Blue Bikes Data Preprocessing

June 30, 2023

[1]:
```
import os
import numpy as np
import pandas as pd
```

[2]:
```
import warnings

warnings.filterwarnings('ignore') # supressing warnings
```

[3]:
```
np.version.version
```

[3]:
```
'1.23.5'
```

[4]:
```
pd.__version__
```

[4]:
```
'2.0.0'
```

[5]:
```
# Blue Bikes tripdata from 2019 to 2022

path = 'data/'
os.listdir(path)
```

[5]:
```
['201901-bluebikes-tripdata.csv',
 '201902-bluebikes-tripdata.csv',
 '201903-bluebikes-tripdata.csv',
 '201904-bluebikes-tripdata.csv',
 '201905-bluebikes-tripdata.csv',
 '201906-bluebikes-tripdata.csv',
 '201907-bluebikes-tripdata.csv',
 '201908-bluebikes-tripdata.csv',
 '201909-bluebikes-tripdata.csv',
 '201910-bluebikes-tripdata.csv',
 '201911-bluebikes-tripdata.csv',
 '201912-bluebikes-tripdata.csv',
 '202001-bluebikes-tripdata.csv',
 '202002-bluebikes-tripdata.csv',
 '202003-bluebikes-tripdata.csv',
 '202004-bluebikes-tripdata.csv',
 '202005-bluebikes-tripdata.csv',
```

```
 '202006-bluebikes-tripdata.csv',
 '202007-bluebikes-tripdata.csv',
 '202008-bluebikes-tripdata.csv',
 '202009-bluebikes-tripdata.csv',
 '202010-bluebikes-tripdata.csv',
 '202011-bluebikes-tripdata.csv',
 '202012-bluebikes-tripdata.csv',
 '202101-bluebikes-tripdata.csv',
 '202102-bluebikes-tripdata.csv',
 '202103-bluebikes-tripdata.csv',
 '202104-bluebikes-tripdata.csv',
 '202105-bluebikes-tripdata.csv',
 '202106-bluebikes-tripdata.csv',
 '202107-bluebikes-tripdata.csv',
 '202108-bluebikes-tripdata.csv',
 '202109-bluebikes-tripdata.csv',
 '202110-bluebikes-tripdata.csv',
 '202111-bluebikes-tripdata.csv',
 '202112-bluebikes-tripdata.csv',
 '202201-bluebikes-tripdata.csv',
 '202202-bluebikes-tripdata.csv',
 '202203-bluebikes-tripdata.csv',
 '202204-bluebikes-tripdata.csv',
 '202205-bluebikes-tripdata.csv',
 '202206-bluebikes-tripdata.csv',
 '202207-bluebikes-tripdata.csv',
 '202208-bluebikes-tripdata.csv',
 '202209-bluebikes-tripdata.csv',
 '202210-bluebikes-tripdata.csv',
 '202211-bluebikes-tripdata.csv',
 '202212-bluebikes-tripdata.csv',
 'zips']
```

[6]:
```
# seperating them in diff. objects

file_list_2019 = [path + f for f in os.listdir(path) if f.startswith('2019')]
file_list_2020 = [path + f for f in os.listdir(path) if f.startswith('2020')]

file_list_2021 = [path + f for f in os.listdir(path) if f.startswith('2021')]
file_list_2022 = [path + f for f in os.listdir(path) if f.startswith('2022')]
```

[7]:
```
csv_list_2019 = []
csv_list_2020 = []

csv_list_2021 = []
csv_list_2022 = []
```

[8]:
```
# appending files to the empty lists

for file in sorted(file_list_2019):
    csv_list_2019.append(pd.read_csv(file))

for file in sorted(file_list_2020):
    csv_list_2020.append(pd.read_csv(file))


for file in sorted(file_list_2021):
    csv_list_2021.append(pd.read_csv(file))

for file in sorted(file_list_2022):
    csv_list_2022.append(pd.read_csv(file))
```

[9]:
```
# merging to dataframe

csv_merged_2019 = pd.concat(csv_list_2019, ignore_index=True)
csv_merged_2020 = pd.concat(csv_list_2020, ignore_index=True)

csv_merged_2021 = pd.concat(csv_list_2021, ignore_index=True)
csv_merged_2022 = pd.concat(csv_list_2022, ignore_index=True)
```

[10]:
```
csv_merged_2019.head(2)
```

[10]:
```
   tripduration                 starttime                  stoptime
0           371  2019-01-01 00:09:13.7980  2019-01-01 00:15:25.3360  \
1           264  2019-01-01 00:33:56.1820  2019-01-01 00:38:20.8800

   start station id                      start station name
0                80  MIT Stata Center at Vassar St / Main St  \
1               117                    Binney St / Sixth St

   start station latitude  start station longitude  end station id
0               42.362131               -71.091156             179  \
1               42.366162               -71.086883             189

  end station name  end station latitude  end station longitude  bikeid
0    MIT Vassar St             42.355601             -71.103945    3689  \
1        Kendall T             42.362428             -71.084955    4142

     usertype  birth year  gender
0  Subscriber        1987       1
1  Subscriber        1990       1
```

[11]:
```
csv_merged_2019.shape
```

[11]: (2522771, 15)

[12]:
```
csv_merged_2020.head(2)
```

[12]:
```
   tripduration                starttime                 stoptime
0           478  2020-01-01 00:04:05.8090  2020-01-01 00:12:04.2370  \
1           363  2020-01-01 00:04:45.6990  2020-01-01 00:10:49.0400

   start station id          start station name  start station latitude
0               366             Broadway T Stop                42.342781  \
1               219  Boston East - 126 Border St                42.373312

   start station longitude  end station id             end station name
0               -71.057473              93             JFK/UMass T Stop  \
1               -71.041020             212  Maverick Square - Lewis Mall

   end station latitude  end station longitude  bikeid    usertype
0             42.320340             -71.051180    6005    Customer  \
1             42.368844             -71.039778    3168  Subscriber

   birth year  gender postal code
0      1969.0     0.0         NaN
1      2000.0     1.0         NaN
```

[13]:
```
csv_merged_2020.shape
```

[13]: (2073448, 16)

[14]:
```
csv_merged_2021.head(2)
```

[14]:
```
   tripduration                starttime                 stoptime
0           914  2021-01-01 00:00:04.5900  2021-01-01 00:15:19.1680  \
1          1085  2021-01-01 00:00:21.8030  2021-01-01 00:18:27.4640

   start station id                                start station name
0                91  One Kendall Square at Hampshire St / Portland St  \
1               370                        Dartmouth St at Newbury St

   start station latitude  start station longitude  end station id
0               42.366277               -71.091690             370  \
1               42.350961               -71.077828             169

                         end station name  end station latitude
0               Dartmouth St at Newbury St             42.350961  \
1  Edwards Playground - Main St at Eden St             42.378965

   end station longitude  bikeid    usertype postal code
0             -71.077828    5316    Customer       02139
```

```
1             -71.068607    4917  Subscriber       02116
```

[15]:
```
csv_merged_2021.shape
```

[15]:
```
(2934378, 14)
```

[16]:
```
csv_merged_2022.head(2)
```

[16]:
```
   tripduration                starttime                 stoptime
0           597  2022-01-01 00:00:25.1660  2022-01-01 00:10:22.1920  \
1           411  2022-01-01 00:00:40.4300  2022-01-01 00:07:32.1980

   start station id              start station name  start station latitude
0               178  MIT Pacific St at Purrington St               42.359573  \
1               189                          Kendall T               42.362428

   start station longitude  end station id
0               -71.101295              74  \
1               -71.084955             178

                     end station name  end station latitude
0  Harvard Square at Mass Ave/ Dunster             42.373268  \
1      MIT Pacific St at Purrington St             42.359573

   end station longitude  bikeid    usertype postal code
0             -71.118579    4923  Subscriber       02139
1             -71.101295    3112  Subscriber       02139
```

[17]:
```
csv_merged_2022.shape
```

[17]:
```
(3757281, 14)
```

[18]:
```
# working with latest data ## 2021-2022

csv_merged_2021.dtypes
```

[18]:
```
tripduration                int64
starttime                  object
stoptime                   object
start station id            int64
start station name         object
start station latitude    float64
start station longitude   float64
end station id              int64
end station name           object
end station latitude      float64
end station longitude     float64
bikeid                      int64
```

```
usertype                   object
postal code                object
dtype: object
```

[19]:
```
csv_merged_2021.isnull().sum()
```

[19]:
```
tripduration                   0
starttime                      0
stoptime                       0
start station id               0
start station name             0
start station latitude         0
start station longitude        0
end station id                 0
end station name               0
end station latitude           0
end station longitude          0
bikeid                         0
usertype                       0
postal code               222076
dtype: int64
```

[20]:
```
csv_merged_2022.dtypes
```

[20]:
```
tripduration                int64
starttime                  object
stoptime                   object
start station id            int64
start station name         object
start station latitude    float64
start station longitude   float64
end station id              int64
end station name           object
end station latitude      float64
end station longitude     float64
bikeid                      int64
usertype                   object
postal code                object
dtype: object
```

[21]:
```
csv_merged_2022.isnull().sum()
```

[21]:
```
tripduration                   0
starttime                      0
stoptime                       0
start station id               0
start station name             0
```

```
start station latitude          0
start station longitude         0
end station id                  0
end station name                0
end station latitude            0
end station longitude           0
bikeid                          0
usertype                        0
postal code                465637
dtype: int64
```

[22]:
```
# comparing datatypes

csv_merged_2021.dtypes == csv_merged_2022.dtypes
```

[22]:
```
tripduration               True
starttime                  True
stoptime                   True
start station id           True
start station name         True
start station latitude     True
start station longitude    True
end station id             True
end station name           True
end station latitude       True
end station longitude      True
bikeid                     True
usertype                   True
postal code                True
dtype: bool
```

[23]:
```
# working with older data ## 2019 - 2020

csv_merged_2019.dtypes
```

[23]:
```
tripduration                int64
starttime                  object
stoptime                   object
start station id            int64
start station name         object
start station latitude    float64
start station longitude   float64
end station id              int64
end station name           object
end station latitude      float64
end station longitude     float64
bikeid                      int64
```

```
usertype                   object
birth year                  int64
gender                      int64
dtype: object
```

[24]:
```
csv_merged_2019.isnull().sum()
```

[24]:
```
tripduration               0
starttime                  0
stoptime                   0
start station id           0
start station name         0
start station latitude     0
start station longitude    0
end station id             0
end station name           0
end station latitude       0
end station longitude      0
bikeid                     0
usertype                   0
birth year                 0
gender                     0
dtype: int64
```

[25]:
```
csv_merged_2020.dtypes
```

[25]:
```
tripduration                int64
starttime                  object
stoptime                   object
start station id            int64
start station name         object
start station latitude    float64
start station longitude   float64
end station id              int64
end station name           object
end station latitude      float64
end station longitude     float64
bikeid                      int64
usertype                   object
birth year                float64
gender                    float64
postal code                object
dtype: object
```

[26]:
```
csv_merged_2020.isnull().sum()
```

[26]:
```
tripduration                    0
starttime                       0
stoptime                        0
start station id                0
start station name              0
start station latitude          0
start station longitude         0
end station id                  0
end station name                0
end station latitude            0
end station longitude           0
bikeid                          0
usertype                        0
birth year                1657472
gender                    1657472
postal code                560202
dtype: int64
```

[27]:
```
'''
dropping postal code from all of them since we'll be collecting them later
and also they're incomplete here.

csv_merged_2019 don't have postal code.
'''

frames = [csv_merged_2020, csv_merged_2021, csv_merged_2022]

for x in (frames):
    x = x.drop('postal code', axis = 1, inplace = True)
```

[28]:
```
# comparing datatypes

csv_merged_2019.dtypes == csv_merged_2020.dtypes
```

[28]:
```
tripduration               True
starttime                  True
stoptime                   True
start station id           True
start station name         True
start station latitude     True
start station longitude    True
end station id             True
end station name           True
end station latitude       True
end station longitude      True
bikeid                     True
usertype                   True
```

```
birth year                 False
gender                     False
dtype: bool
```

[29]:
```
# need to convert birth year and gender datatypes from float64 to int64 for␣
↪csv_merged_2020

csv_merged_2020.head()
```

[29]:
```
   tripduration                 starttime                  stoptime  \
0           478  2020-01-01 00:04:05.8090  2020-01-01 00:12:04.2370   \
1           363  2020-01-01 00:04:45.6990  2020-01-01 00:10:49.0400
2           284  2020-01-01 00:06:07.0630  2020-01-01 00:10:51.9240
3           193  2020-01-01 00:06:13.8550  2020-01-01 00:09:27.8320
4           428  2020-01-01 00:07:25.2950  2020-01-01 00:14:33.7800

   start station id                          start station name  \
0               366                             Broadway T Stop   \
1               219                  Boston East - 126 Border St
2               219                  Boston East - 126 Border St
3               396                        Main St at Beacon St
4                60  Charles Circle - Charles St at Cambridge St

   start station latitude  start station longitude  end station id  \
0               42.342781               -71.057473              93   \
1               42.373312               -71.041020             212
2               42.373312               -71.041020             212
3               42.409330               -71.063819             387
4               42.360793               -71.071190              49

              end station name  end station latitude  end station longitude  \
0              JFK/UMass T Stop             42.320340             -71.051180   \
1  Maverick Square - Lewis Mall             42.368844             -71.039778
2  Maverick Square - Lewis Mall             42.368844             -71.039778
3       Norman St at Kelvin St              42.409859             -71.066319
4      Stuart St at Charles St              42.351146             -71.066289

   bikeid    usertype  birth year  gender
0    6005    Customer      1969.0     0.0
1    3168  Subscriber      2000.0     1.0
2    3985  Subscriber      2001.0     1.0
3    2692  Subscriber      1978.0     1.0
4    4978  Subscriber      1987.0     1.0
```

[30]:
```
csv_merged_2020['birth year'] = csv_merged_2020['birth year'].fillna(0).
↪astype(np.int64) # 0=unknown
```

[31]:
```
csv_merged_2020['gender'] = csv_merged_2020['gender'].fillna(0).astype(np.int64)
```

[32]:
```
csv_merged_2020['gender'].unique() # 0=unknown; 1=male; 2=female
```

[32]:
```
array([0, 1, 2], dtype=int64)
```

[33]:
```
# comparing datatypes again

csv_merged_2019.dtypes == csv_merged_2020.dtypes
```

[33]:
```
tripduration               True
starttime                  True
stoptime                   True
start station id           True
start station name         True
start station latitude     True
start station longitude    True
end station id             True
end station name           True
end station latitude       True
end station longitude      True
bikeid                     True
usertype                   True
birth year                 True
gender                     True
dtype: bool
```

[34]:
```
'''
now we've got two extra features 'birth year' and 'gender' which are present in␣
 ↪only 2019 & 2020 data.
we can extract their age from subtracting birth year from starttime year.

Let's have a look at the starttime column for one.

'''
```

[34]:
```
"\nnow we've got two extra features 'birth year' and 'gender' which are present
in only 2019 & 2020 data.\nwe can extract their age from subtracting birth year
from starttime year.\n\nLet's have a look at the starttime column for one.\n\n"
```

[35]:
```
'''

we only need to split the string and get the year out of the value and then␣
 ↪subtract it from birthyear
if it's zero(0), we're not gonna change that.
```

```
Note: We can use 2019 by default to use here but adding this functionality for␣
 ↪redundancy if we use other data later.

'''
csv_merged_2019['starttime'].head()
```

[35]:
```
0    2019-01-01 00:09:13.7980
1    2019-01-01 00:33:56.1820
2    2019-01-01 00:41:54.6000
3    2019-01-01 00:43:32.5710
4    2019-01-01 00:49:56.4640
Name: starttime, dtype: object
```

[36]:
```
csv_merged_2019['birth year'].unique()
```

[36]:
```
array([1987, 1990, 1977, 1993, 1979, 1969, 1991, 1989, 1973, 1992, 1988,
       1972, 1998, 1997, 1994, 1982, 1995, 1984, 1956, 1966, 1986, 1946,
       1958, 1999, 1965, 1981, 1980, 1985, 1962, 1959, 1976, 1983, 1975,
       1974, 1978, 1964, 1963, 1960, 1967, 1953, 1955, 1961, 1947, 1996,
       1957, 1968, 1971, 2000, 1954, 1952, 1950, 2001, 1970, 1948, 2002,
       1951, 1949, 1943, 1942, 1941, 1944, 1945, 1929, 1937, 1888, 1939,
       1932, 1923, 1907, 1940, 1904, 2003, 1935, 1886, 1901, 1900, 1933,
       1938, 1936, 1889, 1915, 1917], dtype=int64)
```

[37]:
```
csv_merged_2020['birth year'].unique()
```

[37]:
```
array([1969, 2000, 2001, 1978, 1987, 1989, 1990, 1980, 1994, 1972, 1991,
       1993, 1988, 1985, 1977, 1997, 1992, 1975, 1984, 1967, 1996, 1995,
       1986, 1982, 1979, 1976, 1965, 1959, 1983, 1999, 1964, 1973, 1958,
       1998, 1968, 1962, 1955, 1961, 1966, 1981, 1970, 1971, 1963, 1957,
       1960, 1953, 1946, 1974, 1952, 1950, 1944, 1956, 2002, 1954, 1951,
       1949, 1947, 1938, 1941, 1900, 1948, 2003, 1937, 1943, 1940, 1942,
       1904, 1901, 1907, 2004, 1911, 1889, 1945, 1939, 1888, 1935, 1932,
          0], dtype=int64)
```

[38]:
```
## calc -> csv_merged_2019['starttime'].str[:4].astype(np.int64) -␣
 ↪csv_merged_2019['birth year']

csv_merged_2019.loc[csv_merged_2019['birth year'] > 0, 'age'] =␣
 ↪(csv_merged_2019['starttime'].str[:4].astype(np.int64) -␣
 ↪csv_merged_2019['birth year'])

csv_merged_2019['age'] = csv_merged_2019['age'].fillna(0).astype(np.int64)
```

[39]:
```
csv_merged_2019['age'].unique()
```

[39]:
```
array([ 32,  29,  42,  26,  40,  50,  28,  30,  46,  27,  31,  47,  21,
        22,  25,  37,  24,  35,  63,  53,  33,  73,  61,  20,  54,  38,
        39,  34,  57,  60,  43,  36,  44,  45,  41,  55,  56,  59,  52,
        66,  64,  58,  72,  23,  62,  51,  48,  19,  65,  67,  69,  18,
        49,  71,  17,  68,  70,  76,  77,  78,  75,  74,  90,  82, 131,
        80,  87,  96, 112,  79, 115,  16,  84, 133, 118, 119,  86,  81,
        83, 130, 104, 102], dtype=int64)
```

[40]:
```
csv_merged_2020.loc[csv_merged_2020['birth year'] > 0, 'age'] =␣
↪(csv_merged_2020['starttime'].str[:4].astype(np.int64) -␣
↪csv_merged_2020['birth year'])

csv_merged_2020['age'] = csv_merged_2020['age'].fillna(0).astype(np.int64)
```

[41]:
```
csv_merged_2019['age'].unique()
```

[41]:
```
array([ 32,  29,  42,  26,  40,  50,  28,  30,  46,  27,  31,  47,  21,
        22,  25,  37,  24,  35,  63,  53,  33,  73,  61,  20,  54,  38,
        39,  34,  57,  60,  43,  36,  44,  45,  41,  55,  56,  59,  52,
        66,  64,  58,  72,  23,  62,  51,  48,  19,  65,  67,  69,  18,
        49,  71,  17,  68,  70,  76,  77,  78,  75,  74,  90,  82, 131,
        80,  87,  96, 112,  79, 115,  16,  84, 133, 118, 119,  86,  81,
        83, 130, 104, 102], dtype=int64)
```

[42]:
```
# dropping birth year

frames = [csv_merged_2019, csv_merged_2020]

for x in (frames):
    x = x.drop('birth year', axis = 1, inplace = True)
```

[43]:
```
# comparing datatypes again

csv_merged_2019.dtypes == csv_merged_2020.dtypes
```

[43]:
```
tripduration               True
starttime                  True
stoptime                   True
start station id           True
start station name         True
start station latitude     True
start station longitude    True
end station id             True
end station name           True
end station latitude       True
end station longitude      True
bikeid                     True
```

```
usertype                   True
gender                     True
age                        True
dtype: bool
```

[44]:
```
# we need to fill 2021 & 2022 dfs with 0 for 'gender' and 'age' with 0

csv_merged_2021['gender'] = 0
csv_merged_2021['age'] = 0

csv_merged_2022['gender'] = 0
csv_merged_2022['age'] = 0
```

[45]:
```
csv_merged_2022.head(2)
```

[45]:
```
   tripduration                starttime                 stoptime
0           597  2022-01-01 00:00:25.1660  2022-01-01 00:10:22.1920  \
1           411  2022-01-01 00:00:40.4300  2022-01-01 00:07:32.1980

   start station id              start station name  start station latitude
0               178  MIT Pacific St at Purrington St               42.359573  \
1               189                          Kendall T               42.362428

   start station longitude  end station id
0               -71.101295              74  \
1               -71.084955             178

                     end station name  end station latitude
0  Harvard Square at Mass Ave/ Dunster             42.373268  \
1      MIT Pacific St at Purrington St             42.359573

   end station longitude  bikeid    usertype  gender  age
0             -71.118579    4923  Subscriber       0    0
1             -71.101295    3112  Subscriber       0    0
```

[46]:
```
csv_merged_2021.dtypes == csv_merged_2022.dtypes
```

[46]:
```
tripduration               True
starttime                  True
stoptime                   True
start station id           True
start station name         True
start station latitude     True
start station longitude    True
end station id             True
end station name           True
end station latitude       True
```

```
end station longitude      True
bikeid                     True
usertype                   True
gender                     True
age                        True
dtype: bool
```

[47]:
```
csv_merged_2022.dtypes == csv_merged_2019.dtypes
```

[47]:
```
tripduration               True
starttime                  True
stoptime                   True
start station id           True
start station name         True
start station latitude     True
start station longitude    True
end station id             True
end station name           True
end station latitude       True
end station longitude      True
bikeid                     True
usertype                   True
gender                     True
age                        True
dtype: bool
```

[48]:
```
# since they all seem fine, combining all the frames

frames = [csv_merged_2019, csv_merged_2020, csv_merged_2021, csv_merged_2022]
trips_data = pd.concat(frames)
```

[49]:
```
# deleting unused dfs
del [[csv_merged_2019, csv_merged_2020, csv_merged_2021, csv_merged_2022]]
```

[50]:
```
trips_data.shape
```

[50]:
```
(11287878, 15)
```

[51]:
```
trips_data.head()
```

[51]:
```
   tripduration                 starttime                  stoptime  \
0           371  2019-01-01 00:09:13.7980  2019-01-01 00:15:25.3360   \
1           264  2019-01-01 00:33:56.1820  2019-01-01 00:38:20.8800
2           458  2019-01-01 00:41:54.6000  2019-01-01 00:49:33.2730
3           364  2019-01-01 00:43:32.5710  2019-01-01 00:49:37.4260
4           681  2019-01-01 00:49:56.4640  2019-01-01 01:01:17.7010

   start station id                          start station name
```

```
0                80     MIT Stata Center at Vassar St / Main St  \
1               117                         Binney St / Sixth St
2                68       Central Square at Mass Ave / Essex St
3                89  Harvard Law School at Mass Ave / Jarvis St
4                73     Harvard Square at Brattle St / Eliot St

   start station latitude  start station longitude  end station id
0               42.362131               -71.091156             179  \
1               42.366162               -71.086883             189
2               42.365070               -71.103100              96
3               42.379011               -71.119945             334
4               42.373231               -71.120886             367

                                  end station name  end station latitude
0                                    MIT Vassar St             42.355601  \
1                                        Kendall T             42.362428
2  Cambridge Main Library at Broadway / Trowbridg…             42.373379
3                         Mass Ave at Hadley/Walden             42.391210
4                         Vassal Lane at Tobin/VLUS             42.383932

   end station longitude  bikeid    usertype  gender  age
0             -71.103945    3689  Subscriber       1   32
1             -71.084955    4142  Subscriber       1   29
2             -71.111075    1628  Subscriber       1   42
3             -71.122608    2969  Subscriber       1   26
4             -71.139613    3469  Subscriber       2   40
```

[52]:
```
# creating the station data for removing extra cols from tripdata

stations_st = trips_data[['start station id', 'start station name', 'start␣
 ↪station latitude', 'start station longitude']]
stations_en = trips_data[['end station id', 'end station name', 'end station␣
 ↪latitude', 'end station longitude']]
```

[53]:
```
# renaming cols

stations_st.rename(columns = {'start station id' : 'id',
                              'start station name' : 'name',
                              'start station latitude' : 'lat',
                              'start station longitude' : 'long'}, inplace =␣
 ↪True)
stations_st.head()
```

[53]:
```
    id                                    name        lat       long
0   80     MIT Stata Center at Vassar St / Main St  42.362131 -71.091156
1  117                       Binney St / Sixth St  42.366162 -71.086883
2   68       Central Square at Mass Ave / Essex St  42.365070 -71.103100
```

```
3   89  Harvard Law School at Mass Ave / Jarvis St  42.379011 -71.119945
4   73     Harvard Square at Brattle St / Eliot St  42.373231 -71.120886
```

[54]:
```
# renaming cols

stations_en.rename(columns = {'end station id' : 'id',
                              'end station name' : 'name',
                              'end station latitude' : 'lat',
                              'end station longitude' : 'long'}, inplace = True)
stations_en.head()
```

[54]:
```
    id                                          name        lat  \
0  179                                MIT Vassar St  42.355601   \
1  189                                    Kendall T  42.362428
2   96  Cambridge Main Library at Broadway / Trowbridg…  42.373379
3  334                          Mass Ave at Hadley/Walden  42.391210
4  367                          Vassal Lane at Tobin/VLUS  42.383932

        long
0 -71.103945
1 -71.084955
2 -71.111075
3 -71.122608
4 -71.139613
```

[55]:
```
frames2 = [stations_st, stations_en]
result2 = pd.concat(frames2)
```

[56]:
```
# all station data

result2.shape
```

[56]:
```
(22575756, 4)
```

[57]:
```
# removing duplicates

stations = result2.drop_duplicates()
```

[58]:
```
stations.shape
```

[58]:
```
(621, 4)
```

[59]:
```
stations.head()
```

[59]:
```
    id                                    name        lat       long
0   80     MIT Stata Center at Vassar St / Main St  42.362131 -71.091156
1  117                      Binney St / Sixth St  42.366162 -71.086883
2   68       Central Square at Mass Ave / Essex St  42.365070 -71.103100
```

```
3   89  Harvard Law School at Mass Ave / Jarvis St  42.379011 -71.119945
4   73     Harvard Square at Brattle St / Eliot St  42.373231 -71.120886
```

[60]:
```
trips_data.head()
```

[60]:
```
   tripduration                 starttime                  stoptime
0           371  2019-01-01 00:09:13.7980  2019-01-01 00:15:25.3360  \
1           264  2019-01-01 00:33:56.1820  2019-01-01 00:38:20.8800
2           458  2019-01-01 00:41:54.6000  2019-01-01 00:49:33.2730
3           364  2019-01-01 00:43:32.5710  2019-01-01 00:49:37.4260
4           681  2019-01-01 00:49:56.4640  2019-01-01 01:01:17.7010

   start station id                          start station name
0                80     MIT Stata Center at Vassar St / Main St  \
1               117                          Binney St / Sixth St
2                68       Central Square at Mass Ave / Essex St
3                89  Harvard Law School at Mass Ave / Jarvis St
4                73     Harvard Square at Brattle St / Eliot St

   start station latitude  start station longitude  end station id
0               42.362131               -71.091156             179  \
1               42.366162               -71.086883             189
2               42.365070               -71.103100              96
3               42.379011               -71.119945             334
4               42.373231               -71.120886             367

                                  end station name  end station latitude
0                                    MIT Vassar St             42.355601  \
1                                        Kendall T             42.362428
2  Cambridge Main Library at Broadway / Trowbridg…             42.373379
3                          Mass Ave at Hadley/Walden             42.391210
4                          Vassal Lane at Tobin/VLUS             42.383932

   end station longitude  bikeid    usertype  gender  age
0             -71.103945    3689  Subscriber       1   32
1             -71.084955    4142  Subscriber       1   29
2             -71.111075    1628  Subscriber       1   42
3             -71.122608    2969  Subscriber       1   26
4             -71.139613    3469  Subscriber       2   40
```

[61]:
```
# since we've already seperated data
# dropping unnecessary columns

trips = trips_data.drop(['start station name',
                         'start station latitude',
                         'start station longitude',
                         'end station name',
```

```
                          'end station latitude',
                          'end station longitude'], axis=1)

trips.head()
```

[61]:
```
   tripduration                 starttime                  stoptime
0           371  2019-01-01 00:09:13.7980  2019-01-01 00:15:25.3360  \
1           264  2019-01-01 00:33:56.1820  2019-01-01 00:38:20.8800
2           458  2019-01-01 00:41:54.6000  2019-01-01 00:49:33.2730
3           364  2019-01-01 00:43:32.5710  2019-01-01 00:49:37.4260
4           681  2019-01-01 00:49:56.4640  2019-01-01 01:01:17.7010

   start station id  end station id  bikeid    usertype  gender  age
0                80             179    3689  Subscriber       1   32
1               117             189    4142  Subscriber       1   29
2                68              96    1628  Subscriber       1   42
3                89             334    2969  Subscriber       1   26
4                73             367    3469  Subscriber       2   40
```

[62]:
```
trips['usertype'].unique()
```

[62]:
```
array(['Subscriber', 'Customer'], dtype=object)
```

[63]:
```
# encoding Subscriber as 1 and Customer or Casual as 0

trips['usertype'] = trips['usertype'].map({'Subscriber': 1, 'Customer': 0})
```

[64]:
```
trips.dtypes
```

[64]:
```
tripduration         int64
starttime           object
stoptime            object
start station id     int64
end station id       int64
bikeid               int64
usertype             int64
gender               int64
age                  int64
dtype: object
```

[65]:
```
# renaming cols for tableau

trips.rename(columns = {'tripduration' : 'duration',
                        'starttime' : 'st_time',
                        'stoptime' : 'en_time',
                        'start station id' : 'st_id',
                        'end station id' : 'en_id',
```

```
                          'bikeid' : 'bike_id',
                          'usertype' : 'sub_status',
                          'gender' : 'gen',
                          'age': 'age'}, inplace = True)
```

[66]:
```
trips.isnull().sum()    # no null values
```

[66]:
```
duration      0
st_time       0
en_time       0
st_id         0
en_id         0
bike_id       0
sub_status    0
gen           0
age           0
dtype: int64
```

[67]:
```
stations.isnull().sum()    # double checking
```

[67]:
```
id      0
name    0
lat     0
long    0
dtype: int64
```

[68]:
```
trips.shape
```

[68]:
```
(11287878, 9)
```

[69]:
```
trips = trips.reset_index(drop=True)
```

[70]:
```
trips['trip_seq'] = (trips.index+1)
```

[71]:
```
trips.tail()
```

[71]:

	duration	st_time	en_time	st_id
11287873	1901	2022-12-31 23:49:10.5980	2023-01-01 00:20:51.6930	186
11287874	360	2022-12-31 23:51:51.7430	2022-12-31 23:57:52.2170	12
11287875	958	2022-12-31 23:53:42.9690	2023-01-01 00:09:41.0970	49
11287876	995	2022-12-31 23:58:11.1980	2023-01-01 00:14:46.7870	39
11287877	969	2022-12-31 23:58:30.1460	2023-01-01 00:14:39.4400	39

	en_id	bike_id	sub_status	gen	age	trip_seq
11287873	136	4485	0	0	0	11287874
11287874	417	6912	1	0	0	11287875
11287875	16	7898	0	0	0	11287876
11287876	43	8330	1	0	0	11287877

```
11287877     43     3886           1     0     0  11287878
```

[72]:
```
# generate the uuid

import uuid

ids = {trip_seq: str(uuid.uuid4()) for trip_seq in trips['trip_seq'].unique()}

trips['trip_uuid'] = trips['trip_seq'].map(ids)
```

[73]:
```
trips.sample(n = 10)
```

[73]:

	duration	st_time	en_time	st_id	en_id	bike_id	sub_status	gen	age	trip_seq	trip_uuid
4910476	281	2021-04-09 00:49:47.9370	2021-04-09 00:54:28.9800	70	180	2188	1	0	0	4910477	e336a4c2-2b8d-4994-aaf7-690bb98e8efc
584901	3311	2019-05-24 15:06:37.5380	2019-05-24 16:01:48.8230	11	151	3707	0	0	50	584902	d0ded141-8f9d-4ae2-bc4c-07a438254de8
8457724	646	2022-05-26 12:42:14.8720	2022-05-26 12:53:00.9310	12	342	2292	0	0	0	8457725	2e9b9b24-1bb8-4ce3-bcad-a9fa854ae3d0
4198041	396	2020-10-09 23:04:49.1730	2020-10-09 23:11:25.6130	9	446	3793	1	0	0	4198042	4fde4e4e-cbee-47ee-905e-512eb14e1b93
8279401	1374	2022-05-11 13:35:45.6820	2022-05-11 13:58:40.0310	3	35	3052	1	0	0	8279402	2074980f-1c0c-49d3-a11f-a3bc558558b6
5933009	379	2021-07-30 23:21:52.8040	2021-07-30 23:28:12.3340	380	386	2529	1	0	0	5933010	1ad166b1-20ce-4761-ae17-b7885f221a53
7453308	962	2021-12-10 23:54:28.7240	2021-12-11 00:10:31.1330	381	109	3518	0	0	0	7453309	b85458e9-0b59-4fb1-9d5d-a59cf410df54
7040371	2230	2021-10-22 15:14:45.8390	2021-10-22 15:51:56.3970	60	60	5306	1	0	0	7040372	ff641a99-f4f0-4393-af4e-ce86641c1bcc
3066038	784	2020-06-01 16:43:57.4950	2020-06-01 16:57:02.4410	113	24	2500	0	0	0	3066039	a09dd04c-6a3c-430f-9468-b8b25e9480a1
6082242	175	2021-08-13 01:52:56.1450	2021-08-13 01:55:51.4510	183	414	2368	1	0	0	6082243	623d764d-d45d-4ed9-9606-2650cfee90d6

[74]:
```
trips.tail()
```

[74]:

	duration	st_time	en_time	st_id	\
11287873	1901	2022-12-31 23:49:10.5980	2023-01-01 00:20:51.6930	186	\
11287874	360	2022-12-31 23:51:51.7430	2022-12-31 23:57:52.2170	12	
11287875	958	2022-12-31 23:53:42.9690	2023-01-01 00:09:41.0970	49	
11287876	995	2022-12-31 23:58:11.1980	2023-01-01 00:14:46.7870	39	
11287877	969	2022-12-31 23:58:30.1460	2023-01-01 00:14:39.4400	39	

	en_id	bike_id	sub_status	gen	age	trip_seq	\
11287873	136	4485	0	0	0	11287874	\
11287874	417	6912	1	0	0	11287875	
11287875	16	7898	0	0	0	11287876	
11287876	43	8330	1	0	0	11287877	
11287877	43	3886	1	0	0	11287878	

	trip_uuid
11287873	ce7d93aa-5dcf-4891-90d6-55f8c0895349
11287874	12345243-8ffe-46cf-9385-8bddda645ce9
11287875	17e6c8c6-4185-406e-9417-0d39b107aac6
11287876	469c5a14-60d7-42d1-a39e-69374493a7b8
11287877	2cca87b7-606b-4543-920d-5c6de29f59c0

[75]:
```
# path for tableau data

tableau_path = "tableau/"

if not os.path.isdir(tableau_path):
    os.mkdir(tableau_path)
```

[76]:
```
print('Tripdata Ready')
```

```
Tripdata Ready
```

[77]:
```
# saving the tripdata

trips.to_csv(tableau_path + 'bluebikes_trips.csv', index=False)
```

[78]:
```
print('Tripdata Saved')
```

```
Tripdata Saved
```

[79]:
```
'''
geocoding stations data
since we only have the latitude and longitude information,
we can leverage that to get out other relevant features

'''
```

[79]:
```
'\ngeocoding stations data\nsince we only have the latitude and longitude
information,\nwe can leverage that to get out other relevant features\n\n'
```

[80]:
```
stations.shape # 621 unique stations
```

[80]:
```
(621, 4)
```

[81]:
```
# removing rows with invalid coordinates as that will cause issue with the api␣
↪input

stations = stations[stations.lat != 0.0]
stations = stations[stations.long != 0.0]
```

[82]:
```
stations
```

[82]:
```
           id                                                name        lat
0          80           MIT Stata Center at Vassar St / Main St  42.362131  \
1         117                             Binney St / Sixth St  42.366162
2          68            Central Square at Mass Ave / Essex St  42.365070
3          89       Harvard Law School at Mass Ave / Jarvis St  42.379011
4          73          Harvard Square at Brattle St / Eliot St  42.373231
…         …                                                  …         …
3570393   591       515 Somerville Ave (Temp. Winter Location)  42.383227
3583154   590    John Ahern Field at Kennedy-Longfellow School  42.369036
3655034    42  Boylston St at Charles St (Temp Winter Location)  42.352585
1343117   164                                Warehouse Lab PBSC  42.386455
1824917   438                      Mobile Temporary Station 1  42.351478

              long
0       -71.091156
1       -71.086883
2       -71.103100
3       -71.119945
4       -71.120886
…              …
3570393 -71.106069
3583154 -71.086310
3655034 -71.067703
1343117 -71.075420
1824917 -71.044162

[617 rows x 4 columns]
```

[83]:
```
stations = stations.reset_index(drop=True)
```

[84]:
```
stations.shape
```

[84]:
```
(617, 4)
```

[85]:
```
# converting to lat-long

gd = stations[['lat', 'long']]
gd.tail()
```

[85]:
```
           lat       long
612  42.383227 -71.106069
613  42.369036 -71.086310
614  42.352585 -71.067703
615  42.386455 -71.075420
616  42.351478 -71.044162
```

[86]:
```
gd.dtypes
```

[86]:
```
lat     float64
long    float64
dtype: object
```

[87]:
```
gd.shape
```

[87]:
```
(617, 2)
```

[88]:
```
# using geocoder to reverse geocode the pair data
# link: https://www.geocod.io/docs/
# !pip install pygeocodio
```

[89]:
```
# geocodio api

from geocodio import GeocodioClient
client = GeocodioClient('e3e81ac7f4ff374648fa83f97af19561e5de596', timeout=300)
```

[90]:
```
coor_list = gd.values.tolist() # converting to list obj
```

[91]:
```
len(coor_list)
```

[91]:
```
617
```

[92]:
```
firstThree = coor_list[:3]
firstThree
```

[92]:
```
[[42.3621312344991, -71.09115600585936],
 [42.36616223459919, -71.08688293667001],
 [42.36507, -71.1031]]
```

[93]:
```
'''
this will generate a json response containing the suitable responses for each␣
 ↪lat-long pair
'''
```

```
locations = client.reverse(coor_list)   # reverse geocoder
```

[94]:
```
import json

jsonResponse = json.dumps(locations)
```

[95]:
```
df_j = pd.read_json(jsonResponse, orient ='index.address_components') #␣
 ↪address_components in parser from the raw data
```

[96]:
```
# creating a b

nf = pd.DataFrame(columns=['formatted_address',
                           'accuracy',
                           'accuracy_type',
                           'source',
                           'address_components.number',
                           'address_components.street',
                           'address_components.suffix',
                           'address_components.formatted_street',
                           'address_components.city',
                           'address_components.county',
                           'address_components.state',
                           'address_components.zip',
                           'address_components.country',
                           'location.lat',
                           'location.lng'])
```

[97]:
```
nf
```

[97]:
```
Empty DataFrame
Columns: [formatted_address, accuracy, accuracy_type, source,
address_components.number, address_components.street, address_components.suffix,
address_components.formatted_street, address_components.city,
address_components.county, address_components.state, address_components.zip,
address_components.country, location.lat, location.lng]
Index: []
```

[98]:
```
# df -> for each lat-long pair, found data
# nf -> new df which contains the best row out of the returned pair by getting␣
 ↪max(accuracy)

# appending them to create a single df which is going to be merged with␣
 ↪'station' df

for x in range(len(gd)):
    df = pd.json_normalize(df_j.results[x])
    nf = pd.concat([nf, df.loc[df["accuracy"].idxmax()].to_frame().T])
```

[99]:
```
nf = nf.reset_index(drop=True)
```

[100]:
```
nf
```

[100]:
```
                              formatted_address accuracy accuracy_type
0              43 Vassar St, Cambridge, MA 02139        1       rooftop  \
1             290 Binney St, Cambridge, MA 02142        1       rooftop
2      605 Massachusetts Ave, Cambridge, MA 02139        1       rooftop
3     1585 Massachusetts Ave, Cambridge, MA 02138        1       rooftop
4             34 Brattle St, Cambridge, MA 02138        1       rooftop
..                                           …      …           …
612     518 Somerville Ave, Somerville, MA 02143        1       rooftop
613          259 Charles St, Cambridge, MA 02141      1.0       rooftop
614            220 Boylston St, Boston, MA 02116      1.0       rooftop
615        10 Dorrance St, Charlestown, MA 02129        1       rooftop
616            81 Northern Ave, Boston, MA 02210        1       rooftop

                                          source
0                                 City of Cambridge  \
1                                 City of Cambridge
2                                 City of Cambridge
3                                 City of Cambridge
4                                 City of Cambridge
..                                              …
612  Office of Geographic Information (MassGIS), Co…
613  Office of Geographic Information (MassGIS), Co…
614  Office of Geographic Information (MassGIS), Co…
615  Office of Geographic Information (MassGIS), Co…
616  Office of Geographic Information (MassGIS), Co…

    address_components.number address_components.street
0                          43                    Vassar  \
1                         290                    Binney
2                         605             Massachusetts
3                        1585             Massachusetts
4                          34                   Brattle
..                        …                         …
612                       518                Somerville
613                       259                   Charles
614                       220                  Boylston
615                        10                  Dorrance
616                        81                  Northern

    address_components.suffix address_components.formatted_street
0                          St                           Vassar St  \
1                          St                           Binney St
2                         Ave                   Massachusetts Ave
```

```
3                         Ave                   Massachusetts Ave
4                          St                          Brattle St
..                        …                                   …
612                       Ave                      Somerville Ave
613                        St                          Charles St
614                        St                         Boylston St
615                        St                         Dorrance St
616                       Ave                       Northern Ave

    address_components.city address_components.county
0                 Cambridge          Middlesex County  \
1                 Cambridge          Middlesex County
2                 Cambridge          Middlesex County
3                 Cambridge          Middlesex County
4                 Cambridge          Middlesex County
..                      …                       …
612              Somerville          Middlesex County
613               Cambridge          Middlesex County
614                  Boston            Suffolk County
615             Charlestown            Suffolk County
616                  Boston            Suffolk County

    address_components.state address_components.zip
0                         MA                  02139  \
1                         MA                  02142
2                         MA                  02139
3                         MA                  02138
4                         MA                  02138
..                        …                       …
612                       MA                  02143
613                       MA                  02141
614                       MA                  02116
615                       MA                  02129
616                       MA                  02210

    address_components.country location.lat location.lng
0                           US    42.362123   -71.091429  \
1                           US    42.365994   -71.087136
2                           US    42.365159   -71.102976
3                           US    42.379122    -71.11974
4                           US    42.373254   -71.120955
..                         …           …           …
612                         US    42.382989   -71.106153
613                         US    42.368807   -71.086708
614                         US    42.352303   -71.068042
615                         US    42.386557   -71.075731
616                         US    42.351513   -71.044189
```

```
    address_components.predirectional address_components.prefix
0                                  NaN                       NaN  \
1                                  NaN                       NaN
2                                  NaN                       NaN
3                                  NaN                       NaN
4                                  NaN                       NaN
..                                 …                         …
612                                NaN                       NaN
613                                NaN                       NaN
614                                NaN                       NaN
615                                NaN                       NaN
616                                NaN                       NaN

    address_components.postdirectional
0                                  NaN
1                                  NaN
2                                  NaN
3                                  NaN
4                                  NaN
..                                 …
612                                NaN
613                                NaN
614                                NaN
615                                NaN
616                                NaN

[617 rows x 18 columns]
```

[101]:
```
station_data = pd.concat([stations, nf], axis = 1)
```

[102]:
```
list(station_data.columns.values)
```

[102]:
```
['id',
 'name',
 'lat',
 'long',
 'formatted_address',
 'accuracy',
 'accuracy_type',
 'source',
 'address_components.number',
 'address_components.street',
 'address_components.suffix',
 'address_components.formatted_street',
 'address_components.city',
 'address_components.county',
```

```
 'address_components.state',
 'address_components.zip',
 'address_components.country',
 'location.lat',
 'location.lng',
 'address_components.predirectional',
 'address_components.prefix',
 'address_components.postdirectional']
```

[103]:
```
# checking accuracy

station_data['accuracy'].unique()
```

[103]:
```
array([1, 0.99, 0.98], dtype=object)
```

[104]:
```
# dropping unnecessary data cols

bluebikes_stations = station_data.drop(['formatted_address',
                                        'accuracy',
                                        'accuracy_type',
                                        'source',
                                        'address_components.number',
                                        'address_components.street',
                                        'address_components.suffix',
                                        'address_components.country',
                                        'location.lat',
                                        'location.lng',
                                        'address_components.predirectional',
                                        'address_components.prefix',
                                        'address_components.postdirectional'],␣
 ↪axis = 1)
```

[105]:
```
# renaming cols for tableau

bluebikes_stations.rename(columns = {'address_components.formatted_street' :␣
 ↪'street',
                                     'address_components.city' : 'city',
                                     'address_components.county' : 'county',
                                     'address_components.state' : 'state',
                                     'address_components.zip' : 'zip'}, inplace␣
 ↪= True)
```

[106]:
```
print('Stationdata Ready')
```

```
Stationdata Ready
```

[107]:
```
bluebikes_stations.tail()
```

[107]:
```
       id                                              name        lat  \
612  591        515 Somerville Ave (Temp. Winter Location)  42.383227
613  590     John Ahern Field at Kennedy-Longfellow School  42.369036
614   42  Boylston St at Charles St (Temp Winter Location)  42.352585
615  164                                Warehouse Lab PBSC  42.386455
616  438                        Mobile Temporary Station 1  42.351478

          long          street         city            county state    zip
612 -71.106069  Somerville Ave   Somerville  Middlesex County    MA  02143
613 -71.086310      Charles St    Cambridge  Middlesex County    MA  02141
614 -71.067703     Boylston St       Boston    Suffolk County    MA  02116
615 -71.075420     Dorrance St  Charlestown    Suffolk County    MA  02129
616 -71.044162    Northern Ave       Boston    Suffolk County    MA  02210
```

[108]:
```
bluebikes_stations.shape
```

[108]:
```
(617, 9)
```

[109]:
```
# saving the stations

bluebikes_stations.to_csv(tableau_path + 'bluebikes_stations.csv', index=False)
```

[110]:
```
print('Stationdata Saved')
```

```
Stationdata Saved
```

[ ]:
```

```