Extension Project will address Bluebikes correlation with Bus performance.

With this, we will be using Bluebikes station and trip datasets along with previous data about bus given in Bus performance project document.

Subproblem 1: Worst on-time Performance Routes correlation with Bluebikes usage

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from collections import defaultdict
import os

folder_path = "../../data/"
blue_bikes_trips = pd.read_csv(f"{folder_path}/202201-bluebikes-
tripdata.csv")
blue_bikes_stations =
pd.read_csv(f"{folder_path}/current_bluebikes_stations.csv")
census_neighbourhood = pd.read_csv(f"{folder_path}/Census-Boston-
Neighborhood.csv")
mbta_gtfs = pd.read_csv(f"{folder_path}/MBTA_Systemwide_GTFS_Map.csv")
mbta_reliability =
pd.read_csv(f"{folder_path}/MBTA_Bus_Reliability.csv")
mbta_prediction_accuracy =
pd.read_csv(f"{folder_path}/Bus_Prediction_Accuracy.csv")
mbta_bus_ridership =
pd.read_csv(f"{folder_path}/MBTA_Bus_Ridership.csv")
```

```
C:\Users\sataa\AppData\Local\Temp\ipykernel_5364\437775067.py:8:
DtypeWarning: Columns (2,3) have mixed types. Specify dtype option on
import or set low_memory=False.
  mbta_bus_ridership =
pd.read_csv(f"{folder_path}/MBTA_Bus_Ridership.csv")
```

```
def process_blue_bikes_trips(blue_bikes_trips):
      processed_blue_bikes_trips = blue_bikes_trips
      processed_blue_bikes_trips["tripduration"] =
blue_bikes_trips["tripduration"] / 60
      return processed_blue_bikes_trips

processed_blue_bikes_trips =
process_blue_bikes_trips(blue_bikes_trips)
processed_blue_bikes_trips.head()
```

```
   tripduration                 starttime                  stoptime  \
0      9.950000  2022-01-01 00:00:25.1660  2022-01-01 00:10:22.1920
1      6.850000  2022-01-01 00:00:40.4300  2022-01-01 00:07:32.1980
2      7.933333  2022-01-01 00:00:54.8180  2022-01-01 00:08:51.6680
3      7.766667  2022-01-01 00:01:01.6080  2022-01-01 00:08:48.2350
```

```
4     12.533333  2022-01-01 00:01:06.0520  2022-01-01 00:13:38.2300

   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
2                 94             Main St at Austin St
42.375603
3                 94             Main St at Austin St
42.375603
4                 19            Park Dr at Buswell St
42.347241

   start station longitude  end station id  \
0               -71.101295              74
1               -71.084955             178
2               -71.064608             356
3               -71.064608             356
4               -71.105301              41

                                    end station name  end station
latitude  \
0                Harvard Square at Mass Ave/ Dunster
42.373268
1                    MIT Pacific St at Purrington St
42.359573
2                              Charlestown Navy Yard
42.374125
3                              Charlestown Navy Yard
42.374125
4  Packard's Corner - Commonwealth Ave at Brighto...
42.352261

   end station longitude  bikeid    usertype postal code
0             -71.118579    4923  Subscriber       02139
1             -71.101295    3112  Subscriber       02139
2             -71.054812    6901    Customer       02124
3             -71.054812    5214    Customer       02124
4             -71.123831    2214  Subscriber       02215

# print(blue_bikes_stations.head())

def process_blue_bikes_stations(blue_bikes_stations,
blue_bikes_trips):
    # Fixing first row as the column names
    new_column_names = blue_bikes_stations.iloc[0]  # Get the first
row to use as column names
    blue_bikes_stations.columns = new_column_names  # Set new column
```

```
names
     blue_bikes_stations = 
blue_bikes_stations.iloc[1:].reset_index(drop=True)

     # Extracting the unique start station names and IDs
     start_stations = blue_bikes_trips[['start station id', 'start 
station name']].drop_duplicates()
     start_stations = start_stations.rename(columns={'start station 
id': 'station_id', 'start station name': 'station_name'})

     # Extracting the unique end station names and IDs.
     end_stations = blue_bikes_trips[['end station id', 'end station 
name']].drop_duplicates()
     end_stations = end_stations.rename(columns={'end station id': 
'station_id', 'end station name': 'station_name'})

     # Combining the start and end station information.
     combined_stations = pd.concat([start_stations, 
end_stations]).drop_duplicates().set_index('station_name')
     blue_bikes_stations['station_id'] = 
blue_bikes_stations['Name'].map(combined_stations['station_id'])
     blue_bikes_stations = blue_bikes_stations.dropna(subset = 
["station_id"])
     return blue_bikes_stations

processed_blue_bikes_stations = 
process_blue_bikes_stations(blue_bikes_stations, 
processed_blue_bikes_trips)
processed_blue_bikes_stations.head()
```

```
0  Number                               Name     Latitude   
Longitude  \
0  K32015                    1200 Beacon St  42.34414899  -
71.11467361   
1  W32006                       160 Arsenal  42.36466403  -
71.17569387   
2  A32019                  175 N Harvard St  42.36447457  -
71.12840831   
3  S32035                     191 Beacon St  42.38032335  -
71.10878613   
4  C32094  2 Hummingbird Lane at Olmsted Green     42.28887    -
71.095003   

0    District Public Total docks Deployment Year  station_id  
0   Brookline    Yes           1            2021       452.0  
1   Watertown    Yes          11            2021       502.0  
2      Boston    Yes          17            2014       149.0  
3  Somerville    Yes          19            2018       378.0  
4      Boston    Yes          17            2020       493.0  
```

```
census_neighbourhood.head()
```

```
    tract20_nbhd P0020001     P0020005
P0020006  \
0  field concept   Total:  White alone  Black or African American
alone
1        Allston    24904        12536
1326
2       Back Bay    18190        13065
690
3    Beacon Hill     9336         7521
252
4       Brighton    52047        32694
2414

             P0020002
P002aapi  \
0  Hispanic or Latino  Asian, Native Hawaiian and Pacific Islander
al...
1                3259
6271
2                1208
2410
3                 537
630
4                5376
8703

                                 P002others P0040001     P0040005  \
0  Other Races or Multiple Races,  all ages   Total:  White alone
1                                      1512    23140        11976
2                                       817    17042        12349
3                                       396     8603         6980
4                                      2860    47657        30752

                          P0040006  ...  \
0  Black or African American alone  ...
1                             1184  ...
2                              641  ...
3                              231  ...
4                             2076  ...

                                            P0050005  \
0  Nursing facilities/Skilled-nursing facilities
1                                                 0
2                                               269
3                                                 0
4                                               266

                         P0050006                          P0050007  \
0  Other institutional facilities  Noninstitutionalized population:
1                               0                               3281
2                               0                               1610
3                               0                                 33
4                              56                               3796

                              P0050008          P0050009  \
0  College/University student housing  Military quarters
1                                3214                  0
2                                1487                  0
3                                   0                  0
4                                3493                  0

                            P0050010 H0010001  H0010002 H0010003  \
0  Other noninstitutional facilities    Total:  Occupied    Vacant
1                                 67    10748     10027      721
2                                123    11524     10006     1518
3                                 33     6037      5485      552
4                                303    23653     22535     1118

           hhsize
0  household size
1     2.156477511
2     1.630121927
3     1.696080219
4     2.126292434

[5 rows x 34 columns]
```

```
mbta_gtfs.head()

def process_gtfs(MBTA_data):
     MBTA_data = MBTA_data[MBTA_data['Neighborhood'].notnull()]
     MBTA_data = MBTA_data[MBTA_data['Routes'] != '#N/A']
     MBTA_data = MBTA_data[MBTA_data['Routes'].notnull()]

     # Split routes column to separate routes
     MBTA_data['Routes'] = MBTA_data['Routes'].str.split('|')
     MBTA_data = MBTA_data.explode('Routes')
     df = MBTA_data[["stop_id", "stop_name", "stop_lat", "stop_lon",
"Neighborhood", "Routes"]]

     return df

processed_mbta_gtfs = process_gtfs(mbta_gtfs)
processed_mbta_gtfs.head()
```

```
  stop_id                     stop_name   stop_lat   stop_lon
Neighborhood  \
0       1  Washington St opp Ruggles St  42.330957 -71.082754
```

```
Roxbury
0        1  Washington St opp Ruggles St  42.330957 -71.082754
Roxbury
0        1  Washington St opp Ruggles St  42.330957 -71.082754
Roxbury
0        1  Washington St opp Ruggles St  42.330957 -71.082754
Roxbury
0        1  Washington St opp Ruggles St  42.330957 -71.082754
Roxbury

  Routes
0      1
0      8
0     10
0     47
0     19
```

```
mbta_prediction_accuracy.head()
```

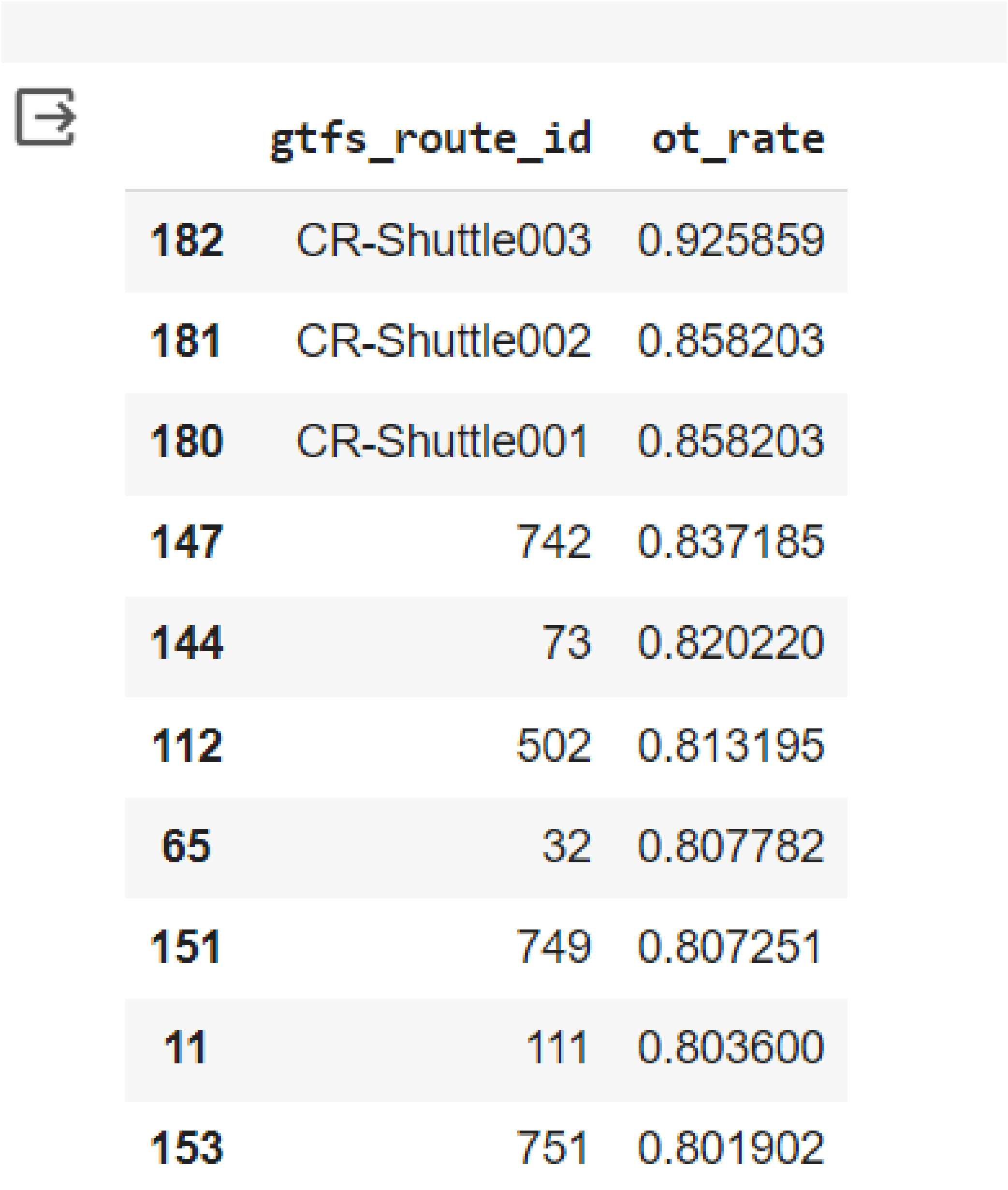
```
                   weekly mode route_id        bin
arrival_departure  \
0  2021/08/13 04:00:00+00  bus       NaN    0-3 min         departure

1  2021/08/13 04:00:00+00  bus       NaN    3-6 min         departure

2  2021/08/13 04:00:00+00  bus       NaN   6-12 min         departure

3  2021/08/13 04:00:00+00  bus       NaN  12-30 min         departure

4  2021/08/20 04:00:00+00  bus       NaN    0-3 min         departure


   num_predictions  num_accurate_predictions  ObjectId
0           293039                    233562         1
1           285817                    229090         2
2           561098                    472923         3
3          1594830                   1405620         4
4           285591                    228653         5
```

```
mbta_reliability.head()

# Code taken from Base Question 2 code
def process_reliability(df):
      new_df = df[df["mode_type"]=="Bus"] # taking only buses
      new_df = new_df.dropna(subset=['otp_denominator',
'otp_numerator','cancelled_numerator']) # No NaN / Null
      new_df['ot_rate'] =
new_df['otp_numerator']/new_df['otp_denominator']
      grouped_route = new_df.groupby('gtfs_route_id')
      grouped_rate = grouped_route['ot_rate'].mean().reset_index()
      rate_sorted = grouped_rate.sort_values(by='ot_rate',
ascending=False)
      return rate_sorted


reliability_rate_sorted = process_reliability(mbta_reliability)

reliability_rate_sorted.head() # best ot_rate
reliability_rate_sorted.tail() # worst ot_rate
```

```
    gtfs_route_id   ot_rate
150           747  0.458202
106           459  0.429970
99            448  0.406302
100           449  0.402552
178          9703  0.320094
```

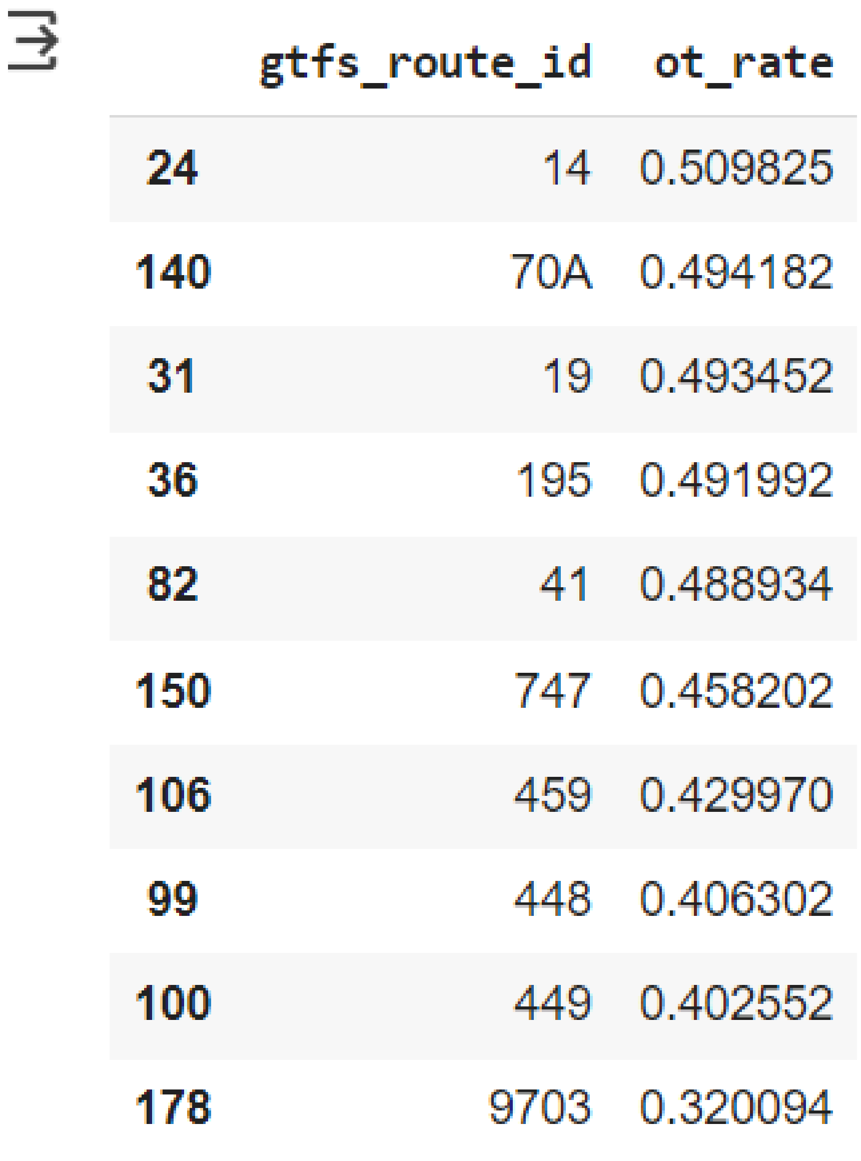
We have the best and worst on-time performance data extracted from base question 2 - Utilizes the MBTA Reliability Dataset:

Best 10:



	gtfs_route_id	ot_rate
182	CR-Shuttle003	0.925859
181	CR-Shuttle002	0.858203
180	CR-Shuttle001	0.858203
147	742	0.837185
144	73	0.820220
112	502	0.813195
65	32	0.807782
151	749	0.807251
11	111	0.803600
153	751	0.801902

Worst 10:



	gtfs_route_id	ot_rate
24	14	0.509825
140	70A	0.494182
31	19	0.493452
36	195	0.491992
82	41	0.488934
150	747	0.458202
106	459	0.429970
99	448	0.406302
100	449	0.402552
178	9703	0.320094

```
merged_data_on_routes = pd.merge(processed_mbta_gtfs,
reliability_rate_sorted, left_on = "Routes", right_on =
"gtfs_route_id")

print(merged_data_on_routes['gtfs_route_id'].isna().sum()) # checking
no bus routes are not included in the relability dataset.
print(merged_data_on_routes['Routes'].isna().sum()) # checking no bus
routes are not included in the GTFS dataset.

merged_data_on_routes.head()

0
0
```

```
  stop_id                          stop_name   stop_lat   stop_lon  \
0       1       Washington St opp Ruggles St  42.330957 -71.082754
1   10003           Albany St opp Randall St  42.331591 -71.076237
2   10100             Albany St @ Randall St  42.331675 -71.076347
3   10101    Melnea Cass Blvd @ Harrison Ave  42.332066 -71.079147
4   10590  Massachusetts Ave @ Washington St  42.336621 -71.076956

  Neighborhood Routes gtfs_route_id   ot_rate
0      Roxbury      1             1  0.744301
1      Roxbury      1             1  0.744301
2      Roxbury      1             1  0.744301
3      Roxbury      1             1  0.744301
4    South End      1             1  0.744301
```

```
# Group by 'Routes'
grouped_by_routes = merged_data_on_routes.groupby('Routes')

grouped_by_routes.head()

# # Aggregate 'ot_rate' for each route, then sort to find the worst 10
# # Assuming 'worst' means the highest values
worst_routes =
grouped_by_routes['ot_rate'].mean().sort_values(ascending=True).head(1
0)

# # Print the worst 10 routes based on ot_rate
print(worst_routes.head())
```

```
Routes
9703    0.320094
449     0.402552
448     0.406302
459     0.429970
747     0.458202
Name: ot_rate, dtype: float64
```

```
worst_routes_loc = pd.merge(worst_routes, merged_data_on_routes,
left_on = ["Routes", "ot_rate"], right_on = ["Routes", "ot_rate"])
worst_routes_loc.rename(columns={"Routes": "route"}, inplace=True)
worst_routes_loc.head(25) # rows are per stop, so showing more rows
ensures the visibility of other routes here beyond route 9703
# print(worst_routes_loc.shape)
```

```
   route   ot_rate stop_id
stop_name  \
0   9703  0.320094    1111                        Cambridge St opp Hano
St
1   9703  0.320094    1112                       Cambridge St @ Harvard
St
2   9703  0.320094    1113                        Cambridge St @ Linden
St
```

```
3   9703  0.320094    1114                    Cambridge St @ N Harvard
St
4   9703  0.320094   11388                  Huntington Ave @ Belvidere
St
5   9703  0.320094    1257                       Tremont St @ Prentiss
St
6   9703  0.320094    1258          Tremont St @ Roxbury Crossing
Station
7   9703  0.320094    1260                    Columbus Ave @ New Cedar
St
8   9703  0.320094    1262                        Columbus Ave @ Heath
St
9   9703  0.320094    1784                    Ruggles St @ Huntington
Ave
10  9703  0.320094    1785                   Ruggles St @ Annunciation
Rd
11  9703  0.320094   31391               Huntington Ave @ Gainsborough
St
12  9703  0.320094   41391                      Huntington Ave @ Opera
Pl
13  9703  0.320094   61391                   Huntington Ave @ Forsyth
Way
14  9703  0.320094   71391                Huntington Ave @ Louis Prang
St
15  9703  0.320094     922                     Cambridge St opp Dustin
St
16  9703  0.320094     924                       Cambridge St @ Gordon
St
17  9703  0.320094     925                      Cambridge St @ Barrows
St
18   448  0.406302   16535                            Otis St @ Summer
St
19   448  0.406302    4727                 McClellan Highway @ Addison
St
20   448  0.406302    4728                McClellan Highway @ Boardman
St
21   448  0.406302    6535                    Franklin St @ Devonshire
St
22   448  0.406302    6564  Summer St @ South Station - Red Line
entrance
23   448  0.406302    7094                  Terminal C - Departures
Level
24   448  0.406302     892                       Summer St @ Atlantic
Ave

     stop_lat   stop_lon  Neighborhood gtfs_route_id
0   42.353931 -71.136365       Allston          9703
1   42.355641 -71.132361       Allston          9703
2   42.355943 -71.131448       Allston          9703
```

```
3   42.357758 -71.126505       Allston          9703
4   42.345344 -71.082045      Back Bay          9703
5   42.332930 -71.092638       Roxbury          9703
6   42.331311 -71.094831       Roxbury          9703
7   42.328067 -71.097310       Roxbury          9703
8   42.325028 -71.098483       Roxbury          9703
9   42.337416 -71.095079  Mission Hill          9703
10  42.336729 -71.093223  Mission Hill          9703
11  42.341443 -71.086788        Fenway          9703
12  42.340553 -71.088908        Fenway          9703
13  42.339219 -71.092168        Fenway          9703
14  42.337684 -71.096046        Fenway          9703
15  42.350692 -71.145688       Allston          9703
16  42.352276 -71.140761       Allston          9703
17  42.353091 -71.138430       Allston          9703
18  42.354243 -71.058557      Downtown           448
19  42.386142 -71.019171   East Boston           448
20  42.391562 -71.012888   East Boston           448
21  42.355521 -71.057253      Downtown           448
22  42.352253 -71.054774      Downtown           448
23  42.366635 -71.017167   East Boston           448
24  42.352480 -71.054849      Downtown           448
```

From here, we will be comparing locations of bus stations of the worst routes and the locations of bluebikes going along those routes. We will then see the average number of rides in that station.

```python
# This formula is used to take distances between locations (using
longitude and latitude)

def haversine(lon1, lat1, lon2, lat2):
    R = 6371  # Earth radius in km
    dlon = np.radians(lon2 - lon1)
    dlat = np.radians(lat2 - lat1)
    a = np.sin(dlat/2)**2 + np.cos(np.radians(lat1)) *
np.cos(np.radians(lat2)) * np.sin(dlon/2)**2
    c = 2 * np.arctan2(np.sqrt(a), np.sqrt(1-a))
    distance = R * c
    return distance


# This formula will be used to test if bluebikes stations are within a
10 minute walk away from any of the worst route stops.
def no_more_than_x_mins(distance, x):
    max_walking_distance = x / 60  * 5 # assuming a walking speed of 5
km/h.
    return distance <= max_walking_distance
```

```python
MAX_WALKING_DISTANCE = 10 # in minutes

close_blue_bikes_list = defaultdict(list)
# Comparing the locations:
for _, bus_stop in worst_routes_loc.iterrows():
    # Extract latitude and longitude for the bus stop
    bus_stop_lat, bus_stop_lon = float(bus_stop['stop_lat']),
float(bus_stop['stop_lon'])

    # Iterate through each blue bike station
    for _, bike_station in processed_blue_bikes_stations.iterrows():
        # Extract latitude and longitude for the bike station
        bike_station_lat, bike_station_lon =
float(bike_station['Latitude']), float(bike_station['Longitude'])

        # Calculate the distance between the bus stop and the bike
station
        distance = haversine(bus_stop_lon, bus_stop_lat,
bike_station_lon, bike_station_lat)

        if no_more_than_x_mins(distance, MAX_WALKING_DISTANCE):
            if (bike_station['station_id'] not in
close_blue_bikes_list[bus_stop["route"]]): # taking only the distinct
stops

close_blue_bikes_list[bus_stop["route"]].append(bike_station["station_
id"])



# Reverse the dictionary
bike_to_bus_station = {bike_station: bus_station for bus_station,
bike_stations in close_blue_bikes_list.items() for bike_station in
bike_stations}

# Filter the DataFrame
filtered_blue_bikes_stations =
processed_blue_bikes_stations[processed_blue_bikes_stations['station_i
d'].isin(bike_to_bus_station.keys())]


# Add the new column for bus_station_id
filtered_blue_bikes_stations['route_id'] =
filtered_blue_bikes_stations['station_id'].map(bike_to_bus_station)
```

```
C:\Users\sataa\AppData\Local\Temp\ipykernel_5364\2799738196.py:9:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead
```

```
See the caveats in the documentation:
https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#
returning-a-view-versus-a-copy
  filtered_blue_bikes_stations['route_id'] =
filtered_blue_bikes_stations['station_id'].map(bike_to_bus_station)
```

```
filtered_station_ids = set(filtered_blue_bikes_stations['station_id'])

# Filter the processed_blue_bikes_trips DataFrame
# Keep rows where either the start or end station is in the list of
filtered station IDs
filtered_trips = processed_blue_bikes_trips[
    processed_blue_bikes_trips['start station
id'].isin(filtered_station_ids) |
    processed_blue_bikes_trips['end station
id'].isin(filtered_station_ids)
]

print(filtered_trips.head())
```

```
    tripduration                 starttime
stoptime  \
4      12.533333  2022-01-01 00:01:06.0520  2022-01-01 00:13:38.2300

6      16.383333  2022-01-01 00:01:24.7490  2022-01-01 00:17:48.1850

8      13.950000  2022-01-01 00:02:30.2650  2022-01-01 00:16:27.6270

10     13.350000  2022-01-01 00:03:23.9650  2022-01-01 00:16:45.4150

11     12.483333  2022-01-01 00:03:37.2630  2022-01-01 00:16:07.1390


    start station id                           start station name  \
4                 19                           Park Dr at Buswell St
6                 36  Copley Square - Dartmouth St at Boylston St
8                 60  Charles Circle - Charles St at Cambridge St
10                60  Charles Circle - Charles St at Cambridge St
11                 4                        Tremont St at E Berkeley St

    start station latitude  start station longitude  end station id  \
4                42.347241               -71.105301              41
6                42.349928               -71.077392              36
8                42.360793               -71.071190             363
10               42.360793               -71.071190             363
11               42.345392               -71.069616               4

                                     end station name  end station
latitude  \
4   Packard's Corner - Commonwealth Ave at Brighto...
42.352261
```

```
6         Copley Square - Dartmouth St at Boylston St
42.349928
8                         Harrison Ave at Mullins Way
42.345216
10                        Harrison Ave at Mullins Way
42.345216
11                        Tremont St at E Berkeley St
42.345392

    end station longitude  bikeid    usertype postal code
4               -71.123831    2214  Subscriber       02215
6               -71.077392    4683  Subscriber       02130
8               -71.063840    3431  Subscriber       02215
10              -71.063840    6614    Customer       02128
11              -71.069616    6588  Subscriber       02116
```

```
avg_trip_duration_start = filtered_trips.groupby('start station id')
['tripduration'].mean().reset_index()
avg_trip_duration_start.rename(columns={'start station id':
'station_id', 'tripduration': 'avg_start_duration'}, inplace=True)

# Calculate average trip duration for end stations
avg_trip_duration_end = filtered_trips.groupby('end station id')
['tripduration'].mean().reset_index()
avg_trip_duration_end.rename(columns={'end station id': 'station_id',
'tripduration': 'avg_end_duration'}, inplace=True)

# Merge the two dataframes on station_id
merged_avg_durations = pd.merge(avg_trip_duration_start,
avg_trip_duration_end, on='station_id', how='outer')

# Calculating the mean of the two averages, handling cases where one
might be NaN
merged_avg_durations['avg_trip_duration'] =
merged_avg_durations[['avg_start_duration',
'avg_end_duration']].mean(axis=1, skipna=True)

all_trip_averages = pd.merge(merged_avg_durations,
filtered_blue_bikes_stations)
print(all_trip_averages[["station_id", "avg_trip_duration",
"avg_start_duration", "avg_end_duration", "route_id"]])
```

```
     station_id  avg_trip_duration  avg_start_duration
avg_end_duration  \
0             3          17.033691           15.262563
18.804818
1             4          13.714921           14.297755
13.132086
2             8          26.393002           36.228563
16.557442
```

```
3             9          13.660425           12.675271
14.645580
4            10          12.328906           12.048364
12.609447
..          ...                ...                 ...                .
..
119         538          14.845833           13.483333
16.208333
120         544          11.534584           11.609878
11.459290
121         547          98.564286           97.273810
99.854762
122         548          27.911111           27.911111
NaN
123         554          13.976831           14.347641
13.606021

    route_id
0         19
1        701
2       9703
3        747
4        747
..       ...
119       14
120      747
121       14
122       14
123      701

[124 rows x 5 columns]

trip_duration_start = filtered_trips.groupby('start station id')
# trip_duration_start.rename(columns={'start station id':
'station_id', 'tripduration': 'trip_duration'}, inplace=True)

# Calculate average trip duration for end stations
trip_duration_end = filtered_trips.groupby('end station id')
# trip_duration_end.rename(columns={'end station id': 'station_id',
'tripduration': 'trip_duration'}, inplace=True)

print(trip_duration_start.head(), trip_duration_end.head())

       tripduration                 starttime
stoptime  \
4         12.533333  2022-01-01 00:01:06.0520  2022-01-01
00:13:38.2300
6         16.383333  2022-01-01 00:01:24.7490  2022-01-01
00:17:48.1850
8         13.950000  2022-01-01 00:02:30.2650  2022-01-01
```

```
00:16:27.6270
10        13.350000  2022-01-01 00:03:23.9650  2022-01-01
00:16:45.4150
11        12.483333  2022-01-01 00:03:37.2630  2022-01-01
00:16:07.1390
...             ...                       ...                       ..
.
75059     23.150000  2022-01-28 11:15:13.7100  2022-01-28
11:38:22.8430
77193     38.716667  2022-01-28 20:07:52.7650  2022-01-28
20:46:36.4570
77749     11.733333  2022-01-28 21:33:41.1780  2022-01-28
21:45:25.6760
79611     52.766667  2022-01-29 04:27:02.8610  2022-01-29
05:19:49.4420
81251     61.066667  2022-01-31 22:33:21.6430  2022-01-31
23:34:26.1770

       start station id                           start station
name  \
4                    19                       Park Dr at Buswell St

6                    36  Copley Square - Dartmouth St at Boylston St

8                    60  Charles Circle - Charles St at Cambridge St

10                   60  Charles Circle - Charles St at Cambridge St

11                    4                 Tremont St at E Berkeley St

...                 ...                                          ...

75059               441                             Sullivan Square

77193               540                    Sumner St at Shirley Ave

77749               214      Airport T Stop - Bremen St at Brooks St

79611               397                      Broadway at Beacham St

81251               236                           Assembly Square T


       start station latitude  start station longitude  end station id
\
4                   42.347241               -71.105301              41

6                   42.349928               -71.077392              36

8                   42.360793               -71.071190             363
```

```
10                   42.360793                -71.071190               363

11                   42.345392                -71.069616                 4

...                        ...                       ...               ...

75059                42.384452                -71.075149               157

77193                42.408337                -70.998048               217

77749                42.374111                -71.032772               355

79611                42.398361                -71.063738                12

81251                42.392233                -71.077466                51


                                        end station name  \
4      Packard's Corner - Commonwealth Ave at Brighto...
6            Copley Square - Dartmouth St at Boylston St
8                              Harrison Ave at Mullins Way
10                             Harrison Ave at Mullins Way
11                             Tremont St at E Berkeley St
...                                                    ...
75059                           Seaport Blvd at Sleeper St
77193  Orient Heights T Stop - Bennington St at Sarat...
77749                Bennington St at Constitution Beach
79611  Ruggles T Stop - Columbus Ave at Melnea Cass Blvd
81251                            Washington St at Lenox St

       end station latitude  end station longitude  bikeid    usertype
\
4                 42.352261             -71.123831    2214  Subscriber

6                 42.349928             -71.077392    4683  Subscriber

8                 42.345216             -71.063840    3431  Subscriber

10                42.345216             -71.063840    6614    Customer

11                42.345392             -71.069616    6588  Subscriber

...                     ...                    ...     ...         ...

75059             42.353178             -71.048174    2754  Subscriber

77193             42.386781             -71.006098    2328  Subscriber

77749             42.385224             -71.010631    6083  Subscriber

79611             42.336244             -71.087986    3482    Customer
```

```
81251             42.335099             -71.079038    3307  Subscriber


      postal code
4           02215
6           02130
8           02215
10          02128
11          02116
...           ...
75059       02451
77193         NaN
77749       02128
79611         NaN
81251       02118

[1393 rows x 14 columns]        tripduration                 starttime
stoptime  \
4         12.533333  2022-01-01 00:01:06.0520  2022-01-01
00:13:38.2300
6         16.383333  2022-01-01 00:01:24.7490  2022-01-01
00:17:48.1850
8         13.950000  2022-01-01 00:02:30.2650  2022-01-01
00:16:27.6270
10        13.350000  2022-01-01 00:03:23.9650  2022-01-01
00:16:45.4150
11        12.483333  2022-01-01 00:03:37.2630  2022-01-01
00:16:07.1390
...             ...                       ...                       ..
.
76635     37.666667  2022-01-28 18:17:04.8630  2022-01-28
18:54:45.3170
78188      8.216667  2022-01-28 22:27:30.0400  2022-01-28
22:35:43.6270
79363     17.416667  2022-01-29 02:09:05.8190  2022-01-29
02:26:31.3640
80613     54.333333  2022-01-31 19:32:57.5730  2022-01-31
20:27:18.0300
81505     28.683333  2022-01-31 23:28:37.5810  2022-01-31
23:57:18.8170

       start station id                          start station
name  \
4                    19                          Park Dr at Buswell St

6                    36  Copley Square - Dartmouth St at Boylston St

8                    60  Charles Circle - Charles St at Cambridge St
```

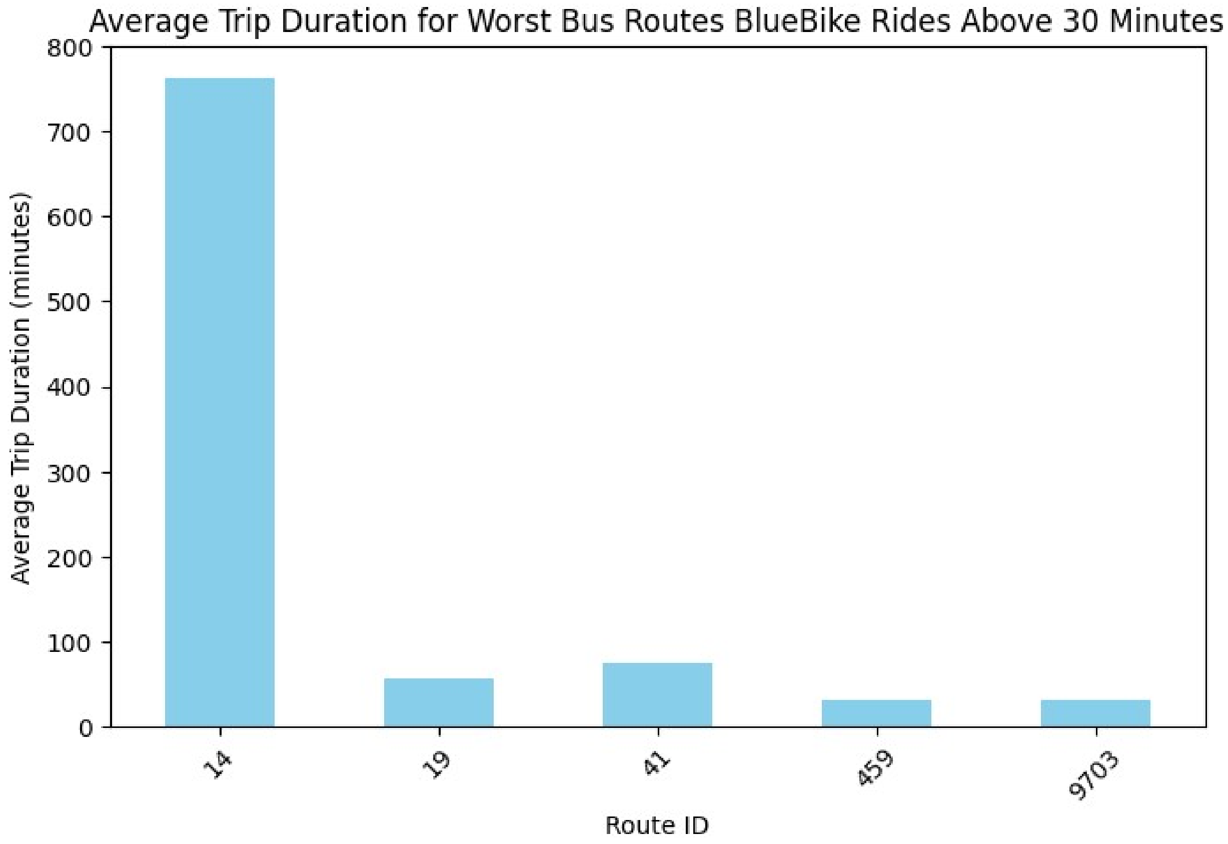
```
10                   60  Charles Circle - Charles St at Cambridge St
11                    4                  Tremont St at E Berkeley St
...                 ...                                          ...
76635               440                               Boston Landing
78188               210                    Bennington St at Byron St
79363               210                    Bennington St at Byron St
80613               370                   Dartmouth St at Newbury St
81505                47                       Cross St at Hanover St

       start station latitude  start station longitude  end station id  
\
4                   42.347241               -71.105301              41
6                   42.349928               -71.077392              36
8                   42.360793               -71.071190             363
10                  42.360793               -71.071190             363
11                  42.345392               -71.069616               4
...                       ...                      ...             ...
76635               42.356561               -71.141675             525
78188               42.383533               -71.016191             489
79363               42.383533               -71.016191             478
80613               42.350961               -71.077828             236
81505               42.362811               -71.056067             482

                                        end station name  \
4      Packard's Corner - Commonwealth Ave at Brighto...
6            Copley Square - Dartmouth St at Boylston St
8                            Harrison Ave at Mullins Way
10                           Harrison Ave at Mullins Way
11                           Tremont St at E Berkeley St
...                                                  ...
76635                               California at Chapel
78188                                             Day Sq
```

```
79363                              Gove St at Orleans St
80613                                  Assembly Square T
81505                                 Kearins Playground

       end station latitude  end station longitude  bikeid    usertype
\
4                 42.352261             -71.123831    2214  Subscriber

6                 42.349928             -71.077392    4683  Subscriber

8                 42.345216             -71.063840    3431  Subscriber

10                42.345216             -71.063840    6614    Customer

11                42.345392             -71.069616    6588  Subscriber

...                     ...                    ...     ...         ...

76635             42.365149             -71.202360    4138    Customer

78188             42.379295             -71.027733    7423  Subscriber

79363             42.370578             -71.035585    7590    Customer

80613             42.392233             -71.077466    3307  Subscriber

81505             42.406161             -71.060417    6843  Subscriber


      postal code
4           02215
6           02130
8           02215
10          02128
11          02116
...           ...
76635       02458
78188       02128
79363         NaN
80613       02118
81505       02149

[1423 rows x 14 columns]
```

```python
average_duration_by_route = all_trip_averages.groupby('route_id')
['avg_trip_duration'].mean()
print(average_duration_by_route.head())
```

```
route_id
14     762.125907
19      55.925891
```

```
41      74.703294
459     31.483486
701     17.881840
Name: avg_trip_duration, dtype: float64

# Filter to include only routes with an average duration above 60
minutes
routes_above_30 = average_duration_by_route[average_duration_by_route
> 30]

# Creating a bar graph for routes with average duration above 60
minutes
plt.figure(figsize=(8, 5))
routes_above_30.plot(kind='bar', color='skyblue')

plt.xlabel('Route ID')
plt.ylabel('Average Trip Duration (minutes)')
plt.title('Average Trip Duration for Worst Bus Routes BlueBike Rides
Above 30 Minutes')
plt.xticks(rotation=45)

plt.show()
```



```
print(len(close_blue_bikes_list["14"]),
len(close_blue_bikes_list["41"]))

29 21
```

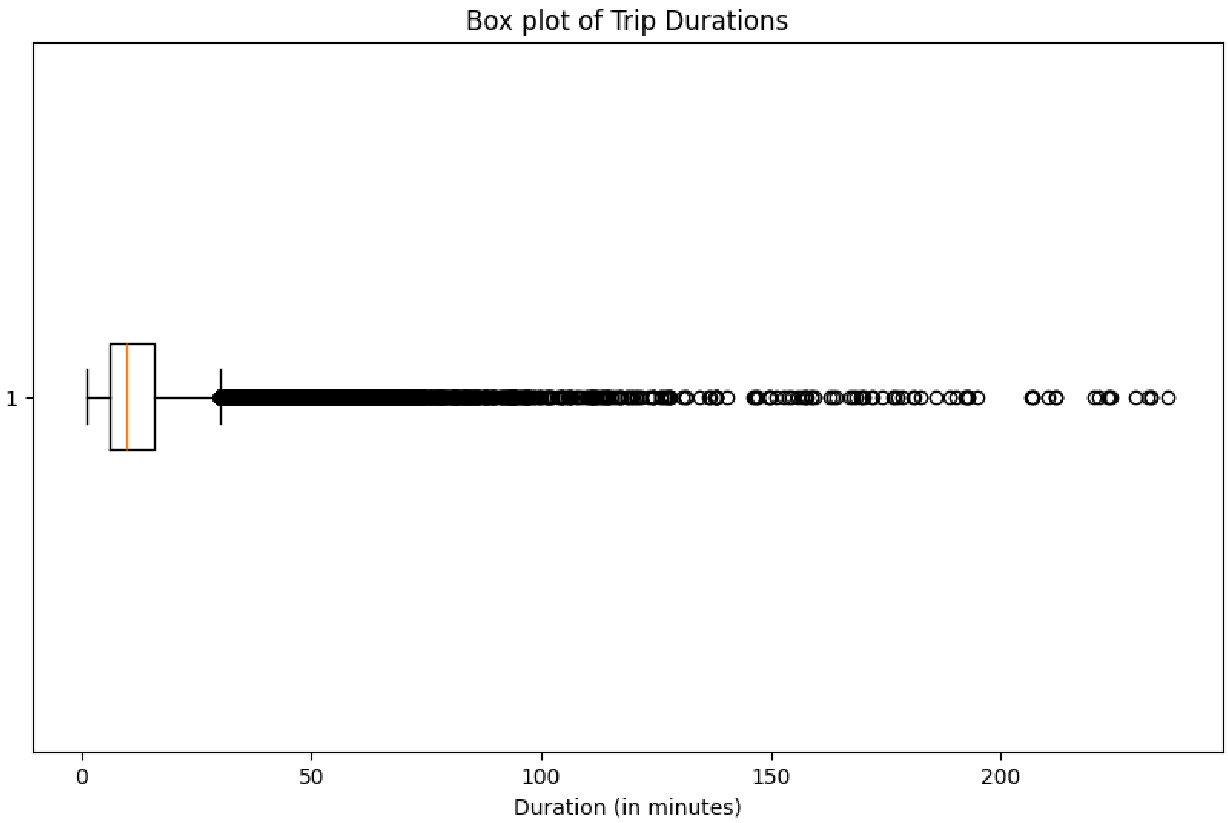
Despite the similarity in number of bluebike stations close by those two stations 41, and 14, we see that the average biking duration for route 14 is much much higher. This may be an indication of either of two things: Either the bus route servicing is subpar in the area, or there may be an abundance of explorers there.

```
start_station_trip_count = blue_bikes_trips.groupby('start station
id')["start station
id"].count().reset_index(name='start_station_trip_count')
end_station_trip_count = blue_bikes_trips.groupby('end station id')
["end station id"].count().reset_index(name='end_station_trip_count')

start_station_trip_count.rename(columns={'start station id':
'station_id'}, inplace=True)
end_station_trip_count.rename(columns={'end station id':
'station_id'}, inplace=True)

# Merge the two dataframes on 'station_id'
trip_counts = pd.merge(start_station_trip_count,
end_station_trip_count)
trip_counts.columns = ['start_station_trip_count',
'end_station_trip_count']
trip_counts["difference"] = trip_counts["end_station_trip_count"] -
trip_counts["start_station_trip_count"] # Negative means more stations
that people pick up bikes from.
print(trip_counts.head())

----------------------------------------------------------------------
-----
ValueError                                Traceback (most recent call
last)
c:\Users\sataa\OneDrive\Documents\Fall 2023\CS 506\Boston Transit
Performance Project\ds-boston-transit-performance\fa23-team-a\
Deliverables\deliverable-3\extension-project.ipynb Cell 20 line 9
      <a
href='vscode-notebook-cell:/c%3A/Users/sataa/OneDrive/Documents/Fall
%202023/CS%20506/Boston%20Transit%20Performance%20Project/ds-boston-
transit-performance/fa23-team-a/Deliverables/deliverable-3/extension-
project.ipynb#X31sZmlsZQ%3D%3D?line=6'>7</a> # Merge the two
dataframes on 'station_id'
      <a
href='vscode-notebook-cell:/c%3A/Users/sataa/OneDrive/Documents/Fall
%202023/CS%20506/Boston%20Transit%20Performance%20Project/ds-boston-
transit-performance/fa23-team-a/Deliverables/deliverable-3/extension-
project.ipynb#X31sZmlsZQ%3D%3D?line=7'>8</a> trip_counts =
pd.merge(start_station_trip_count, end_station_trip_count)
----> <a
href='vscode-notebook-cell:/c%3A/Users/sataa/OneDrive/Documents/Fall
%202023/CS%20506/Boston%20Transit%20Performance%20Project/ds-boston-
transit-performance/fa23-team-a/Deliverables/deliverable-3/extension-
project.ipynb#X31sZmlsZQ%3D%3D?line=8'>9</a> trip_counts.columns =
['start_station_trip_count', 'end_station_trip_count']
     <a
href='vscode-notebook-cell:/c%3A/Users/sataa/OneDrive/Documents/Fall
%202023/CS%20506/Boston%20Transit%20Performance%20Project/ds-boston-
transit-performance/fa23-team-a/Deliverables/deliverable-3/extension-
project.ipynb#X31sZmlsZQ%3D%3D?line=9'>10</a>
trip_counts["difference"] = trip_counts["end_station_trip_count"] -
trip_counts["start_station_trip_count"] # Negative means more stations
that people pick up bikes from.
     <a
href='vscode-notebook-cell:/c%3A/Users/sataa/OneDrive/Documents/Fall
%202023/CS%20506/Boston%20Transit%20Performance%20Project/ds-boston-
transit-performance/fa23-team-a/Deliverables/deliverable-3/extension-
project.ipynb#X31sZmlsZQ%3D%3D?line=10'>11</a>
print(trip_counts.head())

File c:\Users\sataa\.virtualenvs\cs506\Lib\site-packages\pandas\core\
generic.py:6218, in NDFrame.__setattr__(self, name, value)
   6216 try:
   6217     object.__getattribute__(self, name)
-> 6218     return object.__setattr__(self, name, value)
   6219 except AttributeError:
   6220     pass

File properties.pyx:69, in
pandas._libs.properties.AxisProperty.__set__()

File c:\Users\sataa\.virtualenvs\cs506\Lib\site-packages\pandas\core\
generic.py:767, in NDFrame._set_axis(self, axis, labels)
    762 """
    763 This is called from the cython code when we set the `index`
attribute
    764 directly, e.g. `series.index = [1, 2, 3]`.
    765 """
    766 labels = ensure_index(labels)
--> 767 self._mgr.set_axis(axis, labels)
    768 self._clear_item_cache()

File c:\Users\sataa\.virtualenvs\cs506\Lib\site-packages\pandas\core\
internals\managers.py:227, in BaseBlockManager.set_axis(self, axis,
new_labels)
    225 def set_axis(self, axis: AxisInt, new_labels: Index) -> None:
    226     # Caller is responsible for ensuring we have an Index
object.
```

```
--> 227     self._validate_set_axis(axis, new_labels)
    228     self.axes[axis] = new_labels

File c:\Users\sataa\.virtualenvs\cs506\Lib\site-packages\pandas\core\
internals\base.py:85, in DataManager._validate_set_axis(self, axis, 
new_labels)
     82     pass
     84 elif new_len != old_len:
---> 85     raise ValueError(
     86         f"Length mismatch: Expected axis has {old_len}
elements, new "
     87         f"values have {new_len} elements"
     88     )

ValueError: Length mismatch: Expected axis has 3 elements, new values
have 2 elements
```

```
# Reverse the dictionary
station_to_bus_route = {}
for route, stations in close_blue_bikes_list.items():
    for station in stations:
        if station not in station_to_bus_route:
            station_to_bus_route[station] = route

trip_counts['route_id'] = 
trip_counts['station_id'].map(station_to_bus_route)
print(trip_counts.head())

df = pd.DataFrame(processed_blue_bikes_trips)

df = df[df['tripduration'] <= 240]

print(processed_blue_bikes_trips.head())
# Create a box plot for the 'tripduration' column
plt.figure(figsize=(10, 6))
plt.boxplot(df['tripduration'], vert=False)  # 'vert=False' makes the
box plot horizontal
plt.title('Box plot of Trip Durations')
plt.xlabel('Duration (in minutes)')
plt.show()
```

```
(81449, 14)
   tripduration                 starttime                  stoptime  \
0      9.950000  2022-01-01 00:00:25.1660  2022-01-01 00:10:22.1920   
1      6.850000  2022-01-01 00:00:40.4300  2022-01-01 00:07:32.1980   
2      7.933333  2022-01-01 00:00:54.8180  2022-01-01 00:08:51.6680   
3      7.766667  2022-01-01 00:01:01.6080  2022-01-01 00:08:48.2350   
4     12.533333  2022-01-01 00:01:06.0520  2022-01-01 00:13:38.2300   

   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
2                 94             Main St at Austin St
42.375603
3                 94             Main St at Austin St
42.375603
4                 19            Park Dr at Buswell St
42.347241

   start station longitude  end station id  \
0               -71.101295              74
1               -71.084955             178
2               -71.064608             356
3               -71.064608             356
4               -71.105301              41

                                    end station name  end station
latitude  \
0                Harvard Square at Mass Ave/ Dunster
42.373268
1                    MIT Pacific St at Purrington St
42.359573
2                              Charlestown Navy Yard
42.374125
3                              Charlestown Navy Yard
42.374125
4  Packard's Corner - Commonwealth Ave at Brighto...
42.352261

   end station longitude  bikeid    usertype postal code
0             -71.118579    4923  Subscriber       02139
1             -71.101295    3112  Subscriber       02139
2             -71.054812    6901    Customer       02124
3             -71.054812    5214    Customer       02124
4             -71.123831    2214  Subscriber       02215
```


Box plot of Trip Durations
1
0
50
100
150
200
Duration (in minutes)