(evi_select_dbn_stmnt,
                           conn)

evi_df01_s01.head(17)
```

```
SELECT * FROM ads508_t8.evictions
WHERE borough = 'BRONX'
LIMIT 17
```

[40]:

	court_index_number	docket_number
0	56037/17	339568
1	B047517/19	409031
2	15068/17	334442
3	14866/19A	097278
4	66703/18BX	090391
5	B806500/18	396012
6	54026/17	341956
7	69137/18	10335
8	18348/16	324092
9	75943/16A	060118
10	44987/17	069826
11	B55293/17	101573
12	B54741/17	083789

```
13            8911/17        062195
14           72383/16        292013
15            7096/18        077442
16          902063/14        067290

                                    eviction_address  \
0                                547 EAST 168TH STREET
1                                4014 CARPENTER AVENUE
2                                655 EAST 224TH STREET
3                                  718 PENFIELD STREET
4   2032 EAST 177TH ST A /K/A 2032 CROSS BRONX EXP…
5                                281 EAST 143RD STREET
6                               1211 SOUTHERN BOULEVARD
7                             1351 BOSTON ROAD - APT 201
8                                2280 LORING PLACE NORTH
9                           1551    WILLIAMSBRID GE ROAD
10                               1514 SEDGWICK AVENUE
11                               2800 SEDGWICK AVENUE
12                              810 EAST 152ND STREE T
13                                   40 RICHMAN PLAZA
14                                 5 METROPOLITAN OVAL
15                                 3319 BAYCHESTER AVE
16  3488 JEROME AVENUE GROUND FLOOR STORE PREMISES…

   eviction_apartment_number executed_date marshal_first_name  \
0                         3H    02/26/2018             Thomas
1                         4B    11/16/2022            Richard
2                          1    09/29/2017             Thomas
3                        2-F    10/24/2019             Justin
4                         1E    07/30/2019             Justin
5                        07A    01/17/2019            Richard
6                        301    11/19/2018             Thomas
7                        201    07/15/2019             Robert
8                         4B    05/22/2017             Thomas
9                        4-B    08/10/2017             Justin
10                        7C    05/22/2018             Justin
11                        5I    01/19/2018            Darlene
12                       705    07/31/2018             Ileana
13                       11J    05/23/2017             Justin
14                        8D    10/13/2017             George
15                1ST FLOOR    08/21/2018             Justin
16                             02/07/2017              Henry

   marshal_last_name residential_or_commercial borough eviction_postcode  \
0                Bia                Residential   BRONX             10456
1              McCoy                Residential   BRONX             10466
2                Bia                Residential   BRONX             10467
```

```
3          Grossman               Residential   BRONX            10470
4          Grossman               Residential   BRONX            10472
5             McCoy               Residential   BRONX            10451
6               Bia               Residential   BRONX            10459
7          Renzulli               Residential   BRONX            10456
8               Bia               Residential   BRONX            10468
9          Grossman               Residential   BRONX            10461
10         Grossman               Residential   BRONX            10453
11           Barone               Residential   BRONX            10468
12           Rivera               Residential   BRONX            10455
13         Grossman               Residential   BRONX            10453
14           Essock               Residential   BRONX            10462
15         Grossman               Residential   BRONX            10469
16            Daley                Commercial   BRONX            10467

           ejectment eviction_or_legal_possession   latitude   longitude  \
0   Not an Ejectment                   Possession  40.830857  -73.905191
1   Not an Ejectment                   Possession  40.889878  -73.862686
2   Not an Ejectment                   Possession  40.887599  -73.862391
3   Not an Ejectment                   Possession  40.904888  -73.849089
4   Not an Ejectment                   Possession  40.831685  -73.856168
5   Not an Ejectment                   Possession  40.814845  -73.924083
6   Not an Ejectment                   Possession  40.828949  -73.891897
7   Not an Ejectment                   Possession  40.832166  -73.898808
8   Not an Ejectment                   Possession  40.861277  -73.908723
9   Not an Ejectment                   Possession
10  Not an Ejectment                   Possession  40.846731  -73.924961
11  Not an Ejectment                   Possession  40.871918  -73.902287
12  Not an Ejectment                   Possession  40.815435  -73.905199
13  Not an Ejectment                   Possession  40.852084  -73.922436
14  Not an Ejectment                   Possession  40.838831  -73.860013
15  Not an Ejectment                   Possession  40.878064  -73.836946
16  Not an Ejectment                   Possession

   community_board council_district census_tract      bin         bbl  \
0                3                16          145  2004227  2026100065
1               12                12          408  2063060  2048280031
2               12                12          394  2062985  2048260028
3               12                11          442  2071873  2051130039
4                9                18           78  2026230  2038030019
5                1                 8           51  2091116  2023240001
6                3                17          125  2113777  2029750037
7                3                16          151  2128618  2029340050
8                7                14          255  2014918  2032250015
9
10               5                16        20501  2114714  2028800009
11               8                14        26702  2015379  2032490202
```

```
12                1                 8           79  2094310  2026640061
13                5                16           53  2113629  2028820229
14                9                18        21001  2096599  2039447501
15               12                12        46202  2065413  2048810067
16
```

```
                                nta
0                  Claremont-Bathgate
1             Williamsbridge-Olinville
2             Williamsbridge-Olinville
3                   Woodlawn-Wakefield
4                Westchester-Unionport
5               Mott Haven-Port Morris
6                   Morrisania-Melrose
7                   Morrisania-Melrose
8                  Kingsbridge Heights
9
10  University Heights-Morris Heights
11               Van Cortlandt Village
12     Melrose South-Mott Haven North
13  University Heights-Morris Heights
14                         Parkchester
15                          Co-op City
16
```

[41]:
```
if not evi_df01_s01.empty:
    print("[OK]")
else:
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOUR DATA HAS NOT BEEN REGISTERED WITH ATHENA. LOOK IN␣
 ↪PREVIOUS CELLS TO FIND THE ISSUE.")
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

```
[OK]
```

6.8 Create Athena Table from Local TSV File - NYC _Jobs.tsv

[42]:
```
job_tsv_tbl_name = 'jobs'
job_tsv_field_list = """
job_id string,
agency string,
posting_type string,
num_of_positions string,
business_title string,
civil_service_title string,
title_classification string,
title_code_no string,
```

```
level string,
job_category string,
fulltime_or_parttime_indicator string,
career_level string,
salary_range_from string,
salary_range_to string,
salary_frequency string,
work_location string,
division_or_work_unit string,
job_description string,
minimum_qual_requirements string,
preferred_skills string,
additional_information string,
to_apply string,
hours_or_shift string,
work_location_1 string,
recruitment_contact string,
residency_requirement string,
posting_date string,
post_until string,
posting_updated string,
process_date string
"""
job_tsv_s3_raw_data_path = f"s3://{def_bucket}/team_8_data/raw_data/jobs"
print(job_tsv_s3_raw_data_path)

create_athena_tbl_tsv(conn=conn,
                      db=database_name,
                      tbl_name=job_tsv_tbl_name,
                      fields=job_tsv_field_list,
                      s3_path=job_tsv_s3_raw_data_path,
                      delim='\\t',
                      comp='',
                      skip="'skip.header.line.count'='1'")
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/jobs
Create table statement:

        CREATE EXTERNAL TABLE IF NOT EXISTS ads508_t8.jobs(
job_id string,
agency string,
posting_type string,
num_of_positions string,
business_title string,
civil_service_title string,
title_classification string,
title_code_no string,
level string,
```

```
job_category string,
fulltime_or_parttime_indicator string,
career_level string,
salary_range_from string,
salary_range_to string,
salary_frequency string,
work_location string,
division_or_work_unit string,
job_description string,
minimum_qual_requirements string,
preferred_skills string,
additional_information string,
to_apply string,
hours_or_shift string,
work_location_1 string,
recruitment_contact string,
residency_requirement string,
posting_date string,
post_until string,
posting_updated string,
process_date string
)
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY '\t'
            LINES
                TERMINATED BY '\n'
        LOCATION 's3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/jobs'
        TBLPROPERTIES ('skip.header.line.count'='1')


        tab_name
0         census
1   census_block
2          crime
3      crime_pqt
4      evictions
5  grad_outcomes
6        hs_info
7           jobs

Dataframe contains records: True
```

6.8.1 Run A Sample Query

[43]:
```
job_agency01 = "HOUSING"

job_select_dbn_stmnt = f"""
SELECT * FROM {database_name}.{job_tsv_tbl_name}
WHERE agency LIKE '%{job_agency01}%'
LIMIT 17
"""

print(job_select_dbn_stmnt)

job_df01_s01 = pd.read_sql(job_select_dbn_stmnt,
                           conn)

job_df01_s01.head(17)
```

```
SELECT * FROM ads508_t8.jobs
WHERE agency LIKE '%HOUSING%'
LIMIT 17
```

[43]:

	job_id	agency	posting_type	num_of_positions	business_title
0	573469	HOUSING PRESERVATION & DVLPMNT	External	1	Strategic Program Development Analyst for the …
1	568091	HOUSING PRESERVATION & DVLPMNT	External	5	Case Manager for the Division of Tenant Resources
2	576376	NYC HOUSING AUTHORITY	Internal	1	CARETAKER X
3	571769	HOUSING PRESERVATION & DVLPMNT	Internal	2	Case Manager for the Division of Tenant Resources
4	575854	HOUSING PRESERVATION & DVLPMNT	External	1	Data & Analytics Manager, Division of Strategi…
5	554300	NYC HOUSING AUTHORITY	External	1	
6	575870	HOUSING PRESERVATION & DVLPMNT	Internal	1	
7	440244	NYC HOUSING AUTHORITY	External	1	
8	570222	NYC HOUSING AUTHORITY	Internal	1	
9	576674	NYC HOUSING AUTHORITY	External	1	
10	576328	HOUSING PRESERVATION & DVLPMNT	External	1	
11	566107	HOUSING PRESERVATION & DVLPMNT	External	1	
12	563049	NYC HOUSING AUTHORITY	Internal	1	
13	550123	NYC HOUSING AUTHORITY	Internal	1	
14	576272	HOUSING PRESERVATION & DVLPMNT	External	1	
15	576321	NYC HOUSING AUTHORITY	Internal	1	
16	573720	HOUSING PRESERVATION & DVLPMNT	External	1	

```
5   RESIDENT RELOCATION SERVICES COMMUNITY COORDIN…
6   Director of Manhattan Planning for the Divisio…
7                                      Senior Writer
8   SENIOR PROJECT MANAGER, ADULT EDUCATION & TRAI…
9                     SUPERVISOR OF HOUSING CARETAKER
10  Executive Director of Inclusionary Housing for…
11  Production Specialist for the Division of Tena…
12             SUPERVISING HOUSING GROUNDSKEEPER (HA)
13                                  Treasury Analyst
14  Director of the Supportive Housing Loan Progra…
15              ASSISTANT RESIDENT BUILDING SUPT (HA)
16  Housing Connect Project Manager,  Division of …

               civil_service_title title_classification title_code_no level  \
0          CITY RESEARCH SCIENTIST    Non-Competitive-5         21744    02
1              COMMUNITY ASSOCIATE    Non-Competitive-5         56057    00
2                   CARETAKER (HA)              Labor-3         90645    00
3              COMMUNITY ASSOCIATE    Non-Competitive-5         56057    00
4          CITY RESEARCH SCIENTIST    Non-Competitive-5         21744    02
5            COMMUNITY COORDINATOR    Non-Competitive-5         56058    00
6                     CITY PLANNER        Competitive-1         22122    03
7                  AGENCY ATTORNEY    Non-Competitive-5         30087    03
8   ASSOCIATE JOB OPPORTUNITY SPEC        Competitive-1         52316    02
9   SUPERVISOR OF HOUSING CARETAKE        Competitive-1         82011    00
10  ADMINISTRATIVE PROJECT DIRECTO    Non-Competitive-5         95566    M1
11             COMMUNITY ASSOCIATE    Non-Competitive-5         56057    00
12  SUPERVISING HOUSING GROUNDSKEE        Competitive-1         81350    00
13   ADMINISTRATIVE CLAIM EXAMINER        Competitive-1         1004E    00
14   ASSOCIATE HOUSING DEVELOPMENT        Competitive-1         22508    00
15  ASSISTANT RESIDENT BUILDING SU        Competitive-1         80305    00
16           COMMUNITY COORDINATOR    Non-Competitive-5         56058    00

                                       job_category  …  \
0                       Policy, Research & Analysis  …
1            Constituent Services & Community Programs  …
2                    Building Operations & Maintenance  …
3            Constituent Services & Community Programs  …
4                       Policy, Research & Analysis  …
5            Constituent Services & Community Programs  …
6                Engineering, Architecture, & Planning  …
7            Legal Affairs Policy, Research & Analysis  …
8            Constituent Services & Community Programs  …
9   Building Operations & Maintenance Public Safet…  …
10                  Finance, Accounting, & Procurement  …
11           Constituent Services & Community Programs  …
12                   Building Operations & Maintenance  …
13                  Finance, Accounting, & Procurement  …
```

```
14              Engineering, Architecture, & Planning  …
15                  Building Operations & Maintenance  …
16          Constituent Services & Community Programs  …

                                 additional_information  \
0   We engage New Yorkers to build and sustain nei…
1   Determination and verification of eligibility â ¢
2   Prepare apartments for move outs.   Please rea…
3   Determination and verification of eligibility â ¢
4   We engage New Yorkers to build and sustain nei…
5                                                 "1.
6   We engage New Yorkers to build and sustain nei…
7   Conducting operational analysis to understand …
8                                                 "1.
9                          Handle tenant lockouts.  4.
10  We engage New Yorkers to build and sustain nei…
11  We engage New Yorkers to build and sustain nei…
12  Train others in gardening techniques, grounds …
13  Monitor and project the daily flow of funds & …
14  We engage New Yorkers to build and sustain nei…
15  Monitor inventory supply and arrange for reple…
16  We engage New Yorkers to build and sustain nei…

                                               to_apply  \
0   Continue to work on the implementation of Loca…
1   Client briefings {internal and external meetin…
2   Qualification Requirements  There are no forma…
3   Client briefings {internal and external meetin…
4   Gather, prepare, and merge large datasets from…
5   Preference will be given to employees who have…
6   Planning & Predevelopment (P&P) is central to …
7   Conducting independent research of varying dif…
8   Preference will be given to employees who have…
9   Fill out work orders as a result of apartment …
10  1. A baccalaureate degree from an accredited c…
11  Reviewing recertification packages to ensure a…
12  Ensure the proper maintenance of all assigned …
13  Prepare, approve and release the daily Intra A…
14  In collaboration with the Assistant Commission…
15  Conduct building inspections and follow-up on …
16  Facilitating marketing and compliance meetings…

                                           hours_or_shift  \
0   Retrieve and review affordable housing regulat…
1   May perform community outreach to assist Secti…
2
3   May perform community outreach to assist Secti…
```

```
4   Create performance metrics for lottery and hom…
5           NYCHA residents are encouraged to apply."
6   Neighborhood Development & Stabilization (ND&S…
7   Organizing complex text and processes in seque…
8           NYCHA residents are encouraged to apply."
9   Report any hazardous conditions observed in an…
10  Candidates should have a record of achieving r…
11  Sort, record time/date and assign incoming ele…
12  Check repair work being performed by contracto…
13  Familiarity with Cash Management Systems (see …
14  Managing the development pipeline for SHLP and…
15  Monitor work orders in Maximo and deployment o…
16  Monitoring lotteries and reviewing lottery log…

                                      work_location_1  \
0   Research initiatives in other jurisdictions or…
1   Prepare and send appropriate correspondence, t…
2                                                 "1.
3   Prepare and send appropriate correspondence, t…
4   Manage and analyze eviction filing data to und…
5                          Click the Apply Now button.
6   Promote HPD and City policy objectives across …
7   Leading meetings with subject matter experts t…
8                          Click the Apply Now button.
9   One year of permanent service in the title of …
10  This position is also open to qualified person…
11  Maintain electronic files by uploading relevan…
12  Assistant Resident Building Superintendent in …
13  Prepare, approve and release third party wire …
14  Leading negotiations relating to deal structur…
15  Oversee the repair work done by Maintenance Wo…
16  Reviewing applicantsâ  files and required doc…

                                   recruitment_contact  \
0   Understand and leverage existing Agency datase…
1   Document case files and electronic records, fi…
2   Possession of a valid driver's license is requ…
3   Document case files and electronic records, fi…
4   Prepare analytic reports to inform program des…
5
6   Define, manage, and track team priorities and …
7   Editing documents of a high degree of difficul…
8
9
10                                     Apply online
11  Tracking packages and correspondence for accur…
12  Assist tenants in community development projec…
```

```
13  Responsible for making recurring payments to o…
14  Tracking and reporting of key project issues, …
15  Supervise the preparation of move-outs.  NOTE:…
16  Answering the Housing Connect hotline to assis…

                                residency_requirement  \
0   Summarizing and communicating findingsâ quali…
1                              Rent calculations  â ¢
2   Preference will be given to employees who have…
3                              Rent calculations  â ¢
4   Support the implementation of data-driven prog…
5
6   Meet regularly with individual staff members a…
7   Working with the Compliance Integration Report…
8
9                                                 "1.
10
11  Ensuring documents submitted to unit are distr…
12  1. A four-year high school diploma or its educ…
13  Prepare for more than 10 account activities su…
14  Supervising a team of Project Managers who are…
15  1. One year of permanent service in the title …
16  Reviewing and drafting correspondence in respo…

                                         posting_date  \
0   Contributing to the rollout of new initiatives…
1   Review of yearly recertificationâ s of househ…
2           NYCHA residents are encouraged to apply."
3   Review of yearly recertificationâ s of househ…
4   Develop strategies for data integration and au…
5
6   Identify staffing needs and advocate for resou…
7   Working with the Compliance Monitoring Unit to…
8
9   For NYCHA employees: This position is open as …
10                                    100 Gold Street
11  Entering data into Elite or another unit datab…
12  Three years of satisfactory full-time gardenin…
13  Check the daily Wire Register Reports and prep…
14  Identifying opportunities to train and build o…
15
16  1. A baccalaureate degree from an accredited c…

                                            post_until  \
0   Conducting special research, analytical, or co…
1   Demonstrate ability to manage multiple cases w…
2                         Click the Apply now button.
```

```
3   Demonstrate ability to manage multiple cases w…
4   Assist SOA colleagues with quantitative analys…
5                NYCHA has no residency requirements.
6   Ensure that all projects move efficiently thro…
7   Working with Compliance Inquiry Review and Ass…
8                NYCHA has no residency requirements.
9   For NYCHA employees: Preference will be given …
10
11  Contacting participants to inquire about recer…
12                                                 "1.
13  Prepare daily Reconciliation for the Net Chang…
14  Managing special projects, including developme…
15                                                 "1.
16                                                "â ¢

                                      posting_updated  \
0   Adhering to work plans and internal and extern…
1                          Attend mandatory trainings"
2
3                          Attend mandatory trainings"
4   Respond to ad hoc data requests from programs,…
5                                           10/28/2022
6   Identify risks and troubleshoot problems, invo…
7   Coordinating with NYCHA department heads regar…
8                                           02/14/2023
9           NYCHA residents are encouraged to apply."
10  New York City residency is generally required …
11  Assisting clients with inquiries regarding doc…
12  For NYCHA employees, this position is open as …
13  Work closely with bank personnel to resolve an…
14  Communicating with elected officials, other Ci…
15  For NYCHA employees, this position is open as …
16  Strong analytical ability and attention to det…

                                        process_date
0   Participating in meetings, presentations, and …
1   Qualification Requirements  1. High school gra…
2
3   Qualification Requirements  1. High school gra…
4   1.  For Assignment Level I (only physical, bio…
5
6   Create, implement, and maintain consistent, ef…
7   Drafting complex documents based on important …
8
9                          Click the Apply Now button.
10                                          02/24/2023
11  Meeting with participants to assist with compl…
```

```
12  For NYCHA employees, preference will be given …
13  Assist with procurement of cash management ser…
14  Other responsibilities and initiatives as may …
15  For NYCHA employees, preference will be given …
16  Strong time management skills, demonstrated ab…

[17 rows x 30 columns]
```

[44]:
```
if not job_df01_s01.empty:
    print("[OK]")
else:
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOUR DATA HAS NOT BEEN REGISTERED WITH ATHENA. LOOK IN␣
 ↪PREVIOUS CELLS TO FIND THE ISSUE.")
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

```
[OK]
```

7 Create Parquet Files from TSV Table

[45]:
```
ingest_create_athena_table_parquet_passed = False
```

[46]:
```
%store -r ingest_create_athena_table_tsv_passed
```

[47]:
```
try:
    ingest_create_athena_table_tsv_passed
except NameError:
    print("++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOU HAVE TO RUN ALL PREVIOUS NOTEBOOKS.  You did not␣
 ↪register the TSV Data.")
    print("++++++++++++++++++++++++++++++++++++++++++++++++")
```

[48]:
```
print(ingest_create_athena_table_tsv_passed)
```

```
True
```

[49]:
```
if not ingest_create_athena_table_tsv_passed:
    print("++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOU HAVE TO RUN ALL PREVIOUS NOTEBOOKS.  You did not␣
 ↪register the TSV Data.")
    print("++++++++++++++++++++++++++++++++++++++++++++++++")
else:
    print("[OK]")
```

```
[OK]
```

[50]:
```
# Set S3 path to Parquet data
cri_pqt_s3_data_path = f"s3://{def_bucket}/team_8_data/columnar"
```

8 Execute Statement

[51]:
```
cri_pqt_tbl_name = 'crime_pqt'
drop_pqt_tbl_stmnt = f"""DROP TABLE IF EXISTS {database_name}.
 ↪{cri_pqt_tbl_name}"""

# SQL statement to execute
create_pqt_tble_stmnt = f"""
CREATE TABLE IF NOT EXISTS {database_name}.{cri_pqt_tbl_name}
WITH (
    format = 'PARQUET',
    external_location = '{cri_pqt_s3_data_path}',
    partitioned_by = ARRAY['law_cat_cd', 'borough']
    )
AS
SELECT
    cmplnt_num,
    cmplnt_fr_dt,
    cmplnt_fr_tm,
    cmplnt_to_dt,
    cmplnt_to_tm,
    addr_pct_cd,
    rpt_dt,
    ky_cd,
    ofns_desc,
    pd_cd,
    pd_desc,
    crm_atpt_cptd_cd,
    loc_of_occur_desc,
    prem_typ_desc,
    juris_desc,
    jurisdiction_code,
    parks_nm,
    hadevelopt,
    housing_psa,
    x_coord_cd,
    y_coord_cd,
    susp_age_group,
    susp_race,
    susp_sex,
    transit_district,
    latitude,
    longitude,
    lat_lon,
    patrol_boro,
    station_name,
    vic_age_group,
```

```
    vic_race,
    vic_sex,
    law_cat_cd,
    borough
FROM {database_name}.{cri_tsv_tbl_name}
TABLESAMPLE BERNOULLI(2)
"""

print(f'Create table statement:\n{create_pqt_tble_stmnt}')

pd.read_sql(drop_pqt_tbl_stmnt,
            conn)

pd.read_sql(create_pqt_tble_stmnt,
            conn)
```

```
Create table statement:

CREATE TABLE IF NOT EXISTS ads508_t8.crime_pqt
WITH (
    format = 'PARQUET',
    external_location = 's3://sagemaker-us-
east-1-657724983756/team_8_data/columnar',
    partitioned_by = ARRAY['law_cat_cd', 'borough']
    )
AS
SELECT
    cmplnt_num,
    cmplnt_fr_dt,
    cmplnt_fr_tm,
    cmplnt_to_dt,
    cmplnt_to_tm,
    addr_pct_cd,
    rpt_dt,
    ky_cd,
    ofns_desc,
    pd_cd,
    pd_desc,
    crm_atpt_cptd_cd,
    loc_of_occur_desc,
    prem_typ_desc,
    juris_desc,
    jurisdiction_code,
    parks_nm,
    hadevelopt,
    housing_psa,
    x_coord_cd,
    y_coord_cd,
```

```
    susp_age_group,
    susp_race,
    susp_sex,
    transit_district,
    latitude,
    longitude,
    lat_lon,
    patrol_boro,
    station_name,
    vic_age_group,
    vic_race,
    vic_sex,
    law_cat_cd,
    borough
FROM ads508_t8.crime
TABLESAMPLE BERNOULLI(2)
```

[51]:
```
Empty DataFrame
Columns: [rows]
Index: []
```

9 Load partitions by running MSCK REPAIR TABLE

[52]:
```
partition_pqt_stmnt = f"MSCK REPAIR TABLE {database_name}.{cri_pqt_tbl_name}"

print(partition_pqt_stmnt)
```

```
MSCK REPAIR TABLE ads508_t8.crime_pqt
```

[53]:
```
cri_df02 = pd.read_sql(partition_pqt_stmnt,
                       conn)

cri_df02.head(17)
```

[53]:
```
Empty DataFrame
Columns: []
Index: []
```

10 Show the Partitions

[54]:
```
show_part_stmnt = f"SHOW PARTITIONS {database_name}.{cri_pqt_tbl_name}"

print(show_part_stmnt)
```

```
SHOW PARTITIONS ads508_t8.crime_pqt
```

[55]:
```
cri_df02_part = pd.read_sql(show_part_stmnt,
                            conn)

cri_df02_part.head(31)
```

[55]:

	partition
0	law_cat_cd=MISDEMEANOR/borough=BROOKLYN
1	law_cat_cd=VIOLATION/borough=MANHATTAN
2	law_cat_cd=MISDEMEANOR/borough=__HIVE_DEFAULT_…
3	law_cat_cd=FELONY/borough=__HIVE_DEFAULT_PARTI…
4	law_cat_cd=FELONY/borough=BROOKLYN
5	law_cat_cd=MISDEMEANOR/borough=MANHATTAN
6	law_cat_cd=MISDEMEANOR/borough=STATEN ISLAND
7	law_cat_cd=VIOLATION/borough=QUEENS
8	law_cat_cd=MISDEMEANOR/borough=BRONX
9	law_cat_cd=VIOLATION/borough=__HIVE_DEFAULT_PA…
10	law_cat_cd=FELONY/borough=MANHATTAN
11	law_cat_cd=FELONY/borough=BRONX
12	law_cat_cd=VIOLATION/borough=STATEN ISLAND
13	law_cat_cd=FELONY/borough=STATEN ISLAND
14	law_cat_cd=VIOLATION/borough=BROOKLYN
15	law_cat_cd=FELONY/borough=QUEENS
16	law_cat_cd=VIOLATION/borough=BRONX
17	law_cat_cd=MISDEMEANOR/borough=QUEENS

11 Show the Tables

[56]:
```
show_tbl_stmnt = f"SHOW TABLES in {database_name}"
```

[57]:
```
df_tables = pd.read_sql(show_tbl_stmnt,
                        conn)

df_tables.head(17)
```

[57]:

	tab_name
0	census
1	census_block
2	crime
3	crime_pqt
4	evictions
5	grad_outcomes
6	hs_info
7	jobs

[58]:
```
if cri_pqt_tbl_name in df_tables.values:
    ingest_create_athena_table_parquet_passed = True
```

[59]:
```
%store ingest_create_athena_table_parquet_passed
```

```
Stored 'ingest_create_athena_table_parquet_passed' (bool)
```

12 Run Sample Query

[60]:
```
cri_select_dbn_stmnt02 = f"""
SELECT * FROM {database_name}.{cri_pqt_tbl_name}
WHERE LOWER(law_cat_cd) = '{cri_law_cat_cd01}'
    AND LOWER(borough) = '{cri_borough01}'
LIMIT 17
"""

print(cri_select_dbn_stmnt02)

cri_df02_s01 = pd.read_sql(cri_select_dbn_stmnt02,
                           conn)

cri_df02_s01.head(17)
```

```
SELECT * FROM ads508_t8.crime_pqt
WHERE LOWER(law_cat_cd) = 'misdemeanor'
    AND LOWER(borough) = 'bronx'
LIMIT 17
```

[60]:

	cmplnt_num	cmplnt_fr_dt	cmplnt_fr_tm	cmplnt_to_dt	cmplnt_to_tm	addr_pct_cd	\
0	590499848	07/14/2013	05:30:00	07/14/2013	05:43:00	43	
1	675597625	05/27/2017	21:15:00	05/27/2017	21:25:00	45	
2	336036648	03/21/2012	13:00:00			45	
3	388744844	05/25/2015	04:25:00	05/25/2015	04:30:00	52	
4	706022394	02/07/2016	16:00:00			47	
5	281742597	08/29/2013	00:25:00	08/29/2013	00:30:00	40	
6	411962262	04/09/2013	16:55:00			46	
7	545346999	10/01/2017	18:00:00	10/01/2017	18:15:00	52	
8	818313819	05/30/2017	15:15:00	05/30/2017	15:26:00	46	
9	890406291	03/11/2013	21:25:00	03/12/2013	12:00:00	45	
10	973380835	03/31/2015	19:25:00	03/31/2015	19:34:00	42	
11	625369326	03/28/2016	16:10:00	03/28/2016	16:15:00	40	
12	532182401	04/18/2013	21:30:00			42	
13	251257334	04/26/2016	08:30:00	04/26/2016	08:30:00	44	
14	863725090	03/16/2011	18:45:00	03/16/2011	18:50:00	43	
15	127246599	01/18/2011	01:00:00			48	
16	593364190	06/15/2015	09:45:00			43	

	rpt_dt	ky_cd	ofns_desc	pd_cd	…	latitude	\

```
0   07/14/2013   340                          FRAUDS   707  …  40.823101299
1   05/27/2017   235                 DANGEROUS DRUGS   511  …  40.848632895
2   03/21/2012   351  CRIMINAL MISCHIEF & RELATED OF   254  …  40.830889993
3   05/25/2015   351  CRIMINAL MISCHIEF & RELATED OF   259  …  40.868812402
4   02/10/2016   344    ASSAULT 3 & RELATED OFFENSES   114  …  40.886936175
5   08/29/2013   344    ASSAULT 3 & RELATED OFFENSES   101  …  40.811116426
6   04/09/2013   358        OFFENSES INVOLVING FRAUD   705  …  40.861886273
7   10/02/2017   341                   PETIT LARCENY   339  …  40.865155015
8   05/30/2017   352              CRIMINAL TRESPASS   205  …  40.849496743
9   03/12/2013   359  OFFENSES AGAINST PUBLIC ADMINI   748  …  40.827532802
10  03/31/2015   233                       SEX CRIMES   681  …  40.822569916
11  03/28/2016   344    ASSAULT 3 & RELATED OFFENSES   101  …  40.812053484
12  04/18/2013   359  OFFENSES AGAINST PUBLIC ADMINI   748  …   40.82935201
13  04/26/2016   235                 DANGEROUS DRUGS   511  …   40.82386862
14  03/16/2011   235                 DANGEROUS DRUGS   567  …  40.830447407
15  01/18/2011   359  OFFENSES AGAINST PUBLIC ADMINI   749  …   40.84139516
16  06/15/2015   361   OFF. AGNST PUB ORD SENSBLTY &   639  …  40.836612833

       longitude                      lat_lon        patrol_boro  \
0   -73.869690461  (40.823101299, -73.869690461)  PATROL BORO BRONX
1     -73.8279976    (40.848632895, -73.8279976)  PATROL BORO BRONX
2    -73.82728462   (40.830889993, -73.82728462)  PATROL BORO BRONX
3   -73.888723856  (40.868812402, -73.888723856)  PATROL BORO BRONX
4    -73.85249861   (40.886936175, -73.85249861)  PATROL BORO BRONX
5   -73.927329309  (40.811116426, -73.927329309)  PATROL BORO BRONX
6    -73.89320749   (40.861886273, -73.89320749)  PATROL BORO BRONX
7   -73.892996163  (40.865155015, -73.892996163)  PATROL BORO BRONX
8   -73.909315789  (40.849496743, -73.909315789)  PATROL BORO BRONX
9   -73.821613113  (40.827532802, -73.821613113)  PATROL BORO BRONX
10  -73.911307169  (40.822569916, -73.911307169)  PATROL BORO BRONX
11   -73.90950047   (40.812053484, -73.90950047)  PATROL BORO BRONX
12   -73.89188659    (40.82935201, -73.89188659)  PATROL BORO BRONX
13  -73.919402547   (40.82386862, -73.919402547)  PATROL BORO BRONX
14  -73.875790162  (40.830447407, -73.875790162)  PATROL BORO BRONX
15  -73.885520622   (40.84139516, -73.885520622)  PATROL BORO BRONX
16  -73.864315481  (40.836612833, -73.864315481)  PATROL BORO BRONX

   station_name vic_age_group        vic_race vic_sex   law_cat_cd borough
0                                    UNKNOWN       E  MISDEMEANOR   BRONX
1                     UNKNOWN        UNKNOWN       E  MISDEMEANOR   BRONX
2                       45-64  WHITE HISPANIC       M  MISDEMEANOR   BRONX
3                                    UNKNOWN       D  MISDEMEANOR   BRONX
4                       25-44          BLACK       F  MISDEMEANOR   BRONX
5                       18-24          BLACK       F  MISDEMEANOR   BRONX
6                                    UNKNOWN       D  MISDEMEANOR   BRONX
7                         65+  WHITE HISPANIC       F  MISDEMEANOR   BRONX
8                     UNKNOWN  WHITE HISPANIC       M  MISDEMEANOR   BRONX
```

```
9                       25-44           WHITE       F  MISDEMEANOR   BRONX
10                                    UNKNOWN       E  MISDEMEANOR   BRONX
11                      25-44  WHITE HISPANIC       F  MISDEMEANOR   BRONX
12                      18-24           BLACK       F  MISDEMEANOR   BRONX
13                    UNKNOWN         UNKNOWN       E  MISDEMEANOR   BRONX
14                                    UNKNOWN       E  MISDEMEANOR   BRONX
15                      25-44           BLACK       M  MISDEMEANOR   BRONX
16                      25-44           BLACK       F  MISDEMEANOR   BRONX

[17 rows x 35 columns]
```

[61]:
```
if not cri_df02_s01.empty:
    print("[OK]")
else:
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
    print("[ERROR] YOUR DATA HAS NOT BEEN CONVERTED TO PARQUET. LOOK IN␣
 ↪PREVIOUS CELLS TO FIND THE ISSUE.")
    print("++++++++++++++++++++++++++++++++++++++++++++++++++++++++++")
```

```
[OK]
```

12.1 Review the New Athena Table in the Glue Catalog

[62]:
```
display(
    HTML(
        f'<b>Review <a target="top" href="https://console.aws.amazon.com/glue/
 ↪home?region={region}#">AWS Glue Catalog</a></b>'
    )
)
```

```
<IPython.core.display.HTML object>
```

12.2 Store Variables for the Next Notebooks

[63]:
```
%store
```

```
Stored variables and their in-db values:
balance_dataset                                 -> True
balanced_bias_data_jsonlines_s3_uri             -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
balanced_bias_data_s3_uri                       -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
bias_data_s3_uri                                -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
experiment_name                                 -> 'Amazon-Customer-
Reviews-BERT-Experiment-168013737
feature_group_name                              -> 'reviews-feature-
group-1680137375'
feature_store_offline_prefix                    -> 'reviews-feature-
```

```
store-1680137375'
ingest_create_athena_db_passed                          -> True
ingest_create_athena_table_parquet_passed               -> True
ingest_create_athena_table_tsv_passed                   -> True
max_seq_length                                          -> 64
processed_test_data_s3_uri                              -> 's3://sagemaker-us-
east-1-657724983756/sagemaker-s
processed_train_data_s3_uri                             -> 's3://sagemaker-us-
east-1-657724983756/sagemaker-s
processed_validation_data_s3_uri                        -> 's3://sagemaker-us-
east-1-657724983756/sagemaker-s
raw_input_data_s3_uri                                   -> 's3://sagemaker-us-
east-1-657724983756/amazon-revi
s3_private_path_tsv                                     -> 's3://sagemaker-us-
east-1-657724983756/team_8_data
s3_public_path_tsv                                      -> 's3://sagemaker-us-
east-ads508-sp23-t8'
setup_dependencies_passed                               -> True
setup_iam_roles_passed                                  -> True
setup_instance_check_passed                             -> True
setup_s3_bucket_passed                                  -> True
test_split_percentage                                   -> 0.05
train_split_percentage                                  -> 0.9
trial_name                                              -> 'trial-1680137374'
validation_split_percentage                             -> 0.05
```

12.3 Release Resources

[64]:
```
%%html

<p><b>Shutting down your kernel for this notebook to release resources.</b></p>
<button class="sm-command-button" data-commandlinker-command="kernelmenu:
 ↪shutdown" style="display:none;">Shutdown Kernel</button>

<script>
try {
    els = document.getElementsByClassName("sm-command-button");
    els[0].click();
}
catch(err) {
    // NoOp
}
</script>
```

```
<IPython.core.display.HTML object>
```

[65]:
```
%%javascript
```

```
try {
    Jupyter.notebook.save_checkpoint();
    Jupyter.notebook.session.delete();
}
catch(err) {
    // NoOp
}
```

```
<IPython.core.display.Javascript object>
```

00b_S3_Setup_Final

April 14, 2023

1 ADS-508-01-SP23 Team 8: Final Project

2 Setup Database and Athena Tables

Much of the code is modified from `Fregly, C., & Barth, A. (2021). Data science on AWS: Implementing end-to-end, continuous AI and machine learning pipelines. O'Reilly.`

2.1 Install missing dependencies

PyAthena is a Python DB API 2.0 (PEP 249) compliant client for Amazon Athena.

[2]:
```
!pip install --disable-pip-version-check -q PyAthena==2.1.0
```

```
WARNING: The directory '/root/.cache/pip' or its parent directory is not
owned or is not writable by the current user. The cache has been disabled. Check
the permissions and owner of that directory. If executing pip with sudo, you
should use sudo's -H flag.
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
```

2.2 Globally import libraries

[3]:
```
import boto3
from botocore.client import ClientError
import sagemaker
import pandas as pd
from pyathena import connect
from IPython.core.display import display, HTML

%matplotlib inline
```

2.3 Instantiate AWS SageMaker session

[4]:
```
session = boto3.session.Session()
region = session.region_name
sagemaker_session = sagemaker.Session()
def_bucket = sagemaker_session.default_bucket()
bucket = 'sagemaker-us-east-ads508-sp23-t8'

s3 = boto3.Session().client(service_name="s3",
                            region_name=region)
```

[5]:
```
print(f"Default bucket: {def_bucket}")
print(f"Public T8 bucket: {bucket}")
```

```
Default bucket: sagemaker-us-east-1-657724983756
Public T8 bucket: sagemaker-us-east-ads508-sp23-t8
```

2.4 Create Athena Database Table

[6]:
```
database_name = "ads508_t8"
```

[7]:
```
# Set S3 staging directory -- this is a temporary directory used for Athena␣
 ↪queries
s3_staging_dir = f"s3://{def_bucket}/team_8_data/athena/staging"
print(s3_staging_dir)
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/athena/staging
```

[8]:
```
conn = connect(region_name=region,
               s3_staging_dir=s3_staging_dir)
```

2.5 Define custom function to create tables in existing database

[9]:
```
def create_athena_tbl_tsv(conn=None,
                          db=None,
                          tbl_name=None,
                          fields='',
                          s3_path=None,
                          delim=',',
                          ret='',
                          comp='',
                          skip=''):
    # Set Athena parameters

    # SQL statement to execute
    drop_tsv_tbl_stmnt = f"""DROP TABLE IF EXISTS {db}.{tbl_name}"""

    create_tsv_tbl_stmnt = f"""
```

```
        CREATE EXTERNAL TABLE IF NOT EXISTS {db}.{tbl_name}({fields})
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY '{delim}'
            LINES
                TERMINATED BY '{ret}\\n'
        LOCATION '{s3_path}'
        TBLPROPERTIES ({comp}{skip})
        """

    print(f'Create table statement:\n{create_tsv_tbl_stmnt}')

    pd.read_sql(drop_tsv_tbl_stmnt,
                conn)

    pd.read_sql(create_tsv_tbl_stmnt,
                conn)

    # Verify The Table Has Been Created Succesfully
    show_tsv_tbl_stmnt = f"SHOW TABLES IN {db}"

    df_show = pd.read_sql(show_tsv_tbl_stmnt,
                          conn)
    display(df_show.head(17))

    if tbl_name in df_show.values:
        ingest_create_athena_table_tsv_passed = True

    print(f'\nDataframe contains records:␣
↪{ingest_create_athena_table_tsv_passed}')
```

2.6 Create Athena Table from Local TSV File - census_block_loc.csv

2.6.1 Dataset columns

[10]:
```
ceb_tsv_tbl_name = 'census_block'
ceb_tsv_field_list = """
latitude double,
longitude double,
blockCode string,
county string
"""
ceb_tsv_s3_raw_data_path = f"s3://{def_bucket}/team_8_data/raw_data/
↪census_block"
print(ceb_tsv_s3_raw_data_path)

create_athena_tbl_tsv(conn=conn,
```

```
                        db=database_name,
                        tbl_name=ceb_tsv_tbl_name,
                        fields=ceb_tsv_field_list,
                        s3_path=ceb_tsv_s3_raw_data_path,
                        comp='',
                        skip="'skip.header.line.count'='1'")
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/raw_data/census_block
Create table statement:

        CREATE EXTERNAL TABLE IF NOT EXISTS ads508_t8.census_block(
latitude double,
longitude double,
blockCode string,
county string
)
        ROW FORMAT DELIMITED
            FIELDS
                TERMINATED BY ','
            LINES
                TERMINATED BY '\n'
        LOCATION 's3://sagemaker-us-
east-1-657724983756/team_8_data/raw_data/census_block'
        TBLPROPERTIES ('skip.header.line.count'='1')


        tab_name
0         census
1   census_block
2          crime
3      crime_pqt
4      evictions
5  grad_outcomes
6        hs_info
7           jobs


Dataframe contains records: True
```

2.6.2 Run A Sample Query

[11]:
```
ceb_select_dbn_stmnt01 = f"""
SELECT
    substr(blockCode,1,11) AS blockCode,
    count(*),
    min(latitude) AS min_lat,
    max(latitude) AS max_lat,
    min(longitude) AS min_long,
```

```
    max(longitude) AS max_long
FROM {database_name}.{ceb_tsv_tbl_name}
GROUP BY substr(blockCode,1,11)
ORDER BY count(*) DESC
LIMIT 50000
"""

print(ceb_select_dbn_stmnt01)

ceb_df01_s01 = pd.read_sql(ceb_select_dbn_stmnt01,
                           conn)

print(ceb_df01_s01.shape)
display(ceb_df01_s01.head(15))
```

```
SELECT
    substr(blockCode,1,11) AS blockCode,
    count(*),
    min(latitude) AS min_lat,
    max(latitude) AS max_lat,
    min(longitude) AS min_long,
    max(longitude) AS max_long
FROM ads508_t8.census_block
GROUP BY substr(blockCode,1,11)
ORDER BY count(*) DESC
LIMIT 50000

(2995, 6)
```

	blockCode	_col1	min_lat	max_lat	min_long	max_long
0	36081990100	1816	40.491307	40.584020	-74.039397	-73.757638
1	36085990100	1198	40.480000	40.604372	-74.257839	-74.036231
2	34025990000	917	40.480000	40.525226	-74.093216	-73.887437
3	36059990400	690	40.534271	40.579497	-73.767136	-73.650000
4	36059301000	412	40.819196	40.877990	-73.751307	-73.653166
5	36081107202	366	40.586281	40.645075	-73.852613	-73.767136
6	36047070203	327	40.579497	40.642814	-73.890603	-73.833618
7	34017012700	305	40.712915	40.776231	-74.143869	-74.077387
8	34013980200	297	40.674472	40.715176	-74.200854	-74.115377
9	36081071600	286	40.622462	40.663166	-73.830452	-73.748141
10	34039035400	275	40.593065	40.640553	-74.261005	-74.200854
11	36047990100	260	40.552362	40.604372	-74.039397	-73.928593
12	36059300100	252	40.798844	40.841809	-73.773467	-73.713317
13	34039039800	251	40.645075	40.688040	-74.197688	-74.140704
14	36005050400	240	40.839548	40.884774	-73.820955	-73.751307

2.7 Review the New Athena Table in the Glue Catalog

[12]:
```
display(
    HTML(
        f'<b>Review <a target="top" href="https://console.aws.amazon.com/glue/
 ↪home?region={region}#">AWS Glue Catalog</a></b>'
    )
)
```

```
<IPython.core.display.HTML object>
```

2.8 Store Variables for the Next Notebooks

[13]:
```
%store
```

```
Stored variables and their in-db values:
balance_dataset                                 -> True
balanced_bias_data_jsonlines_s3_uri             -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
balanced_bias_data_s3_uri                       -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
bias_data_s3_uri                                -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
experiment_name                                 -> 'Amazon-Customer-
Reviews-BERT-Experiment-168013737
feature_group_name                              -> 'reviews-feature-
group-1680137375'
feature_store_offline_prefix                    -> 'reviews-feature-
store-1680137375'
ingest_create_athena_db_passed                  -> True
ingest_create_athena_table_parquet_passed       -> True
ingest_create_athena_table_tsv_passed           -> True
max_seq_length                                  -> 64
processed_test_data_s3_uri                      -> 's3://sagemaker-us-
east-1-657724983756/sagemaker-s
processed_train_data_s3_uri                     -> 's3://sagemaker-us-
east-1-657724983756/sagemaker-s
processed_validation_data_s3_uri                -> 's3://sagemaker-us-
east-1-657724983756/sagemaker-s
raw_input_data_s3_uri                           -> 's3://sagemaker-us-
east-1-657724983756/amazon-revi
s3_private_path_tsv                             -> 's3://sagemaker-us-
east-1-657724983756/team_8_data
s3_public_path_tsv                              -> 's3://sagemaker-us-
east-ads508-sp23-t8'
setup_dependencies_passed                       -> True
setup_iam_roles_passed                          -> True
setup_instance_check_passed                     -> True
setup_s3_bucket_passed                          -> True
```

```
test_split_percentage                                -> 0.05
train_split_percentage                               -> 0.9
trial_name                                           -> 'trial-1680137374'
validation_split_percentage                          -> 0.05
```

2.9 Release Resources

[14]:
```
%%html

<p><b>Shutting down your kernel for this notebook to release resources.</b></p>
<button class="sm-command-button" data-commandlinker-command="kernelmenu:
 ↪shutdown" style="display:none;">Shutdown Kernel</button>

<script>
try {
    els = document.getElementsByClassName("sm-command-button");
    els[0].click();
}
catch(err) {
    // NoOp
}
</script>
```

```
<IPython.core.display.HTML object>
```

[15]:
```
%%javascript

try {
    Jupyter.notebook.save_checkpoint();
    Jupyter.notebook.session.delete();
}
catch(err) {
    // NoOp
}
```

```
<IPython.core.display.Javascript object>
```

01_Preprocess_and_EDA_Final

April 14, 2023

1 ADS-508-01-SP23 Team 8: Final Project

2 Exploratory Data Analysis (EDA)

Much of the code is modified from `Fregly, C., & Barth, A. (2021). Data science on AWS: Implementing end-to-end, continuous AI and machine learning pipelines. O'Reilly.`

2.1 Install missing dependencies

PyAthena is a Python DB API 2.0 (PEP 249) compliant client for Amazon Athena.

[2]:
```
!pip install --disable-pip-version-check -q PyAthena==2.1.0
!pip install missingno
```

```
WARNING: The directory '/root/.cache/pip' or its parent directory is not
owned or is not writable by the current user. The cache has been disabled. Check
the permissions and owner of that directory. If executing pip with sudo, you
should use sudo's -H flag.
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
WARNING: The directory '/root/.cache/pip' or its parent directory is
not owned or is not writable by the current user. The cache has been disabled.
Check the permissions and owner of that directory. If executing pip with sudo,
you should use sudo's -H flag.
Collecting missingno
  Downloading missingno-0.5.2-py3-none-any.whl (8.7 kB)
Requirement already satisfied: seaborn in /opt/conda/lib/python3.7/site-packages
(from missingno) (0.10.0)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages
(from missingno) (1.21.6)
Requirement already satisfied: scipy in /opt/conda/lib/python3.7/site-packages
```

```
(from missingno) (1.4.1)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.7/site-
packages (from missingno) (3.1.3)
Requirement already satisfied: kiwisolver>=1.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (1.1.0)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.7/site-
packages (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.8.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.4.6)
Requirement already satisfied: pandas>=0.22.0 in /opt/conda/lib/python3.7/site-
packages (from seaborn->missingno) (1.3.5)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages
(from cycler>=0.10->matplotlib->missingno) (1.14.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.7/site-
packages (from kiwisolver>=1.0.1->matplotlib->missingno) (59.3.0)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.22.0->seaborn->missingno) (2019.3)
Installing collected packages: missingno
Successfully installed missingno-0.5.2
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

2.2 Globally import libraries

[3]:
```
import boto3
from botocore.client import ClientError
from IPython.core.display import display, HTML
import pandas as pd
import numpy as np
from pyathena import connect
import matplotlib.pyplot as plt
import missingno as msno
import sagemaker
import seaborn as sns
from sklearn.feature_selection import VarianceThreshold
from scipy import stats
from scipy.stats import skew

%matplotlib inline
```

2.3 Instantiate AWS SageMaker session

[4]:
```
session = boto3.session.Session()
region = session.region_name
sagemaker_session = sagemaker.Session()
def_bucket = sagemaker_session.default_bucket()
bucket = 't8-test-final'
role = sagemaker.get_execution_role()

s3 = boto3.Session().client(service_name="s3", region_name=region)
```

[5]:
```
print(f"Default bucket: {def_bucket}")
print(f"Public T8 bucket: {bucket}")
```

```
Default bucket: sagemaker-us-east-1-657724983756
Public T8 bucket: t8-test-final
```

[6]:
```
# Set S3 staging directory -- this is a temporary directory used for Athena␣
 ↪queries
s3_staging_dir = f"s3://{def_bucket}/team_8_data/athena/staging"
print(s3_staging_dir)
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/athena/staging
```

[7]:
```
conn = connect(region_name=region,
               s3_staging_dir=s3_staging_dir)
```

[8]:
```
database_name = "ads508_t8"
```

2.4 Explore DB tables

2.4.1 census

[9]:
```
cen_tsv_tbl_name = 'census'
```

Explore via SQL SELECT statements

[10]:
```
# Run query to review a sample of records
cen_borough01 = "bronx"

cen_select_borough_stmnt01 = f"""
SELECT * FROM {database_name}.{cen_tsv_tbl_name}
WHERE LOWER(borough) = '{cen_borough01}'
LIMIT 11
"""

# Display SQL statement
print(cen_select_borough_stmnt01)

# Run SQL statement against Athena table
```

```
cen_df01_s01 = pd.read_sql(cen_select_borough_stmnt01,
                           conn)

# Display results
cen_df01_s01.head(11)
```

```
SELECT * FROM ads508_t8.census
WHERE LOWER(borough) = 'bronx'
LIMIT 11
```

[10]:

	censustract	county	borough	totalpop	men	women	hispanic	white	black	\
0	36005000100	Bronx	Bronx	7703	7133	570	29.9	6.1	60.9	
1	36005000200	Bronx	Bronx	5403	2659	2744	75.8	2.3	16.0	
2	36005000400	Bronx	Bronx	5915	2896	3019	62.7	3.6	30.7	
3	36005001600	Bronx	Bronx	5879	2558	3321	65.1	1.6	32.4	
4	36005001900	Bronx	Bronx	2591	1206	1385	55.4	9.0	29.0	
5	36005002000	Bronx	Bronx	8516	3301	5215	61.1	1.6	31.1	
6	36005002300	Bronx	Bronx	4774	2130	2644	62.3	0.2	36.5	
7	36005002400	Bronx	Bronx	150	109	41	0.0	52.0	48.0	
8	36005002500	Bronx	Bronx	5355	2338	3017	76.5	1.5	18.9	
9	36005002701	Bronx	Bronx	3016	1375	1641	68.0	0.0	31.2	
10	36005002702	Bronx	Bronx	4778	2427	2351	71.3	1.6	26.2	

	native	…	walk	othertransp	workathome	meancommute	employed	\
0	0.2	…	NaN	NaN	NaN	NaN	0	
1	0.0	…	2.9	0.0	0.0	43.0	2308	
2	0.0	…	1.4	0.5	2.1	45.0	2675	
3	0.0	…	8.6	1.6	1.7	38.8	2120	
4	0.0	…	3.0	2.4	6.2	45.4	1083	
5	0.3	…	4.3	1.0	0.0	46.0	2508	
6	1.0	…	14.0	1.5	4.1	42.7	1191	
7	0.0	…	0.0	0.0	0.0	NaN	113	
8	0.0	…	17.7	1.8	2.7	35.5	1691	
9	0.0	…	18.0	0.0	1.6	42.8	1102	
10	0.0	…	7.1	0.7	0.5	44.0	1559	

	privatework	publicwork	selfemployed	familywork	unemployment
0	NaN	NaN	NaN	NaN	NaN
1	80.8	16.2	2.9	0.0	7.7
2	71.7	25.3	2.5	0.6	9.5
3	75.0	21.3	3.8	0.0	8.7
4	76.8	15.5	7.7	0.0	19.2
5	71.0	21.3	7.7	0.0	17.2
6	74.2	16.1	9.7	0.0	18.9
7	62.8	37.2	0.0	0.0	0.0

```
8          85.1         8.3          6.1         0.5          9.4
9          86.9         8.5          4.5         0.0         15.2
10         75.0        14.0         11.0         0.0         10.6

[11 rows x 36 columns]
```

Perform aggregated summaries

[11]:
```
# Run query to review a sample of records
cen_select_hispanic_stmnt01 = f"""
SELECT DISTINCT
    hispanic,
    COUNT(*)
FROM {database_name}.{cen_tsv_tbl_name}
WHERE hispanic IS NULL
GROUP BY hispanic
LIMIT 10
"""

# Display SQL statement
print(cen_select_hispanic_stmnt01)

# Run SQL statement against Athena table
cen_df01_s02 = pd.read_sql(cen_select_hispanic_stmnt01,
                           conn)

# Display results
cen_df01_s02.head(11)
```

```
SELECT DISTINCT
    hispanic,
    COUNT(*)
FROM ads508_t8.census
WHERE hispanic IS NULL
GROUP BY hispanic
LIMIT 10
```

[11]:

	hispanic	_col1
0	None	39

[12]:
```
cen_summ_borough_stmnt01 = f"""
SELECT
    borough,
    COUNT(*) AS ctract_count,
    SUM(totalpop) AS bor_pop,
    SUM(round(totalpop*hispanic/100,0))/SUM(totalpop) AS hispanic_perc,
```

```
    SUM(round(totalpop*white/100,0))/SUM(totalpop) AS white_perc,
    SUM(round(totalpop*black/100,0))/SUM(totalpop) AS black_perc,
    SUM(round(totalpop*native/100,0))/SUM(totalpop) AS native_perc,
    SUM(round(totalpop*asian/100,0))/SUM(totalpop) AS asian_perc,
    SUM(round(totalpop*childpoverty/100,0))/SUM(totalpop) AS child_poverty_perc,
    SUM(round(totalpop*income,0))/SUM(totalpop) AS income_avg
FROM {database_name}.{cen_tsv_tbl_name}
GROUP BY borough
LIMIT 100
"""

# Display SQL statement
print(cen_summ_borough_stmnt01)

# Run SQL statement against Athena table
cen_df01_s03 = pd.read_sql(cen_summ_borough_stmnt01,
                           conn)
# Display results
cen_df01_s03.head(11)
```

```
SELECT
    borough,
    COUNT(*) AS ctract_count,
    SUM(totalpop) AS bor_pop,
    SUM(round(totalpop*hispanic/100,0))/SUM(totalpop) AS hispanic_perc,
    SUM(round(totalpop*white/100,0))/SUM(totalpop) AS white_perc,
    SUM(round(totalpop*black/100,0))/SUM(totalpop) AS black_perc,
    SUM(round(totalpop*native/100,0))/SUM(totalpop) AS native_perc,
    SUM(round(totalpop*asian/100,0))/SUM(totalpop) AS asian_perc,
    SUM(round(totalpop*childpoverty/100,0))/SUM(totalpop) AS child_poverty_perc,
    SUM(round(totalpop*income,0))/SUM(totalpop) AS income_avg
FROM ads508_t8.census
GROUP BY borough
LIMIT 100
```

[12]:

	borough	ctract_count	bor_pop	hispanic_perc	white_perc	\
0	Bronx	339	1428357	0.546306	0.102867	
1	Queens	669	2301139	0.279454	0.261341	
2	Brooklyn	761	2595259	0.196219	0.357155	
3	Manhattan	288	1629507	0.257999	0.471131	
4	Staten Island	110	472481	0.178469	0.628205	

	black_perc	native_perc	asian_perc	child_poverty_perc	income_avg
0	0.295901	0.002247	0.035982	0.394679	36822
1	0.173913	0.002189	0.242256	0.195747	59851

2	0.311851	0.001728	0.114031	0.293884	51694
3	0.127395	0.001357	0.115712	0.198270	78003
4	0.095900	0.001179	0.079440	0.155067	73993

[13]:
```
cen_summ_borough_stmnt01 = f"""
SELECT
    censustract,
    COUNT(*) AS ctract_count,
    SUM(totalpop) AS bor_pop,
    SUM(round(totalpop*hispanic/100,0))/SUM(totalpop) AS hispanic_perc,
    SUM(round(totalpop*white/100,0))/SUM(totalpop) AS white_perc,
    SUM(round(totalpop*black/100,0))/SUM(totalpop) AS black_perc,
    SUM(round(totalpop*native/100,0))/SUM(totalpop) AS native_perc,
    SUM(round(totalpop*asian/100,0))/SUM(totalpop) AS asian_perc,
    SUM(round(totalpop*childpoverty/100,0))/SUM(totalpop) AS child_poverty_perc,
    SUM(round(totalpop*income,0))/SUM(totalpop) AS income_avg
FROM {database_name}.{cen_tsv_tbl_name}
GROUP BY censustract
LIMIT 100
"""

# Display SQL statement
print(cen_summ_borough_stmnt01)

# Run SQL statement against Athena table
cen_df01_s04 = pd.read_sql(cen_summ_borough_stmnt01,
                           conn)
# Display results
cen_df01_s04.head(11)
```

```
SELECT
    censustract,
    COUNT(*) AS ctract_count,
    SUM(totalpop) AS bor_pop,
    SUM(round(totalpop*hispanic/100,0))/SUM(totalpop) AS hispanic_perc,
    SUM(round(totalpop*white/100,0))/SUM(totalpop) AS white_perc,
    SUM(round(totalpop*black/100,0))/SUM(totalpop) AS black_perc,
    SUM(round(totalpop*native/100,0))/SUM(totalpop) AS native_perc,
    SUM(round(totalpop*asian/100,0))/SUM(totalpop) AS asian_perc,
    SUM(round(totalpop*childpoverty/100,0))/SUM(totalpop) AS child_poverty_perc,
    SUM(round(totalpop*income,0))/SUM(totalpop) AS income_avg
FROM ads508_t8.census
GROUP BY censustract
LIMIT 100
```

[13]:

	censustract	ctract_count	bor_pop	hispanic_perc	white_perc	black_perc	native_perc	asian_perc	child_poverty_perc	income_avg
0	36005000100	1	7703	0.298974	0.061015	0.608984	0.001947	0.015968	NaN	NaN
1	36005002500	1	5355	0.765079	0.014939	0.188982	0.000000	0.030065	0.624090	17226.0
2	36005005100	1	5653	0.696975	0.001061	0.280913	0.021051	0.000000	0.701044	18633.0
3	36005005902	1	3131	0.703928	0.017886	0.230917	0.025870	0.013095	0.183009	38542.0
4	36005006700	1	7421	0.662983	0.018057	0.294030	0.004986	0.000000	0.670934	21016.0
5	36005007400	1	3635	0.727923	0.015131	0.255021	0.000000	0.001926	0.406052	44455.0
6	36005007600	1	5587	0.637014	0.032934	0.194022	0.015035	0.093073	0.502953	33465.0
7	36005011000	1	153	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	NaN
8	36005011700	1	1603	0.721148	0.021210	0.258890	0.000000	0.000000	0.421709	25483.0
9	36005011800	1	5364	0.200969	0.730984	0.016965	0.000000	0.041946	0.112975	83077.0
10	36005012102	1	1835	0.819074	0.009264	0.160763	0.000000	0.000000	0.609809	20028.0

Load potential predictors and target for further exploration using pandas

[14]:
```
cen_box_stmnt01 = f"""
SELECT
    borough,
    totalpop,
    men,
    women,
    hispanic,
    white,
    black,
    native,
    asian,
    citizen,
    income,
    poverty,
    childpoverty,
    professional,
    service,
    office,
    construction,
    production,
```

```
    drive,
    carpool,
    transit,
    walk,
    othertransp,
    workathome,
    meancommute,
    employed,
    privatework,
    publicwork,
    selfemployed,
    familywork,
    unemployment
FROM {database_name}.{cen_tsv_tbl_name}
WHERE childpoverty IS NOT NULL
LIMIT 5000
"""

# Display SQL statement
print(cen_box_stmnt01)

# Run SQL statement against Athena table
cen_df01_s05 = pd.read_sql(cen_box_stmnt01,
                           conn)

# Display results
cen_df01_s05.head(11)
```

```
SELECT
    borough,
    totalpop,
    men,
    women,
    hispanic,
    white,
    black,
    native,
    asian,
    citizen,
    income,
    poverty,
    childpoverty,
    professional,
    service,
    office,
    construction,
    production,
```

```
    drive,
    carpool,
    transit,
    walk,
    othertransp,
    workathome,
    meancommute,
    employed,
    privatework,
    publicwork,
    selfemployed,
    familywork,
    unemployment
FROM ads508_t8.census
WHERE childpoverty IS NOT NULL
LIMIT 5000
```

[14]:

	borough	totalpop	men	women	hispanic	white	black	native	asian	\
0	Bronx	5403	2659	2744	75.8	2.3	16.0	0.0	4.2	
1	Bronx	5915	2896	3019	62.7	3.6	30.7	0.0	0.3	
2	Bronx	5879	2558	3321	65.1	1.6	32.4	0.0	0.0	
3	Bronx	2591	1206	1385	55.4	9.0	29.0	0.0	2.1	
4	Bronx	8516	3301	5215	61.1	1.6	31.1	0.3	3.3	
5	Bronx	4774	2130	2644	62.3	0.2	36.5	1.0	0.0	
6	Bronx	150	109	41	0.0	52.0	48.0	0.0	0.0	
7	Bronx	5355	2338	3017	76.5	1.5	18.9	0.0	3.0	
8	Bronx	3016	1375	1641	68.0	0.0	31.2	0.0	0.0	
9	Bronx	4778	2427	2351	71.3	1.6	26.2	0.0	0.0	
10	Bronx	5299	2292	3007	23.0	0.2	71.4	0.0	1.7	

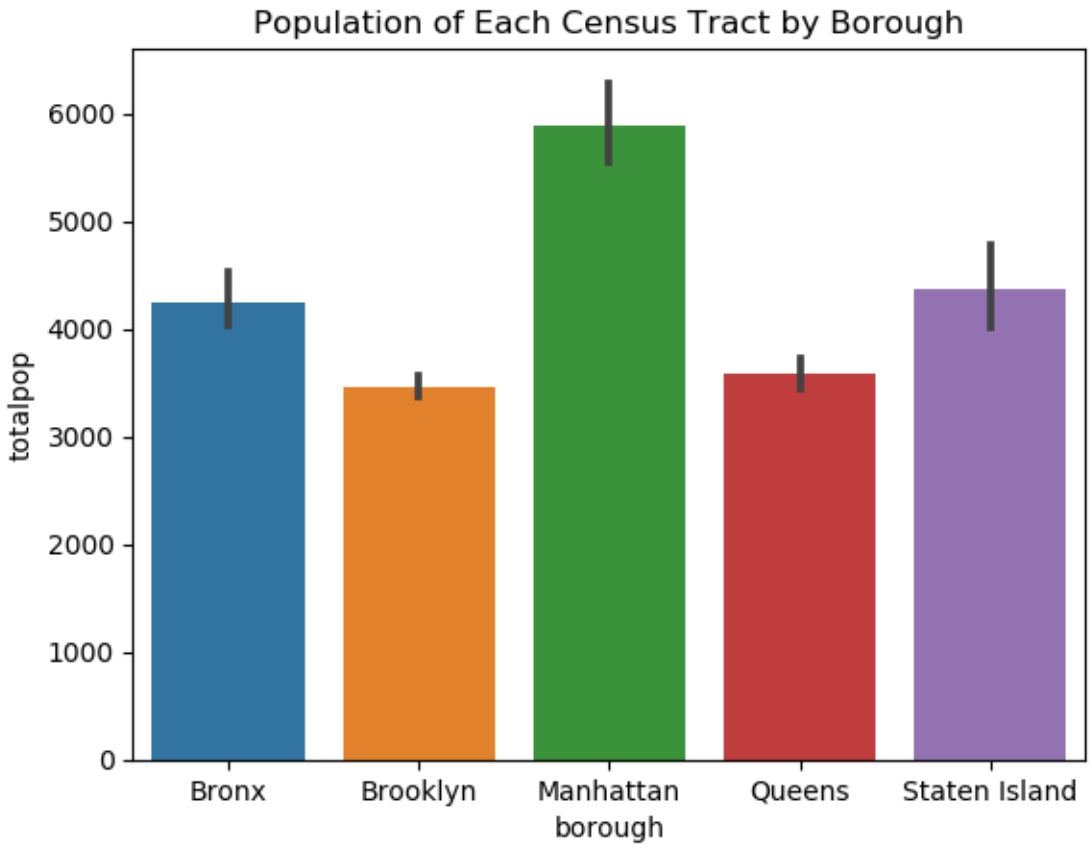
	citizen	…	walk	othertransp	workathome	meancommute	employed	\
0	3639	…	2.9	0.0	0.0	43.0	2308	
1	4100	…	1.4	0.5	2.1	45.0	2675	
2	3536	…	8.6	1.6	1.7	38.8	2120	
3	1557	…	3.0	2.4	6.2	45.4	1083	
4	5436	…	4.3	1.0	0.0	46.0	2508	
5	3056	…	14.0	1.5	4.1	42.7	1191	
6	41	…	0.0	0.0	0.0	NaN	113	
7	2509	…	17.7	1.8	2.7	35.5	1691	
8	1456	…	18.0	0.0	1.6	42.8	1102	
9	2365	…	7.1	0.7	0.5	44.0	1559	
10	4056	…	2.0	0.6	2.7	47.3	2394	

	privatework	publicwork	selfemployed	familywork	unemployment
0	80.8	16.2	2.9	0.0	7.7
1	71.7	25.3	2.5	0.6	9.5

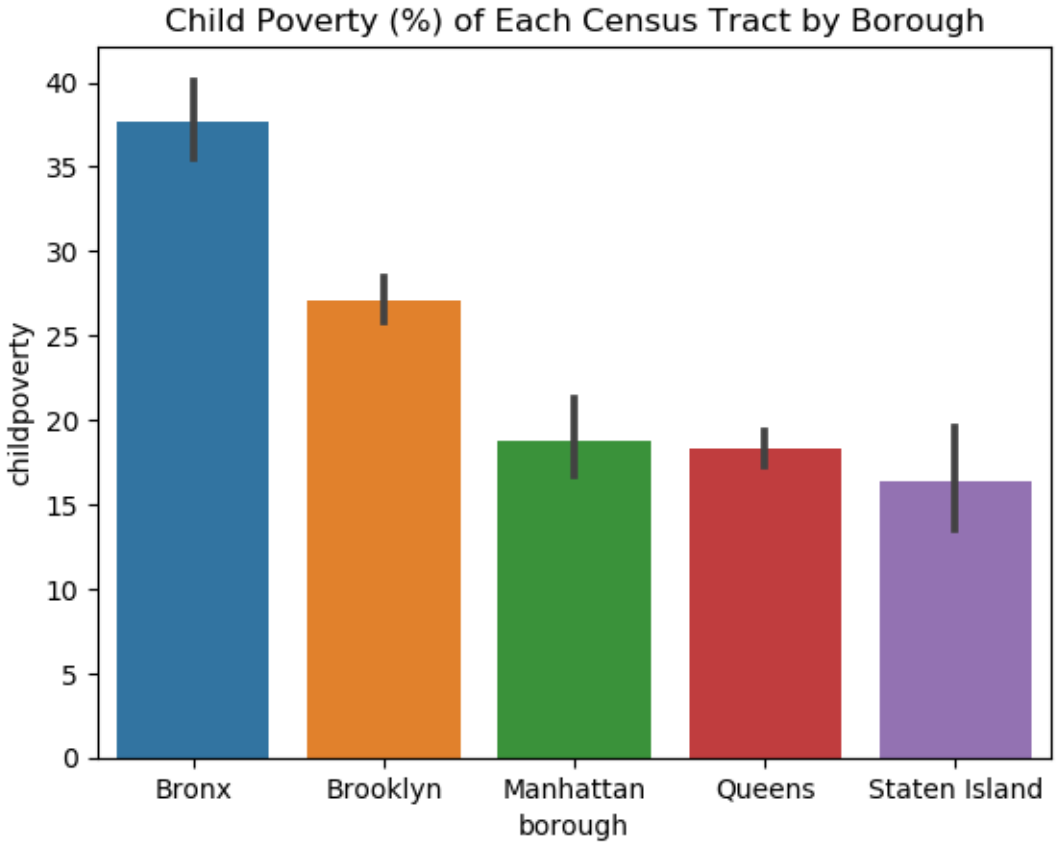
```
2          75.0        21.3          3.8         0.0          8.7
3          76.8        15.5          7.7         0.0         19.2
4          71.0        21.3          7.7         0.0         17.2
5          74.2        16.1          9.7         0.0         18.9
6          62.8        37.2          0.0         0.0          0.0
7          85.1         8.3          6.1         0.5          9.4
8          86.9         8.5          4.5         0.0         15.2
9          75.0        14.0         11.0         0.0         10.6
10         61.9        37.4          0.6         0.0         12.8

[11 rows x 31 columns]
```

Barplots for select features

[15]:
```
popbar = sns.barplot(x='borough',
                     y='totalpop',
                     data=cen_df01_s05).set(title='Population of Each Census␣
↪Tract by Borough')
```



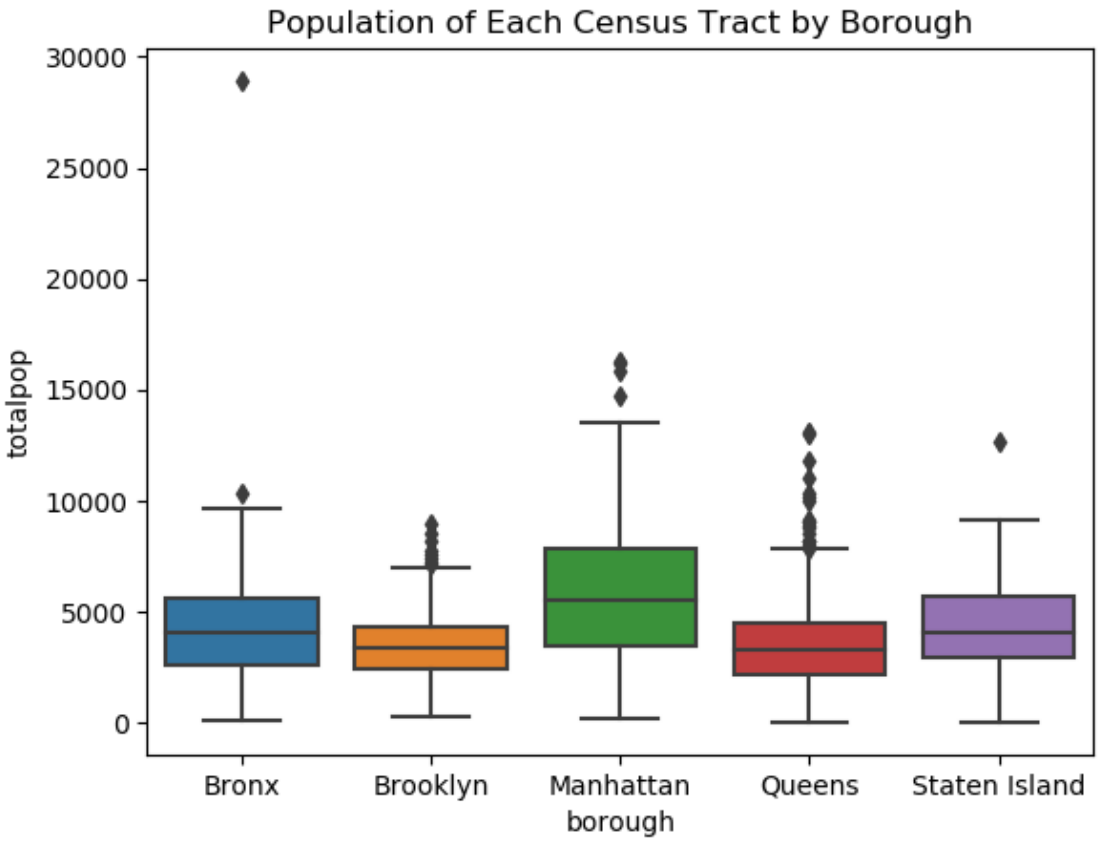
[16]:
```
%matplotlib inline
pov_bar = sns.barplot(x='borough',
                      y='childpoverty',
                      data=cen_df01_s05).set(title='Child Poverty (%) of Each␣
 ↪Census Tract by Borough')
```



Display boxplots for select features

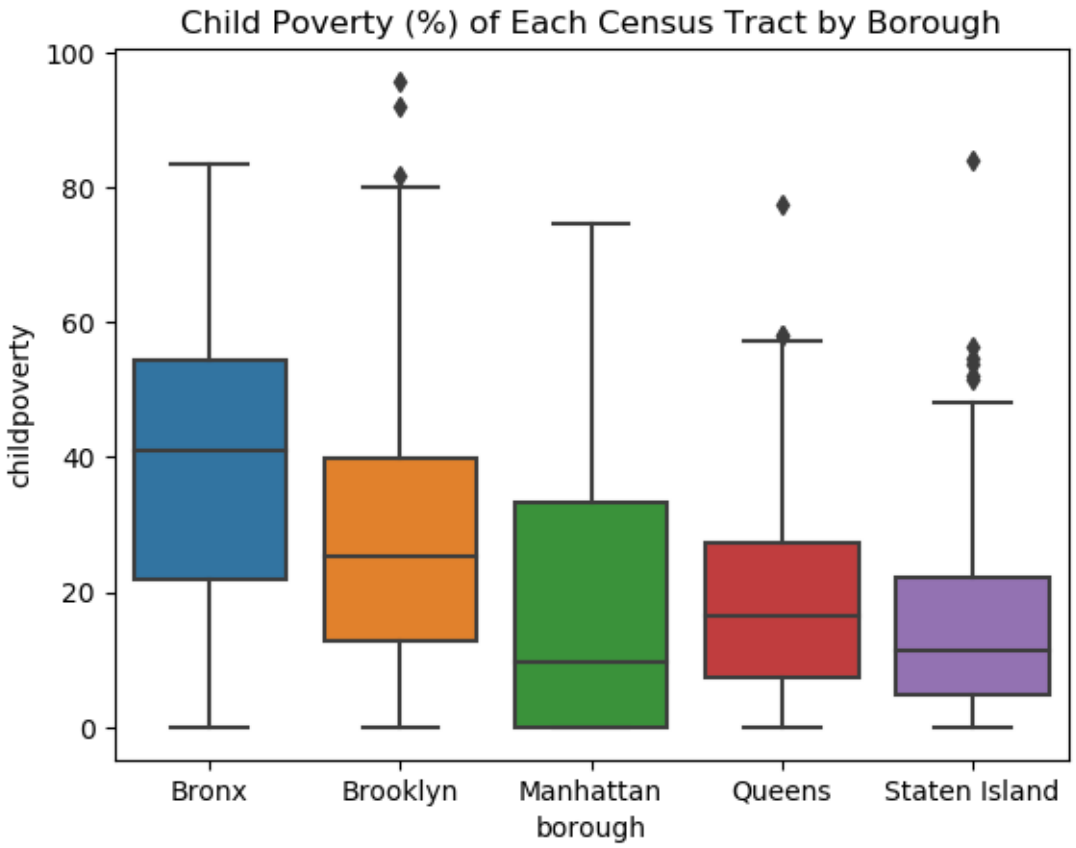
[17]:
```
%matplotlib inline
sns.boxplot(x='borough',
            y='totalpop',
            data=cen_df01_s05).set(title='Population of Each Census Tract by␣
 ↪Borough')
```

[17]: [Text(0.5, 1.0, 'Population of Each Census Tract by Borough')]

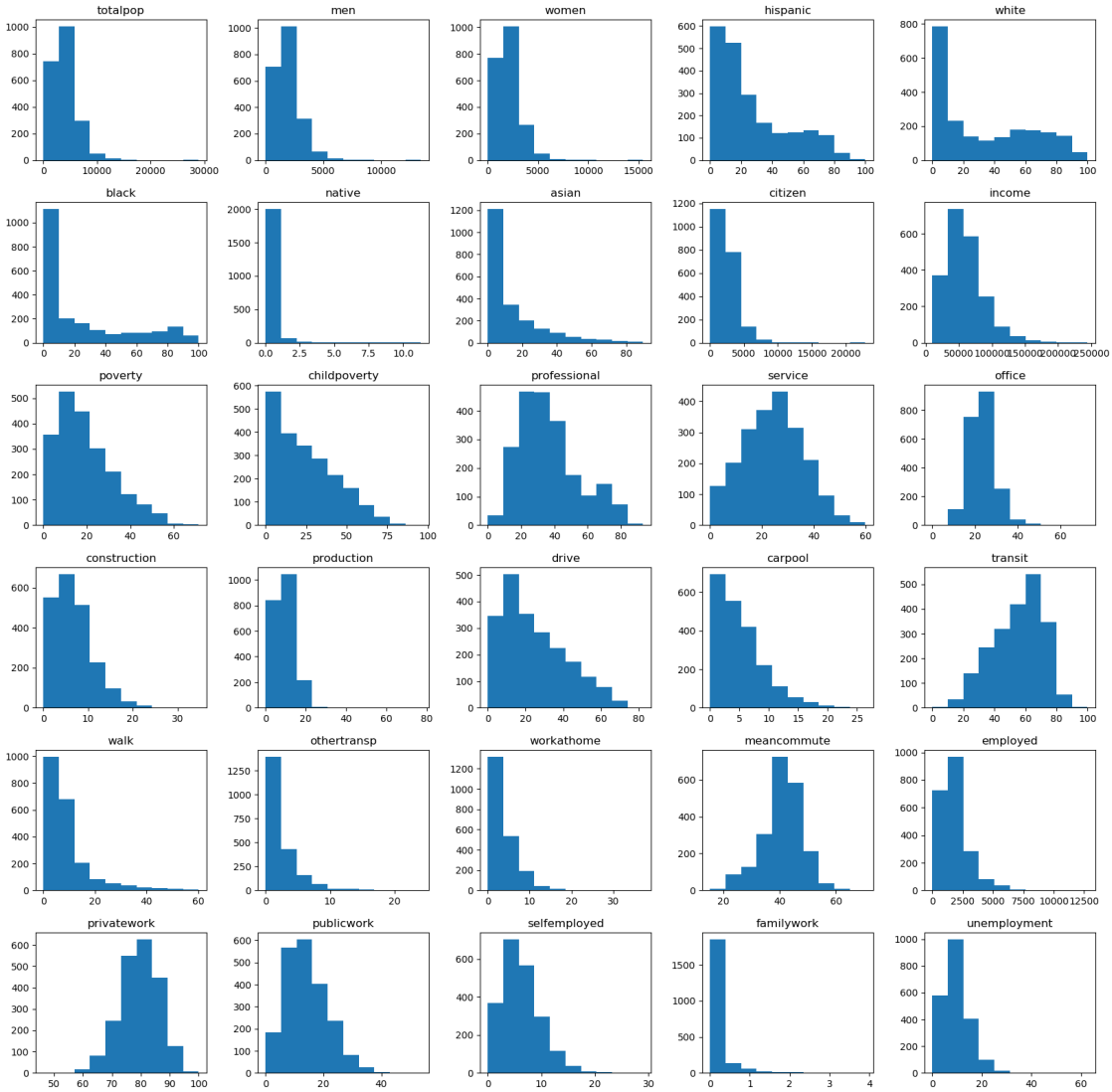


[18]:
```
%matplotlib inline
sns.boxplot(x='borough',
            y='childpoverty',
            data=cen_df01_s05).set(title='Child Poverty (%) of Each Census␣
 ↪Tract by Borough')
```

[18]:
```
[Text(0.5, 1.0, 'Child Poverty (%) of Each Census Tract by Borough')]
```



[19]:
```
# histograms
cen_df01_s05.hist(grid=False, figsize=(20,20))
plt.show()
```



Create subsets of columns for various purposes

[20]:
```
cen_df01_s05_num_lst01 = ['totalpop',
                          'men',
                          'women',
                          'hispanic',
                          'white',
                          'black',
                          'native',
                          'asian',
                          'citizen',
                          'income',
                          'poverty',
                          'childpoverty',
```

```
                            'professional',
                            'service',
                            'office',
                            'construction',
                            'production',
                            'drive',
                            'carpool',
                            'transit',
                            'walk',
                            'othertransp',
                            'workathome',
                            'meancommute',
                            'employed',
                            'privatework',
                            'publicwork',
                            'selfemployed',
                            'familywork',
                            'unemployment'
                           ]

cen_df01_s05_num_lst02 = ['totalpop',
                            'women',
                            'hispanic',
                            'black',
                            'native',
                            'asian',
                            'citizen',
                            'income',
                            'poverty',
                            'childpoverty',
                            'professional',
                            'service',
                            'office',
                            'construction',
                            'meancommute',
                            'employed',
                            'unemployment'
                           ]

cen_df02_s01 = cen_df01_s05[cen_df01_s05_num_lst01]
cen_df03_s01 = cen_df01_s05[cen_df01_s05_num_lst02]

display(cen_df02_s01.head(5))
```

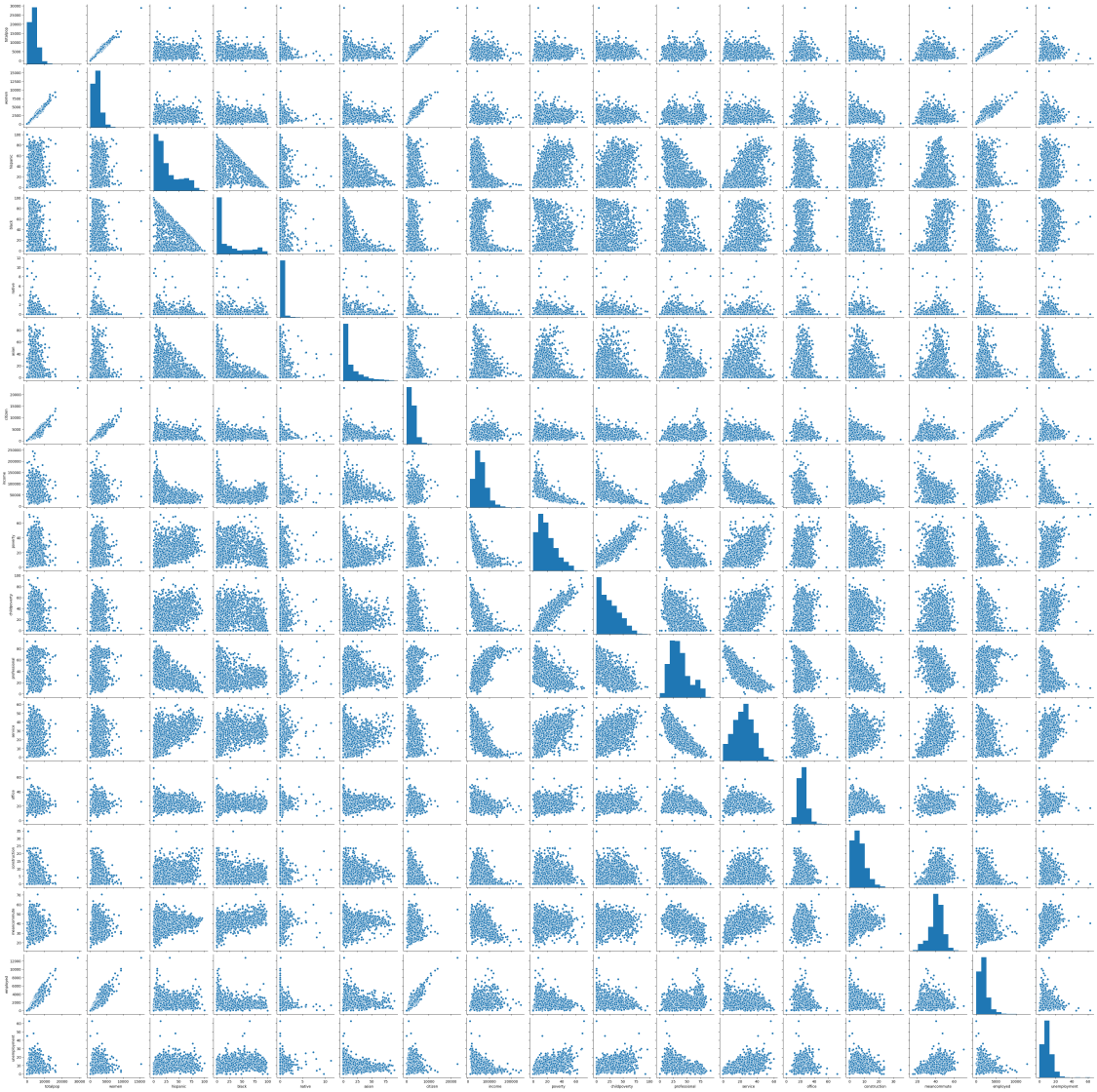
```
   totalpop   men  women  hispanic  white  black  native  asian  citizen  \
0      5403  2659   2744      75.8    2.3   16.0     0.0    4.2     3639
1      5915  2896   3019      62.7    3.6   30.7     0.0    0.3     4100
2      5879  2558   3321      65.1    1.6   32.4     0.0    0.0     3536
```

```
3      2591  1206   1385      55.4    9.0   29.0     0.0    2.1      1557
4      8516  3301   5215      61.1    1.6   31.1     0.3    3.3      5436

    income  …  walk  othertransp  workathome  meancommute  employed  \
0  72034.0  …   2.9          0.0         0.0         43.0      2308
1  74836.0  …   1.4          0.5         2.1         45.0      2675
2  32312.0  …   8.6          1.6         1.7         38.8      2120
3  37936.0  …   3.0          2.4         6.2         45.4      1083
4  18086.0  …   4.3          1.0         0.0         46.0      2508

   privatework  publicwork  selfemployed  familywork  unemployment
0         80.8        16.2           2.9         0.0           7.7
1         71.7        25.3           2.5         0.6           9.5
2         75.0        21.3           3.8         0.0           8.7
3         76.8        15.5           7.7         0.0          19.2
4         71.0        21.3           7.7         0.0          17.2

[5 rows x 30 columns]
```

Scatterplots of numerical features

[21]:
```
# Pair scatter plots for selected features
sns.pairplot(cen_df03_s01)
```

[21]:
```
<seaborn.axisgrid.PairGrid at 0x7fba3e117250>
```



Summary stats for select features

[22]:
```
sum_stat_pov = pd.DataFrame(cen_df01_s05['childpoverty'].describe()).T
sum_stat_pop = pd.DataFrame(cen_df01_s05['totalpop'].describe()).T
sum_stat_emp = pd.DataFrame(cen_df01_s05['employed'].describe()).T
sum_stat = pd.concat([sum_stat_pov, sum_stat_pop, sum_stat_emp])
sum_stat
```

[22]:

	count	mean	std	min	25%	50%	75%	\
childpoverty	2107.0	24.483816	18.948203	0.0	8.3	21.1	37.55	
totalpop	2107.0	3991.738016	2178.239166	15.0	2451.0	3622.0	5003.50	
employed	2107.0	1864.503085	1157.310756	11.0	1098.0	1612.0	2297.50	

	max

```
childpoverty     95.7
totalpop      28926.0
employed      12780.0
```

Outliers for select features

[23]:
```
# Child Poverty
IQR = sum_stat_pov['75%'][0] - sum_stat_pov['25%'][0]
low_outlier = sum_stat_pov['25%'][0] - 1.5*(IQR)
high_outlier = sum_stat_pov['75%'][0] + 1.5*(IQR)
print('Low Outlier (childpoverty):', low_outlier)
print('High Outlier (childpoverty):', high_outlier)

# Population
IQR_pop = sum_stat_pop['75%'][0] - sum_stat_pop['25%'][0]
low_outlier_pop = sum_stat_pop['25%'][0] - 1.5*(IQR_pop)
high_outlier_pop = sum_stat_pop['75%'][0] + 1.5*(IQR_pop)
print('Low Outlier (population):', low_outlier_pop)
print('High Outlier (population):', high_outlier_pop)
```

```
Low Outlier (childpoverty): -35.57499999999999
High Outlier (childpoverty): 81.42499999999998
Low Outlier (population): -1377.75
High Outlier (population): 8832.25
```

[24]:
```
# Identify presence of outliers
outliers_pov = cen_df01_s05.loc[(cen_df01_s05['childpoverty'] < low_outlier) |␣
 ↪(cen_df01_s05['childpoverty'] > high_outlier),
                            ['borough', 'childpoverty', 'totalpop']]
if outliers_pov.empty:
    print('No outliers found in childpoverty column')
else:
    print('Outliers found in childpoverty column ({0}):'.
 ↪format(len(outliers_pov)))
    print(outliers_pov)

outliers_pop = cen_df01_s05.loc[(cen_df01_s05['totalpop'] < low_outlier) |␣
 ↪(cen_df01_s05['totalpop'] > high_outlier),
                            ['borough', 'childpoverty', 'totalpop']]
if outliers_pop.empty:
    print('No outliers found in totalpop column')
else:
    print('Outliers found in totalpop column ({0}):'.format(len(outliers_pop)))
    print(outliers_pop)
```

```
Outliers found in childpoverty column (5):
           borough  childpoverty  totalpop
140          Bronx          83.4      2176
566       Brooklyn          81.7      2626
```

```
622        Brooklyn          92.0      1035
976        Brooklyn          95.7      6094
2043  Staten Island          84.0      1531
Outliers found in totalpop column (2102):
            borough  childpoverty  totalpop
0             Bronx          20.7      5403
1             Bronx          23.6      5915
2             Bronx          35.9      5879
3             Bronx          31.5      2591
4             Bronx          67.7      8516
…               …             …         …
2102  Staten Island          11.1      4895
2103  Staten Island          27.8      6279
2104  Staten Island          51.4      2550
2105  Staten Island          53.8      4611
2106  Staten Island          16.5      1131

[2102 rows x 3 columns]
```

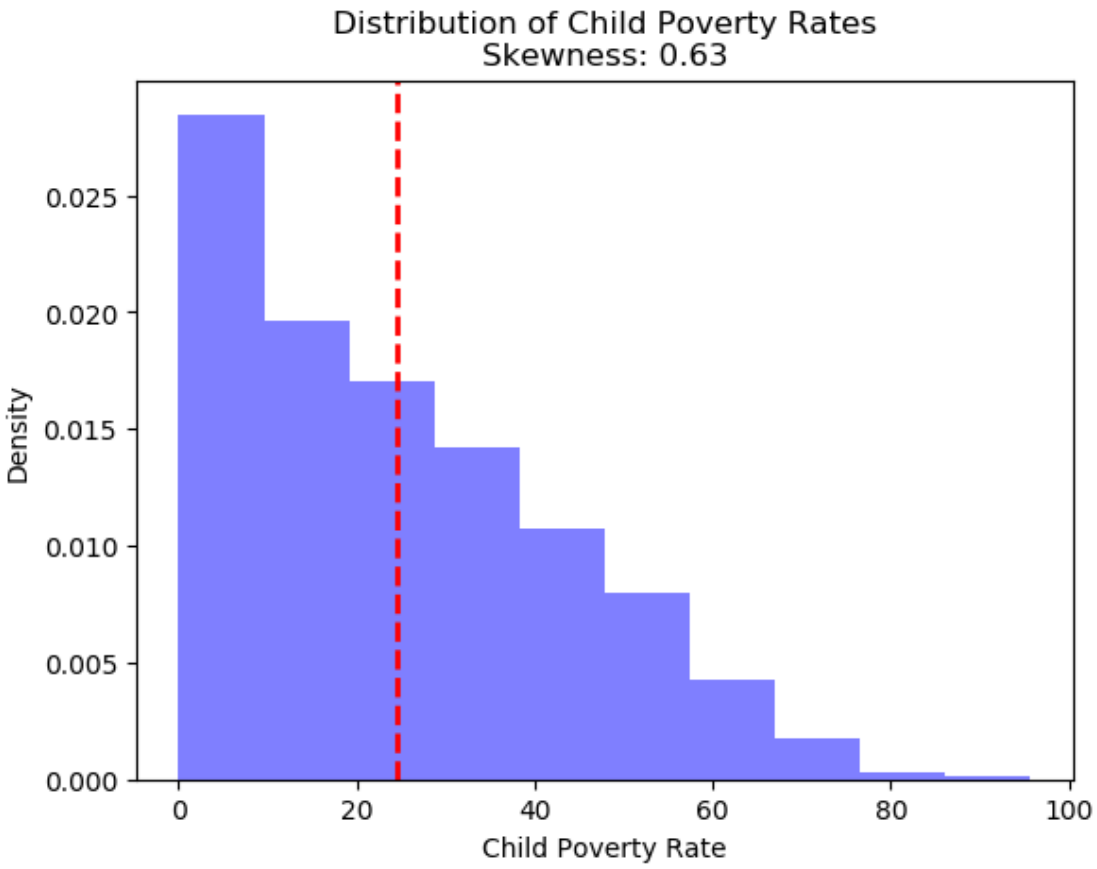
There are a few outliers, especially in our population data. However, these outliers still give a key insight to the differences between communities.

Determine Skew for select features

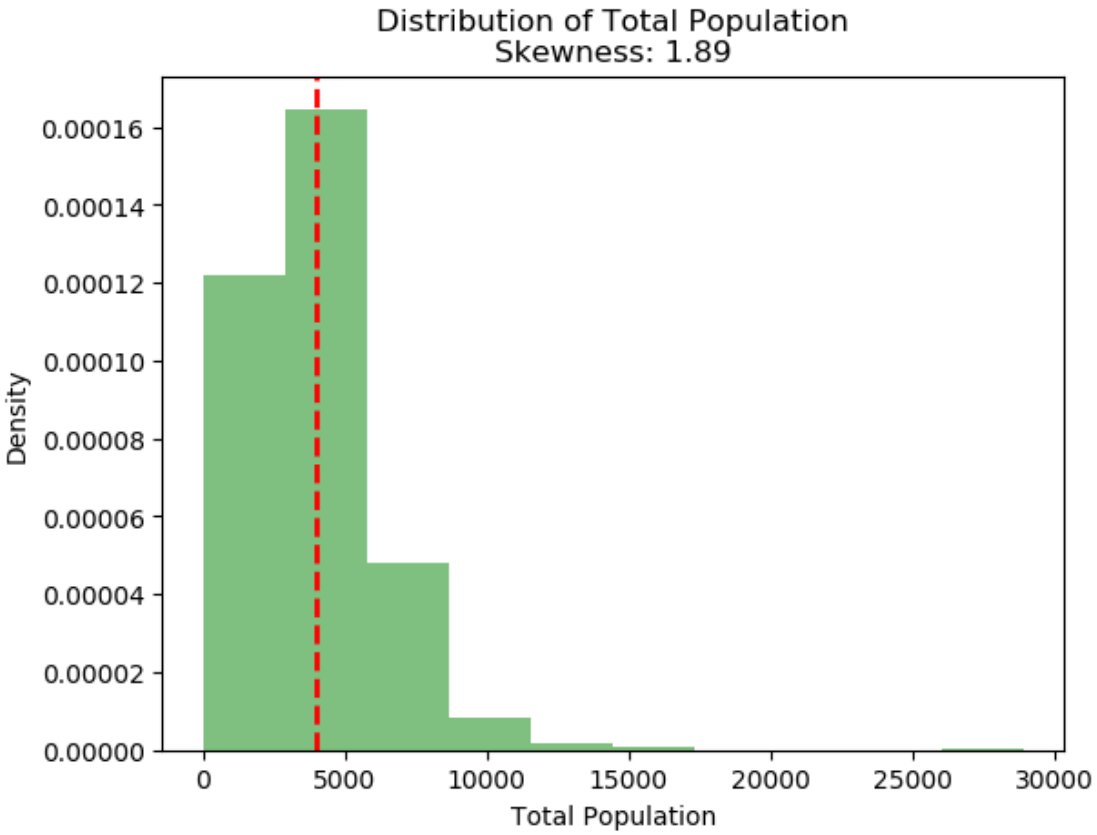
[25]:
```
# For 'childpoverty' column
childpoverty_skew = skew(cen_df01_s05['childpoverty'])
print('Skewness of childpoverty column:', childpoverty_skew)

# For 'population' column
totalpop_skew = skew(cen_df01_s05['totalpop'])
print('Skewness of population column:', totalpop_skew)
```

```
Skewness of childpoverty column: 0.6278449442145481
Skewness of population column: 1.888443026275476
```

[26]:
```
childpoverty_skew = skew(cen_df01_s05['childpoverty'])
plt.hist(cen_df01_s05['childpoverty'], density=True, alpha=0.5, color='blue')
plt.title('Distribution of Child Poverty Rates\nSkewness: {0:.2f}'.
 ↪format(childpoverty_skew))
plt.xlabel('Child Poverty Rate')
plt.ylabel('Density')
plt.axvline(x=cen_df01_s05['childpoverty'].mean(), color='red',␣
 ↪linestyle='dashed', linewidth=2)
plt.show()
```



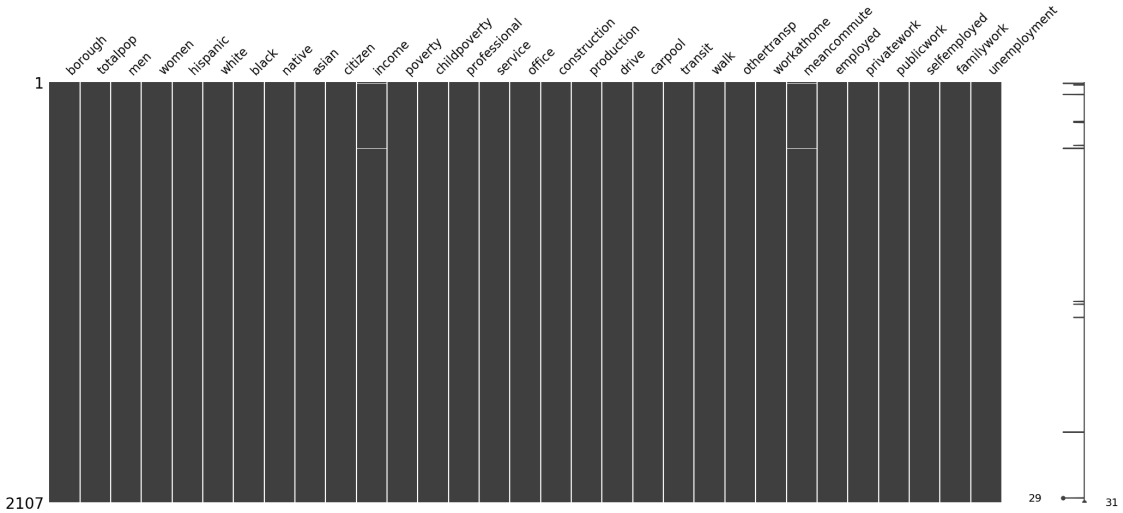
[27]:
```
totalpop_skew = skew(cen_df01_s05['totalpop'])
plt.hist(cen_df01_s05['totalpop'], density=True, alpha=0.5, color='green')
plt.title('Distribution of Total Population\nSkewness: {0:.2f}'.
  ↪format(totalpop_skew))
plt.xlabel('Total Population')
plt.ylabel('Density')
plt.axvline(x=cen_df01_s05['totalpop'].mean(), color='red', linestyle='dashed',␣
  ↪linewidth=2)
plt.show()
```



Examine features with missing values

[28]:
```
# Visualize missing values in each column
msno.matrix(cen_df01_s05)
```

[28]: <matplotlib.axes._subplots.AxesSubplot at 0x7fba26a65950>



[29]:
```
# Remove any features for which the number of null vals exceed a threshold--
#-- (5% of total N)
cen_df01_s05_null_summ01 = pd.DataFrame(cen_df01_s05.isnull().sum(),
                                        columns=['null_count'])

cen_df01_s05_null_summ02 = cen_df01_s05_null_summ01.
 ↪loc[(cen_df01_s05_null_summ01['null_count'] != 0)].sort_values('null_count',
 ↪                                      ascending=False)
cen_df01_s05_null_summ03 = cen_df01_s05_null_summ02.reset_index()
print(cen_df01_s05_null_summ03)

cen_df01_s05_null_summ04 = cen_df01_s05_null_summ03.
 ↪loc[cen_df01_s05_null_summ03['null_count'] > (len(cen_df01_s05)*.05)]
print('\n', cen_df01_s05_null_summ04)

cen_df01_s05_null_summ04_remove_lst01 = list(cen_df01_s05_null_summ04['index'])
print('\n', cen_df01_s05_null_summ04_remove_lst01)
```

```
         index  null_count
0       income          10
1  meancommute           7

 Empty DataFrame
Columns: [index, null_count]
Index: []

 []
```

Examine featues with near zero variances

[30]:
```
# Review near-zero variance (NZV) features for possible removal
cen_df02_s01_nzv_fit = VarianceThreshold().fit(cen_df02_s01)
cen_df02_s01_nzv_vc01 = cen_df02_s01_nzv_fit.transform(cen_df02_s01)

# Get the names of the selected features
cen_df02_s01_nzv_fit_select_features = cen_df02_s01.
 ↪columns[cen_df02_s01_nzv_fit.get_support()]

cen_df02_s01_nzv_df01 = pd.DataFrame(cen_df02_s01_nzv_vc01,
 ↪columns=cen_df02_s01_nzv_fit_select_features)

display(cen_df02_s01_nzv_df01.head(5))
print(f'NZV transformed matrix dimensions = {cen_df02_s01_nzv_df01.shape}')
```

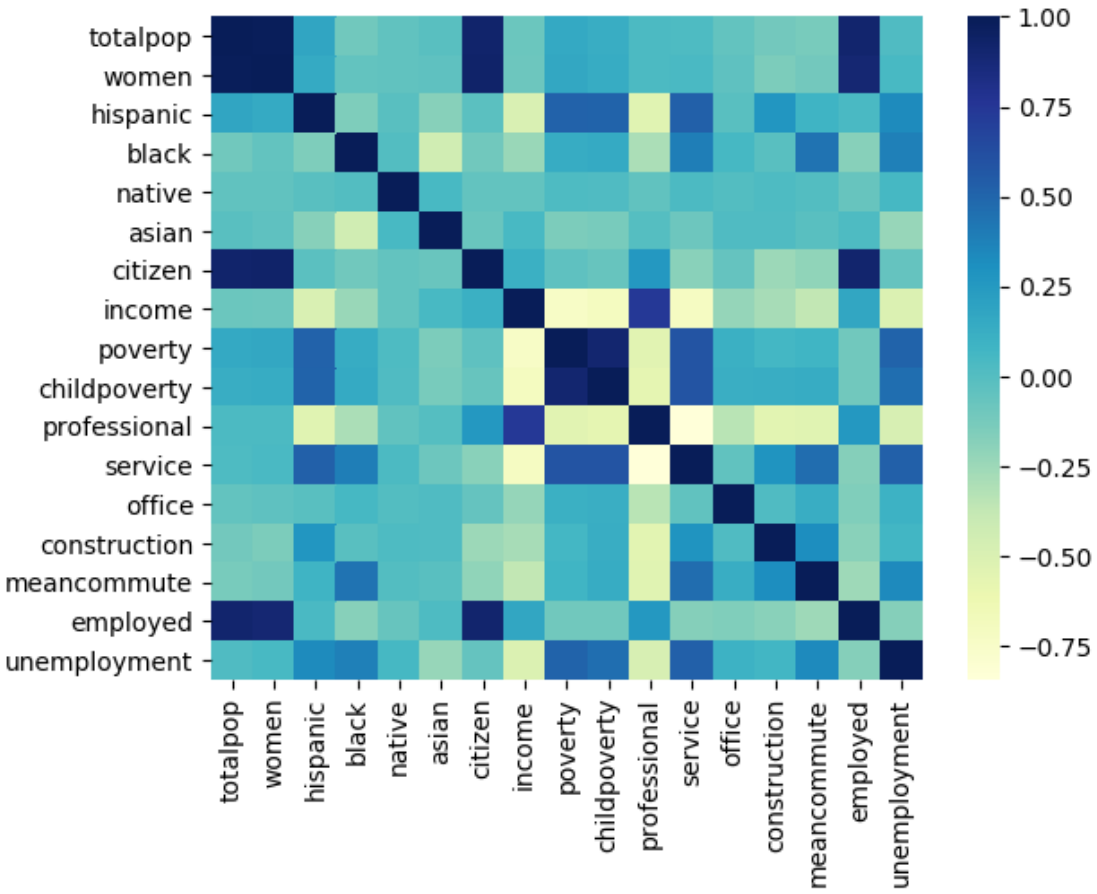
```
print(f'\n{cen_df02_s01.shape[1] - cen_df02_s01_nzv_df01.shape[1]} near zero␣
 ↪variance features were eliminated')
```

```
   totalpop     men   women  hispanic  white  black  native  asian  citizen  \
0    5403.0  2659.0  2744.0      75.8    2.3   16.0     0.0    4.2   3639.0
1    5915.0  2896.0  3019.0      62.7    3.6   30.7     0.0    0.3   4100.0
2    5879.0  2558.0  3321.0      65.1    1.6   32.4     0.0    0.0   3536.0
3    2591.0  1206.0  1385.0      55.4    9.0   29.0     0.0    2.1   1557.0
4    8516.0  3301.0  5215.0      61.1    1.6   31.1     0.3    3.3   5436.0

    income  …  walk  othertransp  workathome  meancommute  employed  \
0  72034.0  …   2.9          0.0         0.0         43.0    2308.0
1  74836.0  …   1.4          0.5         2.1         45.0    2675.0
2  32312.0  …   8.6          1.6         1.7         38.8    2120.0
3  37936.0  …   3.0          2.4         6.2         45.4    1083.0
4  18086.0  …   4.3          1.0         0.0         46.0    2508.0

   privatework  publicwork  selfemployed  familywork  unemployment
0         80.8        16.2           2.9         0.0           7.7
1         71.7        25.3           2.5         0.6           9.5
2         75.0        21.3           3.8         0.0           8.7
3         76.8        15.5           7.7         0.0          19.2
4         71.0        21.3           7.7         0.0          17.2

[5 rows x 30 columns]
```

```
NZV transformed matrix dimensions = (2107, 30)

0 near zero variance features were eliminated
```

Correlations

[31]:
```
# Calculate correlation matrix
cen_corr_matrix = cen_df01_s05[cen_df01_s05_num_lst02].corr()

# Generate heatmap plot
sns.heatmap(cen_corr_matrix, cmap="YlGnBu")

# Show plot
plt.show()
cen_top_corr = cen_corr_matrix['childpoverty'].sort_values(ascending=False)[1:6]
for column in cen_top_corr.index:
    for index in cen_corr_matrix.index:
        correlation = cen_corr_matrix.loc[index, column]
        if correlation > 0.8 and correlation < 1.0:
            print(f"{index} - {column}: {correlation}")
cen_top_corr = cen_corr_matrix.nlargest(4, 'childpoverty')['childpoverty'][1:]
print(cen_top_corr)
```




```
childpoverty - poverty: 0.9111736833012234
poverty     0.911174
service     0.593196
hispanic    0.516168
Name: childpoverty, dtype: float64
```

2.4.2 crime_pqt

[32]:
```
cri_pqt_tbl_name = 'crime_pqt'
```

Explore via SQL SELECT statements

[33]:
```
# Run query to review a sample of records
cri_law_cat_cd01 = "misdemeanor"
cri_bourough01 = "bronx"

cri_select_borough_stmnt01 = f"""
SELECT * FROM {database_name}.{cri_pqt_tbl_name}
WHERE LOWER(law_cat_cd) = '{cri_law_cat_cd01}'
    AND LOWER(borough) = '{cri_bourough01}'
```

```
LIMIT 11
"""

# Display SQL statement
print(cri_select_borough_stmnt01)

# Run SQL statement against Athena table
cri_df01_s01 = pd.read_sql(cri_select_borough_stmnt01,
                           conn)

# Display results
cri_df01_s01.head(5)
```

```
SELECT * FROM ads508_t8.crime_pqt
WHERE LOWER(law_cat_cd) = 'misdemeanor'
    AND LOWER(borough) = 'bronx'
LIMIT 11
```

[33]:

	cmplnt_num	cmplnt_fr_dt	cmplnt_fr_tm	cmplnt_to_dt	cmplnt_to_tm	addr_pct_cd	\
0	976740225	01/29/2020	23:58:00	01/30/2020	00:00:00	44	
1	801051737	01/25/2020	15:50:00	01/25/2020	16:00:00	50	
2	408644965	02/08/2020	09:53:00	02/08/2020	13:57:00	50	
3	768764789	01/28/2020	09:00:00			47	
4	355268543	01/26/2020	06:00:00	01/26/2020	13:00:00	44	

	rpt_dt	ky_cd	ofns_desc	pd_cd	…	\
0	01/30/2020	359	OFFENSES AGAINST PUBLIC ADMINI	748	…	
1	01/25/2020	341	PETIT LARCENY	333	…	
2	02/08/2020	359	OFFENSES AGAINST PUBLIC ADMINI	749	…	
3	01/28/2020	341	PETIT LARCENY	321	…	
4	01/26/2020	361	OFF. AGNST PUB ORD SENSBLTY &	639	…	

	latitude	longitude	\
0	40.84079686100006	-73.91400066299997	
1	40.88258651500007	-73.90221152199997	
2	40.88311992000007	-73.90332096899994	
3	40.88162936400005	-73.84520181999993	
4	40.83658782300005	-73.91214858099994	

	lat_lon	patrol_boro	station_name	\
0	(40.84079686100006, -73.91400066299997)	PATROL BORO BRONX		
1	(40.88258651500007, -73.90221152199997)	PATROL BORO BRONX		
2	(40.88311992000007, -73.90332096899994)	PATROL BORO BRONX		
3	(40.88162936400005, -73.84520181999993)	PATROL BORO BRONX		
4	(40.83658782300005, -73.91214858099994)	PATROL BORO BRONX		

```
  vic_age_group        vic_race vic_sex   law_cat_cd borough
0         45-64           WHITE       F  MISDEMEANOR   BRONX
1       UNKNOWN         UNKNOWN       D  MISDEMEANOR   BRONX
2         25-44  WHITE HISPANIC       F  MISDEMEANOR   BRONX
3           65+           BLACK       F  MISDEMEANOR   BRONX
4         45-64  WHITE HISPANIC       F  MISDEMEANOR   BRONX

[5 rows x 35 columns]
```

Perform aggregated summaries

[34]:
```
# Run query to review a sample of records
cri_select_ofns_desc_stmnt01 = f"""
SELECT DISTINCT
    ofns_desc,
    COUNT(*) AS misdemeanor_offense_count
FROM {database_name}.{cri_pqt_tbl_name}
WHERE LOWER(law_cat_cd) = '{cri_law_cat_cd01}'
GROUP BY ofns_desc
ORDER BY misdemeanor_offense_count DESC
LIMIT 1000
"""

# Display SQL statement
print(cri_select_ofns_desc_stmnt01)

# Run SQL statement against Athena table
cri_df01_s02 = pd.read_sql(cri_select_ofns_desc_stmnt01,
                           conn)

# Display results
cri_df01_s02.head(31)
```

```
SELECT DISTINCT
    ofns_desc,
    COUNT(*) AS misdemeanor_offense_count
FROM ads508_t8.crime_pqt
WHERE LOWER(law_cat_cd) = 'misdemeanor'
GROUP BY ofns_desc
ORDER BY misdemeanor_offense_count DESC
LIMIT 1000
```

[34]:
```
                              ofns_desc  misdemeanor_offense_count
0                         PETIT LARCENY                      26836
1           ASSAULT 3 & RELATED OFFENSES                      16341
```

2	CRIMINAL MISCHIEF & RELATED OF	13220
3	OFF. AGNST PUB ORD SENSBLTY &	8101
4	DANGEROUS DRUGS	6905
5	OFFENSES AGAINST PUBLIC ADMINI	2890
6	VEHICLE AND TRAFFIC LAWS	2047
7	INTOXICATED & IMPAIRED DRIVING	1865
8	DANGEROUS WEAPONS	1802
9	CRIMINAL TRESPASS	1691
10	SEX CRIMES	1528
11	FRAUDS	927
12	OFFENSES INVOLVING FRAUD	508
13	POSSESSION OF STOLEN PROPERTY	490
14	UNAUTHORIZED USE OF A VEHICLE	447
15	OFFENSES AGAINST THE PERSON	401
16		379
17	OTHER OFFENSES RELATED TO THEF	337
18	ADMINISTRATIVE CODE	326
19	OTHER STATE LAWS (NON PENAL LA	153
20	THEFT OF SERVICES	104
21	BURGLAR'S TOOLS	87
22	FRAUDULENT ACCOSTING	68
23	GAMBLING	58
24	PETIT LARCENY OF MOTOR VEHICLE	30
25	OFFENSES RELATED TO CHILDREN	29
26	PROSTITUTION & RELATED OFFENSES	25
27	ALCOHOLIC BEVERAGE CONTROL LAW	24
28	OFFENSES AGAINST PUBLIC SAFETY	19
29	AGRICULTURE & MRKTS LAW-UNCLASSIFIED	17
30	JOSTLING	10

[35]:
```
cri_summ_borough_stmnt01 = f"""
SELECT
    law_cat_cd,
    borough,
    COUNT(*) AS crime_count
FROM {database_name}.{cri_pqt_tbl_name}
GROUP BY law_cat_cd, borough
ORDER BY crime_count DESC
LIMIT 100
"""

# Display SQL statement
print(cri_summ_borough_stmnt01)

# Run SQL statement against Athena table
cri_df01_s03 = pd.read_sql(cri_summ_borough_stmnt01,
                           conn)
```

```
# Display results
cri_df01_s03.head(47)
```

```
SELECT
    law_cat_cd,
    borough,
    COUNT(*) AS crime_count
FROM ads508_t8.crime_pqt
GROUP BY law_cat_cd, borough
ORDER BY crime_count DESC
LIMIT 100
```

[35]:

	law_cat_cd	borough	crime_count
0	MISDEMEANOR	BROOKLYN	25026
1	MISDEMEANOR	MANHATTAN	21620
2	MISDEMEANOR	BRONX	20038
3	MISDEMEANOR	QUEENS	16682
4	FELONY	BROOKLYN	15184
5	FELONY	MANHATTAN	11703
6	FELONY	QUEENS	10396
7	FELONY	BRONX	9287
8	VIOLATION	BROOKLYN	6275
9	VIOLATION	BRONX	4498
10	VIOLATION	QUEENS	4263
11	MISDEMEANOR	STATEN ISLAND	4231
12	VIOLATION	MANHATTAN	4198
13	FELONY	STATEN ISLAND	1642
14	VIOLATION	STATEN ISLAND	1407
15	FELONY	None	164
16	MISDEMEANOR	None	79
17	VIOLATION	None	9

[36]:
```
cri_date_stmnt01 = f"""
SELECT
    cmplnt_fr_dt,
    DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y') AS cmplnt_fr_date,
    COUNT(*) AS daily_misdemeanor_counts
FROM {database_name}.{cri_pqt_tbl_name}
WHERE LOWER(law_cat_cd) = '{cri_law_cat_cd01}'
    AND cmplnt_fr_dt <> ''
GROUP BY cmplnt_fr_dt
ORDER BY cmplnt_fr_dt
LIMIT 10000
"""
```

```
# Display SQL statement
print(cri_date_stmnt01)

# Run SQL statement against Athena table
cri_df01_s04 = pd.read_sql(cri_date_stmnt01,
                           conn)

# Display results
print(cri_df01_s04.shape)
display(cri_df01_s04.head(11))
```

```
SELECT
    cmplnt_fr_dt,
    DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y') AS cmplnt_fr_date,
    COUNT(*) AS daily_misdemeanor_counts
FROM ads508_t8.crime_pqt
WHERE LOWER(law_cat_cd) = 'misdemeanor'
    AND cmplnt_fr_dt <> ''
GROUP BY cmplnt_fr_dt
ORDER BY cmplnt_fr_dt
LIMIT 10000
```

```
(5957, 3)
```

	cmplnt_fr_dt	cmplnt_fr_date	daily_misdemeanor_counts
0	01/01/1978	1978-01-01	1
1	01/01/1999	1999-01-01	2
2	01/01/2000	2000-01-01	1
3	01/01/2001	2001-01-01	1
4	01/01/2002	2002-01-01	1
5	01/01/2003	2003-01-01	1
6	01/01/2004	2004-01-01	2
7	01/01/2005	2005-01-01	2
8	01/01/2006	2006-01-01	22
9	01/01/2007	2007-01-01	21
10	01/01/2008	2008-01-01	23

Load potential predictors and target for further exploration using pandas

[37]:
```
cri_box_stmnt01 = f"""
SELECT
    cmplnt_fr_dt,
    ky_cd,
    ofns_desc,
    pd_cd,
    pd_desc,
    crm_atpt_cptd_cd,
    loc_of_occur_desc,
```

```
    prem_typ_desc,
    jurisdiction_code,
    parks_nm,
    susp_age_group,
    susp_race,
    susp_sex,
    transit_district,
    latitude,
    longitude,
    vic_age_group,
    vic_race,
    vic_sex,
    law_cat_cd,
    borough
FROM {database_name}.{cri_pqt_tbl_name}
WHERE LOWER(law_cat_cd) = '{cri_law_cat_cd01}'
    AND LOWER(borough) = '{cri_bourough01}'
"""

# Display SQL statement
print(cri_box_stmnt01)

# Run SQL statement against Athena table
cri_df01_s05 = pd.read_sql(cri_box_stmnt01,
                           conn)

# Display results
print(cri_df01_s05.shape)
display(cri_df01_s05.head(11))
```

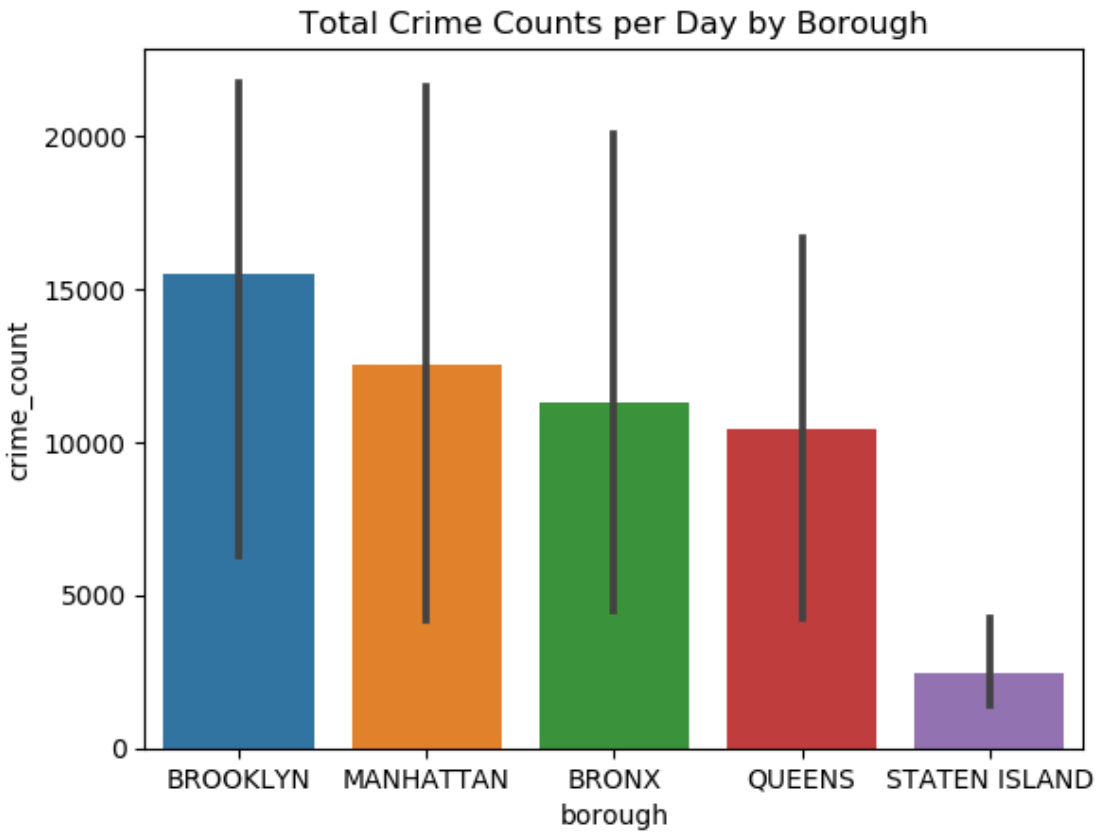
```
SELECT
    cmplnt_fr_dt,
    ky_cd,
    ofns_desc,
    pd_cd,
    pd_desc,
    crm_atpt_cptd_cd,
    loc_of_occur_desc,
    prem_typ_desc,
    jurisdiction_code,
    parks_nm,
    susp_age_group,
    susp_race,
    susp_sex,
    transit_district,
    latitude,
    longitude,
```

```
    vic_age_group,
    vic_race,
    vic_sex,
    law_cat_cd,
    borough
FROM ads508_t8.crime_pqt
WHERE LOWER(law_cat_cd) = 'misdemeanor'
    AND LOWER(borough) = 'bronx'

(20038, 21)

   cmplnt_fr_dt ky_cd                      ofns_desc pd_cd  \
0    09/14/2016   361   OFF. AGNST PUB ORD SENSBLTY &   639
1    10/15/2015   344    ASSAULT 3 & RELATED OFFENSES   101
2    03/08/2015   236              DANGEROUS WEAPONS   782
3    02/15/2017   351  CRIMINAL MISCHIEF & RELATED OF   254
4    12/04/2015   351  CRIMINAL MISCHIEF & RELATED OF   254
5    06/11/2017   344    ASSAULT 3 & RELATED OFFENSES   101
6    06/16/2016   233                     SEX CRIMES   175
7    08/08/2013   351  CRIMINAL MISCHIEF & RELATED OF   258
8    07/17/2015   341                   PETIT LARCENY   338
9    02/13/2014   351  CRIMINAL MISCHIEF & RELATED OF   256
10   11/25/2013   233                     SEX CRIMES   681

                         pd_desc crm_atpt_cptd_cd loc_of_occur_desc  \
0          AGGRAVATED HARASSMENT 2        COMPLETED            INSIDE
1                        ASSAULT 3        COMPLETED            INSIDE
2         WEAPONS, POSSESSION, ETC        COMPLETED
3   MISCHIEF, CRIMINAL 4, OF MOTOR        COMPLETED          FRONT OF
4   MISCHIEF, CRIMINAL 4, OF MOTOR        COMPLETED       OPPOSITE OF
5                        ASSAULT 3        COMPLETED          FRONT OF
6                SEXUAL ABUSE 3,2        COMPLETED            INSIDE
7   CRIMINAL MISCHIEF 4TH, GRAFFIT        COMPLETED          FRONT OF
8   LARCENY,PETIT FROM BUILDING,UN        COMPLETED            INSIDE
9    MISCHIEF, CRIMINAL 4, BY FIRE        COMPLETED            INSIDE
10      CHILD, ENDANGERING WELFARE        COMPLETED            INSIDE

                 prem_typ_desc jurisdiction_code parks_nm  …  \
0       RESIDENCE - APT. HOUSE                 0           …
1           STORE UNCLASSIFIED                 0           …
2                       STREET                 0           …
3                       STREET                 0           …
4                       STREET                 0           …
5              RESIDENCE-HOUSE                 0           …
6                PUBLIC SCHOOL                 0           …
7          COMMERCIAL BUILDING                 0       NA  …
8                  CHAIN STORE                 0           …
9   RESIDENCE - PUBLIC HOUSING                 2           …
```

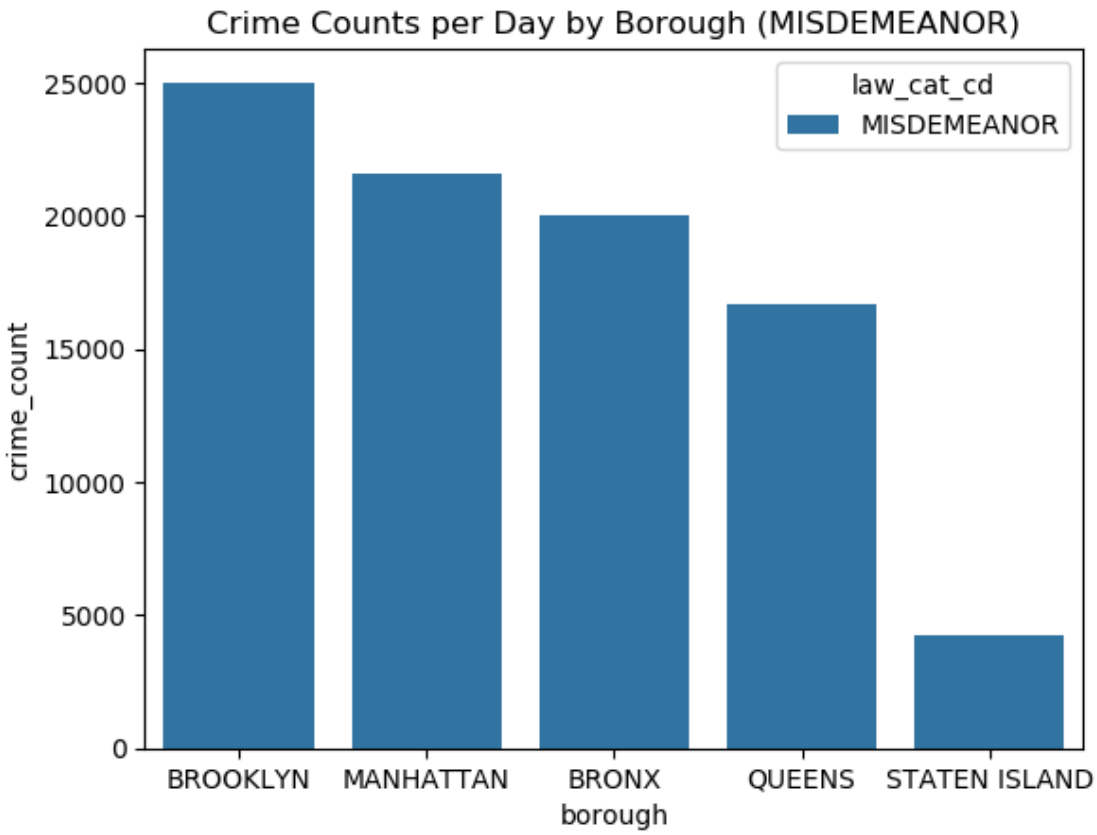
```
10              GROCERY/BODEGA                  0       NA  …

         susp_race susp_sex transit_district      latitude      longitude  \
0   BLACK HISPANIC        M                   40.838778865  -73.864701374
1   WHITE HISPANIC        F                   40.854068298  -73.915049873
2            BLACK        M                    40.83875187  -73.913757556
3          UNKNOWN        U                   40.815695332  -73.928462318
4                                             40.851767407  -73.891185053
5   WHITE HISPANIC        M                   40.822065537  -73.855052319
6                                             40.887451313  -73.847607787
7                                              40.85592989  -73.895794484
8          UNKNOWN        U                   40.861982735  -73.893637549
9                                              40.88367976  -73.833450905
10                                            40.843901255  -73.900504632

   vic_age_group        vic_race vic_sex   law_cat_cd borough
0          25-44  WHITE HISPANIC       F  MISDEMEANOR   BRONX
1          45-64  BLACK HISPANIC       F  MISDEMEANOR   BRONX
2                         UNKNOWN       E  MISDEMEANOR   BRONX
3          45-64  WHITE HISPANIC       M  MISDEMEANOR   BRONX
4          25-44           BLACK       M  MISDEMEANOR   BRONX
5          18-24  WHITE HISPANIC       F  MISDEMEANOR   BRONX
6            <18  WHITE HISPANIC       F  MISDEMEANOR   BRONX
7                         UNKNOWN       D  MISDEMEANOR   BRONX
8          45-64  BLACK HISPANIC       F  MISDEMEANOR   BRONX
9                         UNKNOWN       E  MISDEMEANOR   BRONX
10           <18           BLACK       M  MISDEMEANOR   BRONX

[11 rows x 21 columns]
```

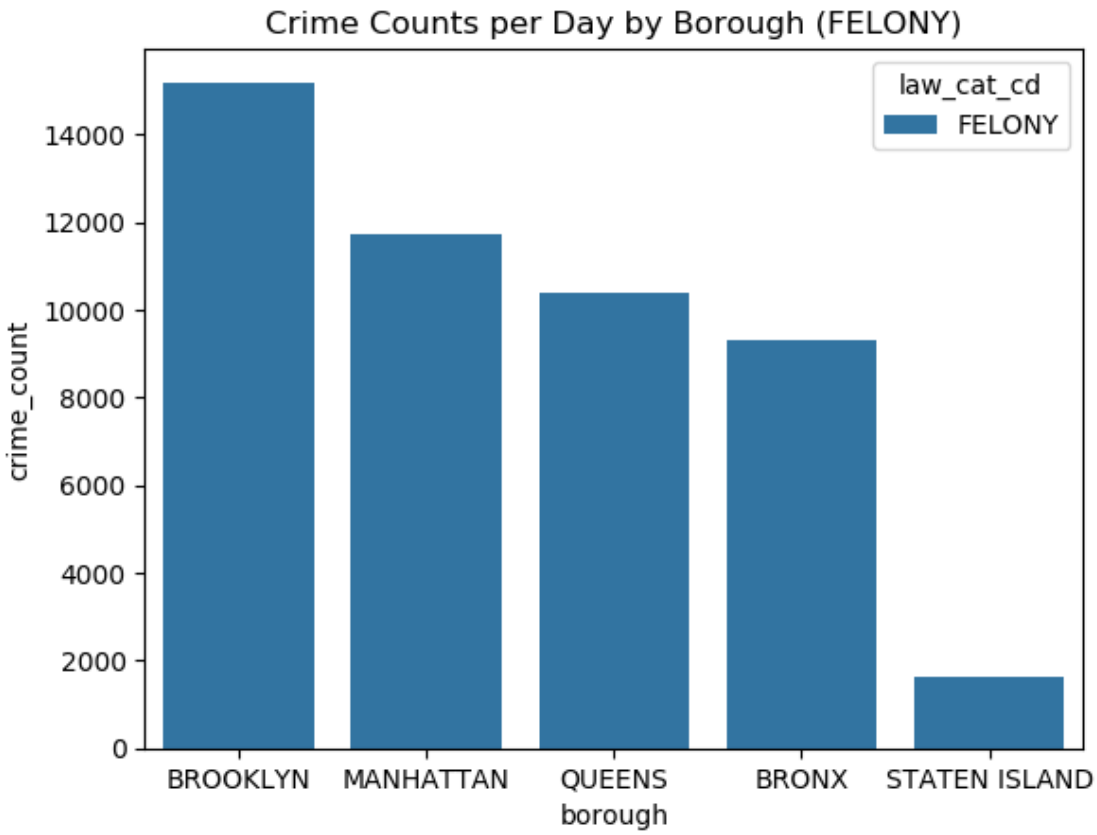
Plots for Various Features

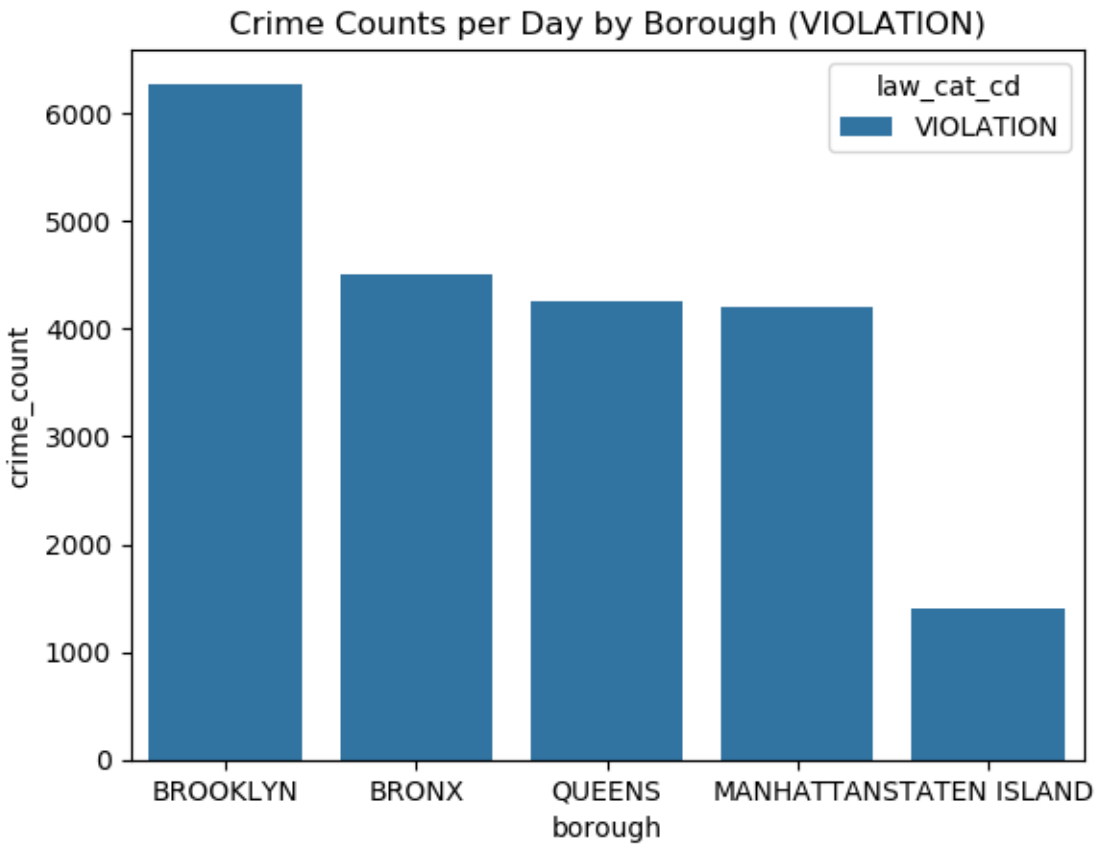
[38]:
```
pqt_bar = sns.barplot(x='borough',
                      y='crime_count',
                      data=cri_df01_s03).set(title='Total Crime Counts per Day␣
↪by Borough')
```



[39]:
```
for law_cat in cri_df01_s03['law_cat_cd'].unique():
    sns.barplot(x='borough',
                y='crime_count',
                data=cri_df01_s03[cri_df01_s03['law_cat_cd']==law_cat],
                hue='law_cat_cd')
    plt.title(f'Crime Counts per Day by Borough ({law_cat})')
    plt.show()
```


Crime Counts per Day by Borough (MISDEMEANOR)
law_cat_cd
MISDEMEANOR
crime_count
25000
20000
15000
10000
5000
0
BROOKLYN
MANHATTAN
BRONX
QUEENS
STATEN ISLAND
borough


Crime Counts per Day by Borough (FELONY)
law_cat_cd
FELONY
crime_count
14000
12000
10000
8000
6000
4000
2000
0
BROOKLYN
MANHATTAN
QUEENS
BRONX
STATEN ISLAND
borough



Create subsets of columns for various purposes

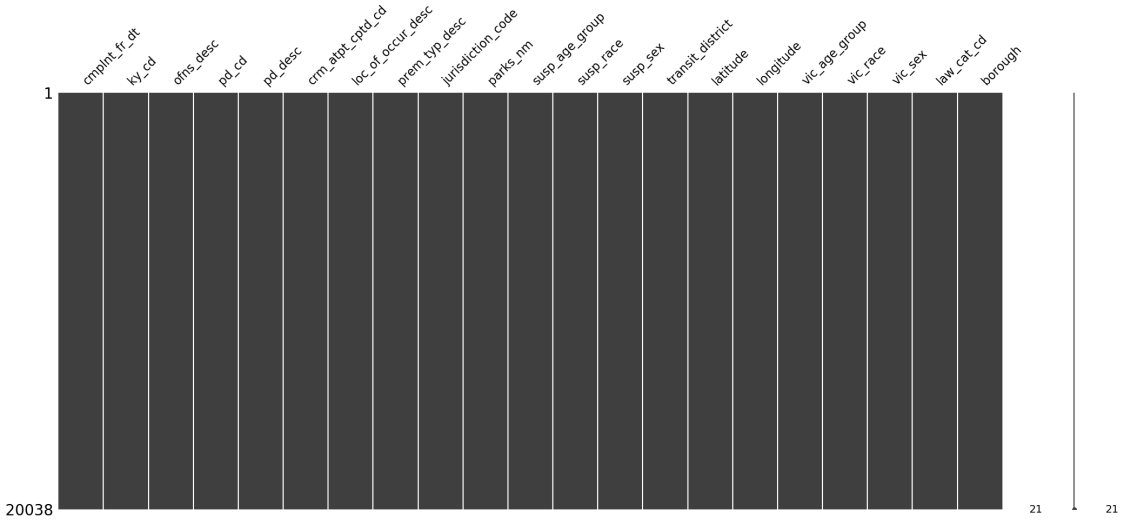
[40]:
```
cri_df01_s05_num_lst01 = []

cri_df01_s05_num_lst02 = []

cri_df02_s01 = cri_df01_s05[cri_df01_s05_num_lst01]
cri_df03_s01 = cri_df01_s05[cri_df01_s05_num_lst02]

display(cri_df02_s01.head(11))
```

```
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Examine features with missing values

[41]:
```
# Visualize missing values in each column
msno.matrix(cri_df01_s05)
```

[41]: <matplotlib.axes._subplots.AxesSubplot at 0x7fba25bce190>



[42]:
```
# Remove any features for which the number of null vals exceed a threshold--
#-- (5% of total N)
cri_df01_s05_null_summ01 = pd.DataFrame(cri_df01_s05.isnull().sum(),
                                        columns=['null_count'])

cri_df01_s05_null_summ02 = cri_df01_s05_null_summ01.
 ↪loc[(cri_df01_s05_null_summ01['null_count'] != 0)].sort_values('null_count',
                                                                 ␣
 ↪                                       ascending=False)
cri_df01_s05_null_summ03 = cri_df01_s05_null_summ02.reset_index()
print(cri_df01_s05_null_summ03)

cri_df01_s05_null_summ04 = cri_df01_s05_null_summ03.
 ↪loc[cri_df01_s05_null_summ03['null_count'] > (len(cri_df01_s05)*.05)]
print('\n', cri_df01_s05_null_summ04)

cri_df01_s05_null_summ04_remove_lst01 = list(cri_df01_s05_null_summ04['index'])
print('\n', cri_df01_s05_null_summ04_remove_lst01)
```

```
Empty DataFrame
Columns: [index, null_count]
Index: []

 Empty DataFrame
Columns: [index, null_count]
Index: []

 []
```

Display time plots for select features

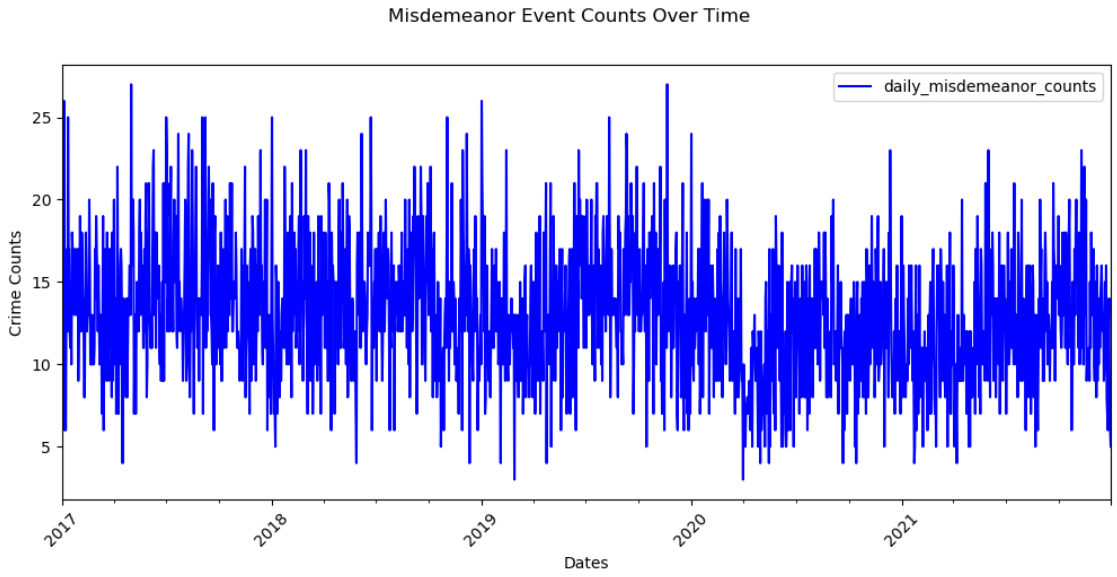
[43]:
```
cri_date_stmnt02 = f"""
SELECT
    cmplnt_fr_dt,
    DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y') AS cmplnt_fr_date,
    COUNT(*) AS daily_misdemeanor_counts
FROM {database_name}.{cri_pqt_tbl_name}
WHERE LOWER(law_cat_cd) = '{cri_law_cat_cd01}'
    AND cmplnt_fr_dt <> ''
    AND YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) BETWEEN 2017 AND 2021
GROUP BY cmplnt_fr_dt
ORDER BY DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')
LIMIT 10000
"""

# Display SQL statement
print(cri_date_stmnt02)

# Run SQL statement against Athena table
cri_df01_s14 = pd.read_sql(cri_date_stmnt02,
                           conn)

# Display results
print(cri_df01_s14.shape)
display(cri_df01_s14.head(11))
```

```
SELECT
    cmplnt_fr_dt,
    DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y') AS cmplnt_fr_date,
    COUNT(*) AS daily_misdemeanor_counts
FROM ads508_t8.crime_pqt
WHERE LOWER(law_cat_cd) = 'misdemeanor'
    AND cmplnt_fr_dt <> ''
    AND YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) BETWEEN 2017 AND 2021
GROUP BY cmplnt_fr_dt
ORDER BY DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')
LIMIT 10000

(1826, 3)
```

	cmplnt_fr_dt	cmplnt_fr_date	daily_misdemeanor_counts
0	01/01/2017	2017-01-01	26
1	01/02/2017	2017-01-02	10
2	01/03/2017	2017-01-03	14
3	01/04/2017	2017-01-04	26
4	01/05/2017	2017-01-05	6
5	01/06/2017	2017-01-06	17
6	01/07/2017	2017-01-07	6

7	01/08/2017	2017-01-08	13
8	01/09/2017	2017-01-09	13
9	01/10/2017	2017-01-10	12
10	01/11/2017	2017-01-11	25

[44]:
```
fig = plt.gcf()
fig.set_size_inches(12, 5)

fig.suptitle("Misdemeanor Event Counts Over Time")

ax = plt.gca()
# ax = plt.gca().set_xticks(df['year'])
ax.locator_params(integer=True)
ax.set_xticks(cri_df01_s14["cmplnt_fr_date"].unique())

cri_df01_s14.plot(kind="line",
        x="cmplnt_fr_date",
        y='daily_misdemeanor_counts',
        color="blue",
        ax=ax)

# plt.xticks(range(1995, 2016, 1))
# plt.yticks(range(0,6,1))
plt.xlabel("Dates")
plt.ylabel("Crime Counts")
plt.xticks(rotation=45)

# fig.savefig('average-rating.png', dpi=300)
plt.show()
```



Summary stats for select features

[45]:
```
sum_stat_cri = pd.DataFrame(cri_df01_s04['daily_misdemeanor_counts'].
 ↪describe()).T
sum_stat_cri
```

[45]:

	count	mean	std	min	25%	50%	75%	\
daily_misdemeanor_counts	5957.0	14.717139	4.893156	1.0	12.0	15.0	18.0	

	max
daily_misdemeanor_counts	36.0

Outliers and Skew Exploration

Outliers for select features

[46]:
```
# Population
IQR_cri = sum_stat_cri['75%'][0] - sum_stat_cri['25%'][0]
low_outlier_cri = sum_stat_cri['25%'][0] - 1.5*(IQR_cri)
high_outlier_cri = sum_stat_cri['75%'][0] + 1.5*(IQR_cri)
print('Low Outlier (daily_misdemeanor_counts):', low_outlier_cri)
print('High Outlier (daily_misdemeanor_counts):', high_outlier_cri)
```

```
Low Outlier (daily_misdemeanor_counts): 3.0
High Outlier (daily_misdemeanor_counts): 27.0
```

[47]:
```
# Identify presence of outliers
outliers_cri = cri_df01_s04.loc[(cri_df01_s04['daily_misdemeanor_counts'] <␣
 ↪low_outlier) | (cri_df01_s04['daily_misdemeanor_counts'] > high_outlier),
                            ['daily_misdemeanor_counts']]
if outliers_cri.empty:
    print('No outliers found in daily_misdemeanor_counts column')
else:
    print('Outliers found in daily_misdemeanor_counts column ({0}):'.
 ↪format(len(outliers_cri)))
    print(outliers_cri)
```

```
No outliers found in daily_misdemeanor_counts column
```

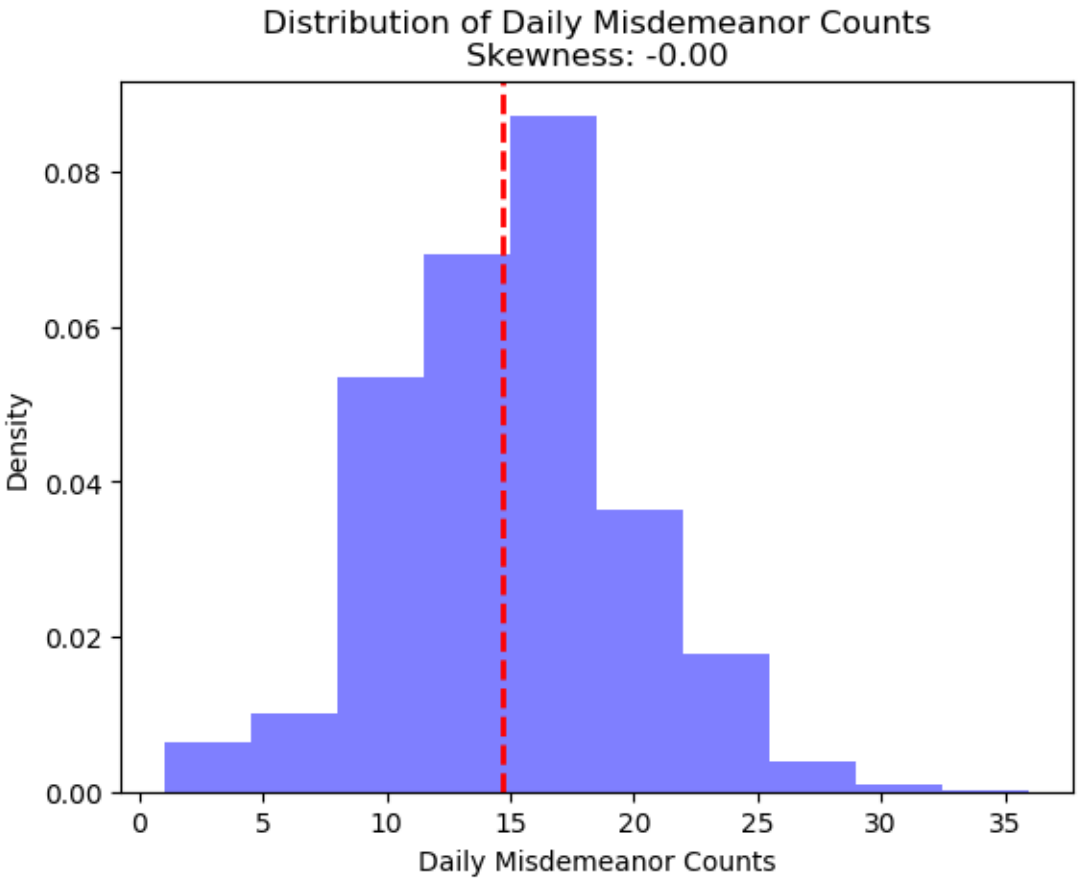
Determine Skew for select features

[48]:
```
# For 'Misdemeanor' column
daily_misdemeanor_skew = skew(cri_df01_s04['daily_misdemeanor_counts'])
print('Skewness of daily_misdemeanor_counts column:', daily_misdemeanor_skew)
```

```
Skewness of daily_misdemeanor_counts column: -0.0019072564182525918
```

[49]:
```
plt.hist(cri_df01_s04['daily_misdemeanor_counts'], density=True, alpha=0.5,␣
 ↪color='blue')
```

```
plt.title('Distribution of Daily Misdemeanor Counts\nSkewness: {0:.2f}'.
  ↪format(daily_misdemeanor_skew))
plt.xlabel('Daily Misdemeanor Counts')
plt.ylabel('Density')
plt.axvline(x=cri_df01_s04['daily_misdemeanor_counts'].mean(), color='red',␣
  ↪linestyle='dashed', linewidth=2)
plt.show()
```



2.4.3 evictions

[50]:
```
evi_tsv_tbl_name = 'evictions'
```

Explore via SQL SELECT statements

[51]:
```
# Run query to review a sample of records
evi_bourough01 = "bronx"

evi_select_borough_stmnt01 = f"""
SELECT * FROM {database_name}.{evi_tsv_tbl_name}
```

```
WHERE LOWER(borough) = '{evi_bourough01}'
LIMIT 11
"""

# Display SQL statement
print(evi_select_borough_stmnt01)

# Run SQL statement against Athena table
evi_df01_s01 = pd.read_sql(evi_select_borough_stmnt01,
                           conn)

# Display results
evi_df01_s01.head(11)
```

```
SELECT * FROM ads508_t8.evictions
WHERE LOWER(borough) = 'bronx'
LIMIT 11
```

[51]:

	court_index_number	docket_number	eviction_address	eviction_apartment_number	executed_date	marshal_first_name
0	56037/17	339568	547 EAST 168TH STREET			
1	B047517/19	409031	4014 CARPENTER AVENUE			
2	15068/17	334442	655 EAST 224TH STREET			
3	14866/19A	097278	718 PENFIELD STREET			
4	66703/18BX	090391	2032 EAST 177TH ST A /K/A 2032 CROSS BRONX EXP…			
5	B806500/18	396012	281 EAST 143RD STREET			
6	54026/17	341956	1211 SOUTHERN BOULEVARD			
7	69137/18	10335	1351 BOSTON ROAD - APT 201			
8	18348/16	324092	2280 LORING PLACE NORTH			
9	75943/16A	060118	1551 WILLIAMSBRID GE ROAD			
10	44987/17	069826	1514 SEDGWICK AVENUE			

0	3H	02/26/2018	Thomas
1	4B	11/16/2022	Richard
2	1	09/29/2017	Thomas
3	2-F	10/24/2019	Justin
4	1E	07/30/2019	Justin
5	07A	01/17/2019	Richard
6	301	11/19/2018	Thomas
7	201	07/15/2019	Robert
8	4B	05/22/2017	Thomas
9	4-B	08/10/2017	Justin
10	7C	05/22/2018	Justin

	marshal_last_name	residential_or_commercial	borough	eviction_postcode
0	Bia	Residential	BRONX	10456
1	McCoy	Residential	BRONX	10466
2	Bia	Residential	BRONX	10467
3	Grossman	Residential	BRONX	10470
4	Grossman	Residential	BRONX	10472
5	McCoy	Residential	BRONX	10451
6	Bia	Residential	BRONX	10459
7	Renzulli	Residential	BRONX	10456
8	Bia	Residential	BRONX	10468
9	Grossman	Residential	BRONX	10461
10	Grossman	Residential	BRONX	10453

	ejectment	eviction_or_legal_possession	latitude	longitude
0	Not an Ejectment	Possession	40.830857	-73.905191
1	Not an Ejectment	Possession	40.889878	-73.862686
2	Not an Ejectment	Possession	40.887599	-73.862391
3	Not an Ejectment	Possession	40.904888	-73.849089
4	Not an Ejectment	Possession	40.831685	-73.856168
5	Not an Ejectment	Possession	40.814845	-73.924083
6	Not an Ejectment	Possession	40.828949	-73.891897
7	Not an Ejectment	Possession	40.832166	-73.898808
8	Not an Ejectment	Possession	40.861277	-73.908723
9	Not an Ejectment	Possession		
10	Not an Ejectment	Possession	40.846731	-73.924961

	community_board	council_district	census_tract	bin	bbl
0	3	16	145	2004227	2026100065
1	12	12	408	2063060	2048280031
2	12	12	394	2062985	2048260028
3	12	11	442	2071873	2051130039
4	9	18	78	2026230	2038030019
5	1	8	51	2091116	2023240001
6	3	17	125	2113777	2029750037
7	3	16	151	2128618	2029340050

```
8                 7                14          255  2014918  2032250015
9
10                5                16        20501  2114714  2028800009

                                  nta
0                    Claremont-Bathgate
1              Williamsbridge-Olinville
2              Williamsbridge-Olinville
3                    Woodlawn-Wakefield
4                 Westchester-Unionport
5                Mott Haven-Port Morris
6                    Morrisania-Melrose
7                    Morrisania-Melrose
8                   Kingsbridge Heights
9
10  University Heights-Morris Heights
```

Perform aggregated summaries

[52]:
```
# Run query to review a sample of records
evi_select_eviction_postcode_stmnt01 = f"""
SELECT DISTINCT
    eviction_postcode,
    COUNT(*)
FROM {database_name}.{evi_tsv_tbl_name}
GROUP BY eviction_postcode
LIMIT 1000
"""

# Display SQL statement
print(evi_select_eviction_postcode_stmnt01)

# Run SQL statement against Athena table
evi_df01_s02 = pd.read_sql(evi_select_eviction_postcode_stmnt01,
                           conn)

# Display results
evi_df01_s02.head(11)
```

```
SELECT DISTINCT
    eviction_postcode,
    COUNT(*)
FROM ads508_t8.evictions
GROUP BY eviction_postcode
LIMIT 1000
```

[52]:

	eviction_postcode	_col1
0	10456	1825
1	10472	1069
2	11221	915
3	10453	1763
4	10455	695
5	11415	181
6	11210	781
7	10458	1941
8	11235	551
9	11223	358
10	10314	234

[53]:
```
evi_summ_borough_stmnt01 = f"""
SELECT
    borough,
    COUNT(*) AS evictions_count
FROM {database_name}.{evi_tsv_tbl_name}
GROUP BY borough
LIMIT 100
"""

# Display SQL statement
print(evi_summ_borough_stmnt01)

# Run SQL statement against Athena table
evi_df01_s03 = pd.read_sql(evi_summ_borough_stmnt01,
                           conn)

# Display results
evi_df01_s03.head(11)
```

```
SELECT
    borough,
    COUNT(*) AS evictions_count
FROM ads508_t8.evictions
GROUP BY borough
LIMIT 100
```

[53]:

	borough	evictions_count
0	MANHATTAN	11321
1	QUEENS	14082
2	BRONX	23519
3	STATEN ISLAND	2500
4	BROOKLYN	21066

[54]:
```
evi_summ_borough_stmnt01 = f"""
SELECT
    borough,
    census_tract,
    COUNT(*) AS ctract_count
FROM {database_name}.{evi_tsv_tbl_name}
GROUP BY borough, census_tract
LIMIT 100
"""

# Display SQL statement
print(evi_summ_borough_stmnt01)

# Run SQL statement against Athena table
evi_df01_s04 = pd.read_sql(evi_summ_borough_stmnt01,
                           conn)

# Display results
evi_df01_s04.head(11)
```

```
SELECT
    borough,
    census_tract,
    COUNT(*) AS ctract_count
FROM ads508_t8.evictions
GROUP BY borough, census_tract
LIMIT 100
```

[54]:

	borough	census_tract	ctract_count
0	BRONX	394	133
1	BRONX		1085
2	QUEENS	919	10
3	BROOKLYN	21	29
4	BROOKLYN	379	45
5	BROOKLYN		2310
6	QUEENS	871	34
7	QUEENS	136	15
8	BRONX	54	95
9	BROOKLYN	234	11
10	BRONX	149	134

[55]:
```
evi_date_stmnt01 = f"""
SELECT
    executed_date,
    DATE_PARSE(executed_date, '%m/%d/%Y') AS evi_date,
    COUNT(*) AS daily_eviction_counts
```

```
FROM {database_name}.{evi_tsv_tbl_name}
WHERE executed_date <> ''
GROUP BY executed_date
ORDER BY executed_date
LIMIT 10000
"""

# Display SQL statement
print(evi_date_stmnt01)

# Run SQL statement against Athena table
evi_df01_s06 = pd.read_sql(evi_date_stmnt01,
                           conn)
# Display results
print(evi_df01_s06.shape)
display(evi_df01_s06.head(11))
```

```
SELECT
    executed_date,
    DATE_PARSE(executed_date, '%m/%d/%Y') AS evi_date,
    COUNT(*) AS daily_eviction_counts
FROM ads508_t8.evictions
WHERE executed_date <> ''
GROUP BY executed_date
ORDER BY executed_date
LIMIT 10000

(1152, 3)
```

	executed_date	evi_date	daily_eviction_counts
0	01/02/2018	2018-01-02	30
1	01/02/2019	2019-01-02	61
2	01/02/2020	2020-01-02	66
3	01/03/2017	2017-01-03	102
4	01/03/2018	2018-01-03	150
5	01/03/2019	2019-01-03	136
6	01/03/2020	2020-01-03	81
7	01/03/2022	2022-01-03	18
8	01/03/2023	2023-01-03	44
9	01/04/2017	2017-01-04	144
10	01/04/2018	2018-01-04	3

[56]:
```
#Adding in borough
evi_date_stmnt01_borough = f"""
SELECT
    borough,
    executed_date,
```

```
    DATE_PARSE(executed_date, '%m/%d/%Y') AS evi_date,
    COUNT(*) AS daily_eviction_counts
FROM {database_name}.{evi_tsv_tbl_name}
WHERE executed_date <> ''
GROUP BY borough, executed_date
ORDER BY borough, executed_date
LIMIT 10000
"""

# Display SQL statement
print(evi_date_stmnt01_borough)

# Run SQL statement against Athena table
evi_df01_s06_borough = pd.read_sql(evi_date_stmnt01_borough,
                           conn)

# Display results
print(evi_df01_s06_borough.shape)
display(evi_df01_s06_borough.head(11))
```

```
SELECT
    borough,
    executed_date,
    DATE_PARSE(executed_date, '%m/%d/%Y') AS evi_date,
    COUNT(*) AS daily_eviction_counts
FROM ads508_t8.evictions
WHERE executed_date <> ''
GROUP BY borough, executed_date
ORDER BY borough, executed_date
LIMIT 10000

(4966, 4)
```

	borough	executed_date	evi_date	daily_eviction_counts
0	BRONX	01/02/2018	2018-01-02	10
1	BRONX	01/02/2019	2019-01-02	23
2	BRONX	01/02/2020	2020-01-02	23
3	BRONX	01/03/2017	2017-01-03	39
4	BRONX	01/03/2018	2018-01-03	68
5	BRONX	01/03/2019	2019-01-03	55
6	BRONX	01/03/2020	2020-01-03	21
7	BRONX	01/03/2022	2022-01-03	4
8	BRONX	01/03/2023	2023-01-03	1
9	BRONX	01/04/2017	2017-01-04	44
10	BRONX	01/04/2019	2019-01-04	30

[57]:
```
#Adding in eviction_postcode
evi_date_stmnt01_post = f"""
SELECT
    eviction_postcode,
    executed_date,
    DATE_PARSE(executed_date, '%m/%d/%Y') AS evi_date,
    COUNT(*) AS daily_eviction_counts
FROM {database_name}.{evi_tsv_tbl_name}
WHERE executed_date <> ''
GROUP BY eviction_postcode, executed_date
ORDER BY eviction_postcode, executed_date
LIMIT 10000
"""

# Display SQL statement
print(evi_date_stmnt01_post)

# Run SQL statement against Athena table
evi_df01_s06_post = pd.read_sql(evi_date_stmnt01_post,
                          conn)

# Display results
print(evi_df01_s06_post.shape)
display(evi_df01_s06_post.head(11))
```

```
SELECT
    eviction_postcode,
    executed_date,
    DATE_PARSE(executed_date, '%m/%d/%Y') AS evi_date,
    COUNT(*) AS daily_eviction_counts
FROM ads508_t8.evictions
WHERE executed_date <> ''
GROUP BY eviction_postcode, executed_date
ORDER BY eviction_postcode, executed_date
LIMIT 10000

(10000, 4)
```

	eviction_postcode	executed_date	evi_date	daily_eviction_counts
0	00000	04/19/2018	2018-04-19	1
1	00000	07/13/2018	2018-07-13	1
2	00000	08/23/2018	2018-08-23	1
3	01000	05/18/2017	2017-05-18	1
4	10000	07/30/2019	2019-07-30	1
5	10001	01/03/2019	2019-01-03	1
6	10001	01/03/2022	2022-01-03	1
7	10001	01/05/2017	2017-01-05	3

```
8               10001     01/05/2023 2023-01-05                    1
9               10001     01/06/2017 2017-01-06                    1
10              10001     01/07/2019 2019-01-07                    1
```

Load potential predictors and target for further exploration using pandas

[58]:
```
evi_box_stmnt01 = f"""
SELECT
    court_index_number,
    docket_number,
    eviction_address,
    eviction_apartment_number,
    executed_date,
    marshal_first_name,
    marshal_last_name,
    residential_or_commercial,
    borough,
    eviction_postcode,
    ejectment,
    eviction_or_legal_possession,
    latitude,
    longitude,
    census_tract
FROM {database_name}.{evi_tsv_tbl_name}
LIMIT 5000
"""

# Display SQL statement
print(evi_box_stmnt01)

# Run SQL statement against Athena table
evi_df01_s05 = pd.read_sql(evi_box_stmnt01,
                           conn)

# Display results
evi_df01_s05.head(11)
```

```
SELECT
    court_index_number,
    docket_number,
    eviction_address,
    eviction_apartment_number,
    executed_date,
    marshal_first_name,
    marshal_last_name,
    residential_or_commercial,
    borough,
    eviction_postcode,
```

```
    ejectment,
    eviction_or_legal_possession,
    latitude,
    longitude,
    census_tract
FROM ads508_t8.evictions
LIMIT 5000
```

[58]:

	court_index_number	docket_number	eviction_address	eviction_apartment_number	executed_date	marshal_first_name
0	56037/17	339568	547 EAST 168TH STREET	3H	02/26/2018	Thomas
1	B047517/19	409031	4014 CARPENTER AVENUE	4B	11/16/2022	Richard
2	15068/17	334442	655 EAST 224TH STREET	1	09/29/2017	Thomas
3	58273/18	025388	1551 DEAN STREET	1ST FLOOR	07/12/2018	Gary
4	14866/19A	097278	718 PENFIELD STREET	2-F	10/24/2019	Justin
5	66703/18BX	090391	2032 EAST 177TH ST A /K/A 2032 CROSS BRONX EXP…	1E	07/30/2019	Justin
6	98925/17	075402	175 WOODRUFF AVENUE	GARDEN APARTMENT	06/01/2018	Justin
7	304057/20	107717	555 TENTH AVENUE	32I	04/18/2022	Justin
8	210706/18	085502	2201 FIRST AVENUE	05B	03/14/2019	Henry
9	B806500/18	396012	281 EAST 143RD STREET	07A	01/17/2019	Richard
10	83995/16	464985	1-11 MARBLE HILL AVE NUE	3F	03/17/2017	Danny

```
   marshal_last_name residential_or_commercial    borough eviction_postcode  \
0                Bia                Residential      BRONX             10456
1              McCoy                Residential      BRONX             10466
2                Bia                Residential      BRONX             10467
3               Rose                Residential   BROOKLYN             11213
4           Grossman                Residential      BRONX             10470
5           Grossman                Residential      BRONX             10472
6           Grossman                Residential   BROOKLYN             11226
7           Grossman                Residential  MANHATTAN             10018
8              Daley                Residential  MANHATTAN             10029
9              McCoy                Residential      BRONX             10451
10          Weinheim                Residential  MANHATTAN             10463

            ejectment eviction_or_legal_possession   latitude   longitude  \
0   Not an Ejectment                     Possession  40.830857  -73.905191
1   Not an Ejectment                     Possession  40.889878  -73.862686
2   Not an Ejectment                     Possession  40.887599  -73.862391
3   Not an Ejectment                     Possession  40.676166  -73.936661
4   Not an Ejectment                     Possession  40.904888  -73.849089
5   Not an Ejectment                     Possession  40.831685  -73.856168
6   Not an Ejectment                     Possession  40.654641  -73.960291
7   Not an Ejectment                     Possession  40.758888  -73.996022
8   Not an Ejectment                     Possession  40.794176  -73.936754
9   Not an Ejectment                     Possession  40.814845  -73.924083
10  Not an Ejectment                     Possession  40.874862  -73.910845

   census_tract
0           145
1           408
2           394
3           311
4           442
5            78
6         50803
7           117
8           180
9            51
10          309
```

Create subsets of columns for various purposes

[59]:
```
evi_df01_s05_num_lst01 = []

evi_df01_s05_num_lst02 = []

evi_df02_s01 = evi_df01_s05[evi_df01_s05_num_lst01]
evi_df03_s01 = evi_df01_s05[evi_df01_s05_num_lst02]
```

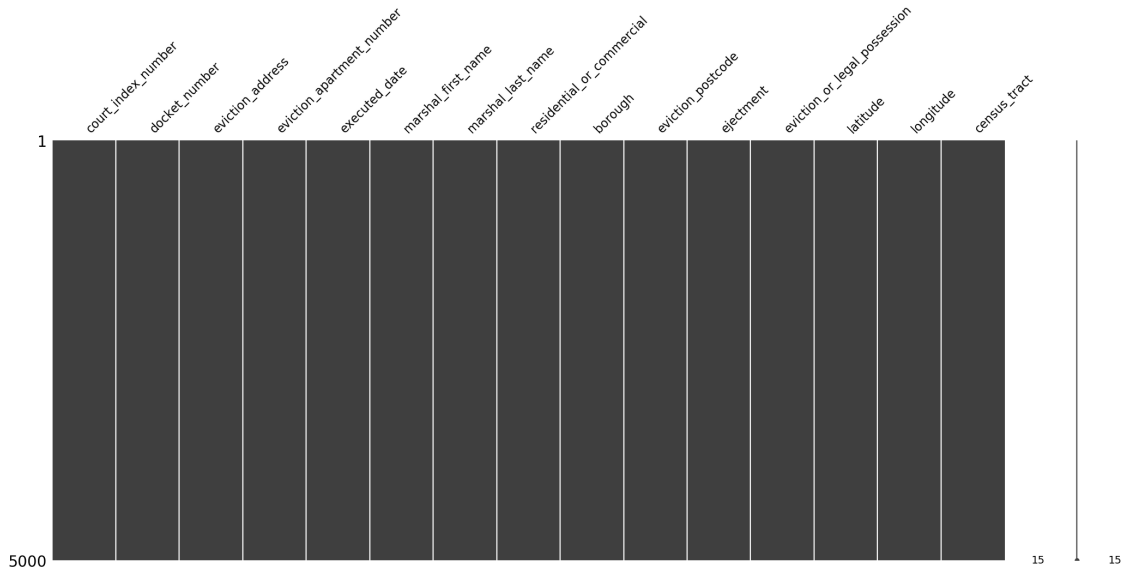
```
display(evi_df02_s01.head(5))
```

```
Empty DataFrame
Columns: []
Index: [0, 1, 2, 3, 4]
```

Examine features with missing values

[60]:
```
# Visualize missing values in each column
msno.matrix(evi_df01_s05)
```

[60]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fba23fcf850>
```



[61]:
```
# Remove any features for which the number of null vals exceed a threshold--
#-- (5% of total N)
evi_df01_s05_null_summ01 = pd.DataFrame(evi_df01_s05.isnull().sum(),
                                        columns=['null_count'])

evi_df01_s05_null_summ02 = evi_df01_s05_null_summ01.
 ↪loc[(evi_df01_s05_null_summ01['null_count'] != 0)].sort_values('null_count',

 ↪                                      ascending=False)
evi_df01_s05_null_summ03 = evi_df01_s05_null_summ02.reset_index()
print(evi_df01_s05_null_summ03)

evi_df01_s05_null_summ04 = evi_df01_s05_null_summ03.
 ↪loc[evi_df01_s05_null_summ03['null_count'] > (len(evi_df01_s05)*.05)]
print('\n', evi_df01_s05_null_summ04)
```

```
evi_df01_s05_null_summ04_remove_lst01 = list(evi_df01_s05_null_summ04['index'])
print('\n', evi_df01_s05_null_summ04_remove_lst01)
```

```
Empty DataFrame
Columns: [index, null_count]
Index: []

 Empty DataFrame
Columns: [index, null_count]
Index: []

 []
```

Display time plots for select features

[62]:
```
#Adding in eviction_postcode
evi_date_stmnt02_post = f"""
SELECT
    executed_date,
    DATE_PARSE(executed_date, '%m/%d/%Y') AS evi_date,
    COUNT(*) AS daily_eviction_counts
FROM {database_name}.{evi_tsv_tbl_name}
WHERE executed_date <> ''
    AND CAST(YEAR(DATE_PARSE(executed_date, '%m/%d/%Y')) AS INT) BETWEEN 2018␣
 ↪AND 2022
GROUP BY executed_date
ORDER BY DATE_PARSE(executed_date, '%m/%d/%Y')
LIMIT 10000
"""

# Display SQL statement
print(evi_date_stmnt02_post)

# Run SQL statement against Athena table
evi_df01_s16_post = pd.read_sql(evi_date_stmnt02_post,
                           conn)

# Display results
print(evi_df01_s16_post.shape)
display(evi_df01_s16_post.head(11))
```

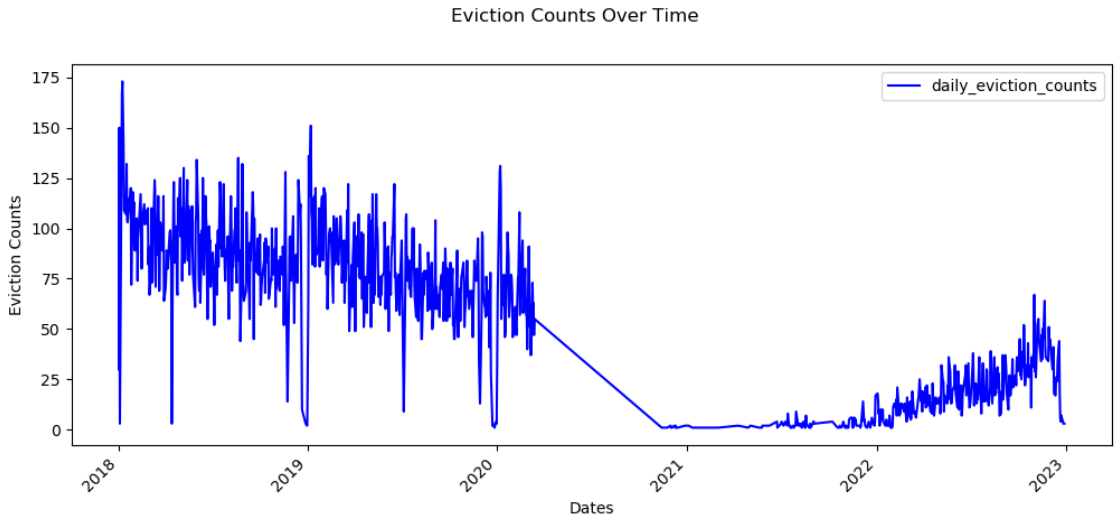
```
SELECT
    executed_date,
    DATE_PARSE(executed_date, '%m/%d/%Y') AS evi_date,
    COUNT(*) AS daily_eviction_counts
FROM ads508_t8.evictions
WHERE executed_date <> ''
```

```
    AND CAST(YEAR(DATE_PARSE(executed_date, '%m/%d/%Y')) AS INT) BETWEEN 2018
AND 2022
GROUP BY executed_date
ORDER BY DATE_PARSE(executed_date, '%m/%d/%Y')
LIMIT 10000
```

```
(895, 3)
```

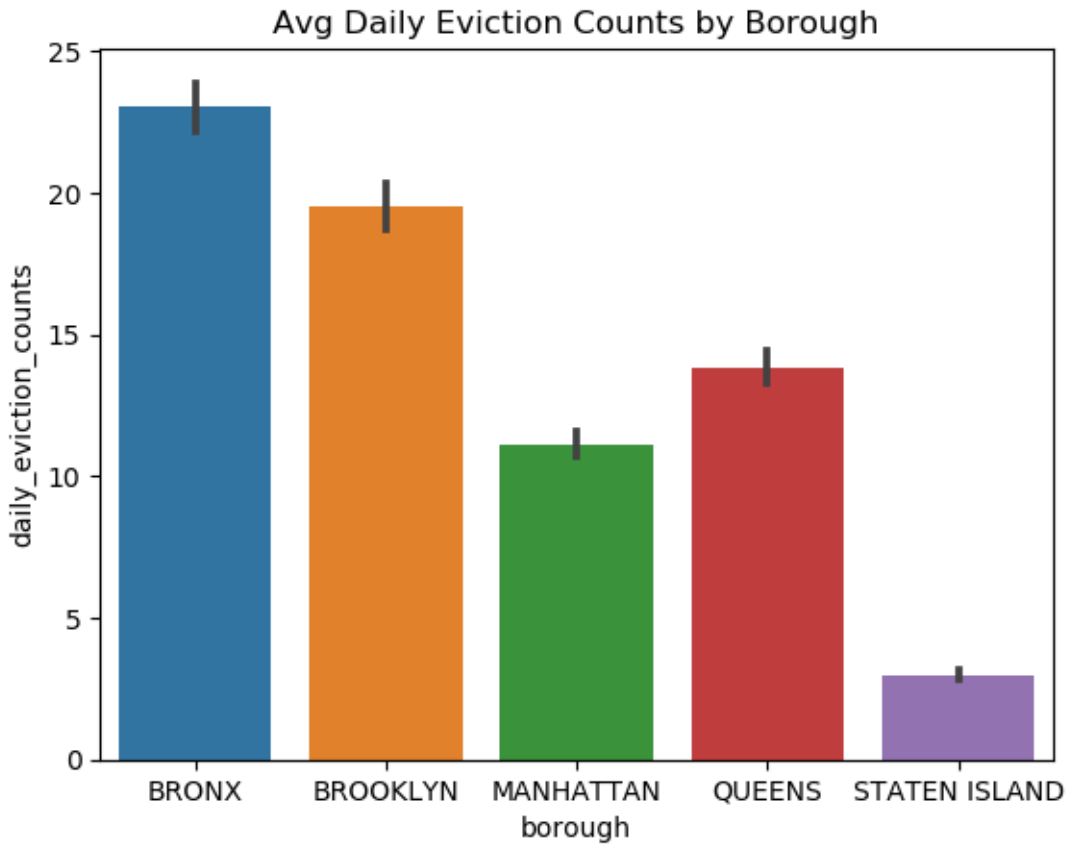
	executed_date	evi_date	daily_eviction_counts
0	01/02/2018	2018-01-02	30
1	01/03/2018	2018-01-03	150
2	01/04/2018	2018-01-04	3
3	01/08/2018	2018-01-08	167
4	01/09/2018	2018-01-09	173
5	01/10/2018	2018-01-10	158
6	01/11/2018	2018-01-11	142
7	01/12/2018	2018-01-12	109
8	01/16/2018	2018-01-16	107
9	01/17/2018	2018-01-17	132
10	01/18/2018	2018-01-18	104

[63]:
```
fig02 = plt.gcf()
fig02.set_size_inches(12, 5)

fig02.suptitle("Eviction Counts Over Time")

ax12 = plt.gca()
# ax = plt.gca().set_xticks(df['year'])
ax12.locator_params(integer=True)
#ax12.set_xticks(evi_df01_s16_post["evi_date"].unique())

evi_df01_s16_post.plot(kind="line",
        x="evi_date",
        y='daily_eviction_counts',
        color="blue",
        ax=ax12)

# plt.xticks(range(1995, 2016, 1))
# plt.yticks(range(0,6,1))
plt.xlabel("Dates")
plt.ylabel("Eviction Counts")
plt.xticks(rotation=45)

# fig.savefig('average-rating.png', dpi=300)
plt.show()
```

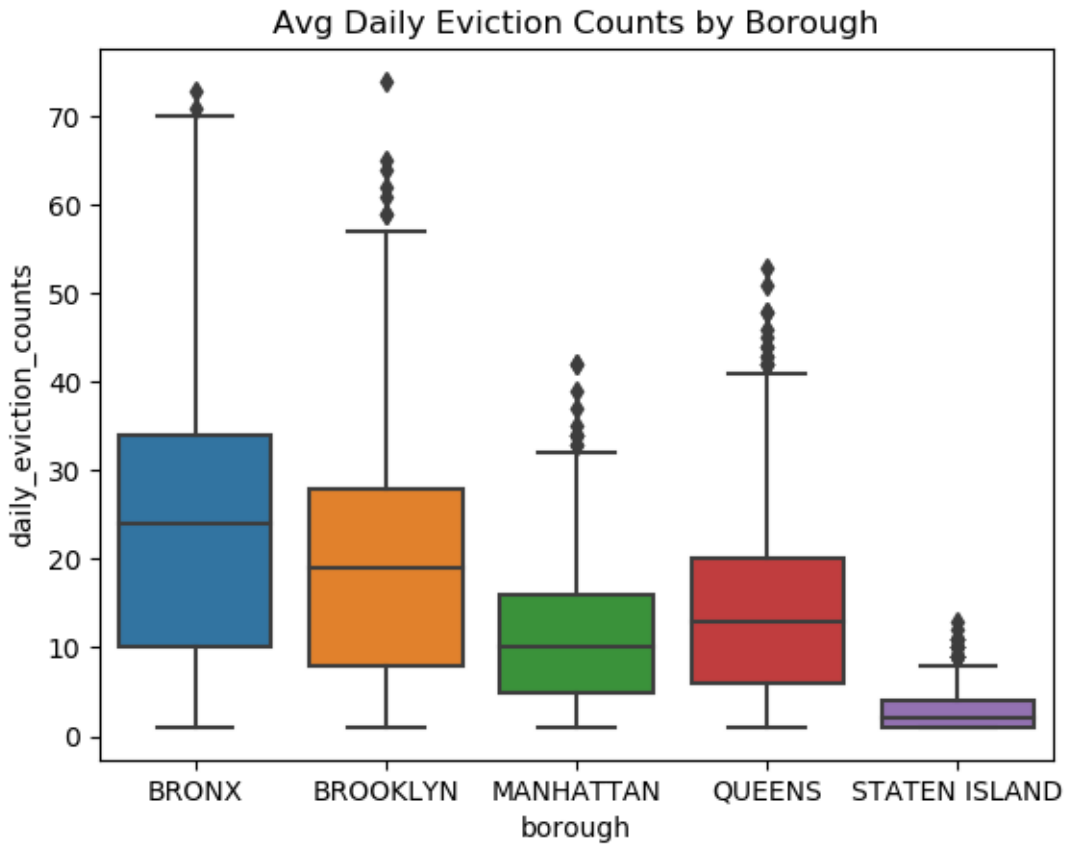


Barplots for select features

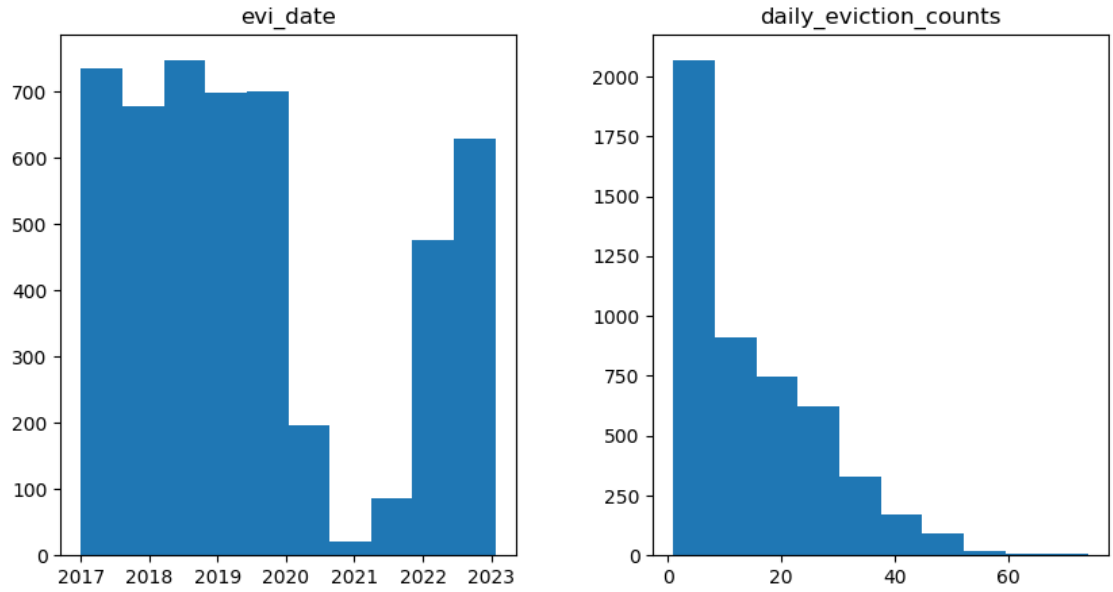
[64]:
```
evbar = sns.barplot(x='borough',
                    y='daily_eviction_counts',
                    data=evi_df01_s06_borough).set(title='Avg Daily Eviction␣
↪Counts by Borough')
```



[65]:
```
evbar = sns.boxplot(x='borough',
                    y='daily_eviction_counts',
                    data=evi_df01_s06_borough).set(title='Avg Daily Eviction␣
↪Counts by Borough')
```



[66]:
```
# histograms
evi_df01_s06_borough.hist(grid=False, figsize=(10,5))
plt.show()
```



Summary stats for select features

[67]:
```
sum_stat_ev = pd.DataFrame(evi_df01_s06_borough['daily_eviction_counts'].
 ↪describe()).T
sum_stat_ev
```

[67]:

	count	mean	std	min	25%	50%	75%	max
daily_eviction_counts	4966.0	14.596859	12.490213	1.0	4.0	12.0	22.75	74.0

Outliers for select features

[68]:
```
# Population
IQR_ev = sum_stat_ev['75%'][0] - sum_stat_ev['25%'][0]
low_outlier_ev = sum_stat_ev['25%'][0] - 1.5*(IQR_ev)
high_outlier_ev = sum_stat_ev['75%'][0] + 1.5*(IQR_ev)
print('Low Outlier (daily_eviction_counts):', low_outlier_ev)
print('High Outlier (daily_eviction_counts):', high_outlier_ev)
```

```
Low Outlier (daily_eviction_counts): -24.125
High Outlier (daily_eviction_counts): 50.875
```

[69]:
```
# Identify presence of outliers
outliers_ev = evi_df01_s06_borough.
 ↪loc[(evi_df01_s06_borough['daily_eviction_counts'] < low_outlier) |␣
 ↪(evi_df01_s06_borough['daily_eviction_counts'] > high_outlier),
                            ['borough', 'daily_eviction_counts']]
```

```
if outliers_ev.empty:
    print('No outliers found in daily_eviction_counts column')
else:
    print('Outliers found in daily_eviction_counts column ({0}):'.
 ↪format(len(outliers_ev)))
    print(outliers_ev)
```

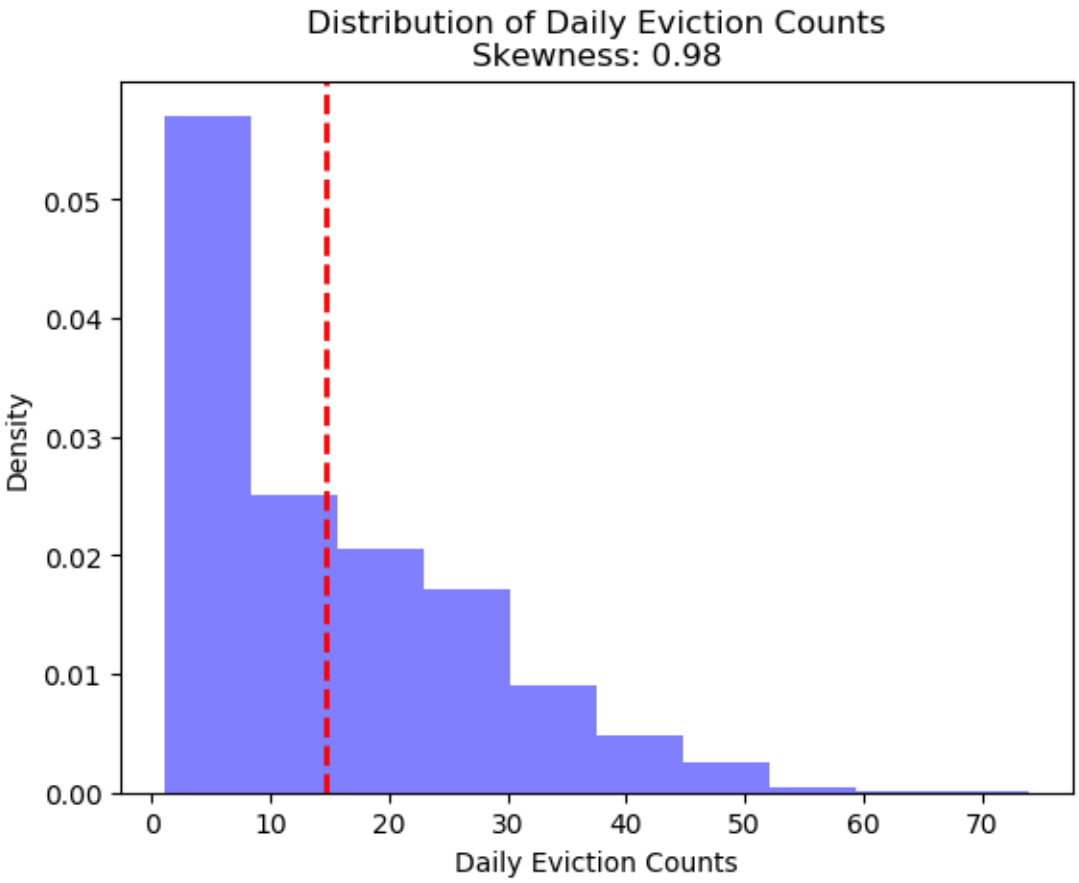
```
No outliers found in daily_eviction_counts column
```

Determine Skew for select features

[70]:
```
# For 'childpoverty' column
daily_eviction_skew = skew(evi_df01_s06_borough['daily_eviction_counts'])
print('Skewness of daily_eviction_counts column:', daily_eviction_skew)
```

```
Skewness of daily_eviction_counts column: 0.9837000262831916
```

[71]:
```
plt.hist(evi_df01_s06_borough['daily_eviction_counts'], density=True, alpha=0.
 ↪5, color='blue')
plt.title('Distribution of Daily Eviction Counts\nSkewness: {0:.2f}'.
 ↪format(daily_eviction_skew))
plt.xlabel('Daily Eviction Counts')
plt.ylabel('Density')
plt.axvline(x=evi_df01_s06_borough['daily_eviction_counts'].mean(),␣
 ↪color='red', linestyle='dashed', linewidth=2)
plt.show()
```



2.4.4 grad_outcomes

[72]:
```
grd_tsv_tbl_name = 'grad_outcomes'
```

Explore via SQL SELECT statements

[73]:
```
# Run query to review a sample of records
grd_total_grads_n01 = "s"

grd_select_borough_stmnt01 = f"""
SELECT * FROM {database_name}.{grd_tsv_tbl_name}
WHERE total_grads_n <> '{grd_total_grads_n01}'
LIMIT 1000
"""

# Display SQL statement
print(grd_select_borough_stmnt01)

# Run SQL statement against Athena table
```

```
grd_df01_s01 = pd.read_sql(grd_select_borough_stmnt01,
                           conn)

# Display results
grd_df01_s01.head(11)
```

```
SELECT * FROM ads508_t8.grad_outcomes
WHERE total_grads_n <> 's'
LIMIT 1000
```

[73]:

	demographic	dbn	school_name	cohort
0	Total Cohort	01M292	HENRY STREET SCHOOL FOR INTERNATIONAL	2004
1	Total Cohort	01M292	HENRY STREET SCHOOL FOR INTERNATIONAL	2005
2	Total Cohort	01M292	HENRY STREET SCHOOL FOR INTERNATIONAL	2006
3	Total Cohort	01M292	HENRY STREET SCHOOL FOR INTERNATIONAL	2006 Aug
4	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	2001
5	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	2002
6	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	2003
7	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	2004
8	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	2005
9	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	2006
10	Total Cohort	01M448	UNIVERSITY NEIGHBORHOOD HIGH SCHOOL	2006 Aug

	total_cohort	total_grads_n	total_grads_perc_cohort	total_regents_n
0	55	37	67.3	17
1	64	43	67.2	27
2	78	43	55.1	36
3	78	44	56.4	37
4	64	46	71.900000000000006	32
5	52	33	63.5	19
6	87	67	77	39
7	112	75	67	36
8	121	64	52.9	35
9	124	53	42.7	42
10	124	60	48.4	42

	total_regents_perc_cohort	total_regents_perc_grads	…
0	30.9	45.9	…
1	42.2	62.8	…
2	46.2	83.7	…
3	47.4	84.1	…
4	50	69.599999999999994	…
5	36.5	57.6	…
6	44.8	58.2	…
7	32.1	48	…

```
8                      28.9                     54.7  …
9                      33.9                     79.2  …
10                     33.9                       70  …

   regents_wo_advanced_n regents_wo_advanced_perc_cohort  \
0                     17                             30.9
1                     27                             42.2
2                     36                             46.2
3                     37                             47.4
4                     25                             39.1
5                     11                             21.2
6                     28              32.200000000000003
7                     30                             26.8
8                     31                             25.6
9                     34                             27.4
10                    34                             27.4

   regents_wo_advanced_perc_grads local_n   local_perc_cohort  \
0                            45.9      20                36.4
1                            62.8      16                  25
2                            83.7       7                   9
3                            84.1       7                   9
4                            54.3      14                21.9
5               33.299999999999997      14                26.9
6                            41.8      28  32.200000000000003
7                              40      39  34.799999999999997
8                            48.4      29                  24
9                            64.2      11                 8.9
10                           56.7      18                14.5

      local_perc_grads still_enrolled_n still_enrolled_perc_cohort  \
0                 54.1               15                        27.3
1   37.200000000000003                9                        14.1
2                 16.3               16                        20.5
3                 15.9               15                        19.2
4                 30.4               10                        15.6
5                 42.4               16                        30.8
6                 41.8                9                        10.3
7                   52               33                        29.5
8                 45.3               41                        33.9
9                 20.8               46                        37.1
10                  30               39                        31.5

   dropped_out_n dropped_out_perc_cohort
0              3                    5.5
1              9                   14.1
2             11                   14.1
```

```
3             11                    14.1
4              6                     9.4
5              1                     1.9
6             11                    12.6
7              4                     3.6
8             11                     9.1
9             20      16.100000000000001
10            20      16.100000000000001

[11 rows x 23 columns]
```

Perform aggregated summaries

[74]:
```
# Run query to review a sample of records
grd_select_hispanic_stmnt01 = f"""
SELECT DISTINCT
    demographic,
    COUNT(*)
FROM {database_name}.{grd_tsv_tbl_name}
GROUP BY demographic
LIMIT 100
"""

# Display SQL statement
print(grd_select_hispanic_stmnt01)

# Run SQL statement against Athena table
grd_df01_s02 = pd.read_sql(grd_select_hispanic_stmnt01,
                           conn)

# Display results
grd_df01_s02.head(11)
```

```
SELECT DISTINCT
    demographic,
    COUNT(*)
FROM ads508_t8.grad_outcomes
GROUP BY demographic
LIMIT 100
```

[74]:

	demographic	_col1
0	Hispanic	2385
1	Asian	1780
2	White	1777
3	Female	2397
4	English Proficient Students	2471

```
5                  Total Cohort   2493
6                          Male   2412
7    Special Education Students   2471
8                         Black   2403
9    General Education Students   2471
10    English Language Learners   2036
```

Load potential predictors and target for further exploration using pandas

[75]:
```
grd_box_stmnt01 = f"""
SELECT
    demographic,
    dbn,
    cohort,
    CAST(total_cohort AS DOUBLE) AS total_cohort,
    CAST(total_grads_n AS DOUBLE) AS total_grads_n,
    CAST(total_regents_n AS DOUBLE) AS total_regents_n,
    CAST(advanced_regents_n AS DOUBLE) AS advanced_regents_n,
    CAST(regents_wo_advanced_n AS DOUBLE) AS regents_wo_advanced_n,
    CAST(local_n AS DOUBLE) AS local_n,
    CAST(still_enrolled_n AS DOUBLE) AS still_enrolled_n,
    CAST(dropped_out_n AS DOUBLE) AS dropped_out_n
FROM {database_name}.{grd_tsv_tbl_name}
WHERE total_grads_n <> '{grd_total_grads_n01}'
LIMIT 50000
"""

# Display SQL statement
print(grd_box_stmnt01)

# Run SQL statement against Athena table
grd_df01_s05 = pd.read_sql(grd_box_stmnt01,
                           conn)

# Display results
grd_df01_s05.head(11)
```

```
SELECT
    demographic,
    dbn,
    cohort,
    CAST(total_cohort AS DOUBLE) AS total_cohort,
    CAST(total_grads_n AS DOUBLE) AS total_grads_n,
    CAST(total_regents_n AS DOUBLE) AS total_regents_n,
    CAST(advanced_regents_n AS DOUBLE) AS advanced_regents_n,
    CAST(regents_wo_advanced_n AS DOUBLE) AS regents_wo_advanced_n,
    CAST(local_n AS DOUBLE) AS local_n,
```

```
    CAST(still_enrolled_n AS DOUBLE) AS still_enrolled_n,
    CAST(dropped_out_n AS DOUBLE) AS dropped_out_n
FROM ads508_t8.grad_outcomes
WHERE total_grads_n <> 's'
LIMIT 50000
```

[75]:

	demographic	dbn	cohort	total_cohort	total_grads_n	\
0	Total Cohort	01M292	2004	55.0	37.0	
1	Total Cohort	01M292	2005	64.0	43.0	
2	Total Cohort	01M292	2006	78.0	43.0	
3	Total Cohort	01M292	2006 Aug	78.0	44.0	
4	Total Cohort	01M448	2001	64.0	46.0	
5	Total Cohort	01M448	2002	52.0	33.0	
6	Total Cohort	01M448	2003	87.0	67.0	
7	Total Cohort	01M448	2004	112.0	75.0	
8	Total Cohort	01M448	2005	121.0	64.0	
9	Total Cohort	01M448	2006	124.0	53.0	
10	Total Cohort	01M448	2006 Aug	124.0	60.0	

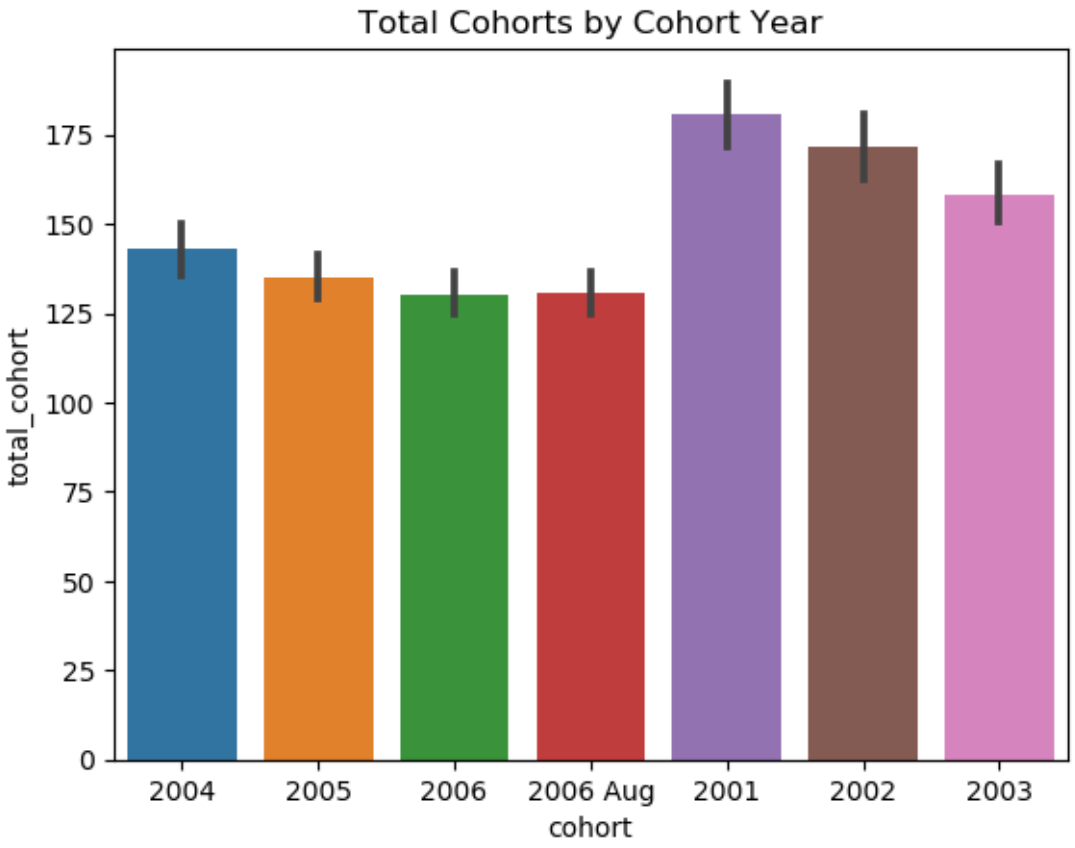
	total_regents_n	advanced_regents_n	regents_wo_advanced_n	local_n	\
0	17.0	0.0	17.0	20.0	
1	27.0	0.0	27.0	16.0	
2	36.0	0.0	36.0	7.0	
3	37.0	0.0	37.0	7.0	
4	32.0	7.0	25.0	14.0	
5	19.0	8.0	11.0	14.0	
6	39.0	11.0	28.0	28.0	
7	36.0	6.0	30.0	39.0	
8	35.0	4.0	31.0	29.0	
9	42.0	8.0	34.0	11.0	
10	42.0	8.0	34.0	18.0	

	still_enrolled_n	dropped_out_n
0	15.0	3.0
1	9.0	9.0
2	16.0	11.0
3	15.0	11.0
4	10.0	6.0
5	16.0	1.0
6	9.0	11.0
7	33.0	4.0
8	41.0	11.0
9	46.0	20.0
10	39.0	20.0

Display barplots for select features

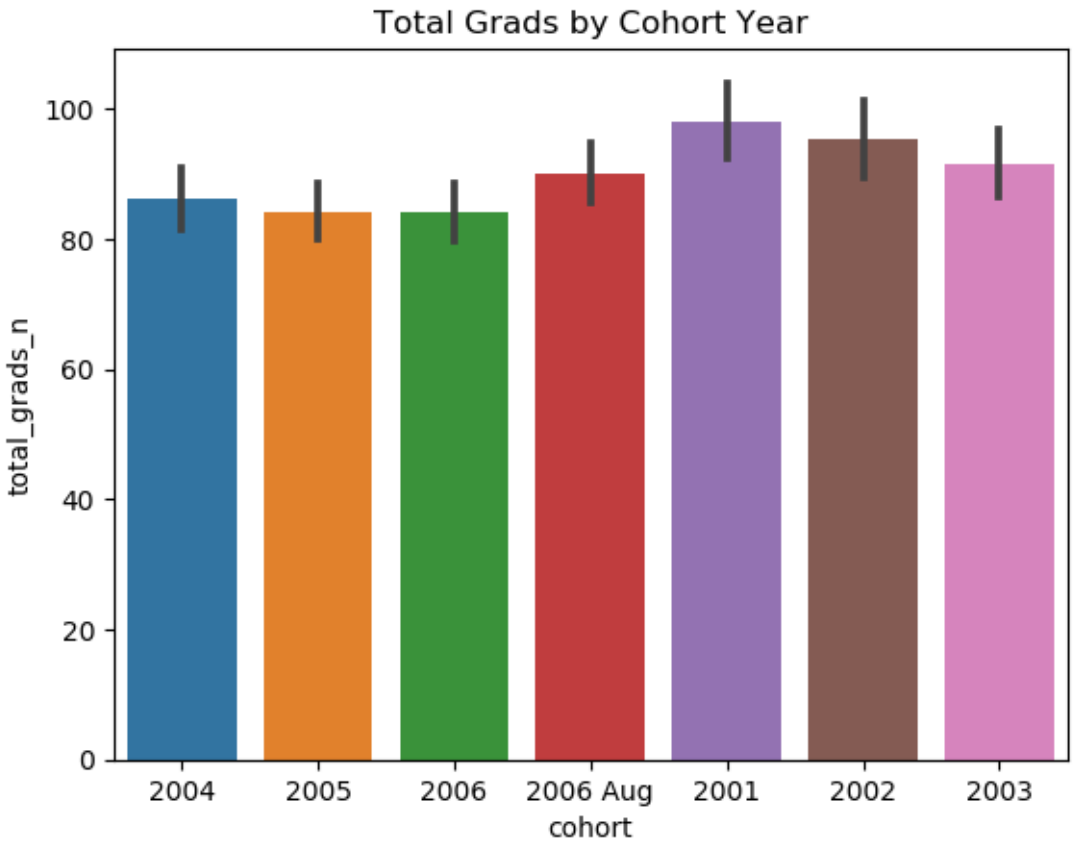
[76]:
```
sns.barplot(x='cohort',
            y='total_cohort',
            data=grd_df01_s05).set(title='Total Cohorts by Cohort Year')
```

[76]: [Text(0.5, 1.0, 'Total Cohorts by Cohort Year')]



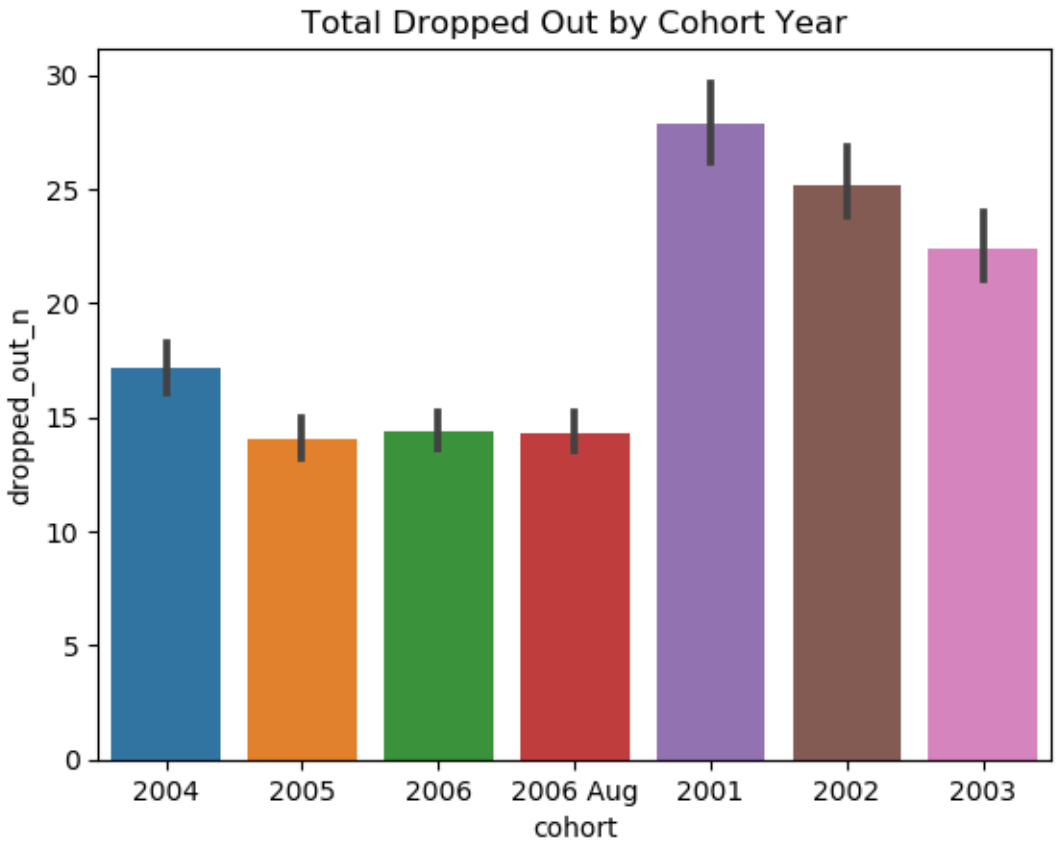
[77]:
```
sns.barplot(x='cohort',
            y='total_grads_n',
            data=grd_df01_s05).set(title='Total Grads by Cohort Year')
```

[77]: [Text(0.5, 1.0, 'Total Grads by Cohort Year')]



[78]:
```
sns.barplot(x='cohort',
            y='dropped_out_n',
            data=grd_df01_s05).set(title='Total Dropped Out by Cohort Year')
```

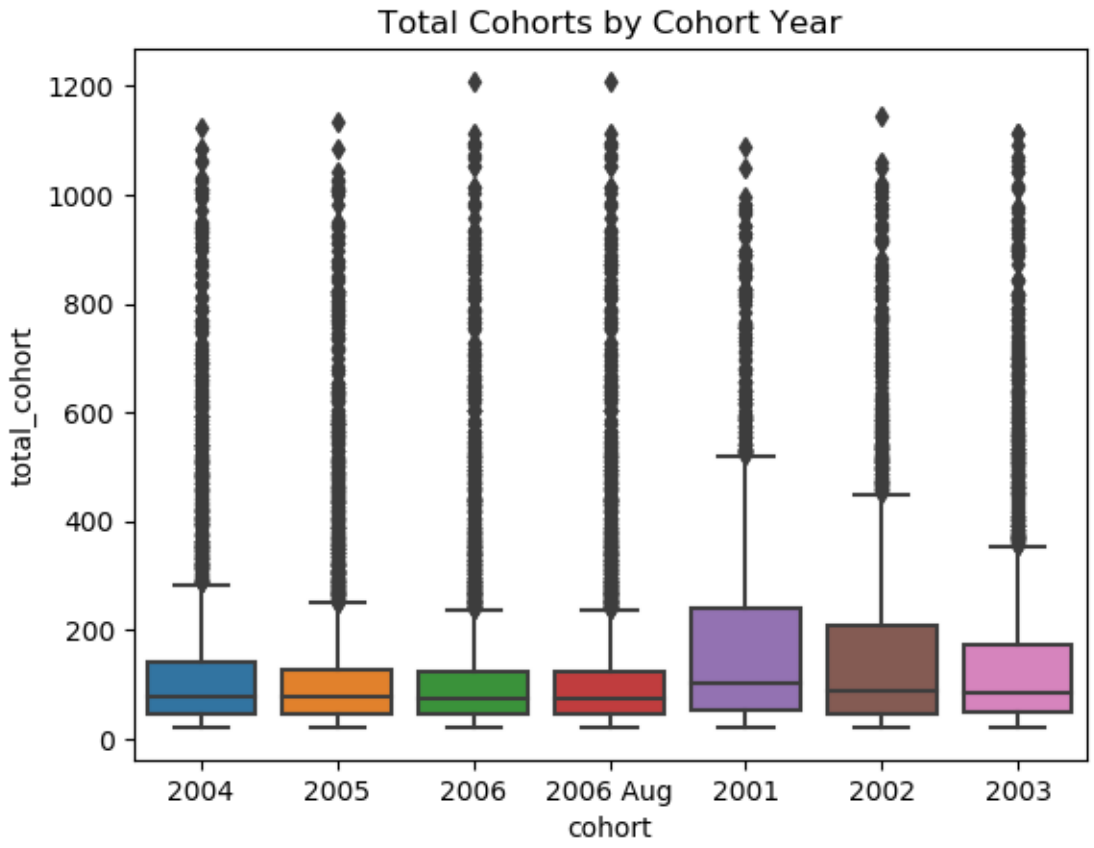
[78]: [Text(0.5, 1.0, 'Total Dropped Out by Cohort Year')]



Display boxplots for select features

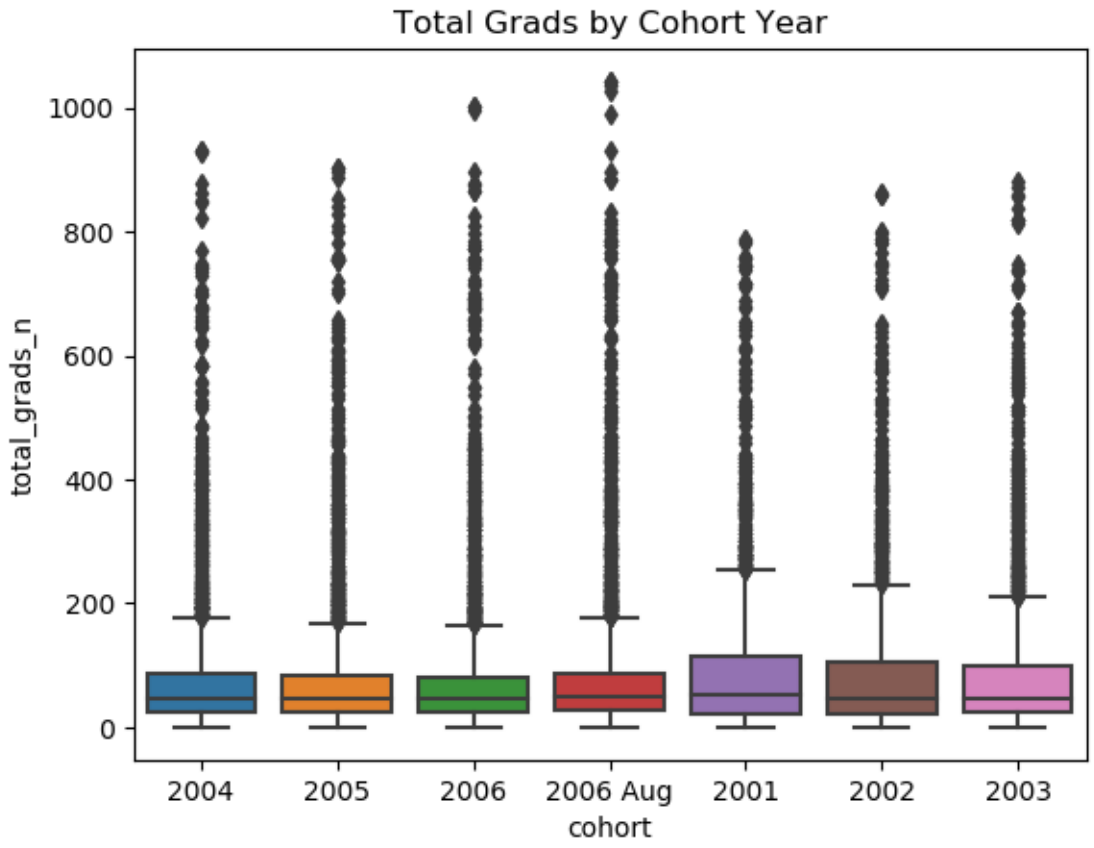
[79]:
```
sns.boxplot(x='cohort',
            y='total_cohort',
            data=grd_df01_s05).set(title='Total Cohorts by Cohort Year')
```

[79]:
```
[Text(0.5, 1.0, 'Total Cohorts by Cohort Year')]
```



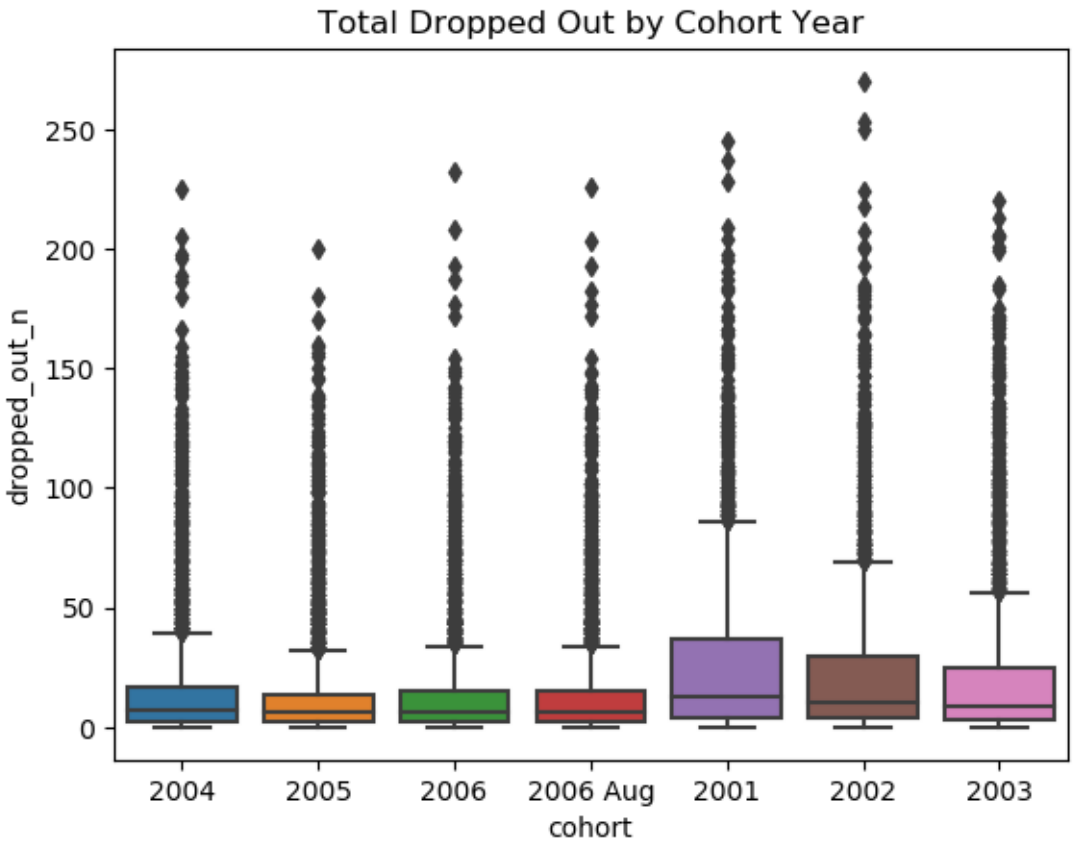
[80]:
```
sns.boxplot(x='cohort',
            y='total_grads_n',
            data=grd_df01_s05).set(title='Total Grads by Cohort Year')
```

[80]: [Text(0.5, 1.0, 'Total Grads by Cohort Year')]

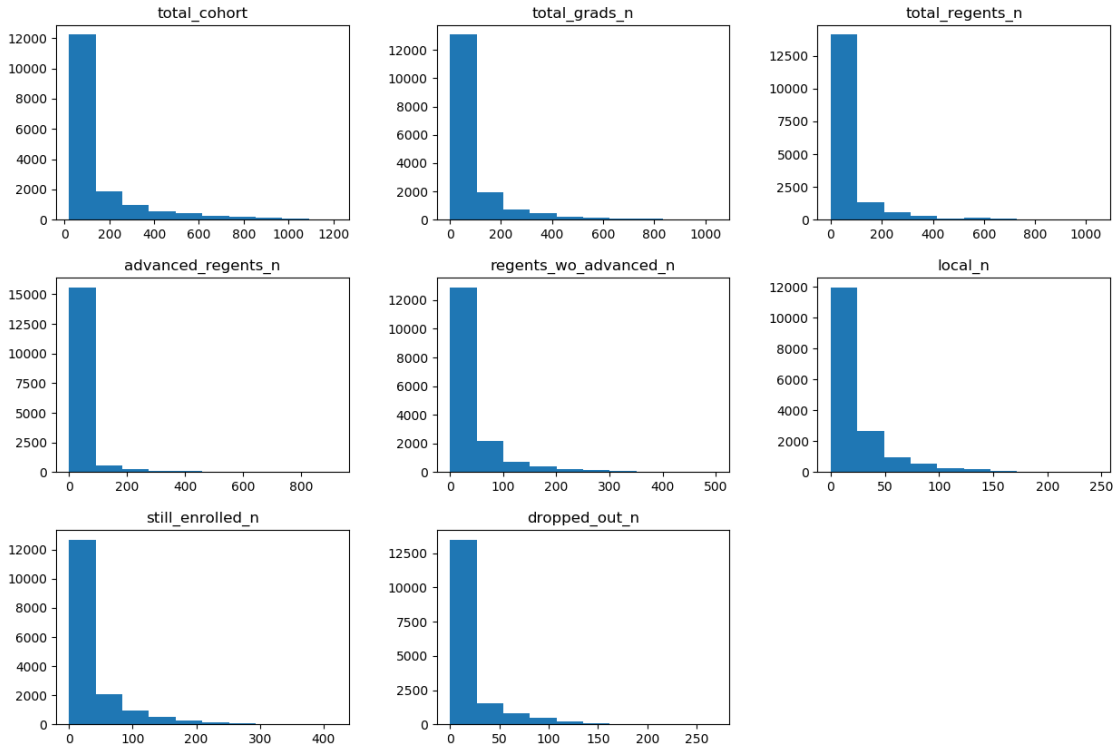


[81]:
```
sns.boxplot(x='cohort',
            y='dropped_out_n',
            data=grd_df01_s05).set(title='Total Dropped Out by Cohort Year')
```

[81]: [Text(0.5, 1.0, 'Total Dropped Out by Cohort Year')]



[82]:
```
# histograms
grd_df01_s05.hist(grid=False, figsize=(15,10))
plt.show()
```



Create subsets of columns for various purposes

[83]:
```
grd_df01_s05_num_lst01 = ['total_cohort',
                          'total_grads_n',
                          'total_regents_n',
                          'advanced_regents_n',
                          'regents_wo_advanced_n',
                          'local_n',
                          'still_enrolled_n',
                          'dropped_out_n'
                         ]

grd_df01_s05_num_lst02 = ['total_cohort',
                          'total_grads_n',
                          'total_regents_n',
                          'advanced_regents_n',
                          'regents_wo_advanced_n',
                          'local_n',
                          'still_enrolled_n',
                          'dropped_out_n'
                         ]

grd_df02_s01 = grd_df01_s05[grd_df01_s05_num_lst01]
```

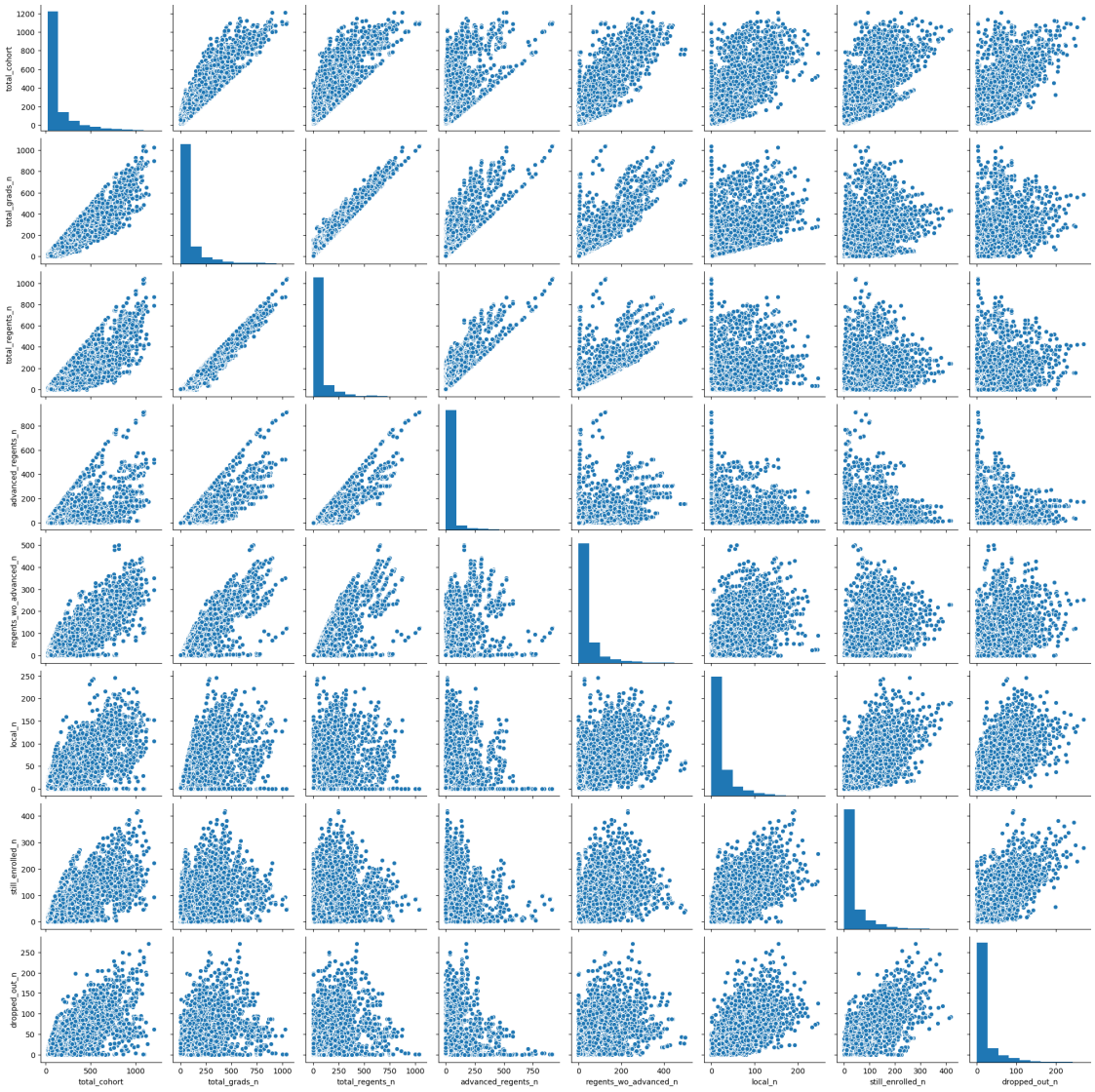
```
grd_df03_s01 = grd_df01_s05[grd_df01_s05_num_lst02]

display(grd_df02_s01.head(5))
```

```
   total_cohort  total_grads_n  total_regents_n  advanced_regents_n  \
0          55.0           37.0             17.0                 0.0
1          64.0           43.0             27.0                 0.0
2          78.0           43.0             36.0                 0.0
3          78.0           44.0             37.0                 0.0
4          64.0           46.0             32.0                 7.0

   regents_wo_advanced_n  local_n  still_enrolled_n  dropped_out_n
0                   17.0     20.0              15.0            3.0
1                   27.0     16.0               9.0            9.0
2                   36.0      7.0              16.0           11.0
3                   37.0      7.0              15.0           11.0
4                   25.0     14.0              10.0            6.0
```

Scatterplots of numerical features

[84]:
```
# Pair scatter plots for selected features
sns.pairplot(grd_df03_s01)
```

[84]: <seaborn.axisgrid.PairGrid at 0x7fba22f48e50>



Summary stats for select features

[85]:
```
sum_stat_totcohort = pd.DataFrame(grd_df01_s05['total_cohort'].describe()).T
sum_stat_totgrad = pd.DataFrame(grd_df01_s05['total_grads_n'].describe()).T
sum_stat_totdrop = pd.DataFrame(grd_df01_s05['dropped_out_n'].describe()).T
sum_stat_grad = pd.concat([sum_stat_totcohort, sum_stat_totgrad,␣
 ↪sum_stat_totdrop])
sum_stat_grad
```

[85]:

	count	mean	std	min	25%	50%	75%	\
total_cohort	16704.0	146.681454	179.345146	21.0	47.0	79.0	151.0	
total_grads_n	16704.0	89.159662	123.704706	0.0	25.0	48.0	93.0	
dropped_out_n	16704.0	18.447079	29.130904	0.0	3.0	7.0	19.0	

```
                  max
total_cohort   1209.0
total_grads_n  1043.0
dropped_out_n   270.0
```

Outliers for select features

[86]:
```
# Total Cohort
IQR_totcohort = sum_stat_totcohort['75%'][0] - sum_stat_totcohort['25%'][0]
low_outlier_totcohort = sum_stat_totcohort['25%'][0] - 1.5*(IQR_totcohort)
high_outlier_totcohort = sum_stat_totcohort['75%'][0] + 1.5*(IQR_totcohort)
print('Low Outlier (total_cohort):', low_outlier_totcohort)
print('High Outlier (total_cohort):', high_outlier_totcohort)

# Total Grads
IQR_totgrad = sum_stat_totgrad['75%'][0] - sum_stat_totgrad['25%'][0]
low_outlier_totgrad = sum_stat_totgrad['25%'][0] - 1.5*(IQR_totgrad)
high_outlier_totgrad = sum_stat_totgrad['75%'][0] + 1.5*(IQR_totgrad)
print('Low Outlier (total_grads_n):', low_outlier_totgrad)
print('High Outlier (total_grads_n):', high_outlier_totgrad)

# Total Dropouts
IQR_totdrop = sum_stat_totdrop['75%'][0] - sum_stat_totdrop['25%'][0]
low_outlier_totdrop = sum_stat_totdrop['25%'][0] - 1.5*(IQR_totdrop)
high_outlier_totdrop = sum_stat_totdrop['75%'][0] + 1.5*(IQR_totdrop)
print('Low Outlier (dropped_out_n):', low_outlier_totdrop)
print('High Outlier (dropped_out_n):', high_outlier_totdrop)
```

```
Low Outlier (total_cohort): -109.0
High Outlier (total_cohort): 307.0
Low Outlier (total_grads_n): -77.0
High Outlier (total_grads_n): 195.0
Low Outlier (dropped_out_n): -21.0
High Outlier (dropped_out_n): 43.0
```

[87]:
```
# Identify presence of outliers
outliers_totcohort = grd_df01_s05.loc[(grd_df01_s05['total_cohort'] <␣
 ↪low_outlier_totcohort) | (grd_df01_s05['total_cohort'] >␣
 ↪high_outlier_totcohort),
                            ['dbn', 'total_cohort']]
if outliers_totcohort.empty:
    print('No outliers found in total_cohort column')
else:
    print('Outliers found in total_cohort column ({0}):'.
 ↪format(len(outliers_totcohort)))
    print(outliers_totcohort)
```

```
outliers_totgrad = grd_df01_s05.loc[(grd_df01_s05['total_grads_n'] <␣
 ↪low_outlier_totgrad) | (grd_df01_s05['total_grads_n'] >␣
 ↪high_outlier_totgrad),
                             ['dbn', 'total_grads_n']]
if outliers_totgrad.empty:
    print('No outliers found in total_grads_n column')
else:
    print('Outliers found in total_grads_n column ({0}):'.
 ↪format(len(outliers_totgrad)))
    print(outliers_totgrad)

outliers_totdrop = grd_df01_s05.loc[(grd_df01_s05['dropped_out_n'] <␣
 ↪low_outlier_totdrop) | (grd_df01_s05['dropped_out_n'] >␣
 ↪high_outlier_totdrop),
                             ['dbn', 'dropped_out_n']]
if outliers_totdrop.empty:
    print('No outliers found in dropped_out_n column')
else:
    print('Outliers found in dropped_out_n column ({0}):'.
 ↪format(len(outliers_totdrop)))
    print(outliers_totdrop)
```

```
Outliers found in total_cohort column (2095):
          dbn  total_cohort
94     02M400         360.0
95     02M400         372.0
96     02M400         312.0
98     02M400         344.0
99     02M400         352.0
…        …             …
16644  31R460         364.0
16645  31R460         353.0
16646  31R460         458.0
16647  31R460         470.0
16648  31R460         470.0

[2095 rows x 2 columns]
Outliers found in total_grads_n column (1852):
          dbn  total_grads_n
94     02M400          271.0
95     02M400          276.0
96     02M400          228.0
97     02M400          236.0
98     02M400          241.0
…        …              …
16644  31R460          247.0
16645  31R460          227.0
```

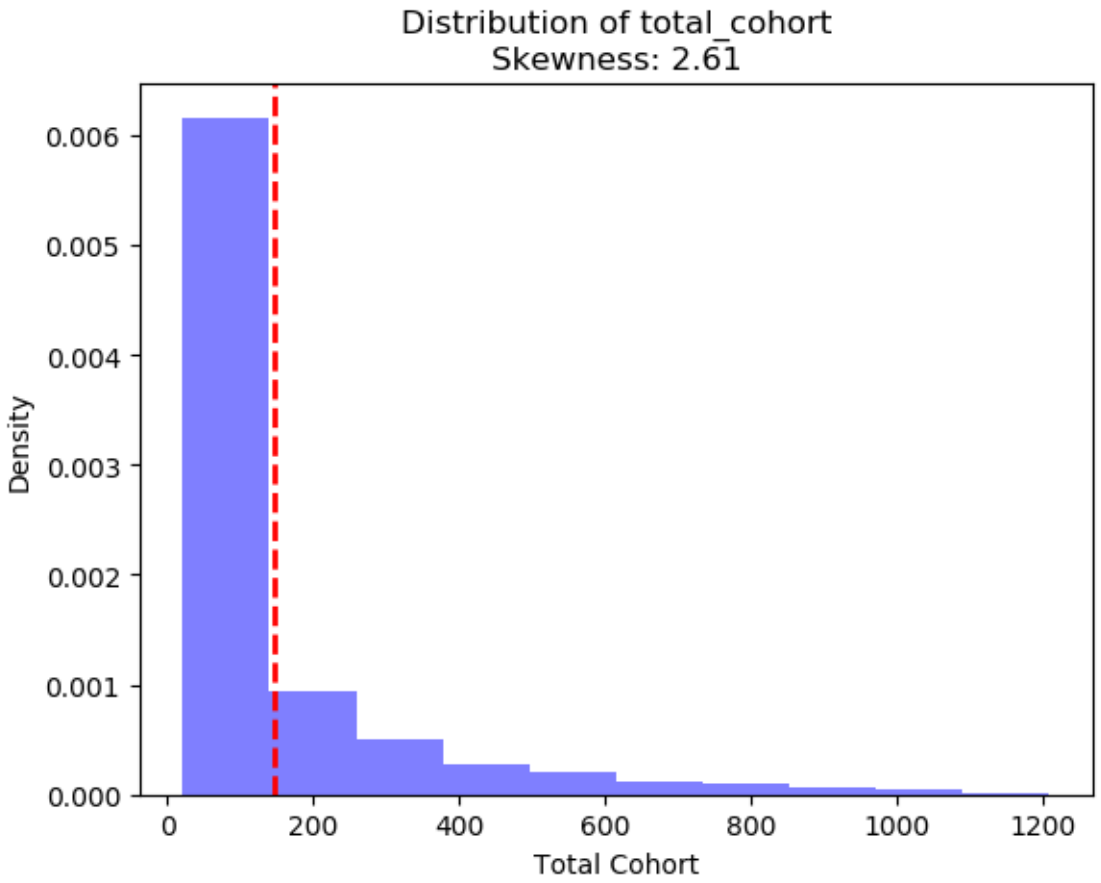
```
16646  31R460          298.0
16647  31R460          328.0
16648  31R460          341.0

[1852 rows x 2 columns]
Outliers found in dropped_out_n column (2111):
          dbn  dropped_out_n
197    02M440          109.0
198    02M440           88.0
199    02M440           92.0
200    02M440           78.0
201    02M440           55.0
…       …             …
16643  31R460           66.0
16647  31R460           45.0
16648  31R460           45.0
16670  32K480           60.0
16671  32K480           77.0

[2111 rows x 2 columns]
```

Determine Skew for select features

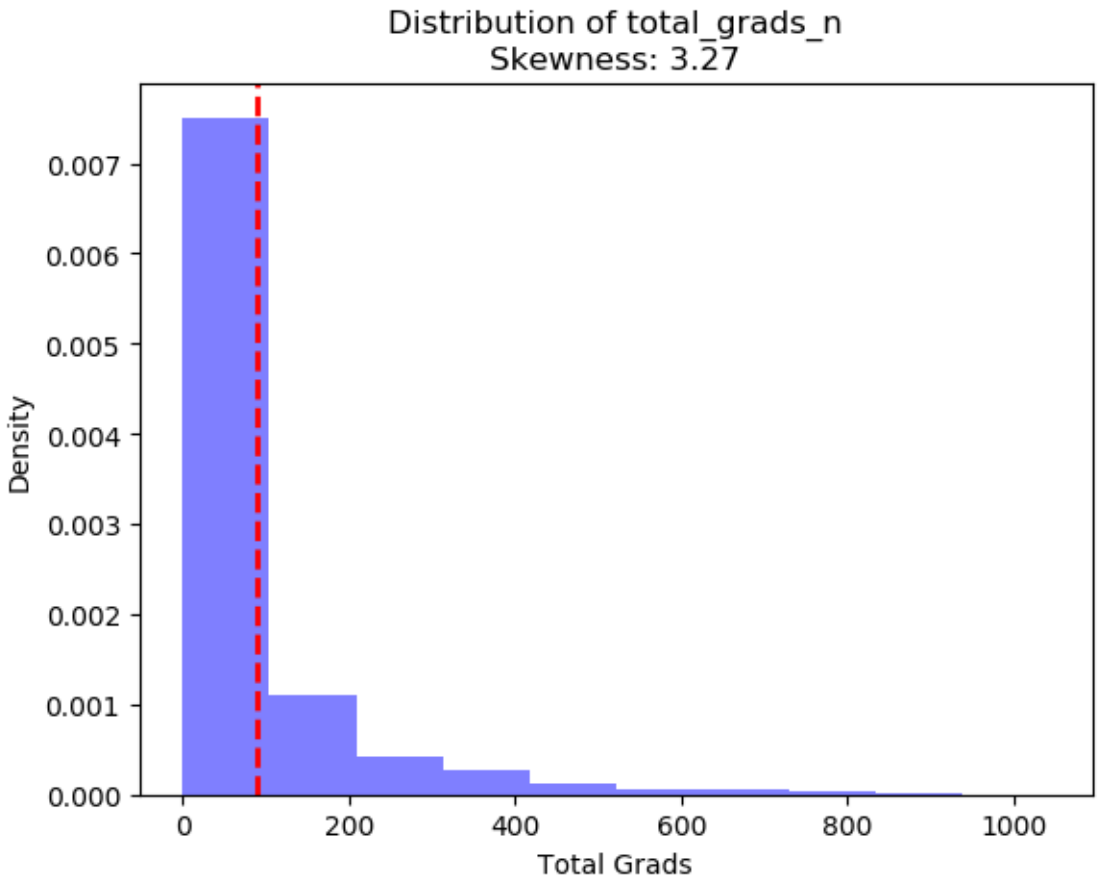
[88]:
```
# For 'total_cohort' column
totcohort_skew = skew(grd_df01_s05['total_cohort'])
print('Skewness of total_cohort column:', totcohort_skew)

# For 'total_grad' column
totgrad_skew = skew(grd_df01_s05['total_grads_n'])
print('Skewness of total_grads_n column:', totgrad_skew)

# For 'dropped_out_n' column
totdrop_skew = skew(grd_df01_s05['dropped_out_n'])
print('Skewness of dropped_out_n column:', totgrad_skew)
```

```
Skewness of total_cohort column: 2.6085451220516616
Skewness of total_grads_n column: 3.2689210059148963
Skewness of dropped_out_n column: 3.2689210059148963
```

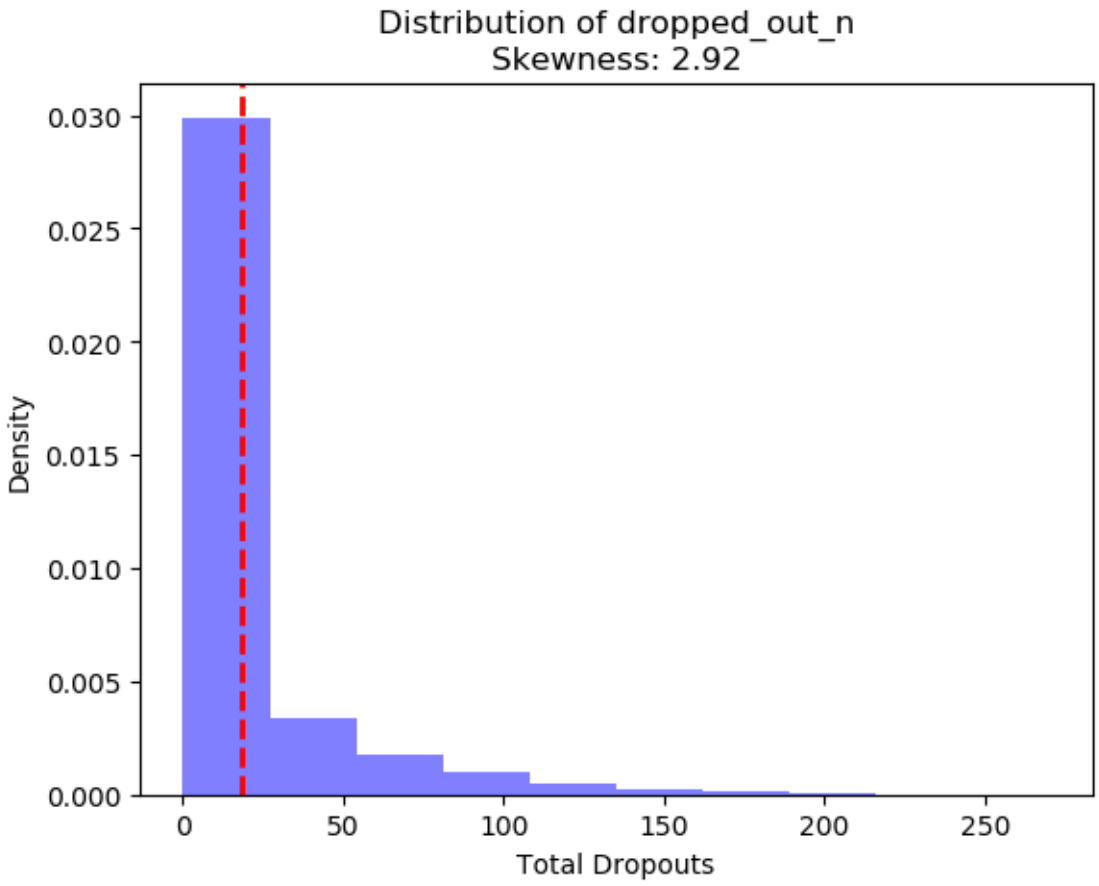
[89]:
```
plt.hist(grd_df01_s05['total_cohort'], density=True, alpha=0.5, color='blue')
plt.title('Distribution of total_cohort\nSkewness: {0:.2f}'.
 ↪format(totcohort_skew))
plt.xlabel('Total Cohort')
plt.ylabel('Density')
plt.axvline(x=grd_df01_s05['total_cohort'].mean(), color='red',␣
 ↪linestyle='dashed', linewidth=2)
plt.show()
```



[90]:
```
plt.hist(grd_df01_s05['total_grads_n'], density=True, alpha=0.5, color='blue')
plt.title('Distribution of total_grads_n\nSkewness: {0:.2f}'.
  ↪format(totgrad_skew))
plt.xlabel('Total Grads')
plt.ylabel('Density')
plt.axvline(x=grd_df01_s05['total_grads_n'].mean(), color='red',␣
  ↪linestyle='dashed', linewidth=2)
plt.show()
```



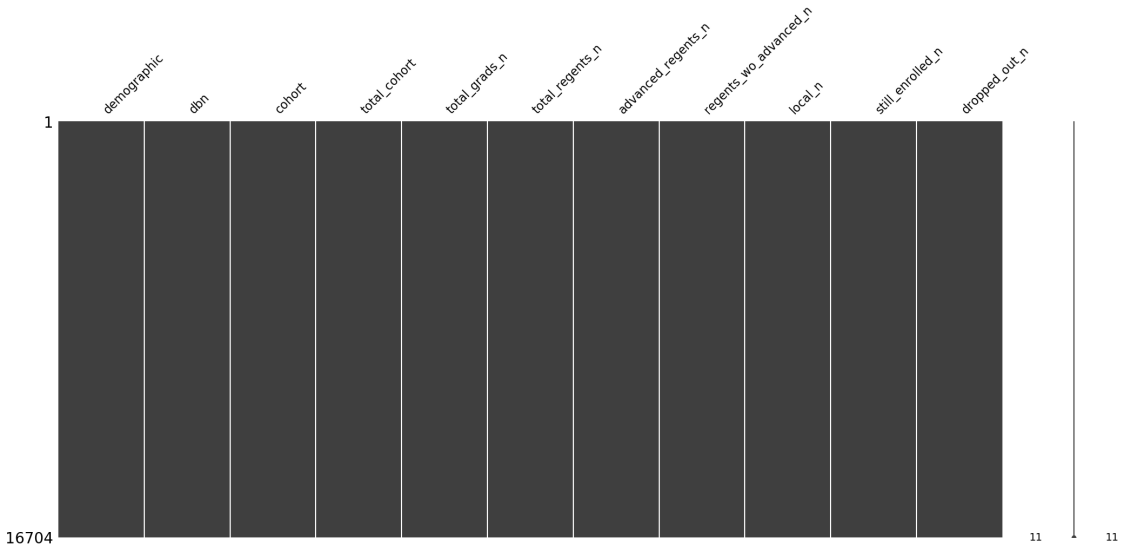
[91]:
```
plt.hist(grd_df01_s05['dropped_out_n'], density=True, alpha=0.5, color='blue')
plt.title('Distribution of dropped_out_n\nSkewness: {0:.2f}'.
  ↪format(totdrop_skew))
plt.xlabel('Total Dropouts')
plt.ylabel('Density')
plt.axvline(x=grd_df01_s05['dropped_out_n'].mean(), color='red',␣
  ↪linestyle='dashed', linewidth=2)
plt.show()
```



Examine features with missing values

[92]:
```
# Visualize missing values in each column
msno.matrix(grd_df01_s05)
```

[92]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fba217470d0>
```



[93]:
```
# Remove any features for which the number of null vals exceed a threshold--
#-- (5% of total N)
grd_df01_s05_null_summ01 = pd.DataFrame(grd_df01_s05.isnull().sum(),
                                        columns=['null_count'])

grd_df01_s05_null_summ02 = grd_df01_s05_null_summ01.
 ↪loc[(grd_df01_s05_null_summ01['null_count'] != 0)].sort_values('null_count',
 ↪                                        ascending=False)
grd_df01_s05_null_summ03 = grd_df01_s05_null_summ02.reset_index()
print(grd_df01_s05_null_summ03)

grd_df01_s05_null_summ04 = grd_df01_s05_null_summ03.
 ↪loc[grd_df01_s05_null_summ03['null_count'] > (len(grd_df01_s05)*.05)]
print('\n', grd_df01_s05_null_summ04)

grd_df01_s05_null_summ04_remove_lst01 = list(grd_df01_s05_null_summ04['index'])
print('\n', grd_df01_s05_null_summ04_remove_lst01)
```

```
Empty DataFrame
Columns: [index, null_count]
Index: []

 Empty DataFrame
Columns: [index, null_count]
Index: []

 []
```

Examine features with near zero variances

[94]:
```
# Review near-zero variance (NZV) features for possible removal
grd_df02_s01_nzv_fit = VarianceThreshold().fit(grd_df02_s01)
grd_df02_s01_nzv_vc01 = grd_df02_s01_nzv_fit.transform(grd_df02_s01)

# Get the names of the selected features
grd_df02_s01_nzv_fit_select_features = grd_df02_s01.
 ↪columns[grd_df02_s01_nzv_fit.get_support()]

grd_df02_s01_nzv_df01 = pd.DataFrame(grd_df02_s01_nzv_vc01,
                                     ␣
 ↪columns=grd_df02_s01_nzv_fit_select_features)

display(grd_df02_s01_nzv_df01.head(5))
print(f'NZV transformed matrix dimensions = {grd_df02_s01_nzv_df01.shape}')

print(f'\n{grd_df02_s01.shape[1] - grd_df02_s01_nzv_df01.shape[1]} near zero␣
 ↪variance features were eliminated')
```

	total_cohort	total_grads_n	total_regents_n	advanced_regents_n	\
0	55.0	37.0	17.0	0.0	
1	64.0	43.0	27.0	0.0	
2	78.0	43.0	36.0	0.0	
3	78.0	44.0	37.0	0.0	
4	64.0	46.0	32.0	7.0	

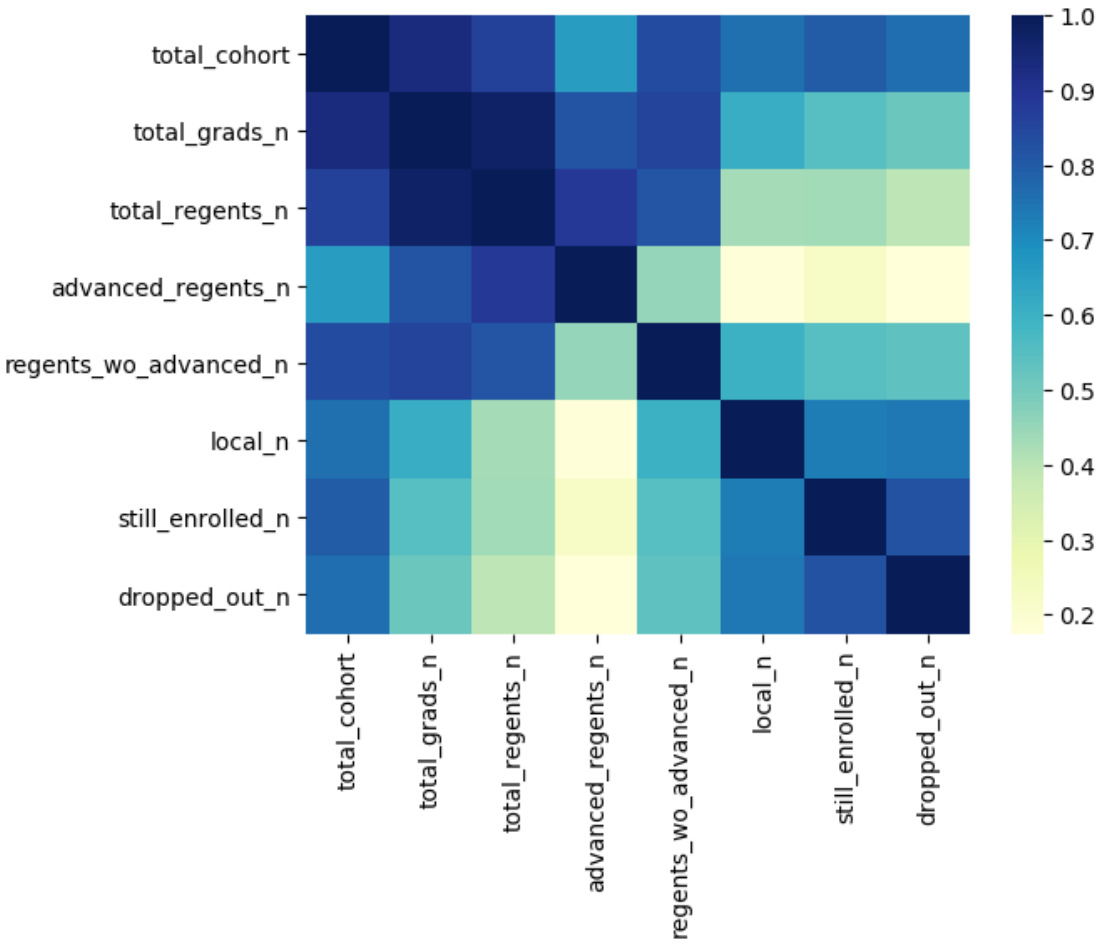
	regents_wo_advanced_n	local_n	still_enrolled_n	dropped_out_n
0	17.0	20.0	15.0	3.0
1	27.0	16.0	9.0	9.0
2	36.0	7.0	16.0	11.0
3	37.0	7.0	15.0	11.0
4	25.0	14.0	10.0	6.0

```
NZV transformed matrix dimensions = (16704, 8)

0 near zero variance features were eliminated
```

Correlations

[95]:
```
# Calculate correlation matrix
grad_corr_matrix = grd_df01_s05.corr()

# Generate heatmap plot
sns.heatmap(grad_corr_matrix, cmap="YlGnBu")
```

[95]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fba21595e10>
```



2.4.5 jobs

[96]:
```
job_tsv_tbl_name = 'jobs'
```

Explore via SQL SELECT statements

[97]:
```
# Run query to review a sample of records
job_agency01 = "housing"

job_select_borough_stmnt01 = f"""
SELECT * FROM {database_name}.{job_tsv_tbl_name}
WHERE LOWER(agency) LIKE '%{job_agency01}%'
LIMIT 100
"""

# Display SQL statement
print(job_select_borough_stmnt01)
```

```
# Run SQL statement against Athena table
job_df01_s01 = pd.read_sql(job_select_borough_stmnt01,
                           conn)

# Display results
job_df01_s01.head(11)
```

```
SELECT * FROM ads508_t8.jobs
WHERE LOWER(agency) LIKE '%housing%'
LIMIT 100
```

[97]:

	job_id	agency	posting_type	num_of_positions	business_title	civil_service_title	title_classification	title_code_no	level
0	573469	HOUSING PRESERVATION & DVLPMNT	External	1	Strategic Program Development Analyst for the …	CITY RESEARCH SCIENTIST	Non-Competitive-5	21744	02
1	568091	HOUSING PRESERVATION & DVLPMNT	External	5	Case Manager for the Division of Tenant Resources	COMMUNITY ASSOCIATE	Non-Competitive-5	56057	00
2	576376	NYC HOUSING AUTHORITY	Internal	1	CARETAKER X	CARETAKER (HA)	Labor-3	90645	00
3	571769	HOUSING PRESERVATION & DVLPMNT	Internal	2	Case Manager for the Division of Tenant Resources	COMMUNITY ASSOCIATE	Non-Competitive-5	56057	00
4	575854	HOUSING PRESERVATION & DVLPMNT	External	1	Data & Analytics Manager, Division of Strategi…	CITY RESEARCH SCIENTIST	Non-Competitive-5	21744	02
5	554300	NYC HOUSING AUTHORITY	External	1	RESIDENT RELOCATION SERVICES COMMUNITY COORDIN…	COMMUNITY COORDINATOR	Non-Competitive-5	56058	00
6	575870	HOUSING PRESERVATION & DVLPMNT	Internal	1	Director of Manhattan Planning for the Divisio…	CITY PLANNER	Competitive-1	22122	03
7	440244	NYC HOUSING AUTHORITY	External	1	Senior Writer				
8	570222	NYC HOUSING AUTHORITY	Internal	1	SENIOR PROJECT MANAGER, ADULT EDUCATION & TRAI…				
9	576674	NYC HOUSING AUTHORITY	External	1	SUPERVISOR OF HOUSING CARETAKER				
10	576328	HOUSING PRESERVATION & DVLPMNT	External	1	Executive Director of Inclusionary Housing for…				

```
7                  AGENCY ATTORNEY    Non-Competitive-5         30087    03
8   ASSOCIATE JOB OPPORTUNITY SPEC        Competitive-1         52316    02
9   SUPERVISOR OF HOUSING CARETAKE        Competitive-1         82011    00
10  ADMINISTRATIVE PROJECT DIRECTO    Non-Competitive-5         95566    M1

                                        job_category  …  \
0                         Policy, Research & Analysis  …
1           Constituent Services & Community Programs  …
2                   Building Operations & Maintenance  …
3           Constituent Services & Community Programs  …
4                         Policy, Research & Analysis  …
5           Constituent Services & Community Programs  …
6               Engineering, Architecture, & Planning  …
7           Legal Affairs Policy, Research & Analysis  …
8           Constituent Services & Community Programs  …
9   Building Operations & Maintenance Public Safet…  …
10                Finance, Accounting, & Procurement  …

                                 additional_information  \
0   We engage New Yorkers to build and sustain nei…
1   Determination and verification of eligibility â ¢
2   Prepare apartments for move outs.   Please rea…
3   Determination and verification of eligibility â ¢
4   We engage New Yorkers to build and sustain nei…
5                                                 "1.
6   We engage New Yorkers to build and sustain nei…
7   Conducting operational analysis to understand …
8                                                 "1.
9                         Handle tenant lockouts.  4.
10  We engage New Yorkers to build and sustain nei…

                                              to_apply  \
0   Continue to work on the implementation of Loca…
1   Client briefings {internal and external meetin…
2   Qualification Requirements  There are no forma…
3   Client briefings {internal and external meetin…
4   Gather, prepare, and merge large datasets from…
5   Preference will be given to employees who have…
6   Planning & Predevelopment (P&P) is central to …
7   Conducting independent research of varying dif…
8   Preference will be given to employees who have…
9   Fill out work orders as a result of apartment …
10  1. A baccalaureate degree from an accredited c…

                                       hours_or_shift  \
0   Retrieve and review affordable housing regulat…
1   May perform community outreach to assist Secti…
```

```
2
3   May perform community outreach to assist Secti…
4   Create performance metrics for lottery and hom…
5           NYCHA residents are encouraged to apply."
6   Neighborhood Development & Stabilization (ND&S…
7   Organizing complex text and processes in seque…
8           NYCHA residents are encouraged to apply."
9   Report any hazardous conditions observed in an…
10  Candidates should have a record of achieving r…

                                      work_location_1  \
0   Research initiatives in other jurisdictions or…
1   Prepare and send appropriate correspondence, t…
2                                                "1.
3   Prepare and send appropriate correspondence, t…
4   Manage and analyze eviction filing data to und…
5                          Click the Apply Now button.
6   Promote HPD and City policy objectives across …
7   Leading meetings with subject matter experts t…
8                          Click the Apply Now button.
9   One year of permanent service in the title of …
10  This position is also open to qualified person…

                                    recruitment_contact  \
0   Understand and leverage existing Agency datase…
1   Document case files and electronic records, fi…
2   Possession of a valid driver's license is requ…
3   Document case files and electronic records, fi…
4   Prepare analytic reports to inform program des…
5
6   Define, manage, and track team priorities and …
7   Editing documents of a high degree of difficul…
8
9
10                                       Apply online

                                   residency_requirement  \
0   Summarizing and communicating findingsâ quali…
1                              Rent calculations  â ¢
2   Preference will be given to employees who have…
3                              Rent calculations  â ¢
4   Support the implementation of data-driven prog…
5
6   Meet regularly with individual staff members a…
7   Working with the Compliance Integration Report…
8
9                                                "1.
```

```
10

                                        posting_date  \
0   Contributing to the rollout of new initiatives…
1   Review of yearly recertificationâ s of househ…
2           NYCHA residents are encouraged to apply."
3   Review of yearly recertificationâ s of househ…
4   Develop strategies for data integration and au…
5
6   Identify staffing needs and advocate for resou…
7   Working with the Compliance Monitoring Unit to…
8
9   For NYCHA employees: This position is open as …
10                                    100 Gold Street

                                          post_until  \
0   Conducting special research, analytical, or co…
1   Demonstrate ability to manage multiple cases w…
2                         Click the Apply now button.
3   Demonstrate ability to manage multiple cases w…
4   Assist SOA colleagues with quantitative analys…
5                NYCHA has no residency requirements.
6   Ensure that all projects move efficiently thro…
7   Working with Compliance Inquiry Review and Ass…
8                NYCHA has no residency requirements.
9   For NYCHA employees: Preference will be given …
10

                                       posting_updated  \
0   Adhering to work plans and internal and extern…
1                         Attend mandatory trainings"
2
3                         Attend mandatory trainings"
4   Respond to ad hoc data requests from programs,…
5                                          10/28/2022
6   Identify risks and troubleshoot problems, invo…
7   Coordinating with NYCHA department heads regar…
8                                          02/14/2023
9           NYCHA residents are encouraged to apply."
10  New York City residency is generally required …

                                         process_date
0   Participating in meetings, presentations, and …
1   Qualification Requirements  1. High school gra…
2
3   Qualification Requirements  1. High school gra…
4   1.  For Assignment Level I (only physical, bio…
```

```
5
6   Create, implement, and maintain consistent, ef…
7   Drafting complex documents based on important …
8
9                          Click the Apply Now button.
10                                          02/24/2023

[11 rows x 30 columns]
```

Perform aggregated summaries

[98]:
```
# Run query to review a sample of records
job_select_job_category_stmnt01 = f"""
SELECT DISTINCT
    job_category,
    COUNT(*)
FROM {database_name}.{job_tsv_tbl_name}
WHERE job_category IS NULL
GROUP BY job_category
LIMIT 100
"""

# Display SQL statement
print(job_select_job_category_stmnt01)

# Run SQL statement against Athena table
job_df01_s02 = pd.read_sql(job_select_job_category_stmnt01,
                           conn)

# Display results
print(job_df01_s02.shape)
display(job_df01_s02.head(11))
```

```
SELECT DISTINCT
    job_category,
    COUNT(*)
FROM ads508_t8.jobs
WHERE job_category IS NULL
GROUP BY job_category
LIMIT 100

(0, 2)
```

```
Empty DataFrame
Columns: [job_category, _col1]
Index: []
```

[99]:
```
job_summ_borough_stmnt01 = f"""
SELECT
    job_id,
    COUNT(*) AS jobs_count
FROM {database_name}.{job_tsv_tbl_name}
GROUP BY job_id
LIMIT 100
"""

# Display SQL statement
print(job_summ_borough_stmnt01)

# Run SQL statement against Athena table
job_df01_s03 = pd.read_sql(job_summ_borough_stmnt01,
                           conn)

# Display results
job_df01_s03.head(11)
```

```
SELECT
    job_id,
    COUNT(*) AS jobs_count
FROM ads508_t8.jobs
GROUP BY job_id
LIMIT 100
```

[99]:

	job_id	jobs_count
0	536549	2
1	572832	2
2	571272	2
3	527818	2
4	536536	2
5	568908	2
6	573773	2
7	556510	2
8	528055	2
9	575992	2
10	500566	2

[100]:
```
job_summ_salary_range_from01 = f"""
SELECT
    job_id,
    job_category,
    salary_range_from,
    salary_range_to
FROM {database_name}.{job_tsv_tbl_name}
```

```
WHERE salary_range_from LIKE '%anager%'
LIMIT 10000
"""

# Display SQL statement
print(job_summ_salary_range_from01)

# Run SQL statement against Athena table
job_df01_s04 = pd.read_sql(job_summ_salary_range_from01,
                           conn)

# Display results
job_df01_s04.head(100)
```

```
SELECT
    job_id,
    job_category,
    salary_range_from,
    salary_range_to
FROM ads508_t8.jobs
WHERE salary_range_from LIKE '%anager%'
LIMIT 10000
```

[100]:

	job_id	job_category	salary_range_from	salary_range_to
0	561954	00	Manager	40000
1	540477	00	Experienced (non-manager)	80440
2	540190	00	Experienced (non-manager)	80440
3	561954	00	Manager	40000
4	566429	02	Experienced (non-manager)	50972
5	566429	02	Experienced (non-manager)	50972
6	540190	00	Experienced (non-manager)	80440
7	540477	00	Experienced (non-manager)	80440

Load potential predictors and target for further exploration using pandas

[101]:
```
job_box_stmnt01 = f"""
SELECT
    agency,
    posting_type,
    num_of_positions,
    business_title,
    civil_service_title,
    title_classification,
    title_code_no,
    level,
    job_category,
```

```
    fulltime_or_parttime_indicator,
    career_level,
    CAST(salary_range_from AS DOUBLE) AS salary_range_from,
    CAST(salary_range_to AS DOUBLE) AS salary_range_to,
    salary_frequency,
    work_location,
    division_or_work_unit,
    job_description,
    minimum_qual_requirements,
    preferred_skills,
    additional_information,
    to_apply,
    hours_or_shift,
    work_location_1,
    recruitment_contact,
    residency_requirement,
    posting_date,
    post_until,
    posting_updated,
    process_date
FROM {database_name}.{job_tsv_tbl_name}
WHERE salary_range_from NOT LIKE '%anager%'
LIMIT 5000
"""

# Display SQL statement
print(job_box_stmnt01)

# Run SQL statement against Athena table
job_df01_s05 = pd.read_sql(job_box_stmnt01,
                           conn)

# Display results
job_df01_s05.head(11)
```

```
SELECT
    agency,
    posting_type,
    num_of_positions,
    business_title,
    civil_service_title,
    title_classification,
    title_code_no,
    level,
    job_category,
    fulltime_or_parttime_indicator,
    career_level,
```

```
    CAST(salary_range_from AS DOUBLE) AS salary_range_from,
    CAST(salary_range_to AS DOUBLE) AS salary_range_to,
    salary_frequency,
    work_location,
    division_or_work_unit,
    job_description,
    minimum_qual_requirements,
    preferred_skills,
    additional_information,
    to_apply,
    hours_or_shift,
    work_location_1,
    recruitment_contact,
    residency_requirement,
    posting_date,
    post_until,
    posting_updated,
    process_date
FROM ads508_t8.jobs
WHERE salary_range_from NOT LIKE '%anager%'
LIMIT 5000
```

[101]:

	agency	posting_type	num_of_positions	business_title
0	DEPARTMENT OF BUILDINGS	Internal	3	Plan Examiner
1	OFFICE OF LABOR RELATIONS	Internal	13	Data Processor
2	ADMIN FOR CHILDREN'S SVCS	External	1	Security Consultant, Horizon
3	DEPARTMENT OF TRANSPORTATION	Internal	1	Civil Engineer Level -3
4	HRA/DEPT OF SOCIAL SERVICES	External	1	DIRECTOR, SNT PROGRAM
5	DEPT OF ENVIRONMENT PROTECTION	Internal	2	CHIEF MARINE ENGINEER (DIESEL)
6	ADMIN FOR CHILDREN'S SVCS	External	1	Child Protective Manager
7	DEPT OF HEALTH/MENTAL HYGIENE	External	1	Payment Analyst
8	DEPT OF ENVIRONMENT PROTECTION	Internal	1	MS4 Deputy Program Manager
9	DEPT OF ENVIRONMENT PROTECTION	Internal	2	2023-BWS-016-Water Treatment Operations Engine…
10	OFFICE OF LABOR RELATIONS	External	13	

```
10                                  Data Processor

              civil_service_title title_classification title_code_no level  \
0            PLAN EXAMINER (BLDGS)        Competitive-1         22410    00
1              COMMUNITY ASSISTANT    Non-Competitive-5         56056    00
2            COMMUNITY COORDINATOR    Non-Competitive-5         56058    00
3                  CIVIL ENGINEER        Competitive-1         20215    03
4         EXECUTIVE AGENCY COUNSEL    Non-Competitive-5         95005    M2
5   CHIEF MARINE ENGINEER (DIESEL)        Competitive-1         91523    00
6   DIRECTOR OF FIELD OPERATIONS (    Non-Competitive-5         95600    M1
7               MANAGEMENT AUDITOR        Competitive-1         40502    01
8          CITY RESEARCH SCIENTIST    Non-Competitive-5         21744    02
9            SUMMER COLLEGE INTERN    Non-Competitive-5         10234    00
10             COMMUNITY ASSISTANT    Non-Competitive-5         56056    00

                                        job_category  \
0               Engineering, Architecture, & Planning
1                    Administration & Human Resources
2            Public Safety, Inspections, & Enforcement
3               Engineering, Architecture, & Planning
4   Administration & Human Resources Constituent S…
5               Engineering, Architecture, & Planning
6                                    Social Services
7                  Finance, Accounting, & Procurement
8   Engineering, Architecture, & Planning Policy, …
9               Engineering, Architecture, & Planning
10                   Administration & Human Resources

   fulltime_or_parttime_indicator  …  \
0                               F  …
1                                  …
2                               P  …
3                               F  …
4                               F  …
5                               F  …
6                               F  …
7                               F  …
8                               F  …
9                               F  …
10                                 …

                              additional_information  \
0                     Reviews site safety plans.  â ¢
1   PLEASE NOTE:  THIS IS A TEMPORARY POSITION UNT…
2   Section 424-A of the New York Social Services …
3   The City of New York is an inclusive equal opp…
4   Establish and maintain professional relationsh…
```

```
5   Appointments are subject to OMB approval.  For…
6   Section 424-A of the New York Social Services …
7   Troubleshoots and resolve stakeholdersâ  issu…
8   Supporting the MS4 program Manager in preparin…
9
10  PLEASE NOTE:  THIS IS A TEMPORARY POSITION UNT…

                                              to_apply  \
0   Engages in research, investigations, studies a…
1   TO APPLY PLEASE SUBMIT YOUR COVER LETTER AND R…
2                      Click on the Apply button now.
3   Resumes may be submitted electronically using …
4   Develop and implement policy changes that make…
5                               Click Apply Now button
6   APPLICATIONS MUST BE SUBMITTED ELECTRONICALLY …
7   Review invoices to ensure accurate information…
8   Leading the implementation of the MS4 Permit P…
9   To Apply click the â Apply Nowâ  button  DEP…
10  TO APPLY PLEASE SUBMIT YOUR COVER LETTER AND R…

                                         hours_or_shift  \
0   Explains and enforces rules and laws to public…
1
2
3                                             35 Hours
4   Use online legal research applications to cond…
5                             40 hours per week / day
6
7   Respond to vendor/program inquiries and concer…
8   Supporting the MS4 Program Manager in preparin…
9                                   35 Hours per week
10

                                       work_location_1  \
0   Testifies as needed to support violations issu…
1
2
3                                   55 Water St Ny Ny
4   Liaise with DSS AO and ITS staff to update rel…
5                                  Wards Island, N.Y.
6
7                  Performs other duties as assigned"
8   Participating in conferences, meetings, semina…
9   This position is located in Westchester, New Y…
10

                                    recruitment_contact  \
```

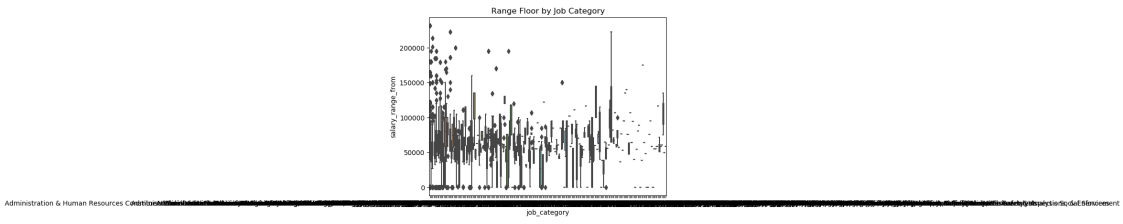
```
0   Trains, mentors and collaborates with new assi…
1
2
3
4   Oversee OLTâ s robust legal internship progra…
5
6
7   1. A baccalaureate degree from an accredited c…
8   collecting and analyzing water quality data, c…
9
10


                               residency_requirement  \
0   May be required to perform fieldwork to make o…
1   New York City residency is generally required …
2   New York City residency is generally required …
3   New York City Residency is not required for th…
4   Admission to the New York State Bar; and four …
5   New York City Residency is not required for th…
6   New York City Residency is not required for th…
7   Our successful candidate should possess; excel…
8   Supporting other BEPA initiatives as required …
9   New York City residency is not required for th…
10  New York City residency is generally required …

                                          posting_date  \
0   May operate a motor vehicle in performance of …
1                                          12/29/2022
2                                          08/19/2022
3                                          09/27/2022
4                                                "â ¢
5                                          06/27/2022
6                                          03/01/2023
7   â ¢ Selected candidates will be required to pr…
8   Assisting MS4 Program Manager in managing MS4-…
9                                          02/19/2023
10                                         12/29/2022

                                           post_until  \
0   1. License or Registration Requirement: A vali…
1
2
3
4                        Knowledge of Medicaid.  â ¢
5
6                                        31-MAR-2023
7   TO APPLY, PLEASE SUBMIT RESUME AND COVER LETTE…
```

```
8   Assisting MS4 Program Manager in the coordinat…
9                                          02-JUN-2023
10

                                      posting_updated  \
0                                                  "â ¢
1                                           03/06/2023
2                                           08/19/2022
3                                           10/04/2022
4   Supplemental Needs Trust knowledge a plus.  â ¢
5                                           09/08/2022
6                                           03/01/2023
7
8      Contract management and budget tracking  â ¢
9                                           02/24/2023
10                                          03/06/2023

                                           process_date
0   Knowledge of the NYC Construction Code and Zon…
1                                            03/07/2023
2                                            03/07/2023
3                                            03/07/2023
4   Strong strategic and creative management skill…
5                                            03/07/2023
6                                            03/07/2023
7
8   Excellent written and oral communications skil…
9                                            03/07/2023
10                                           03/07/2023

[11 rows x 29 columns]
```

Display boxplots for select features

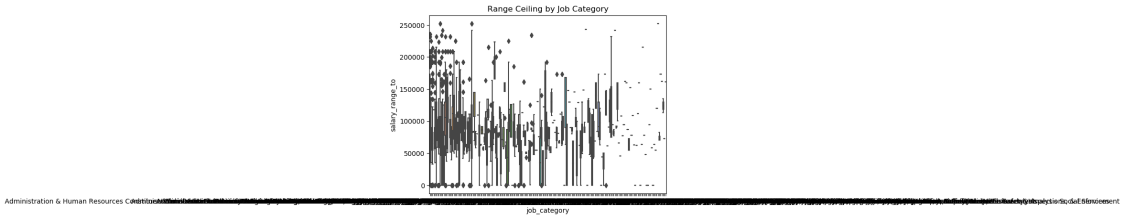
[102]:
```
sns.boxplot(x='job_category',
            y='salary_range_from',
            data=job_df01_s05).set(title='Range Floor by Job Category')
```

[102]:
```
[Text(0.5, 1.0, 'Range Floor by Job Category')]
```

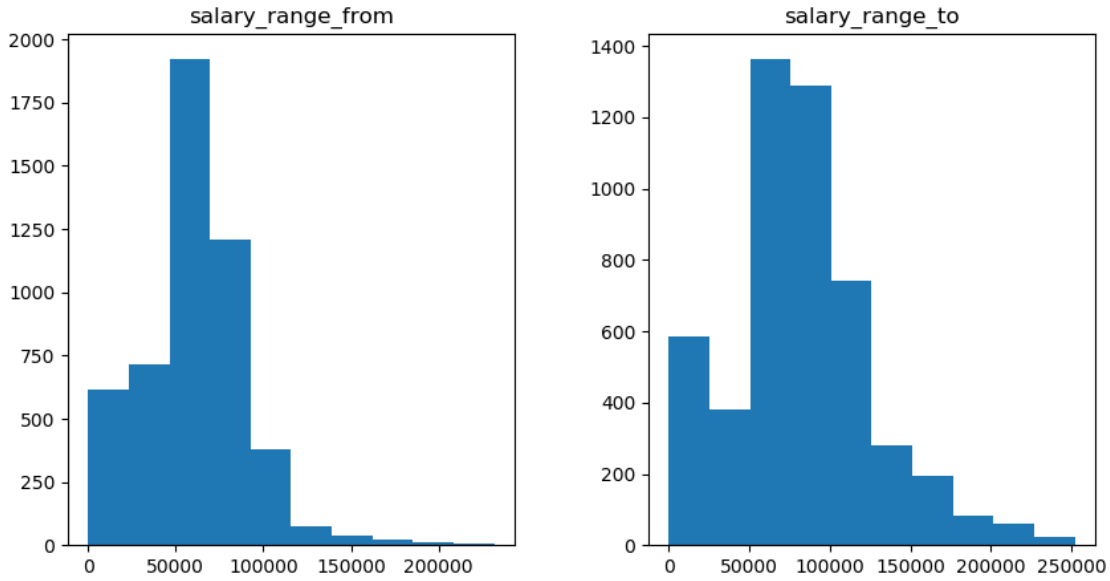


[103]:
```
sns.boxplot(x='job_category',
            y='salary_range_to',
            data=job_df01_s05).set(title='Range Ceiling by Job Category')
```

[103]:
```
[Text(0.5, 1.0, 'Range Ceiling by Job Category')]
```



[104]:
```
# histograms
job_df01_s05.hist(grid=False, figsize=(10,5))
plt.show()
```



Create subsets of columns for various purposes

[105]:
```
job_df01_s05_num_lst01 = ['salary_range_from',
                          'salary_range_to'
                         ]

job_df01_s05_num_lst02 = ['salary_range_from',
                          'salary_range_to'
                         ]
```

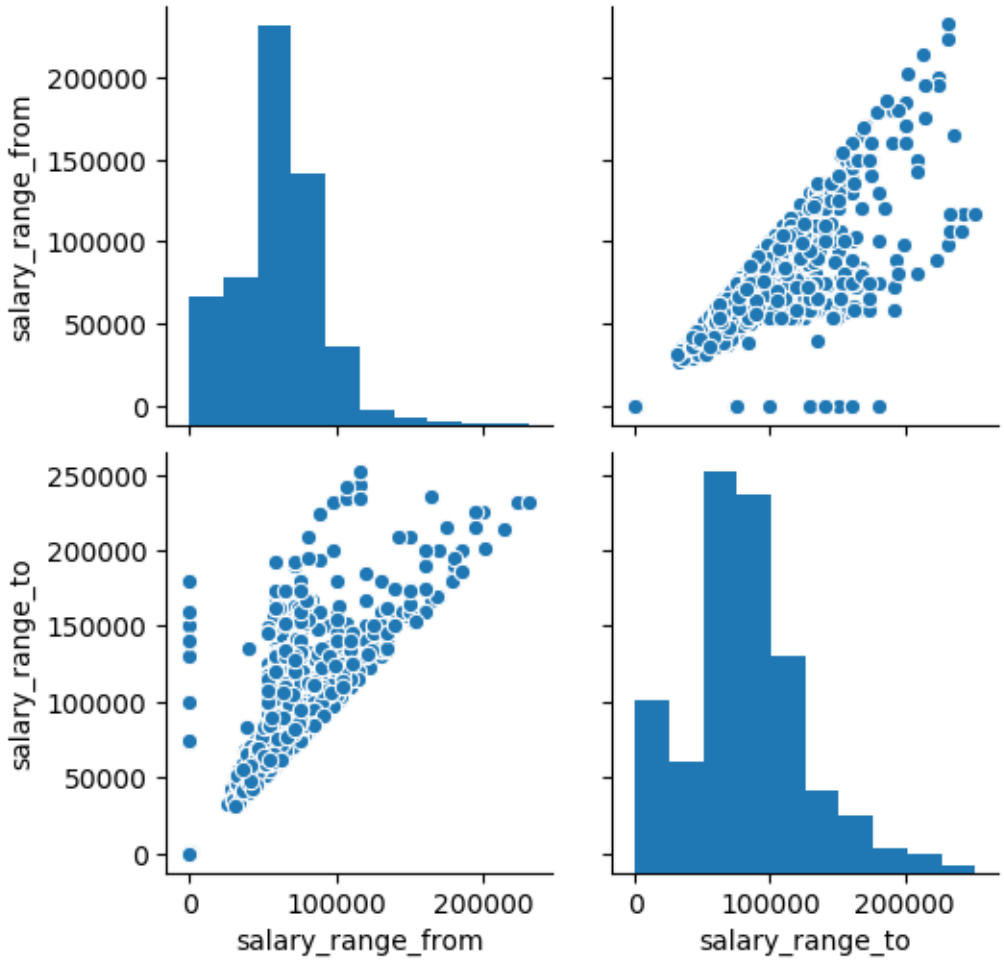
```
job_df02_s01 = job_df01_s05[job_df01_s05_num_lst01]
job_df03_s01 = job_df01_s05[job_df01_s05_num_lst02]

display(job_df02_s01.head(5))
```

	salary_range_from	salary_range_to
0	70341.0	80892.0000
1	32520.0	42191.0000
2	30.0	45.9666
3	90114.0	122168.0000
4	104000.0	118000.0000

Scatterplots of numerical features

[106]:
```
# Pair scatter plots for selected features
sns.pairplot(job_df03_s01)
```

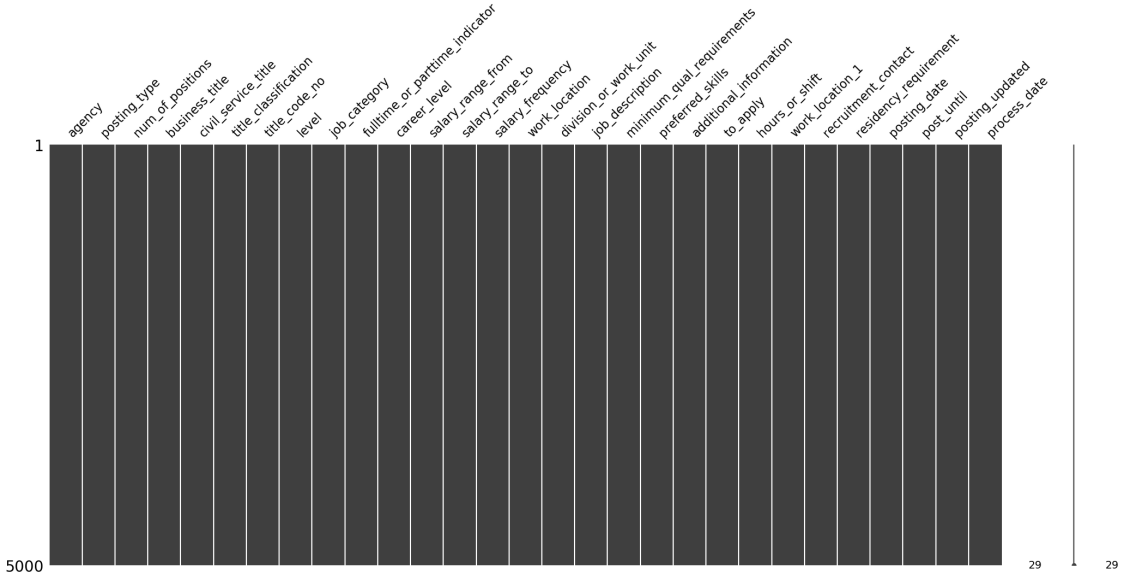
[106]: <seaborn.axisgrid.PairGrid at 0x7fba1e4c88d0>



Examine features with missing values

[107]:
```
# Visualize missing values in each column
msno.matrix(job_df01_s05)
```

[107]: <matplotlib.axes._subplots.AxesSubplot at 0x7fba1e115990>



[108]:
```
# Remove any features for which the number of null vals exceed a threshold--
#-- (5% of total N)
job_df01_s05_null_summ01 = pd.DataFrame(job_df01_s05.isnull().sum(),
                                        columns=['null_count'])

job_df01_s05_null_summ02 = job_df01_s05_null_summ01.
 ↪loc[(job_df01_s05_null_summ01['null_count'] != 0)].sort_values('null_count',

 ↪                                        ascending=False)
job_df01_s05_null_summ03 = job_df01_s05_null_summ02.reset_index()
print(job_df01_s05_null_summ03)

job_df01_s05_null_summ04 = job_df01_s05_null_summ03.
 ↪loc[job_df01_s05_null_summ03['null_count'] > (len(job_df01_s05)*.05)]
print('\n', job_df01_s05_null_summ04)

job_df01_s05_null_summ04_remove_lst01 = list(job_df01_s05_null_summ04['index'])
print('\n', job_df01_s05_null_summ04_remove_lst01)
```

```
Empty DataFrame
```

```
Columns: [index, null_count]
Index: []

 Empty DataFrame
Columns: [index, null_count]
Index: []

 []
```

Examine featues with near zero variances

[109]:
```
# Review near-zero variance (NZV) features for possible removal
job_df02_s01_nzv_fit = VarianceThreshold().fit(job_df02_s01)
job_df02_s01_nzv_vc01 = job_df02_s01_nzv_fit.transform(job_df02_s01)

# Get the names of the selected features
job_df02_s01_nzv_fit_select_features = job_df02_s01.
 ↪columns[job_df02_s01_nzv_fit.get_support()]

job_df02_s01_nzv_df01 = pd.DataFrame(job_df02_s01_nzv_vc01,
                                     ␣
 ↪columns=job_df02_s01_nzv_fit_select_features)

display(job_df02_s01_nzv_df01.head(5))
print(f'NZV transformed matrix dimensions = {job_df02_s01_nzv_df01.shape}')

print(f'\n{job_df02_s01.shape[1] - job_df02_s01_nzv_df01.shape[1]} near zero␣
 ↪variance features were eliminated')
```

	salary_range_from	salary_range_to
0	70341.0	80892.0000
1	32520.0	42191.0000
2	30.0	45.9666
3	90114.0	122168.0000
4	104000.0	118000.0000

```
NZV transformed matrix dimensions = (5000, 2)

0 near zero variance features were eliminated
```

2.5 Release Resources

[110]:
```
%%html

<p><b>Shutting down your kernel for this notebook to release resources.</b></p>
<button class="sm-command-button" data-commandlinker-command="kernelmenu:
 ↪shutdown" style="display:none;">Shutdown Kernel</button>

<script>
```

```
try {
    els = document.getElementsByClassName("sm-command-button");
    els[0].click();
}
catch(err) {
    // NoOp
}
</script>
```

```
<IPython.core.display.HTML object>
```

[111]:
```
%%javascript

try {
    Jupyter.notebook.save_checkpoint();
    Jupyter.notebook.session.delete();
}
catch(err) {
    // NoOp
}
```

```
<IPython.core.display.Javascript object>
```

02_Setup_Join_Final

April 14, 2023

1 ADS-508-01-SP23 Team 8: Final Project

2 Setup Database Joins to Achieve the Analytics Base Table (ABT)

Much of the code is modified from `Fregly, C., & Barth, A. (2021). Data science on AWS: Implementing end-to-end, continuous AI and machine learning pipelines. O'Reilly.`

2.1 Install missing dependencies

PyAthena is a Python DB API 2.0 (PEP 249) compliant client for Amazon Athena.

[2]:
```
!pip install --disable-pip-version-check -q PyAthena==2.1.0
!pip install missingno
```

```
WARNING: The directory '/root/.cache/pip' or its parent directory is not
owned or is not writable by the current user. The cache has been disabled. Check
the permissions and owner of that directory. If executing pip with sudo, you
should use sudo's -H flag.
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
WARNING: The directory '/root/.cache/pip' or its parent directory is
not owned or is not writable by the current user. The cache has been disabled.
Check the permissions and owner of that directory. If executing pip with sudo,
you should use sudo's -H flag.
Requirement already satisfied: missingno in /opt/conda/lib/python3.7/site-
packages (0.5.2)
Requirement already satisfied: seaborn in /opt/conda/lib/python3.7/site-packages
(from missingno) (0.10.0)
Requirement already satisfied: scipy in /opt/conda/lib/python3.7/site-packages
```

```
(from missingno) (1.4.1)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.7/site-
packages (from missingno) (3.1.3)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages
(from missingno) (1.21.6)
Requirement already satisfied: kiwisolver>=1.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (1.1.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.4.6)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.7/site-
packages (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: python-dateutil>=2.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.8.2)
Requirement already satisfied: pandas>=0.22.0 in /opt/conda/lib/python3.7/site-
packages (from seaborn->missingno) (1.3.5)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages
(from cycler>=0.10->matplotlib->missingno) (1.14.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.7/site-
packages (from kiwisolver>=1.0.1->matplotlib->missingno) (59.3.0)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.22.0->seaborn->missingno) (2019.3)
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

2.2 Globally import libraries

[3]:
```
import boto3
from botocore.client import ClientError
import sagemaker
import pandas as pd
from pyathena import connect
from IPython.core.display import display, HTML
import missingno as msno
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import make_pipeline, Pipeline
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.model_selection import train_test_split, cross_val_score,␣
 ↪GridSearchCV
import datetime as dt

%matplotlib inline
```

2.3 Instantiate AWS SageMaker and S3 sessions

[4]:
```
session = boto3.session.Session()
region = session.region_name
sagemaker_session = sagemaker.Session()
def_bucket = sagemaker_session.default_bucket()
bucket = 'sagemaker-us-east-ads508-sp23-t8'

s3 = boto3.Session().client(service_name="s3",
                            region_name=region)
```

[5]:
```
print(f"Default bucket: {def_bucket}")
print(f"Public T8 bucket: {bucket}")
```

```
Default bucket: sagemaker-us-east-1-657724983756
Public T8 bucket: sagemaker-us-east-ads508-sp23-t8
```

2.4 Query Athena Database

[6]:
```
database_name = "ads508_t8"
```

[7]:
```
# Set S3 staging directory -- this is a temporary directory used for Athena␣
 ↪queries
s3_staging_dir = f"s3://{def_bucket}/team_8_data/athena/staging"
print(s3_staging_dir)
```

```
s3://sagemaker-us-east-1-657724983756/team_8_data/athena/staging
```

[8]:
```
conn = connect(region_name=region,
               s3_staging_dir=s3_staging_dir)
```

[9]:
```
cen_tsv_tbl_name = 'census'
ceb_tsv_tbl_name = 'census_block'
evi_tsv_tbl_name = 'evictions'
cri_tsv_tbl_name = 'crime'
cri_pqt_tbl_name = 'crime_pqt'
grd_tsv_tbl_name = 'grad_outcomes'
hsi_tsv_tbl_name = 'hs_info'
job_tsv_tbl_name = 'jobs'
```

2.4.1 Experiment with join using Athena

[10]:
```
abt_select_to_join_stmnt01 = f"""
SELECT
    cen.censustract,
    cen.borough,
    cen.totalpop,
    cen.men,
```

```
    cen.women,
    cen.hispanic,
    cen.white,
    cen.black,
    cen.native,
    cen.asian,
    cen.citizen,
    cen.income,
    cen.poverty,
    cen.childpoverty,
    cen.professional,
    cen.service,
    cen.office,
    cen.construction,
    cen.production,
    cen.drive,
    cen.carpool,
    cen.transit,
    cen.walk,
    cen.othertransp,
    cen.workathome,
    cen.meancommute,
    cen.employed,
    cen.privatework,
    cen.publicwork,
    cen.selfemployed,
    cen.familywork,
    cen.unemployment,
    cvi.blockCode,
    cvi.eviction_count_x_lat_long
FROM {database_name}.{cen_tsv_tbl_name} AS cen
LEFT JOIN (
    SELECT
        ceb.blockCode AS blockCode,
        SUM(evi.eviction_count_x_lat_long) AS eviction_count_x_lat_long
    FROM (
        SELECT
            SUBSTR(blockCode,1,11) AS blockCode,
            COUNT(*),
            MIN(latitude) AS min_lat,
            MAX(latitude) AS max_lat,
            MIN(longitude) AS min_long,
            MAX(longitude) AS max_long
        FROM {database_name}.{ceb_tsv_tbl_name}
        GROUP BY SUBSTR(blockCode,1,11)
        ORDER BY COUNT(*) DESC
        ) AS ceb
```

```
    INNER JOIN (
        SELECT
            CAST(latitude AS DOUBLE) AS latitude,
            CAST(longitude AS DOUBLE) AS longitude,
            COUNT(*) AS eviction_count_x_lat_long
        FROM {database_name}.{evi_tsv_tbl_name}
        WHERE latitude != ''
        GROUP BY latitude, longitude
        ORDER BY COUNT(*) DESC
        ) AS evi
    ON evi.latitude >= ceb.min_lat
        AND evi.latitude <= ceb.max_lat
        AND evi.longitude >= ceb.min_long
        AND evi.longitude <= ceb.max_long
    GROUP BY ceb.blockCode
    LIMIT 50000
    ) AS cvi
    ON cen.censustract = cvi.blockCode
ORDER BY cen.censustract
"""

print(abt_select_to_join_stmnt01)

abt_select_to_join_df01 = pd.read_sql(abt_select_to_join_stmnt01,
                                      conn)

print(abt_select_to_join_df01.shape)
display(abt_select_to_join_df01.head(15))
```

```
SELECT
    cen.censustract,
    cen.borough,
    cen.totalpop,
    cen.men,
    cen.women,
    cen.hispanic,
    cen.white,
    cen.black,
    cen.native,
    cen.asian,
    cen.citizen,
    cen.income,
    cen.poverty,
    cen.childpoverty,
    cen.professional,
    cen.service,
    cen.office,
```

```
    cen.construction,
    cen.production,
    cen.drive,
    cen.carpool,
    cen.transit,
    cen.walk,
    cen.othertransp,
    cen.workathome,
    cen.meancommute,
    cen.employed,
    cen.privatework,
    cen.publicwork,
    cen.selfemployed,
    cen.familywork,
    cen.unemployment,
    cvi.blockCode,
    cvi.eviction_count_x_lat_long
FROM ads508_t8.census AS cen
LEFT JOIN (
    SELECT
        ceb.blockCode AS blockCode,
        SUM(evi.eviction_count_x_lat_long) AS eviction_count_x_lat_long
    FROM (
        SELECT
            SUBSTR(blockCode,1,11) AS blockCode,
            COUNT(*),
            MIN(latitude) AS min_lat,
            MAX(latitude) AS max_lat,
            MIN(longitude) AS min_long,
            MAX(longitude) AS max_long
        FROM ads508_t8.census_block
        GROUP BY SUBSTR(blockCode,1,11)
        ORDER BY COUNT(*) DESC
        ) AS ceb
    INNER JOIN (
        SELECT
            CAST(latitude AS DOUBLE) AS latitude,
            CAST(longitude AS DOUBLE) AS longitude,
            COUNT(*) AS eviction_count_x_lat_long
        FROM ads508_t8.evictions
        WHERE latitude != ''
        GROUP BY latitude, longitude
        ORDER BY COUNT(*) DESC
        ) AS evi
    ON evi.latitude >= ceb.min_lat
        AND evi.latitude <= ceb.max_lat
        AND evi.longitude >= ceb.min_long
        AND evi.longitude <= ceb.max_long
```

```
    GROUP BY ceb.blockCode
    LIMIT 50000
    ) AS cvi
    ON cen.censustract = cvi.blockCode
ORDER BY cen.censustract
```

(2167, 34)

	censustract	borough	totalpop	men	women	hispanic	white	black	\
0	36005000100	Bronx	7703	7133	570	29.9	6.1	60.9	
1	36005000200	Bronx	5403	2659	2744	75.8	2.3	16.0	
2	36005000400	Bronx	5915	2896	3019	62.7	3.6	30.7	
3	36005001600	Bronx	5879	2558	3321	65.1	1.6	32.4	
4	36005001900	Bronx	2591	1206	1385	55.4	9.0	29.0	
5	36005002000	Bronx	8516	3301	5215	61.1	1.6	31.1	
6	36005002300	Bronx	4774	2130	2644	62.3	0.2	36.5	
7	36005002400	Bronx	150	109	41	0.0	52.0	48.0	
8	36005002500	Bronx	5355	2338	3017	76.5	1.5	18.9	
9	36005002701	Bronx	3016	1375	1641	68.0	0.0	31.2	
10	36005002702	Bronx	4778	2427	2351	71.3	1.6	26.2	
11	36005002800	Bronx	5299	2292	3007	23.0	0.2	71.4	
12	36005003100	Bronx	1466	769	697	72.3	0.6	24.6	
13	36005003300	Bronx	3912	1824	2088	65.6	1.0	30.6	
14	36005003500	Bronx	3948	1921	2027	73.5	0.7	25.9	

	native	asian	…	workathome	meancommute	employed	privatework	\
0	0.2	1.6	…	NaN	NaN	0	NaN	
1	0.0	4.2	…	0.0	43.0	2308	80.8	
2	0.0	0.3	…	2.1	45.0	2675	71.7	
3	0.0	0.0	…	1.7	38.8	2120	75.0	
4	0.0	2.1	…	6.2	45.4	1083	76.8	
5	0.3	3.3	…	0.0	46.0	2508	71.0	
6	1.0	0.0	…	4.1	42.7	1191	74.2	
7	0.0	0.0	…	0.0	NaN	113	62.8	
8	0.0	3.0	…	2.7	35.5	1691	85.1	
9	0.0	0.0	…	1.6	42.8	1102	86.9	
10	0.0	0.0	…	0.5	44.0	1559	75.0	
11	0.0	1.7	…	2.7	47.3	2394	61.9	
12	0.0	2.2	…	0.0	40.1	722	79.2	
13	1.7	0.0	…	1.8	42.5	1113	77.2	
14	0.0	0.0	…	4.7	41.6	1360	83.2	

	publicwork	selfemployed	familywork	unemployment	blockCode	\
0	NaN	NaN	NaN	NaN	None	
1	16.2	2.9	0.0	7.7	36005000200	
2	25.3	2.5	0.6	9.5	36005000400	
3	21.3	3.8	0.0	8.7	36005001600	
4	15.5	7.7	0.0	19.2	36005001900	

```
5         21.3           7.7         0.0          17.2  36005002000
6         16.1           9.7         0.0          18.9         None
7         37.2           0.0         0.0           0.0  36005002400
8          8.3           6.1         0.5           9.4  36005002500
9          8.5           4.5         0.0          15.2         None
10        14.0          11.0         0.0          10.6  36005002702
11        37.4           0.6         0.0          12.8  36005002800
12        10.2          10.5         0.0           6.6  36005003100
13        16.9           5.9         0.0          18.5         None
14        13.4           3.4         0.0          11.8  36005003500

    eviction_count_x_lat_long
0                         NaN
1                        31.0
2                        46.0
3                        10.0
4                       230.0
5                        69.0
6                         NaN
7                       169.0
8                        22.0
9                         NaN
10                       32.0
11                       45.0
12                        1.0
13                        NaN
14                       57.0

[15 rows x 34 columns]
```

[11]:
```
ceb_select_to_join_stmnt01 = f"""
SELECT
    substr(blockCode,1,11) AS blockCode,
    COUNT(*),
    MIN(latitude) AS min_lat,
    MAX(latitude) AS max_lat,
    MIN(longitude) AS min_long,
    MAX(longitude) AS max_long
FROM {database_name}.{ceb_tsv_tbl_name}
GROUP BY SUBSTR(blockCode,1,11)
ORDER BY COUNT(*) DESC
"""

print(ceb_select_to_join_stmnt01)

ceb_select_to_join_df01 = pd.read_sql(ceb_select_to_join_stmnt01,
                                      conn)
```

```
print(ceb_select_to_join_df01.shape)
display(ceb_select_to_join_df01.head(15))
```

```
SELECT
    substr(blockCode,1,11) AS blockCode,
    COUNT(*),
    MIN(latitude) AS min_lat,
    MAX(latitude) AS max_lat,
    MIN(longitude) AS min_long,
    MAX(longitude) AS max_long
FROM ads508_t8.census_block
GROUP BY SUBSTR(blockCode,1,11)
ORDER BY COUNT(*) DESC
```

```
(2995, 6)
```

	blockCode	_col1	min_lat	max_lat	min_long	max_long
0	36081990100	1816	40.491307	40.584020	-74.039397	-73.757638
1	36085990100	1198	40.480000	40.604372	-74.257839	-74.036231
2	34025990000	917	40.480000	40.525226	-74.093216	-73.887437
3	36059990400	690	40.534271	40.579497	-73.767136	-73.650000
4	36059301000	412	40.819196	40.877990	-73.751307	-73.653166
5	36081107202	366	40.586281	40.645075	-73.852613	-73.767136
6	36047070203	327	40.579497	40.642814	-73.890603	-73.833618
7	34017012700	305	40.712915	40.776231	-74.143869	-74.077387
8	34013980200	297	40.674472	40.715176	-74.200854	-74.115377
9	36081071600	286	40.622462	40.663166	-73.830452	-73.748141
10	34039035400	275	40.593065	40.640553	-74.261005	-74.200854
11	36047990100	260	40.552362	40.604372	-74.039397	-73.928593
12	36059300100	252	40.798844	40.841809	-73.773467	-73.713317
13	34039039800	251	40.645075	40.688040	-74.197688	-74.140704
14	36005050400	240	40.839548	40.884774	-73.820955	-73.751307

2.4.2 SELECT statements to prepare for full join: evictions table

[12]:
```
# Display full table for review
evi_full_select_stmnt01 = f"""
SELECT * FROM {database_name}.{evi_tsv_tbl_name}
WHERE executed_date <> ''
LIMIT 100
"""

# Display SQL statement
print(evi_full_select_stmnt01)

# Run SQL statement against Athena table
```

```
evi_full_select_df01 = pd.read_sql(evi_full_select_stmnt01,
                                   conn)

# Display results
print(evi_full_select_df01.shape)
display(evi_full_select_df01.head(11))
```

```
SELECT * FROM ads508_t8.evictions
WHERE executed_date <> ''
LIMIT 100
```

```
(100, 20)
```

	court_index_number	docket_number	eviction_address	eviction_apartment_number	executed_date	marshal_first_name
0	56037/17	339568	547 EAST 168TH STREET	3H	02/26/2018	Thomas
1	B047517/19	409031	4014 CARPENTER AVENUE	4B	11/16/2022	Richard
2	15068/17	334442	655 EAST 224TH STREET	1	09/29/2017	Thomas
3	58273/18	025388	1551 DEAN STREET	1ST FLOOR	07/12/2018	Gary
4	14866/19A	097278	718 PENFIELD STREET	2-F	10/24/2019	Justin
5	66703/18BX	090391	2032 EAST 177TH ST A /K/A 2032 CROSS BRONX EXP…	1E	07/30/2019	Justin
6	98925/17	075402	175 WOODRUFF AVENUE	GARDEN APARTMENT	06/01/2018	Justin
7	304057/20	107717	555 TENTH AVENUE			
8	210706/18	085502	2201 FIRST AVENUE			
9	B806500/18	396012	281 EAST 143RD STREET			
10	83995/16	464985	1-11 MARBLE HILL AVE NUE			

```
7                       32I    04/18/2022            Justin
8                       05B    03/14/2019             Henry
9                       07A    01/17/2019           Richard
10                       3F    03/17/2017             Danny

   marshal_last_name residential_or_commercial    borough eviction_postcode  \
0                Bia                Residential      BRONX             10456
1              McCoy                Residential      BRONX             10466
2                Bia                Residential      BRONX             10467
3               Rose                Residential   BROOKLYN             11213
4           Grossman                Residential      BRONX             10470
5           Grossman                Residential      BRONX             10472
6           Grossman                Residential   BROOKLYN             11226
7           Grossman                Residential  MANHATTAN             10018
8              Daley                Residential  MANHATTAN             10029
9              McCoy                Residential      BRONX             10451
10          Weinheim                Residential  MANHATTAN             10463

           ejectment eviction_or_legal_possession   latitude   longitude  \
0   Not an Ejectment                   Possession  40.830857  -73.905191
1   Not an Ejectment                   Possession  40.889878  -73.862686
2   Not an Ejectment                   Possession  40.887599  -73.862391
3   Not an Ejectment                   Possession  40.676166  -73.936661
4   Not an Ejectment                   Possession  40.904888  -73.849089
5   Not an Ejectment                   Possession  40.831685  -73.856168
6   Not an Ejectment                   Possession  40.654641  -73.960291
7   Not an Ejectment                   Possession  40.758888  -73.996022
8   Not an Ejectment                   Possession  40.794176  -73.936754
9   Not an Ejectment                   Possession  40.814845  -73.924083
10  Not an Ejectment                   Possession  40.874862  -73.910845

   community_board council_district census_tract      bin         bbl  \
0                3                16          145  2004227  2026100065
1               12                12          408  2063060  2048280031
2               12                12          394  2062985  2048260028
3                8                36          311  3388499  3013400049
4               12                11          442  2071873  2051130039
5                9                18           78  2026230  2038030019
6               14                40        50803  3115933  3050540052
7                4                 3          117  1089722  1010697501
8               11                 8          180  1081091  1016840001
9                1                 8           51  2091116  2023240001
10               8                10          309  1064643  1022150465

                                    nta
0                  Claremont-Bathgate
1               Williamsbridge-Olinville
2               Williamsbridge-Olinville
```

```
3                          Crown Heights North
4                           Woodlawn-Wakefield
5                        Westchester-Unionport
6                                    Flatbush
7   Hudson Yards-Chelsea-Flatiron-Union Square
8                           East Harlem North
9                       Mott Haven-Port Morris
10                          Marble Hill-Inwood
```

[13]:
```
# Aggregate table based on borough and relative data year
evi_borough_year_stmnt01 = f"""
SELECT
    LOWER(borough) AS borough,
    CAST(YEAR(DATE_PARSE(executed_date, '%m/%d/%Y')) AS INT) - 2022 AS␣
 ↪relative_data_year,
    COUNT(*) AS annual_evictions_x_borough
FROM {database_name}.{evi_tsv_tbl_name}
WHERE executed_date <> ''
    AND CAST(YEAR(DATE_PARSE(executed_date, '%m/%d/%Y')) AS INT) BETWEEN 2018␣
 ↪AND 2022
GROUP BY borough, YEAR(DATE_PARSE(executed_date, '%m/%d/%Y'))
ORDER BY borough, YEAR(DATE_PARSE(executed_date, '%m/%d/%Y'))
LIMIT 10000
"""

# Display SQL statement
print(evi_borough_year_stmnt01)

# Run SQL statement against Athena table
evi_borough_year_df01 = pd.read_sql(evi_borough_year_stmnt01,
                                    conn)

# Display results
print(evi_borough_year_df01.shape)
display(evi_borough_year_df01.head(11))

# Create pivot table
evi_borough_year_df02 = evi_borough_year_df01.pivot_table(index = 'borough',
                                                          columns =␣
 ↪'relative_data_year',
                                                          values =␣
 ↪'annual_evictions_x_borough',
                                                          aggfunc = 'sum',
                                                          fill_value = 0)

print(evi_borough_year_df02.shape)
display(evi_borough_year_df02.head(35))
```

```
SELECT
    LOWER(borough) AS borough,
    CAST(YEAR(DATE_PARSE(executed_date, '%m/%d/%Y')) AS INT) - 2022 AS
relative_data_year,
    COUNT(*) AS annual_evictions_x_borough
FROM ads508_t8.evictions
WHERE executed_date <> ''
    AND CAST(YEAR(DATE_PARSE(executed_date, '%m/%d/%Y')) AS INT) BETWEEN 2018
AND 2022
GROUP BY borough, YEAR(DATE_PARSE(executed_date, '%m/%d/%Y'))
ORDER BY borough, YEAR(DATE_PARSE(executed_date, '%m/%d/%Y'))
LIMIT 10000
```

(25, 3)

	borough	relative_data_year	annual_evictions_x_borough
0	bronx	-4	7140
1	bronx	-3	6244
2	bronx	-2	1088
3	bronx	-1	29
4	bronx	0	1174
5	brooklyn	-4	6157
6	brooklyn	-3	5312
7	brooklyn	-2	1005
8	brooklyn	-1	100
9	brooklyn	0	1864
10	manhattan	-4	3390

(5, 5)

relative_data_year	-4	-3	-2	-1	0
borough					
bronx	7140	6244	1088	29	1174
brooklyn	6157	5312	1005	100	1864
manhattan	3390	2818	521	68	930
queens	4452	3705	696	36	811
staten island	691	636	112	35	271

[14]:
```
# Aggregate table based on census_tract and year
evi_ceb_join_select_stmnt01 = f"""
SELECT
    ceb.blockCode AS census_tract,
    evi.year,
    SUM(evi.eviction_count_x_lat_long) AS annual_evictions_x_census_tract
FROM (
    SELECT
        SUBSTR(blockCode,1,11) AS blockCode,
        COUNT(*),
```

```
        MIN(latitude) AS min_lat,
        MAX(latitude) AS max_lat,
        MIN(longitude) AS min_long,
        MAX(longitude) AS max_long
    FROM {database_name}.{ceb_tsv_tbl_name}
    GROUP BY SUBSTR(blockCode,1,11)
    ORDER BY COUNT(*) DESC
    ) AS ceb
INNER JOIN (
    SELECT
        CAST(latitude AS DOUBLE) AS latitude,
        CAST(longitude AS DOUBLE) AS longitude,
        CAST(YEAR(DATE_PARSE(executed_date, '%m/%d/%Y')) AS INT) AS year,
        COUNT(*) AS eviction_count_x_lat_long
    FROM {database_name}.{evi_tsv_tbl_name}
    WHERE latitude != ''
    GROUP BY latitude, longitude, YEAR(DATE_PARSE(executed_date, '%m/%d/%Y'))
    ORDER BY COUNT(*) DESC
    ) AS evi
    ON evi.latitude >= ceb.min_lat
        AND evi.latitude <= ceb.max_lat
        AND evi.longitude >= ceb.min_long
        AND evi.longitude <= ceb.max_long
GROUP BY ceb.blockCode, evi.year
ORDER BY ceb.blockCode, evi.year
LIMIT 50000
"""

# Display SQL statement
print(evi_ceb_join_select_stmnt01)

evi_ceb_join_select_df01 = pd.read_sql(evi_ceb_join_select_stmnt01,
                           conn)

# Display results
print(evi_ceb_join_select_df01.shape)
display(evi_ceb_join_select_df01.head(11))

# Create pivot table
evi_ceb_join_select_df02 = evi_ceb_join_select_df01.pivot_table(index =␣
 ↪'census_tract',
                                                                 columns =␣
 ↪'year',
                                                                 values =␣
 ↪'annual_evictions_x_census_tract',
                                                                 aggfunc = 'sum',
                                                                 fill_value = 0)
```

```
print(evi_ceb_join_select_df02.shape)
display(evi_ceb_join_select_df02.head(11))
```

```
SELECT
    ceb.blockCode AS census_tract,
    evi.year,
    SUM(evi.eviction_count_x_lat_long) AS annual_evictions_x_census_tract
FROM (
    SELECT
        SUBSTR(blockCode,1,11) AS blockCode,
        COUNT(*),
        MIN(latitude) AS min_lat,
        MAX(latitude) AS max_lat,
        MIN(longitude) AS min_long,
        MAX(longitude) AS max_long
    FROM ads508_t8.census_block
    GROUP BY SUBSTR(blockCode,1,11)
    ORDER BY COUNT(*) DESC
    ) AS ceb
INNER JOIN (
    SELECT
        CAST(latitude AS DOUBLE) AS latitude,
        CAST(longitude AS DOUBLE) AS longitude,
        CAST(YEAR(DATE_PARSE(executed_date, '%m/%d/%Y')) AS INT) AS year,
        COUNT(*) AS eviction_count_x_lat_long
    FROM ads508_t8.evictions
    WHERE latitude != ''
    GROUP BY latitude, longitude, YEAR(DATE_PARSE(executed_date, '%m/%d/%Y'))
    ORDER BY COUNT(*) DESC
    ) AS evi
    ON evi.latitude >= ceb.min_lat
        AND evi.latitude <= ceb.max_lat
        AND evi.longitude >= ceb.min_long
        AND evi.longitude <= ceb.max_long
GROUP BY ceb.blockCode, evi.year
ORDER BY ceb.blockCode, evi.year
LIMIT 50000
```

```
(4870, 3)
```

	census_tract	year	annual_evictions_x_census_tract
0	34003013001	2017	9
1	34003013001	2018	11
2	34003013001	2019	6
3	34003013001	2021	1
4	34003013001	2022	6

```
5    34003013001  2023                                1
6    34003016000  2017                               14
7    34003016000  2018                               13
8    34003016000  2019                               16
9    34003016000  2020                                2
10   34003016000  2021                                2
```

```
(1306, 7)
```

year	2017	2018	2019	2020	2021	2022	2023
census_tract							
34003013001	9	11	6	0	1	6	1
34003016000	14	13	16	2	2	1	0
34017010800	2	2	5	0	0	0	0
36005000200	9	10	10	0	0	1	1
36005000400	10	16	15	0	0	5	0
36005001600	3	2	4	0	0	1	0
36005001900	74	67	53	9	0	22	5
36005002000	20	14	29	3	0	3	0
36005002400	50	43	65	5	0	6	0
36005002500	8	2	9	0	0	3	0
36005002702	10	14	5	1	0	2	0

2.4.3 SELECT statements to prepare for full join: crime_pqt table

[15]:
```
# Display full table for review
cri_full_select_stmnt01 = f"""
SELECT * FROM {database_name}.{cri_pqt_tbl_name}
LIMIT 10000
"""

# Display SQL statement
print(cri_full_select_stmnt01)

# Run SQL statement against Athena table
cri_full_select_df01 = pd.read_sql(cri_full_select_stmnt01,
                                   conn)

# Display results
print(cri_full_select_df01.shape)
display(cri_full_select_df01.head(5))
```

```
SELECT * FROM ads508_t8.crime_pqt
LIMIT 10000

(10000, 35)
```

```
  cmplnt_num cmplnt_fr_dt cmplnt_fr_tm cmplnt_to_dt cmplnt_to_tm addr_pct_cd  \
```

```
0  615895978   02/04/2016     06:44:00                                    44
1  753741689   04/03/2017     23:20:00   04/03/2017     23:30:00          42
2  157660368   06/01/2017     19:00:00   08/08/2017     12:00:00          49
3  638104805   10/04/2014     10:30:00   10/04/2014     14:30:00          43
4  330952034   12/24/2020     16:00:00   12/24/2020     17:00:00          44

       rpt_dt ky_cd      ofns_desc pd_cd  …            latitude  \
0  02/12/2016   578  HARRASSMENT 2   638  …        40.834209452
1  04/03/2017   578  HARRASSMENT 2   637  …        40.830145224
2  08/15/2017   578  HARRASSMENT 2   638  …        40.845844356
3  10/05/2014   578  HARRASSMENT 2   638  …        40.824130774
4  12/28/2020   578  HARRASSMENT 2   638  …  40.826524240000026

            longitude                                    lat_lon  \
0       -73.925706819             (40.834209452, -73.925706819)
1       -73.891878074             (40.830145224, -73.891878074)
2       -73.851790951             (40.845844356, -73.851790951)
3       -73.869872717             (40.824130774, -73.869872717)
4  -73.92154563799994  (40.826524240000026, -73.92154563799994)

         patrol_boro    station_name vic_age_group        vic_race vic_sex  \
0  PATROL BORO BRONX                         25-44  WHITE HISPANIC       F
1  PATROL BORO BRONX  FREEMAN STREET         25-44  WHITE HISPANIC       M
2  PATROL BORO BRONX                         25-44  BLACK HISPANIC       M
3  PATROL BORO BRONX                         25-44           BLACK       F
4  PATROL BORO BRONX                         25-44  BLACK HISPANIC       M

  law_cat_cd borough
0  VIOLATION   BRONX
1  VIOLATION   BRONX
2  VIOLATION   BRONX
3  VIOLATION   BRONX
4  VIOLATION   BRONX

[5 rows x 35 columns]
```

[16]:
```
# Aggregate table based on borough, relative data year, & law_cat_cd
cri_borough_year_type_stmnt01 = f"""
SELECT
    LOWER(borough) AS borough,
    CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS INT) - 2021 AS␣
 ↪relative_data_year,
    law_cat_cd AS complaint_type,
    COUNT(*) AS annual_complaint_counts
FROM {database_name}.{cri_pqt_tbl_name}
WHERE cmplnt_fr_dt <> ''
    AND YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) BETWEEN 2017 AND 2021
```

```
GROUP BY borough, YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')), law_cat_cd
ORDER BY borough, YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')), law_cat_cd
LIMIT 100000
"""

# Display SQL statement
print(cri_borough_year_type_stmnt01)

# Run SQL statement against Athena table
cri_borough_year_type_df01 = pd.read_sql(cri_borough_year_type_stmnt01,
                                         conn)

# Display results
print(cri_borough_year_type_df01.shape)
display(cri_borough_year_type_df01.head(35))
```

```
SELECT
    LOWER(borough) AS borough,
    CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS INT) - 2021 AS
relative_data_year,
    law_cat_cd AS complaint_type,
    COUNT(*) AS annual_complaint_counts
FROM ads508_t8.crime_pqt
WHERE cmplnt_fr_dt <> ''
    AND YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) BETWEEN 2017 AND 2021
GROUP BY borough, YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')), law_cat_cd
ORDER BY borough, YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')), law_cat_cd
LIMIT 100000

(81, 4)
```

	borough	relative_data_year	complaint_type	annual_complaint_counts
0	bronx	-4	FELONY	583
1	bronx	-4	MISDEMEANOR	1179
2	bronx	-4	VIOLATION	289
3	bronx	-3	FELONY	540
4	bronx	-3	MISDEMEANOR	1133
5	bronx	-3	VIOLATION	338
6	bronx	-2	FELONY	558
7	bronx	-2	MISDEMEANOR	1114
8	bronx	-2	VIOLATION	328
9	bronx	-1	FELONY	529
10	bronx	-1	MISDEMEANOR	954
11	bronx	-1	VIOLATION	322
12	bronx	0	FELONY	617
13	bronx	0	MISDEMEANOR	910
14	bronx	0	VIOLATION	320

15	brooklyn	-4	FELONY	862
16	brooklyn	-4	MISDEMEANOR	1512
17	brooklyn	-4	VIOLATION	430
18	brooklyn	-3	FELONY	976
19	brooklyn	-3	MISDEMEANOR	1473
20	brooklyn	-3	VIOLATION	423
21	brooklyn	-2	FELONY	909
22	brooklyn	-2	MISDEMEANOR	1371
23	brooklyn	-2	VIOLATION	435
24	brooklyn	-1	FELONY	803
25	brooklyn	-1	MISDEMEANOR	1168
26	brooklyn	-1	VIOLATION	384
27	brooklyn	0	FELONY	834
28	brooklyn	0	MISDEMEANOR	1185
29	brooklyn	0	VIOLATION	475
30	manhattan	-4	FELONY	701
31	manhattan	-4	MISDEMEANOR	1306
32	manhattan	-4	VIOLATION	267
33	manhattan	-3	FELONY	736
34	manhattan	-3	MISDEMEANOR	1276

[17]:
```
# Aggregate table based on borough, relative data year, & law_cat_cd
cri_borough_year_type_stmnt02 = f"""
SELECT
    LOWER(borough) AS borough,
    CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS VARCHAR), ' - ',␣
 ↪law_cat_cd) AS year_w_complaint,
    COUNT(*) AS annual_complaint_counts
FROM {database_name}.{cri_pqt_tbl_name}
WHERE cmplnt_fr_dt <> ''
    AND YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) >= 2017
GROUP BY borough, CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS␣
 ↪VARCHAR), ' - ', law_cat_cd)
ORDER BY borough, CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS␣
 ↪VARCHAR), ' - ', law_cat_cd)
LIMIT 100000
"""

# Display SQL statement
print(cri_borough_year_type_stmnt02)

# Run SQL statement against Athena table
cri_borough_year_type_df12 = pd.read_sql(cri_borough_year_type_stmnt02,
                                         conn)

# Display results
print(cri_borough_year_type_df12.shape)
```

```
display(cri_borough_year_type_df12.head(11))

# Create pivot table
cri_borough_year_type_df13 = cri_borough_year_type_df12.pivot_table(index =␣
 ↪'borough',
                                                                    columns =␣
 ↪'year_w_complaint',
                                                                    values =␣
 ↪'annual_complaint_counts',
                                                                    aggfunc =␣
 ↪'sum',
                                                                    fill_value␣
 ↪= 0)

print(cri_borough_year_type_df13.shape)
display(cri_borough_year_type_df13.head(11))
```

```
SELECT
    LOWER(borough) AS borough,
    CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS VARCHAR), ' - ',
law_cat_cd) AS year_w_complaint,
    COUNT(*) AS annual_complaint_counts
FROM ads508_t8.crime_pqt
WHERE cmplnt_fr_dt <> ''
    AND YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) >= 2017
GROUP BY borough, CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS
VARCHAR), ' - ', law_cat_cd)
ORDER BY borough, CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS
VARCHAR), ' - ', law_cat_cd)
LIMIT 100000
```

```
(81, 3)
```

	borough	year_w_complaint	annual_complaint_counts
0	bronx	2017 - FELONY	583
1	bronx	2017 - MISDEMEANOR	1179
2	bronx	2017 - VIOLATION	289
3	bronx	2018 - FELONY	540
4	bronx	2018 - MISDEMEANOR	1133
5	bronx	2018 - VIOLATION	338
6	bronx	2019 - FELONY	558
7	bronx	2019 - MISDEMEANOR	1114
8	bronx	2019 - VIOLATION	328
9	bronx	2020 - FELONY	529
10	bronx	2020 - MISDEMEANOR	954

```
(5, 15)
```

```
year_w_complaint  2017 - FELONY  2017 - MISDEMEANOR  2017 - VIOLATION  \
borough
bronx                       583                1179               289
brooklyn                    862                1512               430
manhattan                   701                1306               267
queens                      610                 932               299
staten island               100                 249               112

year_w_complaint  2018 - FELONY  2018 - MISDEMEANOR  2018 - VIOLATION  \
borough
bronx                       540                1133               338
brooklyn                    976                1473               423
manhattan                   736                1276               311
queens                      576                 877               309
staten island               107                 239               109

year_w_complaint  2019 - FELONY  2019 - MISDEMEANOR  2019 - VIOLATION  \
borough
bronx                       558                1114               328
brooklyn                    909                1371               435
manhattan                   665                1302               317
queens                      604                 970               292
staten island               101                 196                61

year_w_complaint  2020 - FELONY  2020 - MISDEMEANOR  2020 - VIOLATION  \
borough
bronx                       529                 954               322
brooklyn                    803                1168               384
manhattan                   605                1085               248
queens                      607                 911               290
staten island               104                 175                57

year_w_complaint  2021 - FELONY  2021 - MISDEMEANOR  2021 - VIOLATION
borough
bronx                       617                 910               320
brooklyn                    834                1185               475
manhattan                   708                1145               295
queens                      646                 995               320
staten island                96                 174                75
```

[18]:
```
# Aggregate table based on census_tract and year
cri_ceb_join_select_stmnt01 = f"""
SELECT
    ceb.blockCode AS census_tract,
    cri.year,
    SUM(cri.complaint_count_x_lat_long) AS annual_complaints_x_census_tract
FROM (
```

```
    SELECT
        SUBSTR(blockCode,1,11) AS blockCode,
        COUNT(*),
        MIN(latitude) AS min_lat,
        MAX(latitude) AS max_lat,
        MIN(longitude) AS min_long,
        MAX(longitude) AS max_long
    FROM {database_name}.{ceb_tsv_tbl_name}
    GROUP BY SUBSTR(blockCode,1,11)
    ORDER BY COUNT(*) DESC
    ) AS ceb
INNER JOIN (
    SELECT
        CAST(latitude AS DOUBLE) AS latitude,
        CAST(longitude AS DOUBLE) AS longitude,
        CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS VARCHAR), ' -␣
 ↪', law_cat_cd) AS year,
        COUNT(*) AS complaint_count_x_lat_long
    FROM {database_name}.{cri_pqt_tbl_name}
    WHERE cmplnt_fr_dt <> ''
        AND latitude != ''
        AND YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) >= 2017
    GROUP BY latitude, longitude, CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/
 ↪%d/%Y')) AS VARCHAR), ' - ', law_cat_cd)
    ORDER BY COUNT(*) DESC
    ) AS cri
    ON cri.latitude >= ceb.min_lat
        AND cri.latitude <= ceb.max_lat
        AND cri.longitude >= ceb.min_long
        AND cri.longitude <= ceb.max_long
GROUP BY ceb.blockCode, cri.year
ORDER BY ceb.blockCode, cri.year
LIMIT 100000
"""

# Display SQL statement
print(cri_ceb_join_select_stmnt01)

# Run SQL statement against Athena table
cri_ceb_join_select_df01 = pd.read_sql(cri_ceb_join_select_stmnt01,
                                       conn)

# Display results
print(cri_ceb_join_select_df01.shape)
display(cri_ceb_join_select_df01.head(15))
```

```
# Create pivot table
cri_ceb_join_select_df02 = cri_ceb_join_select_df01.pivot_table(index =␣
 ↪'census_tract',
                                                                 columns =␣
 ↪'year',
                                                                 values =␣
 ↪'annual_complaints_x_census_tract',
                                                                 aggfunc = 'sum',
                                                                 fill_value = 0)

print(cri_ceb_join_select_df02.shape)
display(cri_ceb_join_select_df02.head(35))
```

```
SELECT
    ceb.blockCode AS census_tract,
    cri.year,
    SUM(cri.complaint_count_x_lat_long) AS annual_complaints_x_census_tract
FROM (
    SELECT
        SUBSTR(blockCode,1,11) AS blockCode,
        COUNT(*),
        MIN(latitude) AS min_lat,
        MAX(latitude) AS max_lat,
        MIN(longitude) AS min_long,
        MAX(longitude) AS max_long
    FROM ads508_t8.census_block
    GROUP BY SUBSTR(blockCode,1,11)
    ORDER BY COUNT(*) DESC
    ) AS ceb
INNER JOIN (
    SELECT
        CAST(latitude AS DOUBLE) AS latitude,
        CAST(longitude AS DOUBLE) AS longitude,
        CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) AS VARCHAR), ' -
', law_cat_cd) AS year,
        COUNT(*) AS complaint_count_x_lat_long
    FROM ads508_t8.crime_pqt
    WHERE cmplnt_fr_dt <> ''
        AND latitude != ''
        AND YEAR(DATE_PARSE(cmplnt_fr_dt, '%m/%d/%Y')) >= 2017
    GROUP BY latitude, longitude, CONCAT(CAST(YEAR(DATE_PARSE(cmplnt_fr_dt,
'%m/%d/%Y')) AS VARCHAR), ' - ', law_cat_cd)
    ORDER BY COUNT(*) DESC
    ) AS cri
    ON cri.latitude >= ceb.min_lat
        AND cri.latitude <= ceb.max_lat
```

```
        AND cri.longitude >= ceb.min_long
        AND cri.longitude <= ceb.max_long
GROUP BY ceb.blockCode, cri.year
ORDER BY ceb.blockCode, cri.year
LIMIT 100000
```

(9555, 3)

	census_tract	year	annual_complaints_x_census_tract
0	34003013001	2017 - FELONY	1
1	34003013001	2017 - MISDEMEANOR	6
2	34003013001	2017 - VIOLATION	4
3	34003013001	2018 - FELONY	3
4	34003013001	2018 - MISDEMEANOR	5
5	34003013001	2019 - FELONY	1
6	34003013001	2019 - MISDEMEANOR	6
7	34003013001	2019 - VIOLATION	1
8	34003013001	2020 - FELONY	3
9	34003013001	2020 - MISDEMEANOR	2
10	34003013001	2020 - VIOLATION	1
11	34003013001	2021 - FELONY	6
12	34003013001	2021 - MISDEMEANOR	5
13	34003013001	2021 - VIOLATION	2
14	34003016000	2017 - MISDEMEANOR	4

(1358, 15)

year	2017 - FELONY	2017 - MISDEMEANOR	2017 - VIOLATION	\
census_tract				
34003013001	1	6	4	
34003016000	0	4	0	
34017010800	0	1	0	
36005000100	16	10	0	
36005000200	0	1	1	
36005000400	0	1	1	
36005001600	0	0	2	
36005001900	14	20	5	
36005002000	3	10	0	
36005002400	13	36	8	
36005002500	0	3	0	
36005002702	1	1	0	
36005002800	1	1	0	
36005003100	1	0	0	
36005003500	0	1	0	
36005003900	0	2	1	
36005004100	2	5	0	
36005004200	4	11	6	
36005004300	1	1	0	
36005004400	1	2	0	

36005005100	2	23	2
36005005200	0	1	0
36005005902	0	2	0
36005006000	0	1	0
36005006100	0	1	0
36005006200	2	1	1
36005006300	1	16	2
36005006500	3	4	0
36005006700	0	0	1
36005006800	1	1	0
36005007100	7	17	1
36005007200	0	0	0
36005007500	2	4	1
36005007600	0	0	1
36005007800	0	1	0

year census_tract	2018 - FELONY	2018 - MISDEMEANOR	2018 - VIOLATION \
34003013001	3	5	0
34003016000	3	0	2
34017010800	0	0	0
36005000100	5	10	0
36005000200	3	3	1
36005000400	2	3	2
36005001600	0	0	1
36005001900	7	15	5
36005002000	1	1	0
36005002400	7	14	7
36005002500	0	2	0
36005002702	0	1	0
36005002800	1	0	1
36005003100	0	0	0
36005003500	1	0	1
36005003900	2	1	0
36005004100	1	5	1
36005004200	1	9	5
36005004300	1	1	0
36005004400	3	3	0
36005005100	10	11	1
36005005200	1	1	1
36005005902	1	0	1
36005006000	0	0	0
36005006100	0	0	1
36005006200	2	1	0
36005006300	13	19	3
36005006500	4	8	2
36005006700	0	4	0
36005006800	0	0	0

36005007100	6	9	5
36005007200	0	0	1
36005007500	0	3	0
36005007600	2	0	0
36005007800	0	2	0

year	2019 - FELONY	2019 - MISDEMEANOR	2019 - VIOLATION	\
census_tract				
34003013001	1	6	1	
34003016000	3	1	0	
34017010800	0	0	0	
36005000100	10	4	1	
36005000200	1	2	2	
36005000400	3	1	1	
36005001600	0	0	0	
36005001900	11	25	12	
36005002000	3	7	2	
36005002400	8	25	6	
36005002500	3	4	1	
36005002702	0	2	0	
36005002800	0	1	0	
36005003100	0	0	0	
36005003500	1	0	0	
36005003900	1	3	1	
36005004100	4	4	1	
36005004200	5	11	1	
36005004300	1	4	0	
36005004400	0	2	1	
36005005100	9	11	1	
36005005200	0	0	0	
36005005902	1	5	1	
36005006000	0	2	0	
36005006100	0	2	0	
36005006200	0	5	1	
36005006300	2	13	4	
36005006500	1	5	2	
36005006700	0	0	0	
36005006800	0	2	0	
36005007100	3	16	4	
36005007200	0	1	0	
36005007500	0	3	0	
36005007600	1	1	0	
36005007800	0	0	2	

year	2020 - FELONY	2020 - MISDEMEANOR	2020 - VIOLATION	\
census_tract				
34003013001	3	2	1	
34003016000	0	0	0	

34017010800	0	1	0
36005000100	3	2	0
36005000200	3	0	0
36005000400	3	2	0
36005001600	1	1	0
36005001900	9	11	7
36005002000	3	3	0
36005002400	12	18	6
36005002500	0	3	0
36005002702	0	0	1
36005002800	1	2	0
36005003100	2	0	0
36005003500	1	1	0
36005003900	2	1	0
36005004100	0	1	2
36005004200	3	6	4
36005004300	1	1	0
36005004400	2	4	3
36005005100	5	13	4
36005005200	0	1	0
36005005902	0	2	0
36005006000	2	0	0
36005006100	0	0	0
36005006200	2	1	4
36005006300	7	13	3
36005006500	4	4	2
36005006700	0	2	0
36005006800	0	1	0
36005007100	3	8	2
36005007200	0	1	0
36005007500	2	3	1
36005007600	0	1	1
36005007800	1	0	0

year	2021 - FELONY	2021 - MISDEMEANOR	2021 - VIOLATION
census_tract			
34003013001	6	5	2
34003016000	3	1	0
34017010800	0	0	0
36005000100	2	1	0
36005000200	3	2	1
36005000400	3	2	1
36005001600	1	1	0
36005001900	8	13	3
36005002000	0	1	1
36005002400	8	15	2
36005002500	0	1	2
36005002702	1	2	0

36005002800	1	5	3
36005003100	0	0	0
36005003500	0	3	2
36005003900	2	1	0
36005004100	2	3	0
36005004200	0	12	3
36005004300	1	1	0
36005004400	2	1	3
36005005100	2	9	3
36005005200	0	1	0
36005005902	1	0	1
36005006000	0	0	0
36005006100	0	0	0
36005006200	2	2	2
36005006300	10	14	3
36005006500	3	2	1
36005006700	2	1	1
36005006800	1	2	0
36005007100	9	10	5
36005007200	0	1	0
36005007500	0	2	1
36005007600	1	0	0
36005007800	0	0	0

2.4.4 SELECT statements to prepare for full join: grad_outcomes table

[19]:
```
# Run query to review a sample of records
grd_full_select_stmnt01 = f"""
SELECT
    grd.dbn,
    grd.school_name,
    grd.cohort,
    grd.total_grads_n,
    grd.dropped_out_n,
    hsi.borough,
    hsi.census_tract,
    hsi.bin
FROM {database_name}.{grd_tsv_tbl_name} AS grd
INNER JOIN {database_name}.{hsi_tsv_tbl_name} AS hsi
    ON grd.dbn = hsi.dbn
WHERE census_tract IS NOT NULL
ORDER BY hsi.census_tract ASC
LIMIT 100000
"""

# Display SQL statement
print(grd_full_select_stmnt01)
```

```
# Run SQL statement against Athena table
grd_full_select_df01 = pd.read_sql(grd_full_select_stmnt01,
                                   conn)

# Display results
print(grd_full_select_df01.shape)
display(grd_full_select_df01.head(7))
```

```
SELECT
    grd.dbn,
    grd.school_name,
    grd.cohort,
    grd.total_grads_n,
    grd.dropped_out_n,
    hsi.borough,
    hsi.census_tract,
    hsi.bin
FROM ads508_t8.grad_outcomes AS grd
INNER JOIN ads508_t8.hs_info AS hsi
    ON grd.dbn = hsi.dbn
WHERE census_tract IS NOT NULL
ORDER BY hsi.census_tract ASC
LIMIT 100000
```

```
(0, 8)
```

```
Empty DataFrame
Columns: [dbn, school_name, cohort, total_grads_n, dropped_out_n, borough,␣
 ↪census_tract, bin]
Index: []
```

[20]:
```
# Run query to review a sample of records
grd_select_borough_stmnt01 = f"""
SELECT
    LOWER(hsi.borough) AS borough,
    CAST(grd.cohort AS INT) - 2006 AS relative_data_year,
    SUM(CAST(grd.total_grads_n AS DOUBLE)) AS annual_grad_n,
    SUM(CAST(grd.dropped_out_n AS DOUBLE)) AS annual_dropped_out_n
FROM {database_name}.{grd_tsv_tbl_name} AS grd
LEFT JOIN {database_name}.{hsi_tsv_tbl_name} AS hsi
    ON grd.dbn = hsi.dbn
WHERE total_grads_n <> 's'
    AND cohort != '2006 Aug'
    AND borough IS NOT null
    AND CAST(grd.cohort AS INT) BETWEEN 2002 AND 2006
GROUP BY hsi.borough, grd.cohort
```

```
ORDER BY hsi.borough, grd.cohort
LIMIT 100000
"""

# Display SQL statement
print(grd_select_borough_stmnt01)

# Run SQL statement against Athena table
grd_select_borough_df01 = pd.read_sql(grd_select_borough_stmnt01,
                                      conn)

# Display results
print(grd_select_borough_df01.shape)
display(grd_select_borough_df01.head(50))

# Create pivot table
grd_select_borough_df02 = grd_select_borough_df01.pivot_table(index = 'borough',
                                                              columns =␣
 ↪'relative_data_year',
                                                              values =␣
 ↪['annual_grad_n', 'annual_dropped_out_n'],
                                                              aggfunc = 'sum',
                                                              fill_value = 0)

print(grd_select_borough_df02.shape)
display(grd_select_borough_df02.head(35))
```

```
SELECT
    LOWER(hsi.borough) AS borough,
    CAST(grd.cohort AS INT) - 2006 AS relative_data_year,
    SUM(CAST(grd.total_grads_n AS DOUBLE)) AS annual_grad_n,
    SUM(CAST(grd.dropped_out_n AS DOUBLE)) AS annual_dropped_out_n
FROM ads508_t8.grad_outcomes AS grd
LEFT JOIN ads508_t8.hs_info AS hsi
    ON grd.dbn = hsi.dbn
WHERE total_grads_n <> 's'
    AND cohort != '2006 Aug'
    AND borough IS NOT null
    AND CAST(grd.cohort AS INT) BETWEEN 2002 AND 2006
GROUP BY hsi.borough, grd.cohort
ORDER BY hsi.borough, grd.cohort
LIMIT 100000

(25, 4)
```

	borough	relative_data_year	annual_grad_n	annual_dropped_out_n
0	bronx	-4	17130.0	2833.0

```
1           bronx                  -3        22123.0                4494.0
2           bronx                  -2        27594.0                4974.0
3           bronx                  -1        31643.0                5112.0
4           bronx                   0        33597.0                6251.0
5        brooklyn                  -4        38539.0                8505.0
6        brooklyn                  -3        44230.0                8488.0
7        brooklyn                  -2        51985.0                8323.0
8        brooklyn                  -1        53783.0                6934.0
9        brooklyn                   0        56436.0                7878.0
10      manhattan                  -4        28721.0                2741.0
11      manhattan                  -3        30950.0                3404.0
12      manhattan                  -2        34333.0                3079.0
13      manhattan                  -1        38817.0                2545.0
14      manhattan                   0        41153.0                3211.0
15         queens                  -4        47027.0               11220.0
16         queens                  -3        50268.0               10593.0
17         queens                  -2        53307.0                9598.0
18         queens                  -1        57757.0                9165.0
19         queens                   0        61196.0                8777.0
20  staten island                  -4        16133.0                2236.0
21  staten island                  -3        16651.0                1852.0
22  staten island                  -2        16391.0                1799.0
23  staten island                  -1        18054.0                1657.0
24  staten island                   0        19483.0                2035.0

(5, 10)

                   annual_dropped_out_n                          \
relative_data_year                   -4     -3    -2    -1     0
borough
bronx                              2833   4494  4974  5112  6251
brooklyn                           8505   8488  8323  6934  7878
manhattan                          2741   3404  3079  2545  3211
queens                            11220  10593  9598  9165  8777
staten island                      2236   1852  1799  1657  2035

                   annual_grad_n
relative_data_year            -4     -3     -2     -1      0
borough
bronx                      17130  22123  27594  31643  33597
brooklyn                   38539  44230  51985  53783  56436
manhattan                  28721  30950  34333  38817  41153
queens                     47027  50268  53307  57757  61196
staten island              16133  16651  16391  18054  19483
```

[21]:
```
# Run query to review a sample of records
grd_select_ct_stmnt01 = f"""
SELECT
```

```
    hsi.census_tract,
    SUM(CAST(grd.total_grads_n AS DOUBLE)) AS annual_grad_n,
    SUM(CAST(grd.dropped_out_n AS DOUBLE)) AS annual_dropped_out_n
FROM {database_name}.{grd_tsv_tbl_name} AS grd
LEFT JOIN {database_name}.{hsi_tsv_tbl_name} AS hsi
    ON grd.dbn = hsi.dbn
WHERE total_grads_n <> 's'
GROUP BY hsi.census_tract
ORDER BY hsi.census_tract
LIMIT 100000
"""

# Display SQL statement
print(grd_select_ct_stmnt01)

# Run SQL statement against Athena table
grd_select_ct_df01 = pd.read_sql(grd_select_ct_stmnt01,
                                 conn)

# Display results
print(grd_select_ct_df01.shape)
display(grd_select_ct_df01.head(50))
```

```
SELECT
    hsi.census_tract,
    SUM(CAST(grd.total_grads_n AS DOUBLE)) AS annual_grad_n,
    SUM(CAST(grd.dropped_out_n AS DOUBLE)) AS annual_dropped_out_n
FROM ads508_t8.grad_outcomes AS grd
LEFT JOIN ads508_t8.hs_info AS hsi
    ON grd.dbn = hsi.dbn
WHERE total_grads_n <> 's'
GROUP BY hsi.census_tract
ORDER BY hsi.census_tract
LIMIT 100000

(1, 3)
```

	census_tract	annual_grad_n	annual_dropped_out_n
0	None	1489323.0	308140.0

2.4.5 SELECT statements to prepare for full join: census table

[22]:
```
# Display full table for review
cen_full_select_stmnt01 = f"""
SELECT
    censustract AS census_tract,
    LOWER(borough) AS borough,
```

```
    totalpop,
    men,
    women,
    hispanic,
    white,
    black,
    native,
    asian,
    citizen,
    income,
    incomeerr,
    incomepercap,
    incomepercaperr,
    poverty,
    childpoverty,
    professional,
    service,
    office,
    construction,
    production,
    drive,
    carpool,
    transit,
    walk,
    othertransp,
    workathome,
    meancommute,
    employed,
    privatework,
    publicwork,
    selfemployed,
    familywork,
    unemployment
FROM {database_name}.{cen_tsv_tbl_name}
WHERE childpoverty IS NOT NULL
LIMIT 10000
"""

# Display SQL statement
print(cen_full_select_stmnt01)

# Run SQL statement against Athena table
cen_full_select_df01 = pd.read_sql(cen_full_select_stmnt01,
                                   conn)

# Display results
print(cen_full_select_df01.shape)
```

```
display(cen_full_select_df01.head(11))
```

```
SELECT
    censustract AS census_tract,
    LOWER(borough) AS borough,
    totalpop,
    men,
    women,
    hispanic,
    white,
    black,
    native,
    asian,
    citizen,
    income,
    incomeerr,
    incomepercap,
    incomepercaperr,
    poverty,
    childpoverty,
    professional,
    service,
    office,
    construction,
    production,
    drive,
    carpool,
    transit,
    walk,
    othertransp,
    workathome,
    meancommute,
    employed,
    privatework,
    publicwork,
    selfemployed,
    familywork,
    unemployment
FROM ads508_t8.census
WHERE childpoverty IS NOT NULL
LIMIT 10000

(2107, 35)
```

	census_tract	borough	totalpop	men	women	hispanic	white	black	\
0	36005000200	bronx	5403	2659	2744	75.8	2.3	16.0	
1	36005000400	bronx	5915	2896	3019	62.7	3.6	30.7	

```
2   36005001600   bronx      5879  2558   3321      65.1    1.6   32.4
3   36005001900   bronx      2591  1206   1385      55.4    9.0   29.0
4   36005002000   bronx      8516  3301   5215      61.1    1.6   31.1
5   36005002300   bronx      4774  2130   2644      62.3    0.2   36.5
6   36005002400   bronx       150   109     41       0.0   52.0   48.0
7   36005002500   bronx      5355  2338   3017      76.5    1.5   18.9
8   36005002701   bronx      3016  1375   1641      68.0    0.0   31.2
9   36005002702   bronx      4778  2427   2351      71.3    1.6   26.2
10  36005002800   bronx      5299  2292   3007      23.0    0.2   71.4

    native  asian  …  walk  othertransp  workathome  meancommute  employed  \
0      0.0    4.2  …   2.9          0.0         0.0         43.0      2308
1      0.0    0.3  …   1.4          0.5         2.1         45.0      2675
2      0.0    0.0  …   8.6          1.6         1.7         38.8      2120
3      0.0    2.1  …   3.0          2.4         6.2         45.4      1083
4      0.3    3.3  …   4.3          1.0         0.0         46.0      2508
5      1.0    0.0  …  14.0          1.5         4.1         42.7      1191
6      0.0    0.0  …   0.0          0.0         0.0          NaN       113
7      0.0    3.0  …  17.7          1.8         2.7         35.5      1691
8      0.0    0.0  …  18.0          0.0         1.6         42.8      1102
9      0.0    0.0  …   7.1          0.7         0.5         44.0      1559
10     0.0    1.7  …   2.0          0.6         2.7         47.3      2394

    privatework  publicwork  selfemployed  familywork  unemployment
0          80.8        16.2           2.9         0.0           7.7
1          71.7        25.3           2.5         0.6           9.5
2          75.0        21.3           3.8         0.0           8.7
3          76.8        15.5           7.7         0.0          19.2
4          71.0        21.3           7.7         0.0          17.2
5          74.2        16.1           9.7         0.0          18.9
6          62.8        37.2           0.0         0.0           0.0
7          85.1         8.3           6.1         0.5           9.4
8          86.9         8.5           4.5         0.0          15.2
9          75.0        14.0          11.0         0.0          10.6
10         61.9        37.4           0.6         0.0          12.8

[11 rows x 35 columns]
```

2.5 Setup ABT version 1 by joining census table with pivot tables of other tables (evictions, crime_pqt, and grad_outcomes) based on borough feature

[23]:
```
evi_borough_year_df03 = evi_borough_year_df02.reset_index()
cri_borough_year_type_df03 = cri_borough_year_type_df13.reset_index()
grd_select_borough_df03 = grd_select_borough_df02.reset_index()

display(cen_full_select_df01.head(11))
```

```
display(evi_borough_year_df03.head(5))
display(cri_borough_year_type_df03.head(5))
display(grd_select_borough_df03.head(5))

abt_df01 = pd.merge(cen_full_select_df01, evi_borough_year_df03,
                    on='borough')

abt_df01 = pd.merge(abt_df01, cri_borough_year_type_df03,
                    on='borough')

abt_df01 = pd.merge(abt_df01, grd_select_borough_df03,
                    on='borough')

display(abt_df01)
```

	census_tract	borough	totalpop	men	women	hispanic	white	black	\
0	36005000200	bronx	5403	2659	2744	75.8	2.3	16.0	
1	36005000400	bronx	5915	2896	3019	62.7	3.6	30.7	
2	36005001600	bronx	5879	2558	3321	65.1	1.6	32.4	
3	36005001900	bronx	2591	1206	1385	55.4	9.0	29.0	
4	36005002000	bronx	8516	3301	5215	61.1	1.6	31.1	
5	36005002300	bronx	4774	2130	2644	62.3	0.2	36.5	
6	36005002400	bronx	150	109	41	0.0	52.0	48.0	
7	36005002500	bronx	5355	2338	3017	76.5	1.5	18.9	
8	36005002701	bronx	3016	1375	1641	68.0	0.0	31.2	
9	36005002702	bronx	4778	2427	2351	71.3	1.6	26.2	
10	36005002800	bronx	5299	2292	3007	23.0	0.2	71.4	

	native	asian	…	walk	othertransp	workathome	meancommute	employed	\
0	0.0	4.2	…	2.9	0.0	0.0	43.0	2308	
1	0.0	0.3	…	1.4	0.5	2.1	45.0	2675	
2	0.0	0.0	…	8.6	1.6	1.7	38.8	2120	
3	0.0	2.1	…	3.0	2.4	6.2	45.4	1083	
4	0.3	3.3	…	4.3	1.0	0.0	46.0	2508	
5	1.0	0.0	…	14.0	1.5	4.1	42.7	1191	
6	0.0	0.0	…	0.0	0.0	0.0	NaN	113	
7	0.0	3.0	…	17.7	1.8	2.7	35.5	1691	
8	0.0	0.0	…	18.0	0.0	1.6	42.8	1102	
9	0.0	0.0	…	7.1	0.7	0.5	44.0	1559	
10	0.0	1.7	…	2.0	0.6	2.7	47.3	2394	

	privatework	publicwork	selfemployed	familywork	unemployment
0	80.8	16.2	2.9	0.0	7.7
1	71.7	25.3	2.5	0.6	9.5
2	75.0	21.3	3.8	0.0	8.7
3	76.8	15.5	7.7	0.0	19.2
4	71.0	21.3	7.7	0.0	17.2
5	74.2	16.1	9.7	0.0	18.9

```
6          62.8        37.2           0.0         0.0           0.0
7          85.1         8.3           6.1         0.5           9.4
8          86.9         8.5           4.5         0.0          15.2
9          75.0        14.0          11.0         0.0          10.6
10         61.9        37.4           0.6         0.0          12.8

[11 rows x 35 columns]
```

relative_data_year	borough	-4	-3	-2	-1	0
0	bronx	7140	6244	1088	29	1174
1	brooklyn	6157	5312	1005	100	1864
2	manhattan	3390	2818	521	68	930
3	queens	4452	3705	696	36	811
4	staten island	691	636	112	35	271

year_w_complaint	borough	2017 - FELONY	2017 - MISDEMEANOR
0	bronx	583	1179
1	brooklyn	862	1512
2	manhattan	701	1306
3	queens	610	932
4	staten island	100	249

year_w_complaint	2017 - VIOLATION	2018 - FELONY	2018 - MISDEMEANOR
0	289	540	1133
1	430	976	1473
2	267	736	1276
3	299	576	877
4	112	107	239

year_w_complaint	2018 - VIOLATION	2019 - FELONY	2019 - MISDEMEANOR
0	338	558	1114
1	423	909	1371
2	311	665	1302
3	309	604	970
4	109	101	196

year_w_complaint	2019 - VIOLATION	2020 - FELONY	2020 - MISDEMEANOR
0	328	529	954
1	435	803	1168
2	317	605	1085
3	292	607	911
4	61	104	175

year_w_complaint	2020 - VIOLATION	2021 - FELONY	2021 - MISDEMEANOR
0	322	617	910
1	384	834	1185
2	248	708	1145
3	290	646	995
4	57	96	174

```
year_w_complaint  2021 - VIOLATION
0                              320
1                              475
2                              295
3                              320
4                               75
```

```
                          borough annual_dropped_out_n                    \
relative_data_year                                     -4     -3    -2    -1
0                            bronx                   2833   4494  4974  5112
1                         brooklyn                   8505   8488  8323  6934
2                        manhattan                   2741   3404  3079  2545
3                           queens                  11220  10593  9598  9165
4                    staten island                   2236   1852  1799  1657

                         annual_grad_n
relative_data_year     0            -4     -3     -2     -1      0
0                   6251         17130  22123  27594  31643  33597
1                   7878         38539  44230  51985  53783  56436
2                   3211         28721  30950  34333  38817  41153
3                   8777         47027  50268  53307  57757  61196
4                   2035         16133  16651  16391  18054  19483
```

```
/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:18: FutureWarning:
merging between different levels is deprecated and will be removed in a future
version. (1 levels on the left,2 on the right)
/opt/conda/lib/python3.7/site-packages/pandas/core/generic.py:4150:
PerformanceWarning: dropping on a non-lexsorted multi-index without a level
parameter may impact performance.
  obj = obj._drop_axis(labels, axis, level=level, errors=errors)
```

```
     census_tract        borough  totalpop   men  women  hispanic  white  \
0     36005000200          bronx      5403  2659   2744      75.8    2.3
1     36005000400          bronx      5915  2896   3019      62.7    3.6
2     36005001600          bronx      5879  2558   3321      65.1    1.6
3     36005001900          bronx      2591  1206   1385      55.4    9.0
4     36005002000          bronx      8516  3301   5215      61.1    1.6
…           …             …        …    …     …        …    …
2102  36085030301  staten island      4895  2371   2524      30.7   40.2
2103  36085030302  staten island      6279  3093   3186      35.8   28.7
2104  36085031901  staten island      2550   953   1597      27.1    6.2
2105  36085031902  staten island      4611  2043   2568      20.9   14.7
2106  36085032300  staten island      1131   597    534      45.5   24.0

      black  native  asian  …  (annual_dropped_out_n, -4)  \
0      16.0     0.0    4.2  …                        2833
1      30.7     0.0    0.3  …                        2833
2      32.4     0.0    0.0  …                        2833
```

```
3      29.0     0.0    2.1  …                          2833
4      31.1     0.3    3.3  …                          2833
…      …     …    …  …                                …
2102   11.6     0.0   16.0  …                          2236
2103   17.6     0.0   14.3  …                          2236
2104   60.4     0.0    6.3  …                          2236
2105   61.9     0.0    0.9  …                          2236
2106   29.7     0.0    0.0  …                          2236

      (annual_dropped_out_n, -3)  (annual_dropped_out_n, -2)  \
0                           4494                        4974
1                           4494                        4974
2                           4494                        4974
3                           4494                        4974
4                           4494                        4974
…                           …                           …
2102                        1852                        1799
2103                        1852                        1799
2104                        1852                        1799
2105                        1852                        1799
2106                        1852                        1799

      (annual_dropped_out_n, -1)  (annual_dropped_out_n, 0)  \
0                           5112                        6251
1                           5112                        6251
2                           5112                        6251
3                           5112                        6251
4                           5112                        6251
…                           …                           …
2102                        1657                        2035
2103                        1657                        2035
2104                        1657                        2035
2105                        1657                        2035
2106                        1657                        2035

      (annual_grad_n, -4)  (annual_grad_n, -3)  (annual_grad_n, -2)  \
0                   17130                22123                27594
1                   17130                22123                27594
2                   17130                22123                27594
3                   17130                22123                27594
4                   17130                22123                27594
…                   …                    …                    …
2102                16133                16651                16391
2103                16133                16651                16391
2104                16133                16651                16391
2105                16133                16651                16391
2106                16133                16651                16391
```

```
      (annual_grad_n, -1)  (annual_grad_n, 0)
0                   31643               33597
1                   31643               33597
2                   31643               33597
3                   31643               33597
4                   31643               33597
…                   …                   …
2102                18054               19483
2103                18054               19483
2104                18054               19483
2105                18054               19483
2106                18054               19483

[2107 rows x 65 columns]
```

2.6 Setup ABT version 2 by joining census table with pivot tables of other tables (evictions, crime_pqt, and grad_outcomes) based on census_tract or borough features

[24]:
```
evi_ceb_join_select_df03 = evi_ceb_join_select_df02.reset_index()
cri_ceb_join_select_df03 = cri_ceb_join_select_df02.reset_index()
grd_select_borough_df03 = grd_select_borough_df02.reset_index()

display(cen_full_select_df01.head(11))

display(evi_ceb_join_select_df03.head(5))
display(cri_ceb_join_select_df03.head(5))
display(grd_select_borough_df03.head(5))

abt_df02 = pd.merge(cen_full_select_df01,
                    evi_ceb_join_select_df03,
                    how='left',
                    on='census_tract')

abt_df02 = pd.merge(abt_df02,
                    cri_ceb_join_select_df03,
                    how='left',
                    on='census_tract')

abt_df02 = pd.merge(abt_df02,
                    grd_select_borough_df03,
                    on='borough')
display(abt_df02)
```

```
   census_tract borough  totalpop   men  women  hispanic  white  black  \
0   36005000200   bronx      5403  2659   2744      75.8    2.3   16.0
1   36005000400   bronx      5915  2896   3019      62.7    3.6   30.7
2   36005001600   bronx      5879  2558   3321      65.1    1.6   32.4
```

```
3   36005001900   bronx      2591  1206   1385      55.4    9.0   29.0
4   36005002000   bronx      8516  3301   5215      61.1    1.6   31.1
5   36005002300   bronx      4774  2130   2644      62.3    0.2   36.5
6   36005002400   bronx       150   109     41       0.0   52.0   48.0
7   36005002500   bronx      5355  2338   3017      76.5    1.5   18.9
8   36005002701   bronx      3016  1375   1641      68.0    0.0   31.2
9   36005002702   bronx      4778  2427   2351      71.3    1.6   26.2
10  36005002800   bronx      5299  2292   3007      23.0    0.2   71.4

    native  asian  …  walk  othertransp  workathome  meancommute  employed  \
0      0.0    4.2  …   2.9          0.0         0.0         43.0      2308
1      0.0    0.3  …   1.4          0.5         2.1         45.0      2675
2      0.0    0.0  …   8.6          1.6         1.7         38.8      2120
3      0.0    2.1  …   3.0          2.4         6.2         45.4      1083
4      0.3    3.3  …   4.3          1.0         0.0         46.0      2508
5      1.0    0.0  …  14.0          1.5         4.1         42.7      1191
6      0.0    0.0  …   0.0          0.0         0.0          NaN       113
7      0.0    3.0  …  17.7          1.8         2.7         35.5      1691
8      0.0    0.0  …  18.0          0.0         1.6         42.8      1102
9      0.0    0.0  …   7.1          0.7         0.5         44.0      1559
10     0.0    1.7  …   2.0          0.6         2.7         47.3      2394

    privatework  publicwork  selfemployed  familywork  unemployment
0          80.8        16.2           2.9         0.0           7.7
1          71.7        25.3           2.5         0.6           9.5
2          75.0        21.3           3.8         0.0           8.7
3          76.8        15.5           7.7         0.0          19.2
4          71.0        21.3           7.7         0.0          17.2
5          74.2        16.1           9.7         0.0          18.9
6          62.8        37.2           0.0         0.0           0.0
7          85.1         8.3           6.1         0.5           9.4
8          86.9         8.5           4.5         0.0          15.2
9          75.0        14.0          11.0         0.0          10.6
10         61.9        37.4           0.6         0.0          12.8

[11 rows x 35 columns]

year census_tract  2017  2018  2019  2020  2021  2022  2023
0     34003013001     9    11     6     0     1     6     1
1     34003016000    14    13    16     2     2     1     0
2     34017010800     2     2     5     0     0     0     0
3     36005000200     9    10    10     0     0     1     1
4     36005000400    10    16    15     0     0     5     0

year census_tract  2017 - FELONY  2017 - MISDEMEANOR  2017 - VIOLATION  \
0     34003013001              1                   6                 4
1     34003016000              0                   4                 0
2     34017010800              0                   1                 0
3     36005000100             16                  10                 0
```

```
4     36005000200              0                   1                1

year  2018 - FELONY  2018 - MISDEMEANOR  2018 - VIOLATION  2019 - FELONY  \
0                 3                   5                 0              1
1                 3                   0                 2              3
2                 0                   0                 0              0
3                 5                  10                 0             10
4                 3                   3                 1              1

year  2019 - MISDEMEANOR  2019 - VIOLATION  2020 - FELONY  2020 - MISDEMEANOR  \
0                      6                 1              3                   2
1                      1                 0              0                   0
2                      0                 0              0                   1
3                      4                 1              3                   2
4                      2                 2              3                   0

year  2020 - VIOLATION  2021 - FELONY  2021 - MISDEMEANOR  2021 - VIOLATION
0                    1              6                   5                 2
1                    0              3                   1                 0
2                    0              0                   0                 0
3                    0              2                   1                 0
4                    0              3                   2                 1

                         borough annual_dropped_out_n                    \
relative_data_year                                   -4     -3    -2    -1
0                           bronx                  2833   4494  4974  5112
1                        brooklyn                  8505   8488  8323  6934
2                       manhattan                  2741   3404  3079  2545
3                          queens                 11220  10593  9598  9165
4                  staten island                  2236   1852  1799  1657

                        annual_grad_n
relative_data_year     0            -4     -3     -2     -1      0
0                   6251         17130  22123  27594  31643  33597
1                   7878         38539  44230  51985  53783  56436
2                   3211         28721  30950  34333  38817  41153
3                   8777         47027  50268  53307  57757  61196
4                   2035         16133  16651  16391  18054  19483

/opt/conda/lib/python3.7/site-packages/ipykernel_launcher.py:23: FutureWarning:
merging between different levels is deprecated and will be removed in a future
version. (1 levels on the left,2 on the right)
/opt/conda/lib/python3.7/site-packages/pandas/core/generic.py:4150:
PerformanceWarning: dropping on a non-lexsorted multi-index without a level
parameter may impact performance.
  obj = obj._drop_axis(labels, axis, level=level, errors=errors)

     census_tract        borough  totalpop   men  women  hispanic  white  \
0     36005000200          bronx      5403  2659   2744      75.8    2.3
```

```
1     36005000400          bronx      5915  2896   3019      62.7    3.6
2     36005001600          bronx      5879  2558   3321      65.1    1.6
3     36005001900          bronx      2591  1206   1385      55.4    9.0
4     36005002000          bronx      8516  3301   5215      61.1    1.6
…           …              …        …    …    …        …    …
2102  36085030301  staten island      4895  2371   2524      30.7   40.2
2103  36085030302  staten island      6279  3093   3186      35.8   28.7
2104  36085031901  staten island      2550   953   1597      27.1    6.2
2105  36085031902  staten island      4611  2043   2568      20.9   14.7
2106  36085032300  staten island      1131   597    534      45.5   24.0

      black  native  asian  …  (annual_dropped_out_n, -4)  \
0      16.0     0.0    4.2  …                        2833
1      30.7     0.0    0.3  …                        2833
2      32.4     0.0    0.0  …                        2833
3      29.0     0.0    2.1  …                        2833
4      31.1     0.3    3.3  …                        2833
…      …      …     …   …                          …
2102   11.6     0.0   16.0  …                        2236
2103   17.6     0.0   14.3  …                        2236
2104   60.4     0.0    6.3  …                        2236
2105   61.9     0.0    0.9  …                        2236
2106   29.7     0.0    0.0  …                        2236

      (annual_dropped_out_n, -3)  (annual_dropped_out_n, -2)  \
0                           4494                        4974
1                           4494                        4974
2                           4494                        4974
3                           4494                        4974
4                           4494                        4974
…                           …                          …
2102                        1852                        1799
2103                        1852                        1799
2104                        1852                        1799
2105                        1852                        1799
2106                        1852                        1799

      (annual_dropped_out_n, -1)  (annual_dropped_out_n, 0)  \
0                           5112                       6251
1                           5112                       6251
2                           5112                       6251
3                           5112                       6251
4                           5112                       6251
…                           …                          …
2102                        1657                       2035
2103                        1657                       2035
2104                        1657                       2035
2105                        1657                       2035
```

```
2106                        1657                        2035

      (annual_grad_n, -4)  (annual_grad_n, -3)  (annual_grad_n, -2)  \
0                   17130                22123                27594
1                   17130                22123                27594
2                   17130                22123                27594
3                   17130                22123                27594
4                   17130                22123                27594
…                     …                    …                    …
2102                16133                16651                16391
2103                16133                16651                16391
2104                16133                16651                16391
2105                16133                16651                16391
2106                16133                16651                16391

      (annual_grad_n, -1)  (annual_grad_n, 0)
0                   31643               33597
1                   31643               33597
2                   31643               33597
3                   31643               33597
4                   31643               33597
…                     …                   …
2102                18054               19483
2103                18054               19483
2104                18054               19483
2105                18054               19483
2106                18054               19483

[2107 rows x 67 columns]
```

2.7 Setup ABT version 3 (FINAL) by joining census table with pivot tables of other tables (evictions, crime_pqt, and grad_outcomes) based on borough feature with or without relative_data_year

[25]:
```
cen_full_select_df02 = cen_full_select_df01.drop(['census_tract'],
                                                 axis=1)

display(cen_full_select_df02.head(11))
print(cen_full_select_df02.shape)

display(evi_borough_year_df01.head(11))
print(evi_borough_year_df01.shape)
display(cri_borough_year_type_df01.head(11))
print(cri_borough_year_type_df01.shape)
display(grd_select_borough_df01.head(11))
print(grd_select_borough_df01.shape)
```

```
   borough  totalpop   men  women  hispanic  white  black  native  asian  \
```

```
0    bronx      5403  2659   2744      75.8    2.3   16.0     0.0    4.2
1    bronx      5915  2896   3019      62.7    3.6   30.7     0.0    0.3
2    bronx      5879  2558   3321      65.1    1.6   32.4     0.0    0.0
3    bronx      2591  1206   1385      55.4    9.0   29.0     0.0    2.1
4    bronx      8516  3301   5215      61.1    1.6   31.1     0.3    3.3
5    bronx      4774  2130   2644      62.3    0.2   36.5     1.0    0.0
6    bronx       150   109     41       0.0   52.0   48.0     0.0    0.0
7    bronx      5355  2338   3017      76.5    1.5   18.9     0.0    3.0
8    bronx      3016  1375   1641      68.0    0.0   31.2     0.0    0.0
9    bronx      4778  2427   2351      71.3    1.6   26.2     0.0    0.0
10   bronx      5299  2292   3007      23.0    0.2   71.4     0.0    1.7

    citizen  …  walk  othertransp  workathome  meancommute  employed  \
0      3639  …   2.9          0.0         0.0         43.0      2308
1      4100  …   1.4          0.5         2.1         45.0      2675
2      3536  …   8.6          1.6         1.7         38.8      2120
3      1557  …   3.0          2.4         6.2         45.4      1083
4      5436  …   4.3          1.0         0.0         46.0      2508
5      3056  …  14.0          1.5         4.1         42.7      1191
6        41  …   0.0          0.0         0.0          NaN       113
7      2509  …  17.7          1.8         2.7         35.5      1691
8      1456  …  18.0          0.0         1.6         42.8      1102
9      2365  …   7.1          0.7         0.5         44.0      1559
10     4056  …   2.0          0.6         2.7         47.3      2394

    privatework  publicwork  selfemployed  familywork  unemployment
0          80.8        16.2           2.9         0.0           7.7
1          71.7        25.3           2.5         0.6           9.5
2          75.0        21.3           3.8         0.0           8.7
3          76.8        15.5           7.7         0.0          19.2
4          71.0        21.3           7.7         0.0          17.2
5          74.2        16.1           9.7         0.0          18.9
6          62.8        37.2           0.0         0.0           0.0
7          85.1         8.3           6.1         0.5           9.4
8          86.9         8.5           4.5         0.0          15.2
9          75.0        14.0          11.0         0.0          10.6
10         61.9        37.4           0.6         0.0          12.8

[11 rows x 34 columns]

(2107, 34)

      borough  relative_data_year  annual_evictions_x_borough
0       bronx                  -4                        7140
1       bronx                  -3                        6244
2       bronx                  -2                        1088
3       bronx                  -1                          29
4       bronx                   0                        1174
5    brooklyn                  -4                        6157
```

```
6    brooklyn                  -3                        5312
7    brooklyn                  -2                        1005
8    brooklyn                  -1                         100
9    brooklyn                   0                        1864
10  manhattan                  -4                        3390
```

```
(25, 3)
```

```
   borough  relative_data_year complaint_type  annual_complaint_counts
0    bronx                  -4         FELONY                      583
1    bronx                  -4    MISDEMEANOR                     1179
2    bronx                  -4      VIOLATION                      289
3    bronx                  -3         FELONY                      540
4    bronx                  -3    MISDEMEANOR                     1133
5    bronx                  -3      VIOLATION                      338
6    bronx                  -2         FELONY                      558
7    bronx                  -2    MISDEMEANOR                     1114
8    bronx                  -2      VIOLATION                      328
9    bronx                  -1         FELONY                      529
10   bronx                  -1    MISDEMEANOR                      954
```

```
(81, 4)
```

```
      borough  relative_data_year  annual_grad_n  annual_dropped_out_n
0       bronx                  -4        17130.0                2833.0
1       bronx                  -3        22123.0                4494.0
2       bronx                  -2        27594.0                4974.0
3       bronx                  -1        31643.0                5112.0
4       bronx                   0        33597.0                6251.0
5    brooklyn                  -4        38539.0                8505.0
6    brooklyn                  -3        44230.0                8488.0
7    brooklyn                  -2        51985.0                8323.0
8    brooklyn                  -1        53783.0                6934.0
9    brooklyn                   0        56436.0                7878.0
10  manhattan                  -4        28721.0                2741.0
```

```
(25, 4)
```

[26]:
```
abt_df03 = pd.merge(evi_borough_year_df01,
                    cri_borough_year_type_df01,
                    how='inner',
                    left_on=['borough', 'relative_data_year'],
                    right_on=['borough', 'relative_data_year'])

abt_df03 = pd.merge(abt_df03,
                    grd_select_borough_df01,
                    how='inner',
                    left_on=['borough', 'relative_data_year'],
                    right_on=['borough', 'relative_data_year'])
```

```
abt_df03 = pd.merge(abt_df03,
                    cen_full_select_df02,
                    how='inner',
                    on='borough')

print(abt_df03.shape)
display(abt_df03.head(11))
```

```
(31605, 40)
```

	borough	relative_data_year	annual_evictions_x_borough	complaint_type	\
0	bronx	-4	7140	FELONY	
1	bronx	-4	7140	FELONY	
2	bronx	-4	7140	FELONY	
3	bronx	-4	7140	FELONY	
4	bronx	-4	7140	FELONY	
5	bronx	-4	7140	FELONY	
6	bronx	-4	7140	FELONY	
7	bronx	-4	7140	FELONY	
8	bronx	-4	7140	FELONY	
9	bronx	-4	7140	FELONY	
10	bronx	-4	7140	FELONY	

	annual_complaint_counts	annual_grad_n	annual_dropped_out_n	totalpop	\
0	583	17130.0	2833.0	5403	
1	583	17130.0	2833.0	5915	
2	583	17130.0	2833.0	5879	
3	583	17130.0	2833.0	2591	
4	583	17130.0	2833.0	8516	
5	583	17130.0	2833.0	4774	
6	583	17130.0	2833.0	150	
7	583	17130.0	2833.0	5355	
8	583	17130.0	2833.0	3016	
9	583	17130.0	2833.0	4778	
10	583	17130.0	2833.0	5299	

	men	women	…	walk	othertransp	workathome	meancommute	employed	\
0	2659	2744	…	2.9	0.0	0.0	43.0	2308	
1	2896	3019	…	1.4	0.5	2.1	45.0	2675	
2	2558	3321	…	8.6	1.6	1.7	38.8	2120	
3	1206	1385	…	3.0	2.4	6.2	45.4	1083	
4	3301	5215	…	4.3	1.0	0.0	46.0	2508	
5	2130	2644	…	14.0	1.5	4.1	42.7	1191	
6	109	41	…	0.0	0.0	0.0	NaN	113	
7	2338	3017	…	17.7	1.8	2.7	35.5	1691	
8	1375	1641	…	18.0	0.0	1.6	42.8	1102	
9	2427	2351	…	7.1	0.7	0.5	44.0	1559	
10	2292	3007	…	2.0	0.6	2.7	47.3	2394	

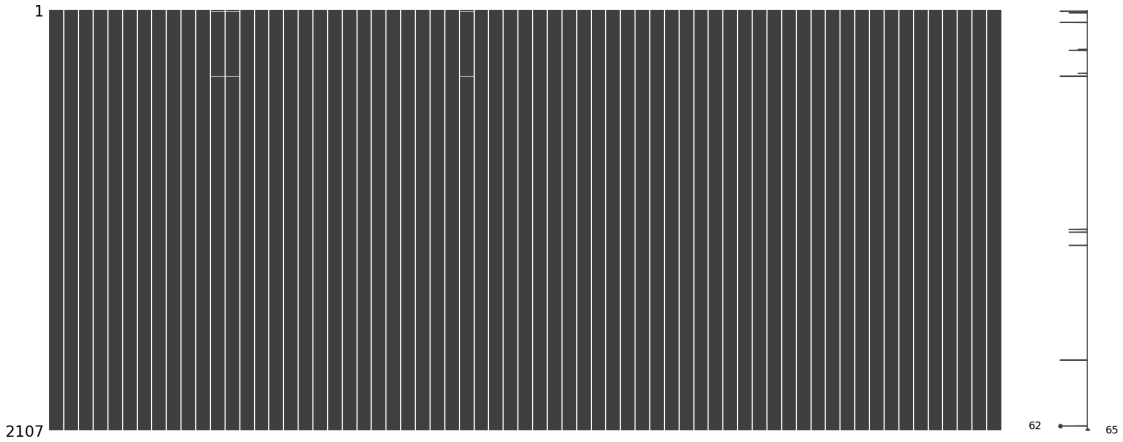
	privatework	publicwork	selfemployed	familywork	unemployment
0	80.8	16.2	2.9	0.0	7.7
1	71.7	25.3	2.5	0.6	9.5
2	75.0	21.3	3.8	0.0	8.7
3	76.8	15.5	7.7	0.0	19.2
4	71.0	21.3	7.7	0.0	17.2
5	74.2	16.1	9.7	0.0	18.9
6	62.8	37.2	0.0	0.0	0.0
7	85.1	8.3	6.1	0.5	9.4
8	86.9	8.5	4.5	0.0	15.2
9	75.0	14.0	11.0	0.0	10.6
10	61.9	37.4	0.6	0.0	12.8

[11 rows x 40 columns]

2.7.1 Check missing values for resulting ABTs

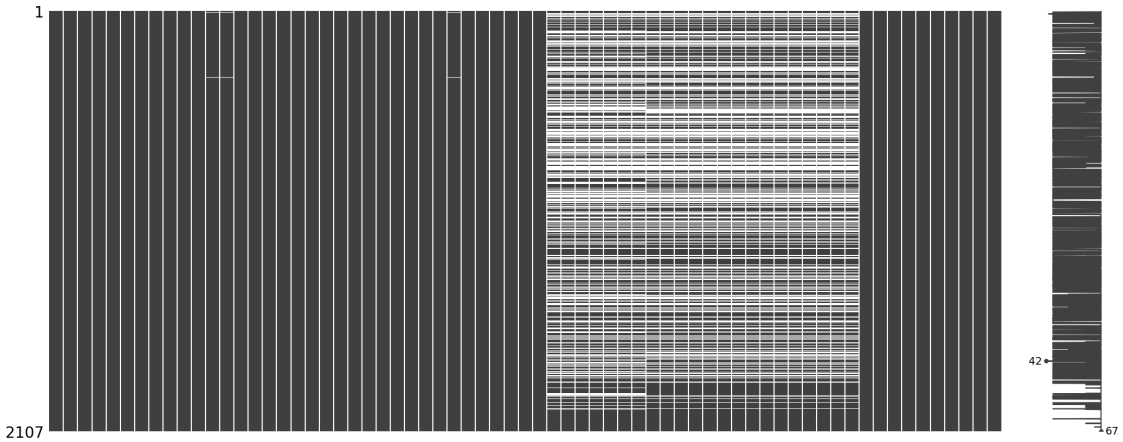
[27]:
```
# Visualize missing values in each column
msno.matrix(abt_df01)
```

[27]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe7fa33eb50>



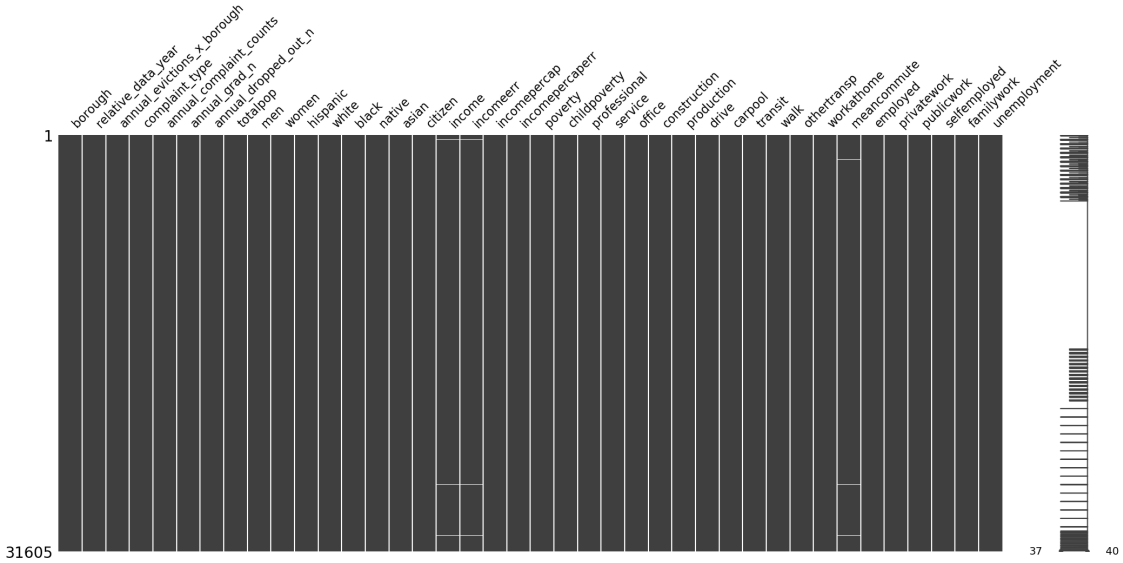
[28]:
```
# Visualize missing values in each column
msno.matrix(abt_df02)
```

[28]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe7fabc62d0>



[29]:
```
# Visualize missing values in each column
msno.matrix(abt_df03)
```

[29]: <matplotlib.axes._subplots.AxesSubplot at 0x7fe7faf91c90>



[30]:
```
# Remove any features for which the number of null vals exceed a threshold--
#-- (5% of total N)
abt_df02_null_summ01 = pd.DataFrame(abt_df02.isnull().sum(),
                                    columns=['null_count'])

abt_df02_null_summ02 = abt_df02_null_summ01.
 ↪loc[(abt_df02_null_summ01['null_count'] != 0)].sort_values('null_count',
```

```
                                                                        ␣
  ↪                                    ascending=False)
abt_df02_null_summ03 = abt_df02_null_summ02.reset_index()
print(abt_df02_null_summ03)

abt_df02_null_summ04 = abt_df02_null_summ03.
  ↪loc[abt_df02_null_summ03['null_count'] > (len(abt_df02)*.05)]
print('\n', abt_df02_null_summ04)

abt_df02_null_summ04_remove_lst01 = list(abt_df02_null_summ04['index'])
print('\n', abt_df02_null_summ04_remove_lst01)
```

```
                index  null_count
0                2017         857
1                2018         857
2                2019         857
3                2020         857
4                2021         857
5                2022         857
6                2023         857
7     2017 - VIOLATION         812
8        2019 - FELONY         812
9   2021 - MISDEMEANOR         812
10       2021 - FELONY         812
11    2020 - VIOLATION         812
12  2020 - MISDEMEANOR         812
13       2020 - FELONY         812
14    2019 - VIOLATION         812
15  2019 - MISDEMEANOR         812
16    2021 - VIOLATION         812
17    2018 - VIOLATION         812
18  2018 - MISDEMEANOR         812
19       2018 - FELONY         812
20  2017 - MISDEMEANOR         812
21       2017 - FELONY         812
22           incomeerr          10
23              income          10
24        meancommute           7

                index  null_count
0                2017         857
1                2018         857
2                2019         857
3                2020         857
4                2021         857
5                2022         857
6                2023         857
```

```
7     2017 - VIOLATION         812
8        2019 - FELONY         812
9   2021 - MISDEMEANOR         812
10       2021 - FELONY         812
11    2020 - VIOLATION         812
12  2020 - MISDEMEANOR         812
13       2020 - FELONY         812
14    2019 - VIOLATION         812
15  2019 - MISDEMEANOR         812
16    2021 - VIOLATION         812
17    2018 - VIOLATION         812
18  2018 - MISDEMEANOR         812
19       2018 - FELONY         812
20  2017 - MISDEMEANOR         812
21       2017 - FELONY         812
```

```
 [2017, 2018, 2019, 2020, 2021, 2022, 2023, '2017 - VIOLATION', '2019 - FELONY',
'2021 - MISDEMEANOR', '2021 - FELONY', '2020 - VIOLATION', '2020 - MISDEMEANOR',
'2020 - FELONY', '2019 - VIOLATION', '2019 - MISDEMEANOR', '2021 - VIOLATION',
'2018 - VIOLATION', '2018 - MISDEMEANOR', '2018 - FELONY', '2017 - MISDEMEANOR',
'2017 - FELONY']
```

2.7.2 Create pipeline for One Hot Encoding

[31]:
```
'''Setup pipelne citation:
OpenAI. (2021). ChatGPT [Computer software]. https://openai.com/'''

print(abt_df03.shape)
display(abt_df03.head(11))

# Define a ColumnTransformer to apply the OneHotEncoder to the selected columns
cols_to_encode = ['borough',
                  'relative_data_year',
                  'complaint_type']

ct = ColumnTransformer(transformers=[('encoder',
                                      OneHotEncoder(),
                                      cols_to_encode)],
                       remainder='passthrough'
)

# Define a Pipeline to apply the ColumnTransformer
abt_pipe = Pipeline(steps=[('preprocessor',
                            ct)])

# Fit and transform the Pipeline to one-hot encode the selected columns
abt_encoded_df01 = pd.DataFrame(abt_pipe.fit_transform(abt_df03))
```

```
# Get the names of the one-hot encoded columns from the OneHotEncoder object
encoder = abt_pipe.named_steps['preprocessor'].named_transformers_['encoder']
print(encoder)

encoded_cols = encoder.get_feature_names(cols_to_encode)
print(encoded_cols)

# Get the names of the non-encoded columns by removing the columns that were␣
 ↪encoded
non_encoded_cols = [col for col in abt_df03.columns if col not in␣
 ↪cols_to_encode]
print(non_encoded_cols)

# Concatenate the one-hot encoded columns with the non-encoded columns to␣
 ↪obtain the new DataFrame with the desired column names
abt_encoded_df01.columns = list(encoded_cols) + non_encoded_cols
print(abt_encoded_df01.head(11))

display(abt_encoded_df01.head(11))
```

```
(31605, 40)
```

	borough	relative_data_year	annual_evictions_x_borough	complaint_type	\
0	bronx	-4	7140	FELONY	
1	bronx	-4	7140	FELONY	
2	bronx	-4	7140	FELONY	
3	bronx	-4	7140	FELONY	
4	bronx	-4	7140	FELONY	
5	bronx	-4	7140	FELONY	
6	bronx	-4	7140	FELONY	
7	bronx	-4	7140	FELONY	
8	bronx	-4	7140	FELONY	
9	bronx	-4	7140	FELONY	
10	bronx	-4	7140	FELONY	

	annual_complaint_counts	annual_grad_n	annual_dropped_out_n	totalpop	\
0	583	17130.0	2833.0	5403	
1	583	17130.0	2833.0	5915	
2	583	17130.0	2833.0	5879	
3	583	17130.0	2833.0	2591	
4	583	17130.0	2833.0	8516	
5	583	17130.0	2833.0	4774	
6	583	17130.0	2833.0	150	
7	583	17130.0	2833.0	5355	
8	583	17130.0	2833.0	3016	
9	583	17130.0	2833.0	4778	
10	583	17130.0	2833.0	5299	

```
     men  women  …  walk  othertransp  workathome  meancommute  employed  \
0   2659   2744  …   2.9          0.0         0.0         43.0      2308
1   2896   3019  …   1.4          0.5         2.1         45.0      2675
2   2558   3321  …   8.6          1.6         1.7         38.8      2120
3   1206   1385  …   3.0          2.4         6.2         45.4      1083
4   3301   5215  …   4.3          1.0         0.0         46.0      2508
5   2130   2644  …  14.0          1.5         4.1         42.7      1191
6    109     41  …   0.0          0.0         0.0          NaN       113
7   2338   3017  …  17.7          1.8         2.7         35.5      1691
8   1375   1641  …  18.0          0.0         1.6         42.8      1102
9   2427   2351  …   7.1          0.7         0.5         44.0      1559
10  2292   3007  …   2.0          0.6         2.7         47.3      2394

    privatework  publicwork  selfemployed  familywork  unemployment
0          80.8        16.2           2.9         0.0           7.7
1          71.7        25.3           2.5         0.6           9.5
2          75.0        21.3           3.8         0.0           8.7
3          76.8        15.5           7.7         0.0          19.2
4          71.0        21.3           7.7         0.0          17.2
5          74.2        16.1           9.7         0.0          18.9
6          62.8        37.2           0.0         0.0           0.0
7          85.1         8.3           6.1         0.5           9.4
8          86.9         8.5           4.5         0.0          15.2
9          75.0        14.0          11.0         0.0          10.6
10         61.9        37.4           0.6         0.0          12.8

[11 rows x 40 columns]
```

```
OneHotEncoder(categories='auto', drop=None, dtype=<class 'numpy.float64'>,
              handle_unknown='error', sparse=True)
['borough_bronx' 'borough_brooklyn' 'borough_manhattan' 'borough_queens'
 'borough_staten island' 'relative_data_year_-4' 'relative_data_year_-3'
 'relative_data_year_-2' 'relative_data_year_-1' 'relative_data_year_0'
 'complaint_type_FELONY' 'complaint_type_MISDEMEANOR'
 'complaint_type_VIOLATION']
['annual_evictions_x_borough', 'annual_complaint_counts', 'annual_grad_n',
'annual_dropped_out_n', 'totalpop', 'men', 'women', 'hispanic', 'white',
'black', 'native', 'asian', 'citizen', 'income', 'incomeerr', 'incomepercap',
'incomepercaperr', 'poverty', 'childpoverty', 'professional', 'service',
'office', 'construction', 'production', 'drive', 'carpool', 'transit', 'walk',
'othertransp', 'workathome', 'meancommute', 'employed', 'privatework',
'publicwork', 'selfemployed', 'familywork', 'unemployment']
    borough_bronx  borough_brooklyn  borough_manhattan  borough_queens  \
0             1.0               0.0                0.0             0.0
1             1.0               0.0                0.0             0.0
2             1.0               0.0                0.0             0.0
3             1.0               0.0                0.0             0.0
```

```
4              1.0                0.0                0.0              0.0
5              1.0                0.0                0.0              0.0
6              1.0                0.0                0.0              0.0
7              1.0                0.0                0.0              0.0
8              1.0                0.0                0.0              0.0
9              1.0                0.0                0.0              0.0
10             1.0                0.0                0.0              0.0

    borough_staten island  relative_data_year_-4  relative_data_year_-3  \
0                     0.0                    1.0                    0.0
1                     0.0                    1.0                    0.0
2                     0.0                    1.0                    0.0
3                     0.0                    1.0                    0.0
4                     0.0                    1.0                    0.0
5                     0.0                    1.0                    0.0
6                     0.0                    1.0                    0.0
7                     0.0                    1.0                    0.0
8                     0.0                    1.0                    0.0
9                     0.0                    1.0                    0.0
10                    0.0                    1.0                    0.0

    relative_data_year_-2  relative_data_year_-1  relative_data_year_0  …  \
0                     0.0                    0.0                   0.0  …
1                     0.0                    0.0                   0.0  …
2                     0.0                    0.0                   0.0  …
3                     0.0                    0.0                   0.0  …
4                     0.0                    0.0                   0.0  …
5                     0.0                    0.0                   0.0  …
6                     0.0                    0.0                   0.0  …
7                     0.0                    0.0                   0.0  …
8                     0.0                    0.0                   0.0  …
9                     0.0                    0.0                   0.0  …
10                    0.0                    0.0                   0.0  …

    walk  othertransp  workathome  meancommute  employed  privatework  \
0    2.9          0.0         0.0         43.0    2308.0         80.8
1    1.4          0.5         2.1         45.0    2675.0         71.7
2    8.6          1.6         1.7         38.8    2120.0         75.0
3    3.0          2.4         6.2         45.4    1083.0         76.8
4    4.3          1.0         0.0         46.0    2508.0         71.0
5   14.0          1.5         4.1         42.7    1191.0         74.2
6    0.0          0.0         0.0          NaN     113.0         62.8
7   17.7          1.8         2.7         35.5    1691.0         85.1
8   18.0          0.0         1.6         42.8    1102.0         86.9
9    7.1          0.7         0.5         44.0    1559.0         75.0
10   2.0          0.6         2.7         47.3    2394.0         61.9

    publicwork  selfemployed  familywork  unemployment
```

```
0         16.2           2.9         0.0           7.7
1         25.3           2.5         0.6           9.5
2         21.3           3.8         0.0           8.7
3         15.5           7.7         0.0          19.2
4         21.3           7.7         0.0          17.2
5         16.1           9.7         0.0          18.9
6         37.2           0.0         0.0           0.0
7          8.3           6.1         0.5           9.4
8          8.5           4.5         0.0          15.2
9         14.0          11.0         0.0          10.6
10        37.4           0.6         0.0          12.8

[11 rows x 50 columns]
    borough_bronx  borough_brooklyn  borough_manhattan  borough_queens  \
0             1.0               0.0                0.0             0.0
1             1.0               0.0                0.0             0.0
2             1.0               0.0                0.0             0.0
3             1.0               0.0                0.0             0.0
4             1.0               0.0                0.0             0.0
5             1.0               0.0                0.0             0.0
6             1.0               0.0                0.0             0.0
7             1.0               0.0                0.0             0.0
8             1.0               0.0                0.0             0.0
9             1.0               0.0                0.0             0.0
10            1.0               0.0                0.0             0.0

    borough_staten island  relative_data_year_-4  relative_data_year_-3  \
0                     0.0                    1.0                    0.0
1                     0.0                    1.0                    0.0
2                     0.0                    1.0                    0.0
3                     0.0                    1.0                    0.0
4                     0.0                    1.0                    0.0
5                     0.0                    1.0                    0.0
6                     0.0                    1.0                    0.0
7                     0.0                    1.0                    0.0
8                     0.0                    1.0                    0.0
9                     0.0                    1.0                    0.0
10                    0.0                    1.0                    0.0

    relative_data_year_-2  relative_data_year_-1  relative_data_year_0  …  \
0                     0.0                    0.0                   0.0  …
1                     0.0                    0.0                   0.0  …
2                     0.0                    0.0                   0.0  …
3                     0.0                    0.0                   0.0  …
4                     0.0                    0.0                   0.0  …
5                     0.0                    0.0                   0.0  …
6                     0.0                    0.0                   0.0  …
```

```
7                    0.0                    0.0                    0.0  …
8                    0.0                    0.0                    0.0  …
9                    0.0                    0.0                    0.0  …
10                   0.0                    0.0                    0.0  …

    walk  othertransp  workathome  meancommute  employed  privatework  \
0    2.9          0.0         0.0         43.0    2308.0         80.8
1    1.4          0.5         2.1         45.0    2675.0         71.7
2    8.6          1.6         1.7         38.8    2120.0         75.0
3    3.0          2.4         6.2         45.4    1083.0         76.8
4    4.3          1.0         0.0         46.0    2508.0         71.0
5   14.0          1.5         4.1         42.7    1191.0         74.2
6    0.0          0.0         0.0          NaN     113.0         62.8
7   17.7          1.8         2.7         35.5    1691.0         85.1
8   18.0          0.0         1.6         42.8    1102.0         86.9
9    7.1          0.7         0.5         44.0    1559.0         75.0
10   2.0          0.6         2.7         47.3    2394.0         61.9

    publicwork  selfemployed  familywork  unemployment
0         16.2           2.9         0.0           7.7
1         25.3           2.5         0.6           9.5
2         21.3           3.8         0.0           8.7
3         15.5           7.7         0.0          19.2
4         21.3           7.7         0.0          17.2
5         16.1           9.7         0.0          18.9
6         37.2           0.0         0.0           0.0
7          8.3           6.1         0.5           9.4
8          8.5           4.5         0.0          15.2
9         14.0          11.0         0.0          10.6
10        37.4           0.6         0.0          12.8

[11 rows x 50 columns]
```

2.7.3 Save ABT to S3

[32]:
```
s3_abt_csv_path = f"s3://{def_bucket}/team_8_data/abt/abt_encoded_df01.csv"
abt_encoded_df01.to_csv(s3_abt_csv_path, index=False, header=True)
```

2.8 Show the Tables

[33]:
```
show_tbl_stmnt = f"SHOW TABLES in {database_name}"
```

[34]:
```
df_tables = pd.read_sql(show_tbl_stmnt,
                        conn)

df_tables.head(17)
```

[34]:
```
         tab_name
0          census
1   census_block
2           crime
3       crime_pqt
4       evictions
5  grad_outcomes
6         hs_info
7            jobs
```

2.9 Review the New Athena Table in the Glue Catalog

[35]:
```
display(
    HTML(
        f'<b>Review <a target="top" href="https://console.aws.amazon.com/glue/
↪home?region={region}#">AWS Glue Catalog</a></b>'
    )
)
```

```
<IPython.core.display.HTML object>
```

2.10 Store Variables for the Next Notebooks

[36]:
```
%store
```

```
Stored variables and their in-db values:
balance_dataset                                 -> True
balanced_bias_data_jsonlines_s3_uri             -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
balanced_bias_data_s3_uri                       -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
bias_data_s3_uri                                -> 's3://sagemaker-us-
east-1-657724983756/bias-detect
experiment_name                                 -> 'Amazon-Customer-
Reviews-BERT-Experiment-168013737
feature_group_name                              -> 'reviews-feature-
group-1680137375'
feature_store_offline_prefix                    -> 'reviews-feature-
store-1680137375'
ingest_create_athena_db_passed                  -> True
ingest_create_athena_table_parquet_passed       -> True
ingest_create_athena_table_tsv_passed           -> True
max_seq_length                                  -> 64
processed_test_data_s3_uri                      -> 's3://sagemaker-us-
east-1-657724983756/sagemaker-s
processed_train_data_s3_uri                     -> 's3://sagemaker-us-
east-1-657724983756/sagemaker-s
processed_validation_data_s3_uri                -> 's3://sagemaker-us-
```

```
east-1-657724983756/sagemaker-s
raw_input_data_s3_uri                                -> 's3://sagemaker-us-
east-1-657724983756/amazon-revi
s3_private_path_tsv                                  -> 's3://sagemaker-us-
east-1-657724983756/team_8_data
s3_public_path_tsv                                   -> 's3://sagemaker-us-
east-ads508-sp23-t8'
setup_dependencies_passed                            -> True
setup_iam_roles_passed                               -> True
setup_instance_check_passed                          -> True
setup_s3_bucket_passed                               -> True
test_split_percentage                                -> 0.05
train_split_percentage                               -> 0.9
trial_name                                           -> 'trial-1680137374'
validation_split_percentage                          -> 0.05
```

2.11 Release Resources

[37]:
```
%%html

<p><b>Shutting down your kernel for this notebook to release resources.</b></p>
<button class="sm-command-button" data-commandlinker-command="kernelmenu:
 ↪shutdown" style="display:none;">Shutdown Kernel</button>

<script>
try {
    els = document.getElementsByClassName("sm-command-button");
    els[0].click();
}
catch(err) {
    // NoOp
}
</script>
```

```
<IPython.core.display.HTML object>
```

[38]:
```
%%javascript

try {
    Jupyter.notebook.save_checkpoint();
    Jupyter.notebook.session.delete();
}
catch(err) {
    // NoOp
}
```

```
<IPython.core.display.Javascript object>
```

03_Split_Final

April 14, 2023

1 ADS-508-01-SP23 Team 8: Final Project

2 Split and Preprocess ABT

Much of the code is modified from `Fregly, C., & Barth, A. (2021). Data science on AWS: Implementing end-to-end, continuous AI and machine learning pipelines. O'Reilly.`

2.1 Install missing dependencies

PyAthena is a Python DB API 2.0 (PEP 249) compliant client for Amazon Athena.

[2]:
```
!pip install --disable-pip-version-check -q PyAthena==2.1.0
!pip install missingno
```

```
WARNING: The directory '/root/.cache/pip' or its parent directory is not
owned or is not writable by the current user. The cache has been disabled. Check
the permissions and owner of that directory. If executing pip with sudo, you
should use sudo's -H flag.
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
WARNING: The directory '/root/.cache/pip' or its parent directory is
not owned or is not writable by the current user. The cache has been disabled.
Check the permissions and owner of that directory. If executing pip with sudo,
you should use sudo's -H flag.
Requirement already satisfied: missingno in /opt/conda/lib/python3.7/site-
packages (0.5.2)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.7/site-
packages (from missingno) (3.1.3)
Requirement already satisfied: scipy in /opt/conda/lib/python3.7/site-packages
(from missingno) (1.4.1)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages
```

```
(from missingno) (1.21.6)
Requirement already satisfied: seaborn in /opt/conda/lib/python3.7/site-packages
(from missingno) (0.10.0)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.7/site-
packages (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (1.1.0)
Requirement already satisfied: python-dateutil>=2.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.8.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.4.6)
Requirement already satisfied: pandas>=0.22.0 in /opt/conda/lib/python3.7/site-
packages (from seaborn->missingno) (1.3.5)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages
(from cycler>=0.10->matplotlib->missingno) (1.14.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.7/site-
packages (from kiwisolver>=1.0.1->matplotlib->missingno) (59.3.0)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.22.0->seaborn->missingno) (2019.3)
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

2.2 Globally import libraries

[3]:
```
import boto3
from botocore.client import ClientError
import sagemaker
import pandas as pd
import numpy as np
from pyathena import connect
from IPython.core.display import display, HTML
import missingno as msno
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import make_pipeline, Pipeline
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.model_selection import train_test_split, cross_val_score,␣
 ↪GridSearchCV
from sklearn.feature_selection import VarianceThreshold
import datetime as dt
from io import BytesIO

%matplotlib inline
```

2.3 Instantiate AWS SageMaker and S3 sessions

[4]:
```
session = boto3.session.Session()
region = session.region_name
sagemaker_session = sagemaker.Session()
def_bucket = sagemaker_session.default_bucket()
bucket = 'sagemaker-us-east-ads508-sp23-t8'

s3 = boto3.Session().client(service_name="s3",
                            region_name=region)
```

[5]:
```
print(f"Default bucket: {def_bucket}")
print(f"Public T8 bucket: {bucket}")
```

```
Default bucket: sagemaker-us-east-1-657724983756
Public T8 bucket: sagemaker-us-east-ads508-sp23-t8
```

2.4 Pass in ABT from CSV

[6]:
```
s3_abt_csv_path = f"s3://{def_bucket}/team_8_data/abt/abt_encoded_df01.csv"
abt_encoded_df01 = pd.read_csv(s3_abt_csv_path)
```

2.4.1 Perform train/test split

[7]:
```
y01 = ['childpoverty']
abt_encoded_y01_vc01 = abt_encoded_df01[y01].to_numpy()
print(abt_encoded_y01_vc01.shape)
display(abt_encoded_y01_vc01[0:11])
abt_encoded_x01_df01 = abt_encoded_df01.drop(y01, axis=1)
print(abt_encoded_x01_df01.shape)
display(abt_encoded_x01_df01.head(11))

'''Citation for stratification by multiple cols:
https://stackoverflow.com/questions/45516424/
 ↪sklearn-train-test-split-on-pandas-stratify-by-multiple-columns'''
abt_encoded_x01_df01['boroughs'] = abt_encoded_x01_df01['borough_bronx'].
 ↪astype(int).astype(str) + abt_encoded_x01_df01['borough_brooklyn'].
 ↪astype(int).astype(str) + abt_encoded_x01_df01['borough_manhattan'].
 ↪astype(int).astype(str) + abt_encoded_x01_df01['borough_queens'].astype(int).
 ↪astype(str) + abt_encoded_x01_df01['borough_staten island'].astype(int).
 ↪astype(str)
display(abt_encoded_x01_df01.head(5))
train_x01, test_x01, train_y01, test_y01 =␣
 ↪train_test_split(abt_encoded_x01_df01,
                                                          ␣
 ↪abt_encoded_y01_vc01,
                                                           test_size=.2,
```

```
 ↪stratify=abt_encoded_x01_df01[['boroughs']],                    ␣
                                                                   shuffle=True,
                                                                   random_state=1699)

train_x01 = train_x01.drop(['boroughs', 'poverty'], axis=1)
test_x01 = test_x01.drop(['boroughs', 'poverty'], axis=1)

print(f'{train_x01.shape}')
print(f'{train_y01.shape}')
print(f'\n{test_x01.shape}')
print(f'{test_y01.shape}')
```

```
(31605, 1)

array([[20.7],
       [23.6],
       [35.9],
       [31.5],
       [67.7],
       [68.3],
       [ 0. ],
       [62.4],
       [64.9],
       [63.1],
       [ 6.5]])

(31605, 49)
```

	borough_bronx	borough_brooklyn	borough_manhattan	borough_queens	\
0	1.0	0.0	0.0	0.0	
1	1.0	0.0	0.0	0.0	
2	1.0	0.0	0.0	0.0	
3	1.0	0.0	0.0	0.0	
4	1.0	0.0	0.0	0.0	
5	1.0	0.0	0.0	0.0	
6	1.0	0.0	0.0	0.0	
7	1.0	0.0	0.0	0.0	
8	1.0	0.0	0.0	0.0	
9	1.0	0.0	0.0	0.0	
10	1.0	0.0	0.0	0.0	

	borough_staten island	relative_data_year_-4	relative_data_year_-3	\
0	0.0	1.0	0.0	
1	0.0	1.0	0.0	
2	0.0	1.0	0.0	
3	0.0	1.0	0.0	
4	0.0	1.0	0.0	
5	0.0	1.0	0.0	

```
6                    0.0                    1.0                    0.0
7                    0.0                    1.0                    0.0
8                    0.0                    1.0                    0.0
9                    0.0                    1.0                    0.0
10                   0.0                    1.0                    0.0

    relative_data_year_-2  relative_data_year_-1  relative_data_year_0  …  \
0                     0.0                    0.0                   0.0  …
1                     0.0                    0.0                   0.0  …
2                     0.0                    0.0                   0.0  …
3                     0.0                    0.0                   0.0  …
4                     0.0                    0.0                   0.0  …
5                     0.0                    0.0                   0.0  …
6                     0.0                    0.0                   0.0  …
7                     0.0                    0.0                   0.0  …
8                     0.0                    0.0                   0.0  …
9                     0.0                    0.0                   0.0  …
10                    0.0                    0.0                   0.0  …

    walk  othertransp  workathome  meancommute  employed  privatework  \
0    2.9          0.0         0.0         43.0    2308.0         80.8
1    1.4          0.5         2.1         45.0    2675.0         71.7
2    8.6          1.6         1.7         38.8    2120.0         75.0
3    3.0          2.4         6.2         45.4    1083.0         76.8
4    4.3          1.0         0.0         46.0    2508.0         71.0
5   14.0          1.5         4.1         42.7    1191.0         74.2
6    0.0          0.0         0.0          NaN     113.0         62.8
7   17.7          1.8         2.7         35.5    1691.0         85.1
8   18.0          0.0         1.6         42.8    1102.0         86.9
9    7.1          0.7         0.5         44.0    1559.0         75.0
10   2.0          0.6         2.7         47.3    2394.0         61.9

    publicwork  selfemployed  familywork  unemployment
0         16.2           2.9         0.0           7.7
1         25.3           2.5         0.6           9.5
2         21.3           3.8         0.0           8.7
3         15.5           7.7         0.0          19.2
4         21.3           7.7         0.0          17.2
5         16.1           9.7         0.0          18.9
6         37.2           0.0         0.0           0.0
7          8.3           6.1         0.5           9.4
8          8.5           4.5         0.0          15.2
9         14.0          11.0         0.0          10.6
10        37.4           0.6         0.0          12.8

[11 rows x 49 columns]

   borough_bronx  borough_brooklyn  borough_manhattan  borough_queens  \
```

```
0            1.0               0.0               0.0            0.0
1            1.0               0.0               0.0            0.0
2            1.0               0.0               0.0            0.0
3            1.0               0.0               0.0            0.0
4            1.0               0.0               0.0            0.0

   borough_staten island  relative_data_year_-4  relative_data_year_-3  \
0                    0.0                    1.0                    0.0
1                    0.0                    1.0                    0.0
2                    0.0                    1.0                    0.0
3                    0.0                    1.0                    0.0
4                    0.0                    1.0                    0.0

   relative_data_year_-2  relative_data_year_-1  relative_data_year_0  …  \
0                    0.0                    0.0                   0.0  …
1                    0.0                    0.0                   0.0  …
2                    0.0                    0.0                   0.0  …
3                    0.0                    0.0                   0.0  …
4                    0.0                    0.0                   0.0  …

   othertransp  workathome  meancommute  employed  privatework  publicwork  \
0          0.0         0.0         43.0    2308.0         80.8        16.2
1          0.5         2.1         45.0    2675.0         71.7        25.3
2          1.6         1.7         38.8    2120.0         75.0        21.3
3          2.4         6.2         45.4    1083.0         76.8        15.5
4          1.0         0.0         46.0    2508.0         71.0        21.3

   selfemployed  familywork  unemployment  boroughs
0           2.9         0.0           7.7     10000
1           2.5         0.6           9.5     10000
2           3.8         0.0           8.7     10000
3           7.7         0.0          19.2     10000
4           7.7         0.0          17.2     10000

[5 rows x 50 columns]
```

```
(25284, 48)
(25284, 1)

(6321, 48)
(6321, 1)
```

Examine featues with near zero variances

[8]:
```
# Review near-zero variance (NZV) features for possible removal
train_x01_nzv_fit = VarianceThreshold(.025).fit(train_x01)
train_x01_nzv_vc01 = train_x01_nzv_fit.transform(train_x01)

# Get the names of the selected features
```

```
train_x01_nzv_fit_select_features = train_x01.columns[train_x01_nzv_fit.
 ↪get_support()]

train_x01_nzv_df01 = pd.DataFrame(train_x01_nzv_vc01,
                                  columns=train_x01_nzv_fit_select_features)

display(train_x01_nzv_df01.head(5))
print(f'NZV transformed matrix dimensions = {train_x01_nzv_df01.shape}')

print(f'\n{train_x01.shape[1] - train_x01_nzv_df01.shape[1]} near zero variance␣
 ↪features were eliminated')

print(train_x01.columns)
print(train_x01_nzv_df01.columns)
```

```
   borough_bronx  borough_brooklyn  borough_manhattan  borough_queens  \
0            0.0               0.0                0.0             1.0
1            0.0               0.0                0.0             1.0
2            0.0               0.0                0.0             1.0
3            0.0               1.0                0.0             0.0
4            0.0               0.0                0.0             0.0

   borough_staten island  relative_data_year_-4  relative_data_year_-3  \
0                    0.0                    1.0                    0.0
1                    0.0                    1.0                    0.0
2                    0.0                    0.0                    1.0
3                    0.0                    1.0                    0.0
4                    1.0                    0.0                    0.0

   relative_data_year_-2  relative_data_year_-1  relative_data_year_0  …  \
0                    0.0                    0.0                   0.0  …
1                    0.0                    0.0                   0.0  …
2                    0.0                    0.0                   0.0  …
3                    0.0                    0.0                   0.0  …
4                    1.0                    0.0                   0.0  …

   walk  othertransp  workathome  meancommute  employed  privatework  \
0   4.6          0.6         3.9         42.9    3293.0         81.3
1   7.2          2.4         3.4         30.4    3872.0         86.0
2   3.7          0.4         3.1         43.0    2014.0         78.6
3   4.1          7.0         6.8         40.4    1782.0         82.7
4   0.6          1.2         0.8         42.0     801.0         76.3

   publicwork  selfemployed  familywork  unemployment
0        13.2           5.6         0.0           3.4
1         5.9           8.1         0.0           6.5
2        13.5           7.9         0.0           6.7
```

```
3         9.2           8.1         0.0           7.8
4        20.1           3.6         0.0           8.9

[5 rows x 48 columns]

NZV transformed matrix dimensions = (25284, 48)

0 near zero variance features were eliminated
Index(['borough_bronx', 'borough_brooklyn', 'borough_manhattan',
       'borough_queens', 'borough_staten island', 'relative_data_year_-4',
       'relative_data_year_-3', 'relative_data_year_-2',
       'relative_data_year_-1', 'relative_data_year_0',
       'complaint_type_FELONY', 'complaint_type_MISDEMEANOR',
       'complaint_type_VIOLATION', 'annual_evictions_x_borough',
       'annual_complaint_counts', 'annual_grad_n', 'annual_dropped_out_n',
       'totalpop', 'men', 'women', 'hispanic', 'white', 'black', 'native',
       'asian', 'citizen', 'income', 'incomeerr', 'incomepercap',
       'incomepercaperr', 'professional', 'service', 'office', 'construction',
       'production', 'drive', 'carpool', 'transit', 'walk', 'othertransp',
       'workathome', 'meancommute', 'employed', 'privatework', 'publicwork',
       'selfemployed', 'familywork', 'unemployment'],
      dtype='object')
Index(['borough_bronx', 'borough_brooklyn', 'borough_manhattan',
       'borough_queens', 'borough_staten island', 'relative_data_year_-4',
       'relative_data_year_-3', 'relative_data_year_-2',
       'relative_data_year_-1', 'relative_data_year_0',
       'complaint_type_FELONY', 'complaint_type_MISDEMEANOR',
       'complaint_type_VIOLATION', 'annual_evictions_x_borough',
       'annual_complaint_counts', 'annual_grad_n', 'annual_dropped_out_n',
       'totalpop', 'men', 'women', 'hispanic', 'white', 'black', 'native',
       'asian', 'citizen', 'income', 'incomeerr', 'incomepercap',
       'incomepercaperr', 'professional', 'service', 'office', 'construction',
       'production', 'drive', 'carpool', 'transit', 'walk', 'othertransp',
       'workathome', 'meancommute', 'employed', 'privatework', 'publicwork',
       'selfemployed', 'familywork', 'unemployment'],
      dtype='object')
```

2.4.2 Save ABT to S3

[9]:
```
print(f'{train_x01.head(5)}')
print(f'{train_y01[0:5]}')
print(f'\n{test_x01.head(5)}')
print(f'{test_y01[0:5]}')

s3_train_x01_csv_path = f"s3://{def_bucket}/team_8_data/modeling_data/training/
 ↪train_x01.csv"
train_x01.to_csv(s3_train_x01_csv_path,
                 index=False,
```

```
                header=True)

s3_test_x01_csv_path = f"s3://{def_bucket}/team_8_data/modeling_data/testing/
 ↪test_x01.csv"
test_x01.to_csv(s3_test_x01_csv_path,
                index=False,
                header=True)
```

```
       borough_bronx  borough_brooklyn  borough_manhattan  borough_queens  \
20821            0.0               0.0                0.0             1.0
20374            0.0               0.0                0.0             1.0
23796            0.0               0.0                0.0             1.0
6080             0.0               1.0                0.0             0.0
30916            0.0               0.0                0.0             0.0

       borough_staten island  relative_data_year_-4  relative_data_year_-3  \
20821                    0.0                    1.0                    0.0
20374                    0.0                    1.0                    0.0
23796                    0.0                    0.0                    1.0
6080                     0.0                    1.0                    0.0
30916                    1.0                    0.0                    0.0

       relative_data_year_-2  relative_data_year_-1  relative_data_year_0  \
20821                    0.0                    0.0                   0.0
20374                    0.0                    0.0                   0.0
23796                    0.0                    0.0                   0.0
6080                     0.0                    0.0                   0.0
30916                    1.0                    0.0                   0.0

       …  walk  othertransp  workathome  meancommute  employed  privatework  \
20821  …   4.6          0.6         3.9         42.9    3293.0         81.3
20374  …   7.2          2.4         3.4         30.4    3872.0         86.0
23796  …   3.7          0.4         3.1         43.0    2014.0         78.6
6080   …   4.1          7.0         6.8         40.4    1782.0         82.7
30916  …   0.6          1.2         0.8         42.0     801.0         76.3

       publicwork  selfemployed  familywork  unemployment
20821        13.2           5.6         0.0           3.4
20374         5.9           8.1         0.0           6.5
23796        13.5           7.9         0.0           6.7
6080          9.2           8.1         0.0           7.8
30916        20.1           3.6         0.0           8.9

[5 rows x 48 columns]
[[ 5.1]
 [ 1.8]
 [33.4]
 [25. ]
```

```
 [18.4]]

       borough_bronx  borough_brooklyn  borough_manhattan  borough_queens  \
3566            1.0               0.0                0.0             0.0
19683           0.0               0.0                1.0             0.0
8603            0.0               1.0                0.0             0.0
19957           0.0               0.0                1.0             0.0
9723            0.0               1.0                0.0             0.0

       borough_staten island  relative_data_year_-4  relative_data_year_-3  \
3566                     0.0                    0.0                    0.0
19683                    0.0                    0.0                    0.0
8603                     0.0                    0.0                    1.0
19957                    0.0                    0.0                    0.0
9723                     0.0                    0.0                    0.0

       relative_data_year_-2  relative_data_year_-1  relative_data_year_0  \
3566                     0.0                    1.0                   0.0
19683                    0.0                    0.0                   1.0
8603                     0.0                    0.0                   0.0
19957                    0.0                    0.0                   1.0
9723                     1.0                    0.0                   0.0

       …  walk  othertransp  workathome  meancommute  employed  privatework  \
3566   …  18.6          2.1         1.3         36.1    1567.0         75.0
19683  …  29.7          9.4         5.7         24.5    2029.0         91.2
8603   …   4.5          1.1         1.8         46.8    2261.0         77.8
19957  …  15.5          3.2         5.7         32.2    3135.0         85.9
9723   …   4.7          1.6         3.6         46.5    1830.0         73.6

       publicwork  selfemployed  familywork  unemployment
3566         18.4           6.5         0.0          15.4
19683         2.3           6.5         0.0           3.8
8603         15.7           6.5         0.0          14.0
19957         8.3           5.8         0.0           2.6
9723         23.7           2.7         0.0          12.8

[5 rows x 48 columns]
[[38.7]
 [ 0.9]
 [36.6]
 [ 0. ]
 [14.8]]
```

[10]:
```
# Define the S3 object key
train_y01_s3_key = 'team_8_data/modeling_data/training/train_y01.npy'
```

```
# Save the numpy array to S3
with BytesIO() as data:
    np.save(data, train_y01)
    data.seek(0)
    s3.upload_fileobj(data, def_bucket, train_y01_s3_key)

# Confirm that the numpy array was saved to S3
train_y01_response = s3.list_objects(Bucket=def_bucket,
                           Prefix=train_y01_s3_key)
print(train_y01_response)

# Define the S3 object key
test_y01_s3_key = 'team_8_data/modeling_data/testing/test_y01.npy'

# Save the numpy array to S3
with BytesIO() as data:
    np.save(data, test_y01)
    data.seek(0)
    s3.upload_fileobj(data, def_bucket, test_y01_s3_key)

# Confirm that the numpy array was saved to S3
test_y01_response = s3.list_objects(Bucket=def_bucket,
                           Prefix=test_y01_s3_key)
print(test_y01_response)
```

```
{'ResponseMetadata': {'RequestId': 'BX6Q8H5DSQWE7YWD', 'HostId':
'L72U9CI7UAiDGjIlqH6y+OINDcJ2jkKtOZHAiHBj7zSNrmBjMqMgOaB4u2Ew/72ZfvQycDX7Y4I=',
'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-2':
'L72U9CI7UAiDGjIlqH6y+OINDcJ2jkKtOZHAiHBj7zSNrmBjMqMgOaB4u2Ew/72ZfvQycDX7Y4I=',
'x-amz-request-id': 'BX6Q8H5DSQWE7YWD', 'date': 'Thu, 13 Apr 2023 17:09:46 GMT',
'x-amz-bucket-region': 'us-east-1', 'content-type': 'application/xml',
'transfer-encoding': 'chunked', 'server': 'AmazonS3'}, 'RetryAttempts': 0},
'IsTruncated': False, 'Marker': '', 'Contents': [{'Key':
'team_8_data/modeling_data/training/train_y01.npy', 'LastModified':
datetime.datetime(2023, 4, 13, 17, 9, 46, tzinfo=tzlocal()), 'ETag':
'"fec93318dc01a0851bfb63e52dcdfd6a"', 'Size': 202400, 'StorageClass':
'STANDARD', 'Owner': {'DisplayName': 'awslabsc0w5192702t1672660495', 'ID':
'6205bdaa014eec2453ddb24fb65480c671e128d98965891a2b0bf2ba5e5cced6'}}], 'Name':
'sagemaker-us-east-1-657724983756', 'Prefix':
'team_8_data/modeling_data/training/train_y01.npy', 'MaxKeys': 1000,
'EncodingType': 'url'}
{'ResponseMetadata': {'RequestId': 'BX6GTM5FTDVZ9TEC', 'HostId':
'NOBWLcj9aYvRJmskDXBbYHlD7AB/l8YVwYB7ZKshzLusq+TcaZhjplr4qA8uW8qICNj2V3AB7b0=',
'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-2':
'NOBWLcj9aYvRJmskDXBbYHlD7AB/l8YVwYB7ZKshzLusq+TcaZhjplr4qA8uW8qICNj2V3AB7b0=',
'x-amz-request-id': 'BX6GTM5FTDVZ9TEC', 'date': 'Thu, 13 Apr 2023 17:09:46 GMT',
'x-amz-bucket-region': 'us-east-1', 'content-type': 'application/xml',
'transfer-encoding': 'chunked', 'server': 'AmazonS3'}, 'RetryAttempts': 0},
```

```
'IsTruncated': False, 'Marker': '', 'Contents': [{'Key':
'team_8_data/modeling_data/testing/test_y01.npy', 'LastModified':
datetime.datetime(2023, 4, 13, 17, 9, 46, tzinfo=tzlocal()), 'ETag':
'"47711adfc2e66766a7a8efe248848df6"', 'Size': 50696, 'StorageClass': 'STANDARD',
'Owner': {'DisplayName': 'awslabsc0w5192702t1672660495', 'ID':
'6205bdaa014eec2453ddb24fb65480c671e128d98965891a2b0bf2ba5e5cced6'}}], 'Name':
'sagemaker-us-east-1-657724983756', 'Prefix':
'team_8_data/modeling_data/testing/test_y01.npy', 'MaxKeys': 1000,
'EncodingType': 'url'}
```

2.5 Release Resources

[11]:
```
%%html

<p><b>Shutting down your kernel for this notebook to release resources.</b></p>
<button class="sm-command-button" data-commandlinker-command="kernelmenu:
 ↪shutdown" style="display:none;">Shutdown Kernel</button>

<script>
try {
    els = document.getElementsByClassName("sm-command-button");
    els[0].click();
}
catch(err) {
    // NoOp
}
</script>
```

```
<IPython.core.display.HTML object>
```

[12]:
```
%%javascript

try {
    Jupyter.notebook.save_checkpoint();
    Jupyter.notebook.session.delete();
}
catch(err) {
    // NoOp
}
```

```
<IPython.core.display.Javascript object>
```

04a_Modeling_Final

April 14, 2023

1 ADS-508-01-SP23 Team 8: Final Project

2 Train model

Much of the code is modified from `Fregly, C., & Barth, A. (2021). Data science on AWS: Implementing end-to-end, continuous AI and machine learning pipelines. O'Reilly.`

2.1 Install missing dependencies

PyAthena is a Python DB API 2.0 (PEP 249) compliant client for Amazon Athena.

[37]:
```
!pip install --disable-pip-version-check -q PyAthena==2.1.0
!pip install --disable-pip-version-check -q sagemaker-experiments==0.1.26
!pip install missingno
!pip install xgboost
!pip install torch==1.8.1
#!pip install torchvision
```

```
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
Requirement already satisfied: missingno in /opt/conda/lib/python3.7/site-
packages (0.5.2)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.7/site-
packages (from missingno) (3.1.3)
Requirement already satisfied: scipy in /opt/conda/lib/python3.7/site-packages
(from missingno) (1.4.1)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages
(from missingno) (1.21.6)
Requirement already satisfied: seaborn in /opt/conda/lib/python3.7/site-packages
(from missingno) (0.10.0)
```

```
Requirement already satisfied: python-dateutil>=2.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (1.1.0)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.7/site-
packages (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.4.6)
Requirement already satisfied: pandas>=0.22.0 in /opt/conda/lib/python3.7/site-
packages (from seaborn->missingno) (1.3.5)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages
(from cycler>=0.10->matplotlib->missingno) (1.14.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.7/site-
packages (from kiwisolver>=1.0.1->matplotlib->missingno) (59.3.0)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.22.0->seaborn->missingno) (2019.3)
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
Requirement already satisfied: xgboost in /opt/conda/lib/python3.7/site-
packages (1.6.2)
Requirement already satisfied: scipy in /opt/conda/lib/python3.7/site-packages
(from xgboost) (1.4.1)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages
(from xgboost) (1.21.6)
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
Collecting torch==1.8.1
  Downloading torch-1.8.1-cp37-cp37m-manylinux1_x86_64.whl (804.1 MB)
                           804.1/804.1
MB 1.0 MB/s eta 0:00:0000:0100:01
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-
packages (from torch==1.8.1) (1.21.6)
Requirement already satisfied: typing-extensions in
/opt/conda/lib/python3.7/site-packages (from torch==1.8.1) (4.5.0)
Installing collected packages: torch
Successfully installed torch-1.8.1
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

2.2 Globally import libraries

[36]:
```
import boto3
from botocore.client import ClientError
import pandas as pd
import numpy as np
from pyathena import connect
from IPython.core.display import display, HTML
import missingno as msno
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import make_pipeline, Pipeline
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.model_selection import train_test_split, cross_val_score,␣
 ↪GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sagemaker.tuner import HyperparameterTuner
from sklearn.linear_model import LinearRegression
from sklearn.neural_network import MLPRegressor
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.impute import SimpleImputer
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.linear_model import Lasso
import datetime as dt
import time
import sagemaker
from smexperiments.experiment import Experiment
from smexperiments.trial import Trial
import joblib
import os
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
<ipython-input-36-5a961501dccc> in <module>
      3 import pandas as pd
      4 import numpy as np
----> 5 import torch
      6 from pyathena import connect
      7 from IPython.core.display import display, HTML

ModuleNotFoundError: No module named 'torch'
```

2.3 Instantiate AWS SageMaker and S3 sessions

[4]:
```
session = boto3.session.Session()
sess = sagemaker.Session()
role = sagemaker.get_execution_role()
region = session.region_name
sagemaker_session = sagemaker.Session()
def_bucket = sagemaker_session.default_bucket()
bucket = 't8-test-final'

s3 = boto3.Session().client(service_name="s3",
                            region_name=region)

sm = boto3.Session().client(service_name="sagemaker",
                            region_name=region)

np.random.seed(1)
```

[5]:
```
setup_s3_bucket_passed = False
ingest_create_athena_db_passed = False
ingest_create_athena_table_tsv_passed = False
```

[6]:
```
print(f"Default bucket: {def_bucket}")
print(f"Public T8 bucket: {bucket}")
```

```
Default bucket: sagemaker-us-east-1-122149314005
Public T8 bucket: t8-test-final
```

2.4 Verify S3 Bucket Creation

[7]:
```
%%bash

aws s3 ls s3://${bucket}/
```

```
2023-03-18 20:53:08 aws-athena-query-results-122149314005-us-east-1
2023-03-05 21:53:59 sagemaker-studio-122149314005-sw16ud198bb
2023-03-05 21:59:02 sagemaker-us-east-1-122149314005
2023-03-19 21:18:45 t8-mc-access
2023-03-23 20:28:25 t8-test-final
```

[8]:
```
response = None

try:
    response = s3.head_bucket(Bucket=bucket)
    print(response)
    setup_s3_bucket_passed = True
except ClientError as e:
    print(f"[ERROR] Cannot find bucket {bucket} in {response} due to {e}.")
```

```
{'ResponseMetadata': {'RequestId': 'ZERW41J0A920GJKT', 'HostId':
'vhG9zTYVva1TPUdeUaFpzfCwVnWsg5Bxs98GWURjnvuv9KcU46GC78IYlE+i1EcaoHaFT1NO0Xs=',
'HTTPStatusCode': 200, 'HTTPHeaders': {'x-amz-id-2':
'vhG9zTYVva1TPUdeUaFpzfCwVnWsg5Bxs98GWURjnvuv9KcU46GC78IYlE+i1EcaoHaFT1NO0Xs=',
'x-amz-request-id': 'ZERW41J0A920GJKT', 'date': 'Tue, 04 Apr 2023 04:05:28 GMT',
'x-amz-bucket-region': 'us-east-1', 'x-amz-access-point-alias': 'false',
'content-type': 'application/xml', 'server': 'AmazonS3'}, 'RetryAttempts': 0}}
```

[9]:
```
%store setup_s3_bucket_passed
```

```
Stored 'setup_s3_bucket_passed' (bool)
```

2.5 Pass in ABT from CSV

[10]:
```
s3_abt_csv_path = f"s3://{def_bucket}/team_8_data/abt/abt_encoded_df01.csv"
abt_encoded_df01 = pd.read_csv(s3_abt_csv_path)
```

[11]:
```
y01 = ['childpoverty']
abt_encoded_y01_vc01 = abt_encoded_df01[y01].to_numpy()
print(abt_encoded_y01_vc01.shape)
display(abt_encoded_y01_vc01[0:11])
abt_encoded_x01_df01 = abt_encoded_df01.drop(y01, axis=1)
print(abt_encoded_x01_df01.shape)
display(abt_encoded_x01_df01.head(11))
```

```
(31605, 1)
```

```
array([[20.7],
       [23.6],
       [35.9],
       [31.5],
       [67.7],
       [68.3],
       [ 0. ],
       [62.4],
       [64.9],
       [63.1],
       [ 6.5]])
```

```
(31605, 49)
```

	borough_bronx	borough_brooklyn	borough_manhattan	borough_queens	\
0	1.0	0.0	0.0	0.0	
1	1.0	0.0	0.0	0.0	
2	1.0	0.0	0.0	0.0	
3	1.0	0.0	0.0	0.0	
4	1.0	0.0	0.0	0.0	
5	1.0	0.0	0.0	0.0	
6	1.0	0.0	0.0	0.0	
7	1.0	0.0	0.0	0.0	

```
8             1.0               0.0                0.0             0.0
9             1.0               0.0                0.0             0.0
10            1.0               0.0                0.0             0.0

    borough_staten island  relative_data_year_-4  relative_data_year_-3  \
0                     0.0                    1.0                    0.0
1                     0.0                    1.0                    0.0
2                     0.0                    1.0                    0.0
3                     0.0                    1.0                    0.0
4                     0.0                    1.0                    0.0
5                     0.0                    1.0                    0.0
6                     0.0                    1.0                    0.0
7                     0.0                    1.0                    0.0
8                     0.0                    1.0                    0.0
9                     0.0                    1.0                    0.0
10                    0.0                    1.0                    0.0

    relative_data_year_-2  relative_data_year_-1  relative_data_year_0  …  \
0                     0.0                    0.0                   0.0  …
1                     0.0                    0.0                   0.0  …
2                     0.0                    0.0                   0.0  …
3                     0.0                    0.0                   0.0  …
4                     0.0                    0.0                   0.0  …
5                     0.0                    0.0                   0.0  …
6                     0.0                    0.0                   0.0  …
7                     0.0                    0.0                   0.0  …
8                     0.0                    0.0                   0.0  …
9                     0.0                    0.0                   0.0  …
10                    0.0                    0.0                   0.0  …

    walk  othertransp  workathome  meancommute  employed  privatework  \
0    2.9          0.0         0.0         43.0    2308.0         80.8
1    1.4          0.5         2.1         45.0    2675.0         71.7
2    8.6          1.6         1.7         38.8    2120.0         75.0
3    3.0          2.4         6.2         45.4    1083.0         76.8
4    4.3          1.0         0.0         46.0    2508.0         71.0
5   14.0          1.5         4.1         42.7    1191.0         74.2
6    0.0          0.0         0.0          NaN     113.0         62.8
7   17.7          1.8         2.7         35.5    1691.0         85.1
8   18.0          0.0         1.6         42.8    1102.0         86.9
9    7.1          0.7         0.5         44.0    1559.0         75.0
10   2.0          0.6         2.7         47.3    2394.0         61.9

    publicwork  selfemployed  familywork  unemployment
0         16.2           2.9         0.0           7.7
1         25.3           2.5         0.6           9.5
2         21.3           3.8         0.0           8.7
3         15.5           7.7         0.0          19.2
```

4	21.3	7.7	0.0	17.2
5	16.1	9.7	0.0	18.9
6	37.2	0.0	0.0	0.0
7	8.3	6.1	0.5	9.4
8	8.5	4.5	0.0	15.2
9	14.0	11.0	0.0	10.6
10	37.4	0.6	0.0	12.8

[11 rows x 49 columns]

[12]:
```
abt_encoded_x01_df02 = abt_encoded_x01_df01.fillna(abt_encoded_x01_df01.
 ↪median())

data_types = abt_encoded_x01_df02.dtypes
data_types = pd.DataFrame(data_types)
data_types = data_types.assign(Null_Values =
                               abt_encoded_x01_df02.isnull().sum())
data_types.reset_index(inplace = True)
data_types.rename(columns={0:'Data Type',
                           'index': 'Column/Variable',
                           'Null_Values': "# of Nulls"})
```

[12]:

	Column/Variable	Data Type	# of Nulls
0	borough_bronx	float64	0
1	borough_brooklyn	float64	0
2	borough_manhattan	float64	0
3	borough_queens	float64	0
4	borough_staten island	float64	0
5	relative_data_year_-4	float64	0
6	relative_data_year_-3	float64	0
7	relative_data_year_-2	float64	0
8	relative_data_year_-1	float64	0
9	relative_data_year_0	float64	0
10	complaint_type_FELONY	float64	0
11	complaint_type_MISDEMEANOR	float64	0
12	complaint_type_VIOLATION	float64	0
13	annual_evictions_x_borough	float64	0
14	annual_complaint_counts	float64	0
15	annual_grad_n	float64	0
16	annual_dropped_out_n	float64	0
17	totalpop	float64	0
18	men	float64	0
19	women	float64	0
20	hispanic	float64	0
21	white	float64	0
22	black	float64	0
23	native	float64	0

```
24                      asian   float64           0
25                    citizen   float64           0
26                     income   float64           0
27                  incomeerr   float64           0
28               incomepercap   float64           0
29          incomepercaperr   float64           0
30                    poverty   float64           0
31               professional   float64           0
32                    service   float64           0
33                     office   float64           0
34               construction   float64           0
35                 production   float64           0
36                      drive   float64           0
37                    carpool   float64           0
38                    transit   float64           0
39                       walk   float64           0
40                 othertransp   float64           0
41                  workathome   float64           0
42                 meancommute   float64           0
43                    employed   float64           0
44                 privatework   float64           0
45                  publicwork   float64           0
46                selfemployed   float64           0
47                  familywork   float64           0
48                unemployment   float64           0
```

[13]:
```
abt_encoded_x01_df02['boroughs'] = abt_encoded_x01_df02['borough_bronx'].
 ↪astype(int).astype(str) + abt_encoded_x01_df02['borough_brooklyn'].
 ↪astype(int).astype(str) + abt_encoded_x01_df02['borough_manhattan'].
 ↪astype(int).astype(str) + abt_encoded_x01_df02['borough_queens'].astype(int).
 ↪astype(str) + abt_encoded_x01_df02['borough_staten island'].astype(int).
 ↪astype(str)
display(abt_encoded_x01_df02.head(5))

scaler = StandardScaler()
abt_encoded_x01_df02_scaled = scaler.fit_transform(abt_encoded_x01_df02.
 ↪drop(['boroughs', 'poverty'], axis=1))

train_x01, test_x01, train_y01, test_y01 =␣
 ↪train_test_split(abt_encoded_x01_df02_scaled,
                                                                  ␣
 ↪abt_encoded_y01_vc01,
                                                                   test_size=.2,
                                                                  ␣
 ↪stratify=abt_encoded_x01_df02[['boroughs']],
                                                                   shuffle=True,
                                                                   random_state=1699)
```

```
train_x01 = pd.DataFrame(train_x01, columns=abt_encoded_x01_df02.
 ↪drop(['boroughs', 'poverty'], axis=1).columns)
test_x01 = pd.DataFrame(test_x01, columns=abt_encoded_x01_df02.
 ↪drop(['boroughs', 'poverty'], axis=1).columns)

train_y01 = train_y01.ravel()
test_y01 = test_y01.ravel()


print(f'{train_x01.shape}')
print(f'{train_y01.shape}')
print(f'\n{test_x01.shape}')
print(f'{test_y01.shape}')
```

```
   borough_bronx  borough_brooklyn  borough_manhattan  borough_queens  \
0            1.0               0.0                0.0             0.0
1            1.0               0.0                0.0             0.0
2            1.0               0.0                0.0             0.0
3            1.0               0.0                0.0             0.0
4            1.0               0.0                0.0             0.0

   borough_staten island  relative_data_year_-4  relative_data_year_-3  \
0                    0.0                    1.0                    0.0
1                    0.0                    1.0                    0.0
2                    0.0                    1.0                    0.0
3                    0.0                    1.0                    0.0
4                    0.0                    1.0                    0.0

   relative_data_year_-2  relative_data_year_-1  relative_data_year_0  …  \
0                    0.0                    0.0                   0.0  …
1                    0.0                    0.0                   0.0  …
2                    0.0                    0.0                   0.0  …
3                    0.0                    0.0                   0.0  …
4                    0.0                    0.0                   0.0  …

   othertransp  workathome  meancommute  employed  privatework  publicwork  \
0          0.0         0.0         43.0    2308.0         80.8        16.2
1          0.5         2.1         45.0    2675.0         71.7        25.3
2          1.6         1.7         38.8    2120.0         75.0        21.3
3          2.4         6.2         45.4    1083.0         76.8        15.5
4          1.0         0.0         46.0    2508.0         71.0        21.3

   selfemployed  familywork  unemployment  boroughs
0           2.9         0.0           7.7     10000
1           2.5         0.6           9.5     10000
2           3.8         0.0           8.7     10000
```

```
3           7.7         0.0          19.2     10000
4           7.7         0.0          17.2     10000

[5 rows x 50 columns]
```

```
(25284, 48)
(25284,)

(6321, 48)
(6321,)
```

Untuned Linear Regression Model

[14]:
```
# create an un-tuned linear regression model
lin_reg = LinearRegression()

# fit the model to the training set
lin_reg.fit(train_x01, train_y01)

# make predictions on the validation set
lin_reg_pred = lin_reg.predict(test_x01)
```

[15]:
```
# calculate the model performance metrics
lin_reg_mse = mean_squared_error(test_y01, lin_reg_pred)
lin_reg_rmse = lin_reg_mse ** 0.5
lin_reg_r2 = r2_score(test_y01, lin_reg_pred)

# print the performance metrics
print(f"Un-tuned Linear Regression Model: RMSE = {lin_reg_rmse:.3f}, R-squared␣
 ↪= {lin_reg_r2:.3f}")
```

```
Un-tuned Linear Regression Model: RMSE = 11.331, R-squared = 0.648
```

Random Forest

[40]:
```
# Define the hyperparameters to tune
param_dist = {
    'n_estimators': randint(50, 200),
    'max_depth': randint(3, 10),
    'min_samples_split': uniform(0.01, 0.19),
    'min_samples_leaf': uniform(0.01, 0.19),
    'max_features': ['auto', 'sqrt']
}

# Create the Random Forest Regressor model
rf = RandomForestRegressor(random_state=42)

# Perform a randomized search to find the best hyperparameters
random_search = RandomizedSearchCV(estimator=rf,␣
 ↪param_distributions=param_dist, cv=5, n_iter=10, n_jobs=-1)
```

```
random_search.fit(train_x01, train_y01)

# Print the best hyperparameters found
print("Best hyperparameters: ", random_search.best_params_)
```

```
Best hyperparameters:  {'max_depth': 5, 'max_features': 'auto',
'min_samples_leaf': 0.05482542923388117, 'min_samples_split':
0.12414899564785374, 'n_estimators': 153}
```

[41]:
```
# Make predictions on the train and test data
rf_train_preds = random_search.predict(train_x01)
rf_test_preds = random_search.predict(test_x01)
```

[45]:
```
# Evaluate the performance of the model using mean squared error
train_rf_mse = mean_squared_error(train_y01, rf_train_preds)
train_rf_rmse = train_rf_mse ** 0.5
train_rf_r2 = r2_score(train_y01, rf_train_preds)
print(f"XGBoost train model performance: RMSE = {train_rf_rmse:.3f}, R-squared␣
 ↪= {train_rf_r2:.3f}")

# Evaluate the performance of the model using mean squared error
test_rf_mse = mean_squared_error(test_y01, rf_test_preds)
test_rf_rmse = test_rf_mse ** 0.5
test_rf_r2 = r2_score(test_y01, rf_test_preds)
print(f"XGBoost test model performance: RMSE = {test_rf_rmse:.3f}, R-squared =␣
 ↪{test_rf_r2:.3f}")
```

```
XGBoost train model performance: RMSE = 11.276, R-squared = 0.644
XGBoost test model performance: RMSE = 11.301, R-squared = 0.650
```

XGBoost

[16]:
```
# Set the hyperparameters for the Gradient Boosting Regressor model
params = {
    'max_depth': 5,
    'learning_rate': 0.1,
    'loss': 'ls'
}

# Train the Gradient Boosting Regressor model
model = GradientBoostingRegressor(**params)
model.fit(train_x01, train_y01)

# Make predictions on the train data
xgb_y__train_pred = model.predict(train_x01)

# Make predictions on the test data
xgb_y_pred = model.predict(test_x01)
```

[18]:
```
# Evaluate the performance of the model using mean squared error
train_xgb_mse = mean_squared_error(train_y01, xgb_y__train_pred)
train_xgb_rmse = train_xgb_mse ** 0.5
train_xgb_r2 = r2_score(train_y01, xgb_y__train_pred)
print(f"XGBoost train model performance: RMSE = {train_xgb_rmse:.3f}, R-squared␣
 ↪= {train_xgb_r2:.3f}")

# Evaluate the performance of the model using mean squared error
test_xgb_mse = mean_squared_error(test_y01, xgb_y_pred)
test_xgb_rmse = test_xgb_mse ** 0.5
test_xgb_r2 = r2_score(test_y01, xgb_y_pred)
print(f"XGBoost test model performance: RMSE = {test_xgb_rmse:.3f}, R-squared =␣
 ↪{test_xgb_r2:.3f}")
```

```
XGBoost train model performance: RMSE = 5.204, R-squared = 0.924
XGBoost test model performance: RMSE = 5.327, R-squared = 0.922
```

Tuned XGBoost

[19]:
```
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import randint, uniform

# Define the hyperparameters to tune
param_dist = {
    'max_depth': randint(3, 10),
    'learning_rate': uniform(0.05, 0.5),
    'loss': ['ls', 'lad', 'huber']
}

# Create the Gradient Boosting Regressor model
gbm = GradientBoostingRegressor()

# Perform a randomized search to find the best hyperparameters
random_search = RandomizedSearchCV(estimator=gbm,␣
 ↪param_distributions=param_dist, cv=3, n_iter=10, n_jobs=-1)
random_search.fit(train_x01, train_y01)

# Print the best hyperparameters found
print("Best hyperparameters: ", random_search.best_params_)
```

```
Best hyperparameters:  {'learning_rate': 0.4911814805544623, 'loss': 'ls',
'max_depth': 8}
```

[20]:
```
# Train the model with the best hyperparameters found
xgb_tune_model = GradientBoostingRegressor(**random_search.best_params_)
xgb_tune_model.fit(train_x01, train_y01)
```

[20]:
```
GradientBoostingRegressor(alpha=0.9, ccp_alpha=0.0, criterion='friedman_mse',
                          init=None, learning_rate=0.4911814805544623,
```

```
                                   loss='ls', max_depth=8, max_features=None,
                                   max_leaf_nodes=None, min_impurity_decrease=0.0,
                                   min_impurity_split=None, min_samples_leaf=1,
                                   min_samples_split=2, min_weight_fraction_leaf=0.0,
                                   n_estimators=100, n_iter_no_change=None,
                                   presort='deprecated', random_state=None,
                                   subsample=1.0, tol=0.0001, validation_fraction=0.1,
                                   verbose=0, warm_start=False)
```

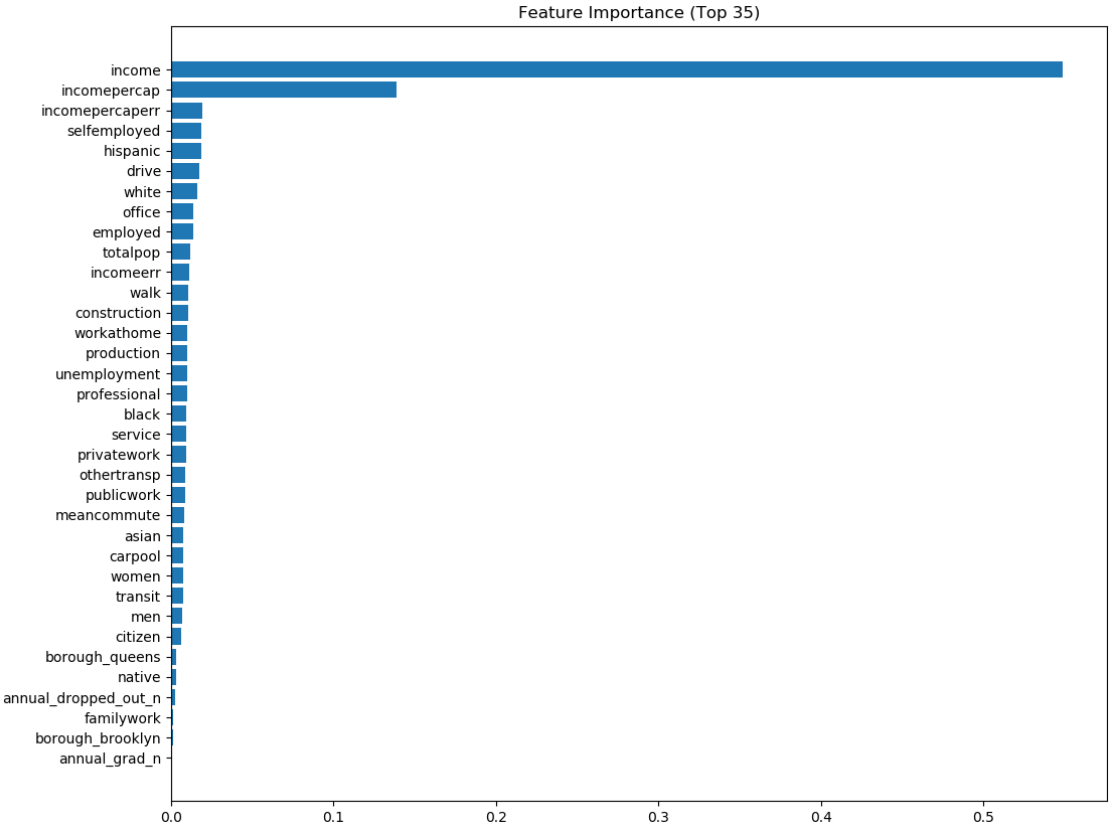
[21]:
```
# Make predictions on the train data
xgb_tune_ytrain_pred = xgb_tune_model.predict(train_x01)

# Evaluate the performance of the model using mean squared error
xgb_tune_ytrain_mse = mean_squared_error(train_y01, xgb_tune_ytrain_pred)
xgb_tune_ytrain_rmse = xgb_tune_ytrain_mse ** 0.5
xgb_tune_ytrain_r2 = r2_score(train_y01, xgb_tune_ytrain_pred)
print(f"XGBoost model train performance: RMSE = {xgb_tune_ytrain_rmse:.3f},␣
 ↪R-squared = {xgb_tune_ytrain_r2:.3f}")

# Make predictions on the test data
xgb_tune_ytest_pred = xgb_tune_model.predict(test_x01)

# Evaluate the performance of the model using mean squared error
xgb_tune_ytest_mse = mean_squared_error(test_y01, xgb_tune_ytest_pred)
xgb_tune_ytest_rmse = xgb_tune_ytest_mse ** 0.5
xgb_tune_ytest_r2 = r2_score(test_y01, xgb_tune_ytest_pred)
print(f"XGBoost model test performance: RMSE = {xgb_tune_ytest_rmse:.3f},␣
 ↪R-squared = {xgb_tune_ytest_r2:.3f}")
```

```
XGBoost model train performance: RMSE = 0.002, R-squared = 1.000
XGBoost model test performance: RMSE = 0.002, R-squared = 1.000
```

[22]:
```
# r2 of 1 is a bit... suspicious.
```

[23]:
```
'''Citations:
https://www.rasgoml.com/feature-engineering-tutorials/
 ↪how-to-generate-feature-importance-plots-from-scikit-learn;
OpenAI. (2021). ChatGPT [Computer software]. https://openai.com/'''
import matplotlib.pyplot as plt

feature_importance = xgb_tune_model.feature_importances_
feature_idx = np.argsort(feature_importance)
fig = plt.figure(figsize=(12, 10))
plt.barh(range(-1, -36, -1),
         feature_importance[feature_idx][-35:][::-1],
         align='center')
```

```
plt.yticks(range(-1, -36, -1), np.array(train_x01.columns)[feature_idx][-35:][::
 ↪-1])
plt.title('Feature Importance (Top 35)')
plt.show()
```



Weighted Ensemble of XGBoost - Reccomended by AutoML App

[24]:
```
# Train multiple Gradient Boosting Regressor models with different␣
 ↪hyperparameters
models = []
models.append(('model1', GradientBoostingRegressor(max_depth=2, learning_rate=0.
 ↪2, loss='ls')))
models.append(('model2', GradientBoostingRegressor(max_depth=3, learning_rate=0.
 ↪1, loss='ls')))
models.append(('model3', GradientBoostingRegressor(max_depth=4, learning_rate=0.
 ↪05, loss='ls')))
models.append(('model4', GradientBoostingRegressor(max_depth=5, learning_rate=0.
 ↪01, loss='ls')))
models.append(('model5', GradientBoostingRegressor(max_depth=6, learning_rate=0.
 ↪01, loss='ls')))
```

```
# Train the models and make predictions on the train data
predictions = []
weights = [0.2, 0.2, 0.2, 0.2, 0.2]
for name, model in models:
    model.fit(train_x01, train_y01)
    wr_y_pred = model.predict(train_x01)
    predictions.append(wr_y_pred)

# Combine the predictions with the weights to get the final prediction
wr_final_train_pred = np.average(predictions, axis=0, weights=weights)
```

[25]:
```
# Train the models and make predictions on the test data
predictions = []
weights = [0.2, 0.2, 0.2, 0.2, 0.2]
for name, model in models:
    model.fit(train_x01, train_y01)
    wr_y_pred = model.predict(test_x01)
    predictions.append(wr_y_pred)

# Combine the predictions with the weights to get the final prediction
wr_final_test_pred = np.average(predictions, axis=0, weights=weights)
```

[26]:
```
# Evaluate the performance of the model using root mean squared error
wr_train_mse = mean_squared_error(train_y01, wr_final_train_pred)
wr_train_rmse = wr_train_mse ** 0.5
wr_train_r2 = r2_score(train_y01, wr_final_train_pred)
print(f"Weighted Ensemble model train performance: RMSE = {wr_train_rmse:.3f},␣
 ↪R-squared = {wr_train_r2:.3f}")

# Evaluate the performance of the model using root mean squared error
wr_test_mse = mean_squared_error(test_y01, wr_final_test_pred)
wr_test_rmse = wr_test_mse ** 0.5
wr_test_r2 = r2_score(test_y01, wr_final_test_pred)
print(f"Weighted Ensemble model test performance: RMSE = {wr_test_rmse:.3f},␣
 ↪R-squared = {wr_test_r2:.3f}")
```

```
Weighted Ensemble model train performance: RMSE = 8.954, R-squared = 0.776
Weighted Ensemble model test performance: RMSE = 9.050, R-squared = 0.776
```

PyTorch Neural Network

[38]:
```
import torch
import torch.nn as nn
import torch.optim as optim

# Convert Pandas dataframes to numpy arrays
np_train_x01 = train_x01.to_numpy()
np_train_y01 = train_y01
```

```
np_test_x01 = test_x01.to_numpy()
np_test_y01 = test_y01

# Define PyTorch neural network model
class Net(nn.Module):
    def __init__(self):
        super(Net, self).__init__()
        self.fc1 = nn.Linear(np_train_x01.shape[1], 10)
        self.fc2 = nn.Linear(10, 7)
        self.fc3 = nn.Linear(7, 5)
        self.fc4 = nn.Linear(5, 1)

    def forward(self, x):
        x = nn.functional.relu(self.fc1(x))
        x = nn.functional.relu(self.fc2(x))
        x = nn.functional.relu(self.fc3(x))
        x = self.fc4(x)
        return x

# Instantiate model, loss function, and optimizer
model = Net()
criterion = nn.MSELoss()
optimizer = optim.Adam(model.parameters(), lr=0.01)

# Train model for data
for epoch in range(300):
    inputs = torch.Tensor(np_train_x01)
    targets = torch.Tensor(np_train_y01).view(-1, 1)
    optimizer.zero_grad()
    outputs = model(inputs)
    loss = criterion(outputs, targets)
    loss.backward()
    optimizer.step()
```

[39]:
```
# Evaluate model performance
inputs = torch.Tensor(np_test_x01)
targets = torch.Tensor(np_test_y01).view(-1, 1)
outputs = model(inputs)
Py_y_pred = outputs.detach().numpy().reshape(-1,)
Py_rmse = np.sqrt(mean_squared_error(np_test_y01, Py_y_pred))
Py_r2 = r2_score(np_test_y01, Py_y_pred)

print(f"PyTorch neural network model performance: RMSE = {Py_rmse:.3f},␣
 ↪R-squared = {Py_r2:.3f}")
```

```
PyTorch neural network model performance: RMSE = 10.353, R-squared = 0.707
```

2.6 Release Resources

[ ]:
```
%%html

<p><b>Shutting down your kernel for this notebook to release resources.</b></p>
<button class="sm-command-button" data-commandlinker-command="kernelmenu:
 ↪shutdown" style="display:none;">Shutdown Kernel</button>

<script>
try {
    els = document.getElementsByClassName("sm-command-button");
    els[0].click();
}
catch(err) {
    // NoOp
}
</script>
```

[ ]:
```
%%javascript

try {
    Jupyter.notebook.save_checkpoint();
    Jupyter.notebook.session.delete();
}
catch(err) {
    // NoOp
}
```

04b_Modeling_Final

April 14, 2023

1 ADS-508-01-SP23 Team 8: Final Project

2 Train model

Much of the code is modified from `Fregly, C., & Barth, A. (2021). Data science on AWS: Implementing end-to-end, continuous AI and machine learning pipelines. O'Reilly.`

2.1 Install missing dependencies

PyAthena is a Python DB API 2.0 (PEP 249) compliant client for Amazon Athena.

[2]:
```
!pip install --disable-pip-version-check -q PyAthena==2.1.0
!pip install --disable-pip-version-check -q sagemaker-experiments==0.1.26
!pip install missingno
!pip install scikit-optimize
```

```
WARNING: The directory '/root/.cache/pip' or its parent directory is not
owned or is not writable by the current user. The cache has been disabled. Check
the permissions and owner of that directory. If executing pip with sudo, you
should use sudo's -H flag.
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
WARNING: The directory '/root/.cache/pip' or its parent directory is
not owned or is not writable by the current user. The cache has been disabled.
Check the permissions and owner of that directory. If executing pip with sudo,
you should use sudo's -H flag.
WARNING: Running pip as the 'root' user can result in broken
permissions and conflicting behaviour with the system package manager. It is
recommended to use a virtual environment instead:
https://pip.pypa.io/warnings/venv
WARNING: The directory '/root/.cache/pip' or its parent directory is
not owned or is not writable by the current user. The cache has been disabled.
Check the permissions and owner of that directory. If executing pip with sudo,
you should use sudo's -H flag.
Collecting missingno
  Downloading missingno-0.5.2-py3-none-any.whl (8.7 kB)
Requirement already satisfied: numpy in /opt/conda/lib/python3.7/site-packages
(from missingno) (1.21.6)
Requirement already satisfied: seaborn in /opt/conda/lib/python3.7/site-packages
(from missingno) (0.10.0)
Requirement already satisfied: matplotlib in /opt/conda/lib/python3.7/site-
packages (from missingno) (3.1.3)
Requirement already satisfied: scipy in /opt/conda/lib/python3.7/site-packages
(from missingno) (1.4.1)
Requirement already satisfied: kiwisolver>=1.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (1.1.0)
Requirement already satisfied: python-dateutil>=2.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.8.2)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in
/opt/conda/lib/python3.7/site-packages (from matplotlib->missingno) (2.4.6)
Requirement already satisfied: cycler>=0.10 in /opt/conda/lib/python3.7/site-
```

```
packages (from matplotlib->missingno) (0.10.0)
Requirement already satisfied: pandas>=0.22.0 in /opt/conda/lib/python3.7/site-
packages (from seaborn->missingno) (1.3.5)
Requirement already satisfied: six in /opt/conda/lib/python3.7/site-packages
(from cycler>=0.10->matplotlib->missingno) (1.14.0)
Requirement already satisfied: setuptools in /opt/conda/lib/python3.7/site-
packages (from kiwisolver>=1.0.1->matplotlib->missingno) (59.3.0)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/lib/python3.7/site-
packages (from pandas>=0.22.0->seaborn->missingno) (2019.3)
Installing collected packages: missingno
Successfully installed missingno-0.5.2
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
WARNING: The directory '/root/.cache/pip' or its parent directory is
not owned or is not writable by the current user. The cache has been disabled.
Check the permissions and owner of that directory. If executing pip with sudo,
you should use sudo's -H flag.

Collecting scikit-optimize
  Downloading scikit_optimize-0.9.0-py2.py3-none-any.whl (100 kB)
                         100.3/100.3 kB
222.4 MB/s eta 0:00:00
Requirement already satisfied: scikit-learn>=0.20.0 in
/opt/conda/lib/python3.7/site-packages (from scikit-optimize) (0.22.1)
Requirement already satisfied: numpy>=1.13.3 in /opt/conda/lib/python3.7/site-
packages (from scikit-optimize) (1.21.6)
Requirement already satisfied: joblib>=0.11 in /opt/conda/lib/python3.7/site-
packages (from scikit-optimize) (1.2.0)
Requirement already satisfied: scipy>=0.19.1 in /opt/conda/lib/python3.7/site-
packages (from scikit-optimize) (1.4.1)
Collecting pyaml>=16.9
  Downloading pyaml-21.10.1-py2.py3-none-any.whl (24 kB)
Requirement already satisfied: PyYAML in /opt/conda/lib/python3.7/site-packages
(from pyaml>=16.9->scikit-optimize) (6.0)
Installing collected packages: pyaml, scikit-optimize
Successfully installed pyaml-21.10.1 scikit-optimize-0.9.0
WARNING: Running pip as the 'root' user can result in broken permissions
and conflicting behaviour with the system package manager. It is recommended to
use a virtual environment instead: https://pip.pypa.io/warnings/venv
```

2.2 Globally import libraries

[3]:
```
import boto3
from botocore.client import ClientError
import pandas as pd
import numpy as np
from pyathena import connect
from IPython.core.display import display, HTML
import missingno as msno
from skopt import BayesSearchCV
from skopt.space import Real, Categorical, Integer
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import make_pipeline, Pipeline
from sklearn.preprocessing import StandardScaler, OneHotEncoder
from sklearn.model_selection import train_test_split, cross_val_score,␣
 ↪GridSearchCV
from sklearn.ensemble import RandomForestRegressor
from sklearn.neural_network import MLPRegressor
from sklearn.impute import SimpleImputer
from sklearn.metrics import r2_score, mean_squared_error
from sklearn.linear_model import Lasso
import datetime as dt
import time
import sagemaker
from smexperiments.experiment import Experiment
from smexperiments.trial import Trial
import joblib
import os
from io import BytesIO

%matplotlib inline
```

2.3 Instantiate AWS SageMaker and S3 sessions

[4]:
```
session = boto3.session.Session()
role = sagemaker.get_execution_role()
region = session.region_name
sagemaker_session = sagemaker.Session()
def_bucket = sagemaker_session.default_bucket()
bucket = 'sagemaker-us-east-ads508-sp23-t8'

s3 = boto3.Session().client(service_name="s3",
                            region_name=region)

sm = boto3.Session().client(service_name="sagemaker",
                            region_name=region)
```

[5]:
```
print(f"Default bucket: {def_bucket}")
print(f"Public T8 bucket: {bucket}")
```

```
Default bucket: sagemaker-us-east-1-657724983756
Public T8 bucket: sagemaker-us-east-ads508-sp23-t8
```

2.4 Pass in train and test X from CSV

[6]:
```
s3_train_x01_csv_path = f"s3://{def_bucket}/team_8_data/modeling_data/training/
 ↪train_x01.csv"
train_x01 = pd.read_csv(s3_train_x01_csv_path)
s3_test_x01_csv_path = f"s3://{def_bucket}/team_8_data/modeling_data/testing/
 ↪test_x01.csv"
test_x01 = pd.read_csv(s3_test_x01_csv_path)

print(f'{train_x01.shape}')
print(f'\n{test_x01.shape}')
```

```
(25284, 48)

(6321, 48)
```

2.5 Pass in train and test y from np array

[7]:
```
# Define the S3 object key
train_y01_s3_key = 'team_8_data/modeling_data/training/train_y01.npy'

# Load the numpy array from S3
with BytesIO() as data:
    s3.download_fileobj(def_bucket, train_y01_s3_key, data)
    data.seek(0)
    train_y01 = np.load(data)

# Define the S3 object key
test_y01_s3_key = 'team_8_data/modeling_data/testing/test_y01.npy'

# Load the numpy array from S3
with BytesIO() as data:
    s3.download_fileobj(def_bucket, test_y01_s3_key, data)
    data.seek(0)
    test_y01 = np.load(data)

train_y01 = train_y01.ravel()
test_y01 = test_y01.ravel()

# Confirm that the numpy array was loaded from S3
print(f'{train_y01.shape}')
print(f'{test_y01.shape}')
```

```
(25284,)
(6321,)
```

2.6 Model Training using Grid search with 5-fold cross-validation

2.6.1 Neural Network

[8]:
```
# Start timer script
start_time = dt.datetime.today()

# Citation: Hochberg, 2018; Shanmukh, 2021
m1v1_nn_pip = Pipeline([('si', SimpleImputer(strategy='median')),
                        ('ss', StandardScaler()),
                        ('nn', MLPRegressor(random_state=1699))])

nodes_h = 3
predictors_p = 49

hidden_layer_sizes_hparam = [[100,],
                             [(nodes_h*(predictors_p+1))+nodes_h+1,],
                             [50, 50]
                            ]
activation_hparam = ['logistic', 'relu']
solver_hparam = ['adam']
alpha_hparam = [.0001, .0005, .001]
learn_rate_hparam = ['constant', 'invscaling']

#hidden_layer_sizes_hparam = [[100,]]
#activation_hparam = ['relu']
#solver_hparam = ['adam']
#alpha_hparam = [.0001]
#learn_rate_hparam = ['invscaling']

m1v1_nn_grd = {'nn__hidden_layer_sizes': hidden_layer_sizes_hparam,
               'nn__activation': activation_hparam,
               'nn__solver': solver_hparam,
               'nn__alpha': alpha_hparam,
               'nn__learning_rate': learn_rate_hparam
              }

m1v1_nn = GridSearchCV(m1v1_nn_pip,
                       m1v1_nn_grd,
                       scoring='neg_root_mean_squared_error',
                       n_jobs=2,
                       refit=True,
                       verbose=2)

m1v1_nn.fit(train_x01, train_y01)
```

```
print(f'Best Estimator:\n{m1v1_nn.best_estimator_}')

print(pd.DataFrame(m1v1_nn.cv_results_))

train_m1v1_nn_y01_pred = m1v1_nn.predict(train_x01)
print(train_m1v1_nn_y01_pred)

test_m1v1_nn_y01_pred = m1v1_nn.predict(test_x01)
print(test_m1v1_nn_y01_pred)

# Display evaluation metrics
# R-sq
train_m1v1_nn_r2 = r2_score(train_y01, train_m1v1_nn_y01_pred)
test_m1v1_nn_r2 = r2_score(test_y01, test_m1v1_nn_y01_pred)

print(f'Train R-sq:\n{train_m1v1_nn_r2}')
print(f'Test R-sq:\n{test_m1v1_nn_r2}')

# RMSE
train_m1v1_nn_rmse = mean_squared_error(train_y01, train_m1v1_nn_y01_pred,␣
 ↪squared=False)
test_m1v1_nn_rmse = mean_squared_error(test_y01, test_m1v1_nn_y01_pred,␣
 ↪squared=False)

print(f'Train RMSE:\n{train_m1v1_nn_rmse}')
print(f'Test RMSE:\n{test_m1v1_nn_rmse}')

# End timer script
end_time = dt.datetime.today()
time_elapse = end_time - start_time
print(f'End Time = {end_time}')
print(f'Script Time = {time_elapse}')
```

```
Fitting 5 folds for each of 36 candidates, totalling 180 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done  37 tasks      | elapsed: 12.0min
[Parallel(n_jobs=2)]: Done 158 tasks      | elapsed: 45.9min
[Parallel(n_jobs=2)]: Done 180 out of 180 | elapsed: 51.7min finished

Best Estimator:
Pipeline(memory=None,
         steps=[('si',
                 SimpleImputer(add_indicator=False, copy=True, fill_value=None,
                               missing_values=nan, strategy='median',
                               verbose=0)),
                ('ss',
```

```
                 StandardScaler(copy=True, with_mean=True, with_std=True)),
                ('nn',
                 MLPRegressor(activation='relu', alpha=0.0005,
                              batch_size='auto', beta_1=0.9, beta_2=0.999,
                              early_stopping=False, epsilon=1e-08,
                              hidden_layer_sizes=[50, 50],
                              learning_rate='constant',
                              learning_rate_init=0.001, max_fun=15000,
                              max_iter=200, momentum=0.9, n_iter_no_change=10,
                              nesterovs_momentum=True, power_t=0.5,
                              random_state=1699, shuffle=True, solver='adam',
                              tol=0.0001, validation_fraction=0.1,
                              verbose=False, warm_start=False))],
         verbose=False)
    mean_fit_time  std_fit_time  mean_score_time  std_score_time  \
0       33.143548      0.529329         0.032452        0.003388
1       34.235025      0.891932         0.034432        0.003698
2       45.957392      0.721042         0.042339        0.000732
3       46.024776      0.781211         0.050556        0.017036
4       38.072548      0.445525         0.033994        0.003400
5       36.995700      0.322889         0.032575        0.000501
6       32.707927      0.229827         0.032212        0.000665
7       32.493836      0.077391         0.032501        0.000327
8       45.085183      0.421935         0.044698        0.003383
9       45.192096      0.353731         0.045635        0.003949
10      37.167754      0.503359         0.034419        0.004507
11      37.187461      0.628572         0.033794        0.003179
12      32.951811      0.651863         0.032234        0.000757
13      32.465124      0.270572         0.032322        0.000358
14      47.043259      2.372003         0.042367        0.000092
15      45.504390      0.374914         0.042963        0.001024
16      37.119073      0.681981         0.033486        0.001257
17      37.297852      0.599174         0.032584        0.000332
18      25.550653      0.229620         0.020148        0.003253
19      25.787540      0.218238         0.017209        0.000411
20      33.771521      0.603930         0.024322        0.009828
21      33.597681      0.430749         0.019367        0.000481
22      31.016342      0.396437         0.017053        0.001032
23      31.029871      0.554800         0.018221        0.001761
24      26.011305      0.443071         0.017106        0.000348
25      25.785980      0.471428         0.016555        0.001151
26      33.899021      0.349528         0.020482        0.001773
27      33.563256      0.528809         0.019615        0.000963
28      31.141742      0.566396         0.020337        0.006852
29      30.997358      0.365988         0.017249        0.000827
30      25.684840      0.256868         0.017028        0.000479
31      25.798281      0.302832         0.017244        0.000354
32      33.996955      0.638432         0.019921        0.000500
```

```
33      33.543362      0.277484         0.019681        0.000308
34      30.799481      0.347098         0.017663        0.000995
35      30.831124      0.491073         0.015790        0.002267

   param_nn__activation param_nn__alpha param_nn__hidden_layer_sizes  \
0              logistic          0.0001                        [100]
1              logistic          0.0001                        [100]
2              logistic          0.0001                        [154]
3              logistic          0.0001                        [154]
4              logistic          0.0001                      [50, 50]
5              logistic          0.0001                      [50, 50]
6              logistic          0.0005                        [100]
7              logistic          0.0005                        [100]
8              logistic          0.0005                        [154]
9              logistic          0.0005                        [154]
10             logistic          0.0005                      [50, 50]
11             logistic          0.0005                      [50, 50]
12             logistic           0.001                        [100]
13             logistic           0.001                        [100]
14             logistic           0.001                        [154]
15             logistic           0.001                        [154]
16             logistic           0.001                      [50, 50]
17             logistic           0.001                      [50, 50]
18                 relu          0.0001                        [100]
19                 relu          0.0001                        [100]
20                 relu          0.0001                        [154]
21                 relu          0.0001                        [154]
22                 relu          0.0001                      [50, 50]
23                 relu          0.0001                      [50, 50]
24                 relu          0.0005                        [100]
25                 relu          0.0005                        [100]
26                 relu          0.0005                        [154]
27                 relu          0.0005                        [154]
28                 relu          0.0005                      [50, 50]
29                 relu          0.0005                      [50, 50]
30                 relu           0.001                        [100]
31                 relu           0.001                        [100]
32                 relu           0.001                        [154]
33                 relu           0.001                        [154]
34                 relu           0.001                      [50, 50]
35                 relu           0.001                      [50, 50]

   param_nn__learning_rate param_nn__solver  \
0                 constant             adam
1              invscaling             adam
2                 constant             adam
3              invscaling             adam
4                 constant             adam
```

```
5               invscaling            adam
6                 constant            adam
7               invscaling            adam
8                 constant            adam
9               invscaling            adam
10                constant            adam
11              invscaling            adam
12                constant            adam
13              invscaling            adam
14                constant            adam
15              invscaling            adam
16                constant            adam
17              invscaling            adam
18                constant            adam
19              invscaling            adam
20                constant            adam
21              invscaling            adam
22                constant            adam
23              invscaling            adam
24                constant            adam
25              invscaling            adam
26                constant            adam
27              invscaling            adam
28                constant            adam
29              invscaling            adam
30                constant            adam
31              invscaling            adam
32                constant            adam
33              invscaling            adam
34                constant            adam
35              invscaling            adam

                                                 params  split0_test_score  \
0   {'nn__activation': 'logistic', 'nn__alpha': 0…          -7.886156
1   {'nn__activation': 'logistic', 'nn__alpha': 0…          -7.886156
2   {'nn__activation': 'logistic', 'nn__alpha': 0…          -6.666789
3   {'nn__activation': 'logistic', 'nn__alpha': 0…          -6.666789
4   {'nn__activation': 'logistic', 'nn__alpha': 0…          -4.942271
5   {'nn__activation': 'logistic', 'nn__alpha': 0…          -4.942271
6   {'nn__activation': 'logistic', 'nn__alpha': 0…          -7.885017
7   {'nn__activation': 'logistic', 'nn__alpha': 0…          -7.885017
8   {'nn__activation': 'logistic', 'nn__alpha': 0…          -6.660854
9   {'nn__activation': 'logistic', 'nn__alpha': 0…          -6.660854
10  {'nn__activation': 'logistic', 'nn__alpha': 0…          -4.943635
11  {'nn__activation': 'logistic', 'nn__alpha': 0…          -4.943635
12  {'nn__activation': 'logistic', 'nn__alpha': 0…          -7.886704
13  {'nn__activation': 'logistic', 'nn__alpha': 0…          -7.886704
14  {'nn__activation': 'logistic', 'nn__alpha': 0…          -6.656225
```

15	{'nn__activation': 'logistic', 'nn__alpha': 0…	-6.656225
16	{'nn__activation': 'logistic', 'nn__alpha': 0…	-4.945009
17	{'nn__activation': 'logistic', 'nn__alpha': 0…	-4.945009
18	{'nn__activation': 'relu', 'nn__alpha': 0.0001…	-4.755215
19	{'nn__activation': 'relu', 'nn__alpha': 0.0001…	-4.755215
20	{'nn__activation': 'relu', 'nn__alpha': 0.0001…	-3.408820
21	{'nn__activation': 'relu', 'nn__alpha': 0.0001…	-3.408820
22	{'nn__activation': 'relu', 'nn__alpha': 0.0001…	-2.138443
23	{'nn__activation': 'relu', 'nn__alpha': 0.0001…	-2.138443
24	{'nn__activation': 'relu', 'nn__alpha': 0.0005…	-4.793217
25	{'nn__activation': 'relu', 'nn__alpha': 0.0005…	-4.793217
26	{'nn__activation': 'relu', 'nn__alpha': 0.0005…	-3.415002
27	{'nn__activation': 'relu', 'nn__alpha': 0.0005…	-3.415002
28	{'nn__activation': 'relu', 'nn__alpha': 0.0005…	-2.072638
29	{'nn__activation': 'relu', 'nn__alpha': 0.0005…	-2.072638
30	{'nn__activation': 'relu', 'nn__alpha': 0.001,…	-4.732671
31	{'nn__activation': 'relu', 'nn__alpha': 0.001,…	-4.732671
32	{'nn__activation': 'relu', 'nn__alpha': 0.001,…	-3.442814
33	{'nn__activation': 'relu', 'nn__alpha': 0.001,…	-3.442814
34	{'nn__activation': 'relu', 'nn__alpha': 0.001,…	-2.158172
35	{'nn__activation': 'relu', 'nn__alpha': 0.001,…	-2.158172

	split1_test_score	split2_test_score	split3_test_score	\
0	-8.056679	-7.571552	-7.917553	
1	-8.056679	-7.571552	-7.917553	
2	-6.611240	-6.292724	-6.284235	
3	-6.611240	-6.292724	-6.284235	
4	-4.733368	-4.564217	-4.642696	
5	-4.733368	-4.564217	-4.642696	
6	-8.054838	-7.574916	-7.917823	
7	-8.054838	-7.574916	-7.917823	
8	-6.610766	-6.290245	-6.284635	
9	-6.610766	-6.290245	-6.284635	
10	-4.732347	-4.564699	-4.641053	
11	-4.732347	-4.564699	-4.641053	
12	-8.054588	-7.577553	-7.916608	
13	-8.054588	-7.577553	-7.916608	
14	-6.610904	-6.288636	-6.287919	
15	-6.610904	-6.288636	-6.287919	
16	-4.731240	-4.566586	-4.652802	
17	-4.731240	-4.566586	-4.652802	
18	-4.727183	-4.493818	-4.706454	
19	-4.727183	-4.493818	-4.706454	
20	-3.261693	-3.143096	-3.177231	
21	-3.261693	-3.143096	-3.177231	
22	-2.074308	-1.988741	-2.209510	
23	-2.074308	-1.988741	-2.209510	
24	-4.620354	-4.531985	-4.668786	

25	-4.620354	-4.531985	-4.668786
26	-3.165363	-3.111582	-3.153980
27	-3.165363	-3.111582	-3.153980
28	-1.978748	-1.917381	-2.099757
29	-1.978748	-1.917381	-2.099757
30	-4.652343	-4.511500	-4.645985
31	-4.652343	-4.511500	-4.645985
32	-3.205790	-3.099954	-3.160798
33	-3.205790	-3.099954	-3.160798
34	-1.995288	-1.845477	-2.203662
35	-1.995288	-1.845477	-2.203662

	split4_test_score	mean_test_score	std_test_score	rank_test_score
0	-7.868349	-7.860058	0.158714	31
1	-7.868349	-7.860058	0.158714	31
2	-6.383989	-6.447796	0.160961	29
3	-6.383989	-6.447796	0.160961	29
4	-4.418050	-4.660120	0.174970	19
5	-4.418050	-4.660120	0.174970	19
6	-7.869608	-7.860440	0.157032	33
7	-7.869608	-7.860440	0.157032	33
8	-6.380977	-6.445496	0.159897	27
9	-6.380977	-6.445496	0.159897	27
10	-4.419229	-4.660193	0.174983	21
11	-4.419229	-4.660193	0.174983	21
12	-7.871709	-7.861432	0.156003	35
13	-7.871709	-7.861432	0.156003	35
14	-6.378792	-6.444495	0.158522	25
15	-6.378792	-6.444495	0.158522	25
16	-4.421981	-4.663523	0.174169	23
17	-4.421981	-4.663523	0.174169	23
18	-4.317670	-4.600068	0.168870	17
19	-4.317670	-4.600068	0.168870	17
20	-3.227816	-3.243731	0.092065	11
21	-3.227816	-3.243731	0.092065	11
22	-1.943277	-2.070856	0.096772	5
23	-1.943277	-2.070856	0.096772	5
24	-4.332465	-4.589361	0.153697	15
25	-4.332465	-4.589361	0.153697	15
26	-3.171301	-3.203446	0.107820	7
27	-3.171301	-3.203446	0.107820	7
28	-2.024192	-2.018543	0.065378	1
29	-2.024192	-2.018543	0.065378	1
30	-4.367697	-4.582040	0.128561	13
31	-4.367697	-4.582040	0.128561	13
32	-3.234924	-3.228856	0.116271	9
33	-3.234924	-3.228856	0.116271	9
34	-2.029527	-2.046425	0.126879	3

```
35          -2.029527        -2.046425        0.126879                3
[ 5.77257103  2.71321492 34.23584791 …  2.24031845 14.15423095
  0.35154327]
[41.00555646 -0.83261161 37.19630089 … -1.23440346  6.85440592
 17.10392427]
Train R-sq:
0.9953169265206908
Test R-sq:
0.9946084935331966
Train RMSE:
1.2934970735367237
Test RMSE:
1.4032555490004501
End Time = 2023-04-13 19:29:37.552207
Script Time = 0:52:22.942444
```

```
/opt/conda/lib/python3.7/site-
packages/sklearn/neural_network/_multilayer_perceptron.py:571:
ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reached and
the optimization hasn't converged yet.
  % self.max_iter, ConvergenceWarning)
```

[9]:
```python
s3_m1v1_nn_pqt_base_path = f"../models"

if not os.path.exists(s3_m1v1_nn_pqt_base_path):
    os.makedirs(s3_m1v1_nn_pqt_base_path)

s3_m1v1_nn_pqt_path = os.path.join(s3_m1v1_nn_pqt_base_path,
                                   'm1v1_nn.parquet')

# save the model to disk using joblib
joblib.dump(m1v1_nn,
            s3_m1v1_nn_pqt_path)

# load the saved model from disk using joblib
m1v1_nn_fitted = joblib.load(s3_m1v1_nn_pqt_path)
```

[10]:
```python
# specify the S3 bucket and key where you want to save the model
m1v1_nn_key_name = 'team_8_data/models/m1v1_nn.parquet'

# save the model to an in-memory buffer
buffer = BytesIO()
joblib.dump(m1v1_nn, buffer)

# upload the buffer to S3
buffer.seek(0)
s3.upload_fileobj(buffer, def_bucket, m1v1_nn_key_name)
```

```
# load the saved model from S3
#buffer = BytesIO()
#s3.download_fileobj(def_bucket, m1v1_nn_key_name, buffer)
#buffer.seek(0)
#m1v1_nn_fitted = joblib.load(buffer)
```

2.6.2 Lasso - Using `GridSearchCV`

[11]:
```
# Start timer script
start_time = dt.datetime.today()

# Citation: Hochberg, 2018; Shanmukh, 2021
m2v1_ls_pip = Pipeline([('si', SimpleImputer(strategy='median')),
                        ('ss', StandardScaler()),
                        ('ls', Lasso(random_state=1699))])

alpha_hparam = [.01, .05, .1, .5, 1, 2]
selection_hparam = ['cyclic', 'random']


m2v1_ls_grd = {'ls__alpha': alpha_hparam,
               'ls__selection': selection_hparam
           }

m2v1_ls = GridSearchCV(m2v1_ls_pip,
                       m2v1_ls_grd,
                       scoring='neg_root_mean_squared_error',
                       n_jobs=2,
                       refit=True,
                       verbose=2)

m2v1_ls.fit(train_x01, train_y01)

print(f'Best Estimator:\n{m2v1_ls.best_estimator_}')
print(f'Coefficients:\n{m2v1_ls.best_estimator_.named_steps["ls"].coef_}')

print(pd.DataFrame(m2v1_ls.cv_results_))

train_m2v1_ls_y01_pred = m2v1_ls.predict(train_x01)
print(train_m2v1_ls_y01_pred)

test_m2v1_ls_y01_pred = m2v1_ls.predict(test_x01)
print(test_m2v1_ls_y01_pred)

# Display evaluation metrics
# R-sq
train_m2v1_ls_r2 = r2_score(train_y01, train_m2v1_ls_y01_pred)
```

```
test_m2v1_ls_r2 = r2_score(test_y01, test_m2v1_ls_y01_pred)

print(f'Train R-sq:\n{train_m2v1_ls_r2}')
print(f'Test R-sq:\n{test_m2v1_ls_r2}')

# RMSE
train_m2v1_ls_rmse = mean_squared_error(train_y01, train_m2v1_ls_y01_pred,␣
 ↪squared=False)
test_m2v1_ls_rmse = mean_squared_error(test_y01, test_m2v1_ls_y01_pred,␣
 ↪squared=False)

print(f'Train RMSE:\n{train_m2v1_ls_rmse}')
print(f'Test RMSE:\n{test_m2v1_ls_rmse}')

# End timer script
end_time = dt.datetime.today()
time_elapse = end_time - start_time
print(f'End Time = {end_time}')
print(f'Script Time = {time_elapse}')
```

```
Fitting 5 folds for each of 12 candidates, totalling 60 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done  37 tasks      | elapsed:   15.4s
[Parallel(n_jobs=2)]: Done  60 out of  60 | elapsed:   18.6s finished

Best Estimator:
Pipeline(memory=None,
         steps=[('si',
                 SimpleImputer(add_indicator=False, copy=True, fill_value=None,
                               missing_values=nan, strategy='median',
                               verbose=0)),
                ('ss',
                 StandardScaler(copy=True, with_mean=True, with_std=True)),
                ('ls',
                 Lasso(alpha=0.01, copy_X=True, fit_intercept=True,
                       max_iter=1000, normalize=False, positive=False,
                       precompute=False, random_state=1699, selection='random',
                       tol=0.0001, warm_start=False))],
         verbose=False)
Coefficients:
[-0.04698528  1.1655262   0.         -1.02395897  1.13308637 -0.
 -0.          0.          0.02814286 -0.         -0.0307876   0.
  0.         -0.         -0.01445511  0.         -0.          2.92859099
  2.50110937  5.60822761  1.40168756 -2.34296164 -1.18943262 -0.33822063
 -1.3657179  -3.4737577  -6.79353668  0.12912462  0.20841108 -0.04818647
 -0.51860673  2.15041078  0.72946348 -0.44590033  0.         -1.47779489
 -0.48312123  0.          1.54018429  0.49113008  0.56200594 -0.72542784
```

```
 -8.16308228  0.40673301 -1.2849073   0.46802939  0.73592459  1.6206294 ]
    mean_fit_time  std_fit_time  mean_score_time  std_score_time  \
0        1.025094      0.110517         0.009961        0.001149
1        1.579347      0.095895         0.010434        0.000936
2        0.512077      0.018247         0.010032        0.001147
3        0.840451      0.166026         0.009896        0.001420
4        0.431524      0.015982         0.008961        0.001188
5        1.213542      0.502992         0.010119        0.001508
6        0.287477      0.024440         0.008343        0.000368
7        0.303767      0.005579         0.010572        0.001484
8        0.253030      0.003281         0.010716        0.000799
9        0.258238      0.008042         0.010476        0.000802
10       0.242289      0.005520         0.010471        0.000613
11       0.238470      0.028518         0.008936        0.001677

   param_ls__alpha param_ls__selection  \
0             0.01              cyclic
1             0.01              random
2             0.05              cyclic
3             0.05              random
4              0.1              cyclic
5              0.1              random
6              0.5              cyclic
7              0.5              random
8                1              cyclic
9                1              random
10               2              cyclic
11               2              random

                                        params  split0_test_score  \
0   {'ls__alpha': 0.01, 'ls__selection': 'cyclic'}         -11.378789
1   {'ls__alpha': 0.01, 'ls__selection': 'random'}         -11.378654
2   {'ls__alpha': 0.05, 'ls__selection': 'cyclic'}         -11.381059
3   {'ls__alpha': 0.05, 'ls__selection': 'random'}         -11.380958
4    {'ls__alpha': 0.1, 'ls__selection': 'cyclic'}         -11.396508
5    {'ls__alpha': 0.1, 'ls__selection': 'random'}         -11.396574
6    {'ls__alpha': 0.5, 'ls__selection': 'cyclic'}         -11.806394
7    {'ls__alpha': 0.5, 'ls__selection': 'random'}         -11.806288
8      {'ls__alpha': 1, 'ls__selection': 'cyclic'}         -12.008540
9      {'ls__alpha': 1, 'ls__selection': 'random'}         -12.008700
10     {'ls__alpha': 2, 'ls__selection': 'cyclic'}         -12.496541
11     {'ls__alpha': 2, 'ls__selection': 'random'}         -12.496534

    split1_test_score  split2_test_score  split3_test_score  \
0          -11.391382         -11.411164         -11.505775
1          -11.391378         -11.411135         -11.505295
2          -11.384965         -11.423042         -11.519926
3          -11.384860         -11.423228         -11.520242
```

```
4          -11.408320         -11.450277         -11.549378
5          -11.408500         -11.450586         -11.550226
6          -11.866156         -11.935607         -12.014084
7          -11.866341         -11.935859         -12.014298
8          -12.112724         -12.186260         -12.226056
9          -12.112880         -12.186494         -12.226203
10         -12.607476         -12.682111         -12.710059
11         -12.607472         -12.682084         -12.710063

    split4_test_score  mean_test_score  std_test_score  rank_test_score
0          -11.386550       -11.414732        0.046761                2
1          -11.386614       -11.414615        0.046588                1
2          -11.395211       -11.420840        0.051671                3
3          -11.395450       -11.420948        0.051800                4
4          -11.417847       -11.444466        0.055417                5
5          -11.418167       -11.444811        0.055679                6
6          -11.850882       -11.894625        0.072744                7
7          -11.851056       -11.894768        0.072833                8
8          -12.081994       -12.123115        0.076825                9
9          -12.082092       -12.123274        0.076840               10
10         -12.596496       -12.618537        0.074712               11
11         -12.596565       -12.618544        0.074706               12
[ 1.00885432  2.1155105  20.65181841 …  5.04395528 20.70033313
  9.2996565 ]
[43.29645545  3.29328038 36.54323059 … -0.46893988  1.17345827
  3.2687973 ]
Train R-sq:
0.6369229292317483
Test R-sq:
0.6479556475979478
Train RMSE:
11.389361452679568
Test RMSE:
11.339147217962033
End Time = 2023-04-13 19:29:57.540560
Script Time = 0:00:19.711830
```

[12]:
```
coef_intercept = np.hstack((m2v1_ls.best_estimator_.named_steps["ls"].coef_,
                            m2v1_ls.best_estimator_.named_steps["ls"].
 ↪intercept_))
#print(coef_intercept)

coef_intercept_df01 = pd.DataFrame(coef_intercept)
#display(coef_intercept_df01)

train_x01_col_names = list(train_x01.columns)
train_x01_col_names.append('intercept')
```

```
train_x01_col_names_df01 = pd.DataFrame(train_x01_col_names)
#display(train_x01_col_names_df01)

model_params = pd.concat([train_x01_col_names_df01, coef_intercept_df01],␣
 ↪axis=1)
display(model_params)
```

	0	0
0	borough_bronx	-0.046985
1	borough_brooklyn	1.165526
2	borough_manhattan	0.000000
3	borough_queens	-1.023959
4	borough_staten island	1.133086
5	relative_data_year_-4	-0.000000
6	relative_data_year_-3	-0.000000
7	relative_data_year_-2	0.000000
8	relative_data_year_-1	0.028143
9	relative_data_year_0	-0.000000
10	complaint_type_FELONY	-0.030788
11	complaint_type_MISDEMEANOR	0.000000
12	complaint_type_VIOLATION	0.000000
13	annual_evictions_x_borough	-0.000000
14	annual_complaint_counts	-0.014455
15	annual_grad_n	0.000000
16	annual_dropped_out_n	-0.000000
17	totalpop	2.928591
18	men	2.501109
19	women	5.608228
20	hispanic	1.401688
21	white	-2.342962
22	black	-1.189433
23	native	-0.338221
24	asian	-1.365718
25	citizen	-3.473758
26	income	-6.793537
27	incomeerr	0.129125
28	incomepercap	0.208411
29	incomepercaperr	-0.048186
30	professional	-0.518607
31	service	2.150411
32	office	0.729463
33	construction	-0.445900
34	production	0.000000
35	drive	-1.477795
36	carpool	-0.483121
37	transit	0.000000
38	walk	1.540184

```
39                 othertransp   0.491130
40                  workathome   0.562006
41                meancommute  -0.725428
42                   employed  -8.163082
43                privatework   0.406733
44                 publicwork  -1.284907
45               selfemployed   0.468029
46                 familywork   0.735925
47               unemployment   1.620629
48                  intercept  24.475273
```

[13]:
```
# specify the S3 bucket and key where you want to save the model
m2v1_ls_key_name = 'team_8_data/models/m2v1_ls.parquet'

# save the model to an in-memory buffer
buffer = BytesIO()
joblib.dump(m2v1_ls, buffer)

# upload the buffer to S3
buffer.seek(0)
s3.upload_fileobj(buffer, def_bucket, m2v1_ls_key_name)

# load the saved model from S3
#buffer = BytesIO()
#s3.download_fileobj(def_bucket, m2v1_ls_key_name, buffer)
#buffer.seek(0)
#m2v1_ls_fitted = joblib.load(buffer)
```

2.6.3 Lasso - Using BayesSearchCV

[14]:
```
# Start timer script
start_time = dt.datetime.today()

# Citation: Hochberg, 2018; Shanmukh, 2021
m2v2_ls_pip = Pipeline([('si', SimpleImputer(strategy='median')),
                        ('ss', StandardScaler()),
                        ('ls', Lasso(random_state=1699))])

alpha_hparam = Real(1e-3, 1e3, prior='log-uniform')
selection_hparam = Categorical(['cyclic', 'random'])
max_iter_hparam = Integer(100, 5000, prior='log-uniform')
warm_start_hparam = Categorical([False, True])


m2v2_ls_grd = {'ls__alpha': alpha_hparam,
               'ls__selection': selection_hparam,
               'ls__max_iter': max_iter_hparam,
```

```
                'ls__warm_start': warm_start_hparam
            }

m2v2_ls = BayesSearchCV(m2v2_ls_pip,
                        m2v2_ls_grd,
                        scoring='neg_root_mean_squared_error',
                         cv=5,
                        n_jobs=2,
                        refit=True,
                        verbose=2)

m2v2_ls.fit(train_x01, train_y01)

print(f'Best Estimator:\n{m2v2_ls.best_estimator_}')
print(f'Coefficients:\n{m2v2_ls.best_estimator_.named_steps["ls"].coef_}')

print(pd.DataFrame(m2v2_ls.cv_results_))

train_m2v2_ls_y01_pred = m2v2_ls.predict(train_x01)
print(train_m2v2_ls_y01_pred)

test_m2v2_ls_y01_pred = m2v2_ls.predict(test_x01)
print(test_m2v2_ls_y01_pred)

# Display evaluation metrics
# R-sq
train_m2v2_ls_r2 = r2_score(train_y01, train_m2v2_ls_y01_pred)
test_m2v2_ls_r2 = r2_score(test_y01, test_m2v2_ls_y01_pred)

print(f'Train R-sq:\n{train_m2v2_ls_r2}')
print(f'Test R-sq:\n{test_m2v2_ls_r2}')

# RMSE
train_m2v2_ls_rmse = mean_squared_error(train_y01, train_m2v2_ls_y01_pred,␣
 ↪squared=False)
test_m2v2_ls_rmse = mean_squared_error(test_y01, test_m2v2_ls_y01_pred,␣
 ↪squared=False)

print(f'Train RMSE:\n{train_m2v2_ls_rmse}')
print(f'Test RMSE:\n{test_m2v2_ls_rmse}')

# End timer script
end_time = dt.datetime.today()
time_elapse = end_time - start_time
print(f'End Time = {end_time}')
print(f'Script Time = {time_elapse}')
```

```
Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s finished
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.6s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.6s finished
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s finished
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.8s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.8s finished
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.3s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.3s finished
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    3.4s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    3.4s finished
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.4s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.4s finished
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s finished
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits
```

```
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    0.7s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.6s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.6s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.7s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.3s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.3s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   14.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   14.7s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.9s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.9s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    7.5s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    7.5s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished
```

```
Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.3s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.3s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    4.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    4.7s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    8.4s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    8.4s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.3s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.3s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    6.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    6.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   26.6s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   26.6s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits
```

```
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   10.9s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   10.9s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    5.7s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    5.7s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   16.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   16.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    4.2s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    4.2s finished
```

```
Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    3.5s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    3.5s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    8.4s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    8.4s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.4s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.4s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.9s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.9s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    3.8s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    3.8s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits
```

```
[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   14.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:   14.1s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.0s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    2.0s finished

Fitting 5 folds for each of 1 candidates, totalling 5 fits

[Parallel(n_jobs=2)]: Using backend LokyBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s remaining:    0.0s
[Parallel(n_jobs=2)]: Done   5 out of   5 | elapsed:    1.1s finished

Best Estimator:
Pipeline(memory=None,
         steps=[('si',
                 SimpleImputer(add_indicator=False, copy=True, fill_value=None,
                               missing_values=nan, strategy='median',
                               verbose=0)),
                ('ss',
                 StandardScaler(copy=True, with_mean=True, with_std=True)),
                ('ls',
                 Lasso(alpha=0.00312004499292689, copy_X=True,
                       fit_intercept=True, max_iter=5000, normalize=False,
                       positive=False, precompute=False, random_state=1699,
                       selection='cyclic', tol=0.0001, warm_start=False))],
         verbose=False)
Coefficients:
[-5.90772722e-02  1.20356958e+00  0.00000000e+00 -1.03729787e+00
  1.13319125e+00 -0.00000000e+00 -0.00000000e+00  2.94787994e-03
  3.38749310e-02 -2.05060234e-04 -2.82296287e-02  2.06826899e-02
  0.00000000e+00 -0.00000000e+00 -4.34594017e-02  0.00000000e+00
 -0.00000000e+00  8.48098006e+00  0.00000000e+00  2.76832909e+00
  6.97559873e-01 -3.30318362e+00 -2.14827171e+00 -3.71075424e-01
 -1.90796058e+00 -3.65510811e+00 -6.88025498e+00  1.54667377e-01
  4.43630919e-01 -1.76161247e-01 -1.81767382e+00  1.33115461e+00
  2.87452875e-01 -7.98665794e-01 -3.86055676e-01 -1.45597342e+00
 -4.86681209e-01  0.00000000e+00  1.52719040e+00  4.85589992e-01
  5.68145407e-01 -7.49278734e-01 -8.29828524e+00  0.00000000e+00
 -1.69877386e+00  2.42544215e-01  7.16688567e-01  1.60868297e+00]
    mean_fit_time  std_fit_time  mean_score_time  std_score_time  \
0        0.252393      0.044313         0.008819        0.001531
1        0.211456      0.029397         0.009242        0.001724
2        0.238254      0.037010         0.010291        0.002250
3        0.241629      0.017929         0.010836        0.002084
4        0.371777      0.079431         0.015980        0.004156
5        1.241521      0.785646         0.011383        0.002698
```

```
6        0.476834      0.089144         0.009877        0.001961
7        0.227144      0.035521         0.009860        0.002405
8        0.213959      0.030165         0.009917        0.002330
9        0.243137      0.040122         0.009101        0.001723
10       0.543308      0.085319         0.009564        0.001922
11       0.606364      0.112065         0.008954        0.001687
12       0.449924      0.079588         0.010595        0.003273
13       5.474890      2.152558         0.010317        0.002130
14       1.031375      0.176444         0.009314        0.001718
15       2.952774      1.177147         0.010025        0.002432
16       0.334927      0.052971         0.009441        0.001776
17       0.371395      0.064502         0.009407        0.001627
18       0.370219      0.069268         0.008443        0.001334
19       0.356148      0.062020         0.009524        0.002007
20       0.456968      0.099663         0.013050        0.004496
21       1.705872      0.324273         0.010758        0.004344
22       0.355035      0.064276         0.008949        0.001570
23       3.311975      0.950526         0.010320        0.002188
24       0.873525      0.198976         0.010325        0.003037
25       2.112508      0.328161         0.008992        0.002086
26       9.484036      1.248022         0.010280        0.001913
27       0.354692      0.058464         0.009997        0.001848
28       0.367649      0.061606         0.009237        0.001774
29       0.378164      0.070533         0.009635        0.002129
30       4.151578      2.864317         0.010427        0.002207
31       0.683975      0.111741         0.008902        0.001899
32       0.352353      0.063039         0.009234        0.001608
33       0.376071      0.064972         0.009555        0.001731
34       0.367897      0.065547         0.009819        0.002583
35       2.044964      0.413021         0.009135        0.002086
36       6.089054      1.929319         0.010797        0.002346
37       1.577792      0.276699         0.010048        0.001905
38       1.364861      0.264036         0.010090        0.001880
39       3.132879      0.511682         0.009999        0.002032
40       0.367394      0.062305         0.008894        0.001489
41       0.473200      0.077456         0.010350        0.002553
42       0.362873      0.056866         0.009705        0.002118
43       1.114455      0.251687         0.010146        0.002999
44       0.342149      0.052996         0.009398        0.001434
45       0.369063      0.061133         0.009580        0.001755
46       1.316128      0.174785         0.009341        0.001797
47       4.643917      1.957186         0.010486        0.002173
48       0.740760      0.162634         0.009555        0.002145
49       0.371850      0.062724         0.009669        0.001504

   param_ls__alpha param_ls__max_iter param_ls__selection  \
0         0.919229                372              cyclic
1        20.785016                193              cyclic
```

2	1.469648	441	random
3	9.380698	174	random
4	520.400979	2697	cyclic
5	0.002528	165	random
6	0.252225	949	cyclic
7	2.444393	4088	cyclic
8	107.97026	357	cyclic
9	0.859011	467	cyclic
10	0.020092	5000	cyclic
11	0.003409	246	random
12	0.056266	5000	cyclic
13	0.001	5000	cyclic
14	0.008623	5000	cyclic
15	0.001422	5000	cyclic
16	0.10612	100	random
17	0.001179	100	random
18	0.005593	100	cyclic
19	0.030282	100	random
20	0.012457	100	random
21	0.006117	5000	random
22	0.040493	100	cyclic
23	0.00312	5000	cyclic
24	0.040469	5000	random
25	0.004728	5000	cyclic
26	0.001	5000	random
27	0.061997	100	random
28	0.016873	100	random
29	0.001819	100	random
30	0.00183	5000	cyclic
31	0.013661	5000	cyclic
32	0.046368	100	random
33	0.006918	100	random
34	0.001	100	cyclic
35	0.004217	5000	random
36	0.001	5000	cyclic
37	0.010632	5000	random
38	0.015439	5000	random
39	0.002274	5000	random
40	0.002292	100	cyclic
41	0.033243	5000	cyclic
42	0.009614	100	cyclic
43	0.023583	5000	random
44	0.080033	100	cyclic
45	0.026002	100	cyclic
46	0.007288	5000	cyclic
47	0.001158	5000	cyclic
48	0.080142	5000	random
49	0.003943	100	cyclic

	param_ls__warm_start	params
0	True	{'ls__alpha': 0.9192293938041608, 'ls__max_ite…
1	True	{'ls__alpha': 20.785016353070272, 'ls__max_ite…
2	True	{'ls__alpha': 1.4696478744138841, 'ls__max_ite…
3	True	{'ls__alpha': 9.380698318075792, 'ls__max_iter…
4	True	{'ls__alpha': 520.4009792978661, 'ls__max_iter…
5	True	{'ls__alpha': 0.0025276741815852596, 'ls__max_…
6	True	{'ls__alpha': 0.2522250210775078, 'ls__max_ite…
7	True	{'ls__alpha': 2.4443925994929305, 'ls__max_ite…
8	False	{'ls__alpha': 107.97025991821005, 'ls__max_ite…
9	False	{'ls__alpha': 0.8590108805849646, 'ls__max_ite…
10	False	{'ls__alpha': 0.020091744449482356, 'ls__max_i…
11	True	{'ls__alpha': 0.003409256952063032, 'ls__max_i…
12	True	{'ls__alpha': 0.056266004320859396, 'ls__max_i…
13	False	{'ls__alpha': 0.001, 'ls__max_iter': 5000, 'ls…
14	True	{'ls__alpha': 0.008623045081308458, 'ls__max_i…
15	False	{'ls__alpha': 0.0014221967527329972, 'ls__max_…
16	False	{'ls__alpha': 0.10611954673637193, 'ls__max_it…
17	True	{'ls__alpha': 0.0011791282733941873, 'ls__max_…
18	False	{'ls__alpha': 0.005592992209800713, 'ls__max_i…
19	True	{'ls__alpha': 0.030281735565660575, 'ls__max_i…
20	False	{'ls__alpha': 0.012457400581715239, 'ls__max_i…
21	False	{'ls__alpha': 0.006117246602685628, 'ls__max_i…
22	False	{'ls__alpha': 0.04049343717819463, 'ls__max_it…
23	False	{'ls__alpha': 0.00312004499292689, 'ls__max_it…
24	False	{'ls__alpha': 0.04046875004317316, 'ls__max_it…
25	True	{'ls__alpha': 0.004728162592110137, 'ls__max_i…
26	False	{'ls__alpha': 0.001, 'ls__max_iter': 5000, 'ls…
27	False	{'ls__alpha': 0.06199705881601676, 'ls__max_it…
28	True	{'ls__alpha': 0.01687341186753009, 'ls__max_it…
29	False	{'ls__alpha': 0.0018185560271089783, 'ls__max_…
30	True	{'ls__alpha': 0.0018303658598422532, 'ls__max_…
31	True	{'ls__alpha': 0.013661350637267831, 'ls__max_i…
32	True	{'ls__alpha': 0.04636797353331355, 'ls__max_it…
33	True	{'ls__alpha': 0.006918445469792199, 'ls__max_i…
34	False	{'ls__alpha': 0.001, 'ls__max_iter': 100, 'ls_…
35	False	{'ls__alpha': 0.00421690504388654, 'ls__max_it…
36	True	{'ls__alpha': 0.001, 'ls__max_iter': 5000, 'ls…
37	True	{'ls__alpha': 0.010632072820022796, 'ls__max_i…
38	False	{'ls__alpha': 0.015438722672047024, 'ls__max_i…
39	False	{'ls__alpha': 0.0022740964287695036, 'ls__max_…
40	False	{'ls__alpha': 0.0022923216227139103, 'ls__max_…
41	True	{'ls__alpha': 0.03324342462416447, 'ls__max_it…
42	False	{'ls__alpha': 0.009614323672543546, 'ls__max_i…
43	True	{'ls__alpha': 0.02358308086767287, 'ls__max_it…
44	True	{'ls__alpha': 0.08003292102153944, 'ls__max_it…
45	False	{'ls__alpha': 0.02600213461730265, 'ls__max_it…

```
46               False  {'ls__alpha': 0.0072884291262491525, 'ls__max_…
47               False  {'ls__alpha': 0.0011575416936216076, 'ls__max_…
48                True  {'ls__alpha': 0.08014202444546731, 'ls__max_it…
49               False  {'ls__alpha': 0.003942785721175991, 'ls__max_i…

    split0_test_score  split1_test_score  split2_test_score  \
0          -11.977457         -12.082645         -12.155722
1          -18.828273         -18.816528         -18.873667
2          -12.217014         -12.327770         -12.406181
3          -16.469237         -16.476761         -16.486357
4          -18.828273         -18.816528         -18.873667
5          -11.380662         -11.395944         -11.410907
6          -11.507037         -11.556216         -11.604021
7          -12.747390         -12.846645         -12.912869
8          -18.828273         -18.816528         -18.873667
9          -11.955538         -12.058478         -12.134445
10         -11.380589         -11.386755         -11.415462
11         -11.379845         -11.396099         -11.409756
12         -11.381898         -11.386847         -11.425501
13         -11.379818         -11.398849         -11.406317
14         -11.378792         -11.392354         -11.410356
15         -11.379706         -11.398501         -11.406496
16         -11.400669         -11.413166         -11.456496
17         -11.382560         -11.395759         -11.412853
18         -11.380164         -11.395063         -11.410263
19         -11.380880         -11.382676         -11.418229
20         -11.381084         -11.389948         -11.414542
21         -11.378914         -11.394251         -11.408826
22         -11.381145         -11.383385         -11.418390
23         -11.379440         -11.396892         -11.407253
24         -11.380317         -11.382732         -11.420022
25         -11.379224         -11.395523         -11.408173
26         -11.379918         -11.398930         -11.406477
27         -11.384259         -11.389523         -11.429869
28         -11.381018         -11.388484         -11.415529
29         -11.382439         -11.395379         -11.413013
30         -11.379603         -11.398171         -11.406777
31         -11.379134         -11.389215         -11.413041
32         -11.381865         -11.384254         -11.423136
33         -11.381857         -11.392092         -11.413386
34         -11.379834         -11.397019         -11.409298
35         -11.379235         -11.395867         -11.407826
36         -11.379818         -11.398849         -11.406317
37         -11.378559         -11.390977         -11.411400
38         -11.379298         -11.388353         -11.414127
39         -11.379691         -11.397716         -11.407038
40         -11.379802         -11.396319         -11.409414
41         -11.380290         -11.382304         -11.417945
```

42	-11.380949	-11.394112	-11.411515
43	-11.380230	-11.385291	-11.416149
44	-11.388175	-11.396595	-11.436773
45	-11.381719	-11.385534	-11.414868
46	-11.378868	-11.393370	-11.409555
47	-11.379802	-11.398752	-11.406384
48	-11.388123	-11.396783	-11.437692
49	-11.379968	-11.395684	-11.409814

	split3_test_score	split4_test_score	mean_test_score	std_test_score	\
0	-12.199795	-12.052402	-12.093604	0.078029	
1	-18.934900	-19.055553	-18.901784	0.087412	
2	-12.424359	-12.291528	-12.333370	0.076028	
3	-16.600301	-16.742277	-16.554987	0.105179	
4	-18.934900	-19.055553	-18.901784	0.087412	
5	-11.504640	-11.387121	-11.415855	0.045537	
6	-11.705059	-11.555625	-11.585592	0.067147	
7	-12.945755	-12.854852	-12.861502	0.067858	
8	-18.934900	-19.055553	-18.901784	0.087412	
9	-12.179510	-12.029071	-12.071409	0.078797	
10	-11.510673	-11.389846	-11.416665	0.048486	
11	-11.503854	-11.386609	-11.415232	0.045434	
12	-11.522605	-11.397110	-11.422792	0.052142	
13	-11.500890	-11.385568	-11.414288	0.044305	
14	-11.504997	-11.386338	-11.414567	0.046402	
15	-11.501125	-11.385462	-11.414258	0.044446	
16	-11.555529	-11.423823	-11.449937	0.055954	
17	-11.506265	-11.388701	-11.417228	0.045658	
18	-11.503716	-11.386209	-11.415083	0.045460	
19	-11.513630	-11.391245	-11.417332	0.049969	
20	-11.509086	-11.388563	-11.416645	0.047573	
21	-11.503610	-11.385897	-11.414300	0.045754	
22	-11.515097	-11.391125	-11.417828	0.050411	
23	-11.502023	-11.385352	-11.414192	0.044947	
24	-11.516632	-11.393011	-11.418543	0.051028	
25	-11.502993	-11.385667	-11.414316	0.045402	
26	-11.501023	-11.385705	-11.414411	0.044312	
27	-11.526411	-11.399865	-11.425986	0.052637	
28	-11.510211	-11.389163	-11.416881	0.048111	
29	-11.506586	-11.388744	-11.417232	0.045829	
30	-11.501183	-11.385397	-11.414226	0.044508	
31	-11.507987	-11.387405	-11.415356	0.047671	
32	-11.519297	-11.394784	-11.420667	0.051450	
33	-11.508549	-11.388722	-11.416921	0.047009	
34	-11.503119	-11.385764	-11.415007	0.045193	
35	-11.502647	-11.385574	-11.414230	0.045258	
36	-11.500890	-11.385568	-11.414288	0.044305	
37	-11.505984	-11.386739	-11.414732	0.046892	

38	-11.508911	-11.388004	-11.415739	0.048022
39	-11.501598	-11.385604	-11.414330	0.044654
40	-11.503101	-11.385844	-11.414896	0.045235
41	-11.514004	-11.391430	-11.417194	0.050231
42	-11.504158	-11.387072	-11.415561	0.045465
43	-11.511637	-11.390373	-11.416736	0.049037
44	-11.533929	-11.406318	-11.432358	0.053377
45	-11.511093	-11.389242	-11.416491	0.048708
46	-11.504270	-11.386175	-11.414448	0.046047
47	-11.500968	-11.385536	-11.414288	0.044347
48	-11.536468	-11.407262	-11.433265	0.054250
49	-11.503447	-11.385891	-11.414961	0.045382

	rank_test_score
0	44
1	48
2	45
3	47
4	48
5	23
6	42
7	46
8	48
9	43
10	26
11	19
12	37
13	5
14	13
15	4
16	41
17	31
18	18
19	33
20	25
21	8
22	34
23	1
24	35
25	9
26	11
27	38
28	28
29	32
30	2
31	20
32	36
33	29

```
34                17
35                 3
36                 5
37                14
38                22
39                10
40                15
41                30
42                21
43                27
44                39
45                24
46                12
47                 7
48                40
49                16
[ 0.80813214  2.15188504 20.47191416 …  4.87480063 20.68240508
  9.20470403]
[43.3663052   3.55764009 36.54084305 … -0.57203178  1.09980524
  3.14855382]
Train R-sq:
0.6370404443324977
Test R-sq:
0.6480368310543366
Train RMSE:
11.387518138430686
Test RMSE:
11.337839706231485
End Time = 2023-04-13 19:34:32.039668
Script Time = 0:04:34.328540
```

[15]:
```
coef_intercept = np.hstack((m2v2_ls.best_estimator_.named_steps["ls"].coef_,
                            m2v2_ls.best_estimator_.named_steps["ls"].
 ↪intercept_))
#print(coef_intercept)

coef_intercept_df01 = pd.DataFrame(coef_intercept)
#display(coef_intercept_df01)

train_x01_col_names = list(train_x01.columns)
train_x01_col_names.append('intercept')

train_x01_col_names_df01 = pd.DataFrame(train_x01_col_names)
#display(train_x01_col_names_df01)

model_params = pd.concat([train_x01_col_names_df01, coef_intercept_df01],␣
 ↪axis=1)
```

```
display(model_params)
```

	0	0
0	borough_bronx	-0.059077
1	borough_brooklyn	1.203570
2	borough_manhattan	0.000000
3	borough_queens	-1.037298
4	borough_staten island	1.133191
5	relative_data_year_-4	-0.000000
6	relative_data_year_-3	-0.000000
7	relative_data_year_-2	0.002948
8	relative_data_year_-1	0.033875
9	relative_data_year_0	-0.000205
10	complaint_type_FELONY	-0.028230
11	complaint_type_MISDEMEANOR	0.020683
12	complaint_type_VIOLATION	0.000000
13	annual_evictions_x_borough	-0.000000
14	annual_complaint_counts	-0.043459
15	annual_grad_n	0.000000
16	annual_dropped_out_n	-0.000000
17	totalpop	8.480980
18	men	0.000000
19	women	2.768329
20	hispanic	0.697560
21	white	-3.303184
22	black	-2.148272
23	native	-0.371075
24	asian	-1.907961
25	citizen	-3.655108
26	income	-6.880255
27	incomeerr	0.154667
28	incomepercap	0.443631
29	incomepercaperr	-0.176161
30	professional	-1.817674
31	service	1.331155
32	office	0.287453
33	construction	-0.798666
34	production	-0.386056
35	drive	-1.455973
36	carpool	-0.486681
37	transit	0.000000
38	walk	1.527190
39	othertransp	0.485590
40	workathome	0.568145
41	meancommute	-0.749279
42	employed	-8.298285
43	privatework	0.000000
44	publicwork	-1.698774

```
45                selfemployed   0.242544
46                  familywork   0.716689
47                unemployment   1.608683
48                   intercept  24.475273
```

[16]:
```
# specify the S3 bucket and key where you want to save the model
m2v2_ls_key_name = 'team_8_data/models/m2v2_ls.parquet'

# save the model to an in-memory buffer
buffer = BytesIO()
joblib.dump(m2v2_ls, buffer)

# upload the buffer to S3
buffer.seek(0)
s3.upload_fileobj(buffer, def_bucket, m2v2_ls_key_name)

# load the saved model from S3
#buffer = BytesIO()
#s3.download_fileobj(def_bucket, m2v2_ls_key_name, buffer)
#buffer.seek(0)
#m2v2_ls_fitted = joblib.load(buffer)
```

2.7 Release Resources

[17]:
```
%%html

<p><b>Shutting down your kernel for this notebook to release resources.</b></p>
<button class="sm-command-button" data-commandlinker-command="kernelmenu:
 ↪shutdown" style="display:none;">Shutdown Kernel</button>

<script>
try {
    els = document.getElementsByClassName("sm-command-button");
    els[0].click();
}
catch(err) {
    // NoOp
}
</script>
```

```
<IPython.core.display.HTML object>
```

[18]:
```
%%javascript

try {
    Jupyter.notebook.save_checkpoint();
    Jupyter.notebook.session.delete();
}
```

```
catch(err) {
    // NoOp
}
```

```
<IPython.core.display.Javascript object>
```