In [242]:

```
import pandas as pd
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns

import scipy.stats as stats
import statsmodels.api as sm
import pylab as py
from scipy.stats import boxcox
from scipy.stats import f_oneway
from scipy.stats import levene
```

In [213]:

```
pd.options.display.max_columns = 40
```

1. EDA

Problem Statement

Data from Delhivery. We need to sanitize, aggregate the data to get insights and prepare for forecasting. Aggregate to create 1 row for one trip. Currently, the data is at more granular level.

1.1 Basic Stats

In [3]:

```
df = pd.read_csv('../../../data/casestudy/5.delhivery_data.csv')
```

In [4]:

```
df.shape
```

Out[4]:

```
(144867, 24)
```

In [5]:

```
df.head()
```

Out[5]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	source_(
0	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
1	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
2	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
3	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
4	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812

In [6]:

```
df.dtypes.value_counts()
```

Out[6]:

```
object     12
float64    10
int64       1
bool        1
dtype: int64
```

In [7]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144867 entries, 0 to 144866
Data columns (total 24 columns):
 #   Column                          Non-Null Count   Dtype  
---  ------                          --------------   -----  
 0   data                            144867 non-null  object 
 1   trip_creation_time              144867 non-null  object 
 2   route_schedule_uuid             144867 non-null  object 
 3   route_type                      144867 non-null  object 
 4   trip_uuid                       144867 non-null  object 
 5   source_center                   144867 non-null  object 
 6   source_name                     144574 non-null  object 
 7   destination_center              144867 non-null  object 
 8   destination_name                144606 non-null  object 
 9   od_start_time                   144867 non-null  object 
 10  od_end_time                     144867 non-null  object 
 11  start_scan_to_end_scan          144867 non-null  float64
 12  is_cutoff                       144867 non-null  bool   
 13  cutoff_factor                   144867 non-null  int64  
 14  cutoff_timestamp                144867 non-null  object 
 15  actual_distance_to_destination  144867 non-null  float64
 16  actual_time                     144867 non-null  float64
 17  osrm_time                       144867 non-null  float64
 18  osrm_distance                   144867 non-null  float64
 19  factor                          144867 non-null  float64
 20  segment_actual_time             144867 non-null  float64
 21  segment_osrm_time               144867 non-null  float64
 22  segment_osrm_distance           144867 non-null  float64
 23  segment_factor                  144867 non-null  float64
dtypes: bool(1), float64(10), int64(1), object(12)
memory usage: 25.6+ MB
```

In [8]:

```
df.describe()
```

Out[8]:

ne	osrm_time	osrm_distance	factor	segment_actual_time	segment_osrm_time	segr
00	144867.000000	144867.000000	144867.000000	144867.000000	144867.000000	
27	213.868272	284.771297	2.120107	36.196111	18.507548	
21	308.011085	421.119294	1.715421	53.571158	14.775960	
00	6.000000	9.008200	0.144000	-244.000000	0.000000	
00	27.000000	29.914700	1.604264	20.000000	11.000000	
00	64.000000	78.525800	1.857143	29.000000	17.000000	
00	257.000000	343.193250	2.213483	40.000000	22.000000	
00	1686.000000	2326.199100	77.387097	3051.000000	1611.000000	

Outliers possible: difference in mean and median start_scan_to_end_scan, cutoff_factor, actual_distance_to_destination, actual_time, osrm_time, osrm_distance, segment_actual_time

In [9]:

```
df.describe(include='object')
```

Out[9]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	sc
count	144867	144867	144867	144867	144867	
unique	2	14817	1504	2	14817	
top	training	2018-09-24 05:12:53.848469	thanos::sroute:4029a8a2-6c74-4b7e-a6d8-f9e069f...	FTL	trip-153802363942560700	IN
freq	104858	101	1812	99660	101	

In [10]:

```
df.dtypes
```

Out[10]:

```
data                              object
trip_creation_time                object
route_schedule_uuid               object
route_type                        object
trip_uuid                         object
source_center                     object
source_name                       object
destination_center                object
destination_name                  object
od_start_time                     object
od_end_time                       object
start_scan_to_end_scan           float64
is_cutoff                           bool
cutoff_factor                      int64
cutoff_timestamp                  object
actual_distance_to_destination   float64
actual_time                      float64
osrm_time                        float64
osrm_distance                    float64
factor                           float64
segment_actual_time              float64
segment_osrm_time                float64
segment_osrm_distance            float64
segment_factor                   float64
dtype: object
```

In [11]:

```
df.select_dtypes(['object']).columns
```

Out[11]:

```
Index(['data', 'trip_creation_time', 'route_schedule_uuid', 'route_type',
       'trip_uuid', 'source_center', 'source_name', 'destination_center',
       'destination_name', 'od_start_time', 'od_end_time', 'cutoff_timestam
p'],
      dtype='object')
```

In [12]:

```
df.head()
```

Out[12]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	source_c
0	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
1	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
2	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
3	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
4	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812

In [13]:

```
obj_cols = ['data','route_schedule_uuid','route_type','trip_uuid','source_center','source_n
           'destination_name',]
date_cols = ['trip_creation_time','od_start_time','od_end_time','cutoff_timestamp']
num_cols = ['start_scan_to_end_scan', 'actual_distance_to_destination',
       'actual_time', 'osrm_time', 'osrm_distance', 'factor',
       'segment_actual_time', 'segment_osrm_time', 'segment_osrm_distance',
       'segment_factor']
bool_cols = ['is_cutoff']
```

In [14]:

```
for col in date_cols:
    df[col] = pd.to_datetime(df[col])
```

In [15]:

```
df.dtypes
```

Out[15]:

```
data                                      object
trip_creation_time                datetime64[ns]
route_schedule_uuid                       object
route_type                                object
trip_uuid                                 object
source_center                             object
source_name                               object
destination_center                        object
destination_name                          object
od_start_time                     datetime64[ns]
od_end_time                       datetime64[ns]
start_scan_to_end_scan                   float64
is_cutoff                                   bool
cutoff_factor                              int64
cutoff_timestamp                  datetime64[ns]
actual_distance_to_destination           float64
actual_time                              float64
osrm_time                                float64
osrm_distance                            float64
factor                                   float64
segment_actual_time                      float64
segment_osrm_time                        float64
segment_osrm_distance                    float64
segment_factor                           float64
dtype: object
```

In [16]:

```
for col in df.columns:
    print("***"*30)
    print(col)
    print(df[col].value_counts())
```

```
******************************************************************************
**************
data
training    104858
test         40009
Name: data, dtype: int64
******************************************************************************
**************
trip_creation_time
2018-09-24 05:12:53.848469    101
2018-09-25 04:21:12.551117    101
2018-09-29 05:04:57.639067    101
2018-09-22 04:55:04.835022    101
2018-09-19 04:07:34.091798    101
                             ...
2018-09-30 21:41:51.314395      1
2018-09-22 06:42:14.980815      1
2018-09-27 16:32:14.784918      1
2018-09-16 05:31:31.532090      1
2018-09-20 07:13:31.478665      1
Name: trip_creation_time, Length: 14817, dtype: int64
******************************************************************************
**************
route_schedule_uuid
thanos::sroute:4029a8a2-6c74-4b7e-a6d8-f9e069fbcea9    1812
thanos::sroute:0456b740-1dad-4929-bbe0-87d8843f5a10    1608
thanos::sroute:dca6268f-741a-4d1a-b1b0-aab13095a366    1605
thanos::sroute:a1b25549-1e77-498f-8538-00292e5bd5a2    1285
thanos::sroute:de5e208e-7641-45e6-8100-4d9fb1e5720d    1280
                                                       ...
thanos::sroute:0a05c445-d943-4f05-82ed-fd90a6d31e87       1
thanos::sroute:a2c15d09-9bd2-4d29-a3fb-3dbab548800a       1
thanos::sroute:d29fd731-9f1f-490c-922e-6d79d166db24       1
thanos::sroute:238d7712-f567-405b-bad5-874ba36fb3c4       1
thanos::sroute:036f372d-28d8-4d19-877c-6277077ad09e       1
Name: route_schedule_uuid, Length: 1504, dtype: int64
******************************************************************************
**************
route_type
FTL        99660
Carting    45207
Name: route_type, dtype: int64
******************************************************************************
**************
trip_uuid
trip-153802363942560700    101
trip-153793758186488532    101
trip-153819749763881430    101
trip-153854305492910872    101
trip-153741795740530104    101
                          ...
trip-153779228813052637      1
trip-153762302742584394      1
```

```
trip-153759657429809563      1
trip-153813029204130085      1
trip-153809265997799146      1
Name: trip_uuid, Length: 14817, dtype: int64
***************************************************************************
**************
source_center
IND000000ACB    23347
IND562132AAA     9975
IND421302AAG     9088
IND411033AAA     4061
IND501359AAE     3340
                ...
IND400072AAI        1
IND733202AAC        1
IND733202AAB        1
IND741101AAB        1
IND493445AAB        1
Name: source_center, Length: 1508, dtype: int64
***************************************************************************
**************
source_name
Gurgaon_Bilaspur_HB (Haryana)            23347
Bangalore_Nelmngla_H (Karnataka)          9975
Bhiwandi_Mankoli_HB (Maharashtra)         9088
Pune_Tathawde_H (Maharashtra)             4061
Hyderabad_Shamshbd_H (Telangana)          3340
                                         ...
Chikhli_KKndrDPP_D (Maharashtra)             1
Kasganj_BnkrGate_D (Uttar Pradesh)           1
Jetpur_DC (Gujarat)                          1
Islampure_ShbdnDPP_D (West Bengal)           1
Islampure_Central_DPP_2 (West Bengal)        1
Name: source_name, Length: 1498, dtype: int64
***************************************************************************
**************
destination_center
IND000000ACB    15192
IND562132AAA    11019
IND421302AAG     5492
IND501359AAE     5142
IND712311AAA     4892
                ...
IND686141AAA        1
IND396210AAA        1
IND421302AAF        1
IND221401AAA        1
IND761020AAA        1
Name: destination_center, Length: 1481, dtype: int64
***************************************************************************
**************
destination_name
Gurgaon_Bilaspur_HB (Haryana)          15192
Bangalore_Nelmngla_H (Karnataka)       11019
Bhiwandi_Mankoli_HB (Maharashtra)       5492
Hyderabad_Shamshbd_H (Telangana)        5142
Kolkata_Dankuni_HB (West Bengal)        4892
                                       ...
Bhadohi_Rajpura_D (Uttar Pradesh)          1
Ranaghat_ArickDPP_D (West Bengal)          1
Vaikom_KotyamRD_D (Kerala)                 1
```

```
Khatauli_TilakNgr_D (Uttar Pradesh)        1
Mumbai_Skynet_INT (Maharashtra)            1
Name: destination_name, Length: 1468, dtype: int64
****************************************************************************
**************
od_start_time
2018-09-21 18:37:09.322207    81
2018-09-13 04:57:25.708746    79
2018-09-15 06:03:01.496238    79
2018-09-21 06:51:50.532257    79
2018-09-26 05:33:10.899941    79
                              ..
2018-10-03 23:10:20.756105     1
2018-09-22 02:24:22.946198     1
2018-09-13 23:38:08.229837     1
2018-09-23 03:28:58.460080     1
2018-09-30 03:42:39.055097     1
Name: od_start_time, Length: 26369, dtype: int64
****************************************************************************
**************
od_end_time
2018-09-24 09:59:15.691618    81
2018-09-29 12:11:18.125562    79
2018-09-28 12:13:41.675546    79
2018-10-02 10:36:25.970169    79
2018-09-15 10:44:35.527660    79
                              ..
2018-09-26 08:02:10.953476     1
2018-09-13 05:49:21.813943     1
2018-09-13 03:40:53.355885     1
2018-09-18 08:48:09.325396     1
2018-09-15 10:52:18.794809     1
Name: od_end_time, Length: 26369, dtype: int64
****************************************************************************
**************
start_scan_to_end_scan
110.0     459
72.0      424
99.0      411
95.0      405
86.0      399
         ...
1336.0      1
1296.0      1
2045.0      1
1167.0      1
2701.0      1
Name: start_scan_to_end_scan, Length: 1915, dtype: int64
****************************************************************************
**************
is_cutoff
True     118749
False     26118
Name: is_cutoff, dtype: int64
****************************************************************************
**************
cutoff_factor
22      13157
9       12378
44       8334
18       8263
```

```
66       5795
        ...
368        1
240        1
310        1
1461       1
239        1
Name: cutoff_factor, Length: 501, dtype: int64
****************************************************************************
**************
cutoff_timestamp
2018-09-24 05:19:20    40
2018-09-24 05:19:21    33
2018-09-14 05:29:26    19
2018-09-24 07:21:24    18
2018-09-18 05:19:27    17
                       ..
2018-10-04 04:33:53     1
2018-09-19 21:40:11     1
2018-09-20 16:48:28     1
2018-09-17 08:10:30     1
2018-09-28 09:59:34     1
Name: cutoff_timestamp, Length: 93180, dtype: int64
****************************************************************************
**************
actual_distance_to_destination
18.036366      2
195.585266     2
19.574937      2
100.282892     2
25.757877      2
              ..
27.530666      1
9.606108       1
1540.042272    1
11.365110      1
903.265473     1
Name: actual_distance_to_destination, Length: 144515, dtype: int64
****************************************************************************
**************
actual_time
32.0      1443
36.0      1420
30.0      1350
38.0      1329
42.0      1241
          ...
2788.0       1
3031.0       1
3498.0       1
2896.0       1
3860.0       1
Name: actual_time, Length: 3182, dtype: int64
****************************************************************************
**************
osrm_time
21.0      2414
20.0      2361
18.0      2253
22.0      2147
17.0      2098
```

```
          ...
1284.0      1
1293.0      1
1505.0      1
1491.0      1
1080.0      1
Name: osrm_time, Length: 1531, dtype: int64
***************************************************************************
**************
osrm_distance
48.0394     11
11.2300      5
15.6814      4
24.3558      4
23.5248      4
            ..
163.8151     1
114.7974     1
1258.7837    1
63.8134      1
32.2584      1
Name: osrm_distance, Length: 138046, dtype: int64
***************************************************************************
**************
factor
2.000000     2351
1.500000     1278
1.666667      830
1.750000      667
1.333333      599
             ...
1.794562        1
2.087010        1
12.882353       1
1.765258        1
1.875758        1
Name: factor, Length: 45641, dtype: int64
***************************************************************************
**************
segment_actual_time
24.0     6188
26.0     5479
30.0     4903
27.0     4439
23.0     4401
         ...
479.0       1
345.0       1
736.0       1
546.0       1
718.0       1
Name: segment_actual_time, Length: 747, dtype: int64
***************************************************************************
**************
segment_osrm_time
16.0     11483
17.0     10856
18.0      8734
19.0      6925
15.0      6846
         ...
```

```
211.0        1
254.0        1
997.0        1
370.0        1
294.0        1
Name: segment_osrm_time, Length: 214, dtype: int64
****************************************************************************
**************
segment_osrm_distance
0.0000     1536
22.6267       8
25.6081       8
26.5134       7
26.6974       7
           ...
18.2517       1
27.6473       1
45.7671       1
22.1294       1
6.3429        1
Name: segment_osrm_distance, Length: 113799, dtype: int64
****************************************************************************
**************
segment_factor
 2.000000     6001
 1.500000     4637
 1.000000     2371
 1.666667     2370
-1.000000     2347
              ...
 0.520833        1
 1.051724        1
 1.963415        1
 1.306122        1
 10.416667       1
Name: segment_factor, Length: 5675, dtype: int64
```

In [17]:

```
for col in df.columns:
    print("***"*30)
    print(col)
    print(df[col].value_counts(normalize=True))
```

```
******************************************************************************
**************
data
training    0.723823
test        0.276177
Name: data, dtype: float64
******************************************************************************
**************
trip_creation_time
2018-09-24 05:12:53.848469    0.000697
2018-09-25 04:21:12.551117    0.000697
2018-09-29 05:04:57.639067    0.000697
2018-09-22 04:55:04.835022    0.000697
2018-09-19 04:07:34.091798    0.000697
                                ...
2018-09-30 21:41:51.314395    0.000007
2018-09-22 06:42:14.980815    0.000007
2018-09-27 16:32:14.784918    0.000007
2018-09-16 05:31:31.532090    0.000007
2018-09-20 07:13:31.478665    0.000007
Name: trip_creation_time, Length: 14817, dtype: float64
******************************************************************************
**************
route_schedule_uuid
thanos::sroute:4029a8a2-6c74-4b7e-a6d8-f9e069fbcea9    0.012508
thanos::sroute:0456b740-1dad-4929-bbe0-87d8843f5a10    0.011100
thanos::sroute:dca6268f-741a-4d1a-b1b0-aab13095a366    0.011079
thanos::sroute:a1b25549-1e77-498f-8538-00292e5bd5a2    0.008870
thanos::sroute:de5e208e-7641-45e6-8100-4d9fb1e5720d    0.008836
                                                         ...
thanos::sroute:0a05c445-d943-4f05-82ed-fd90a6d31e87    0.000007
thanos::sroute:a2c15d09-9bd2-4d29-a3fb-3dbab548800a    0.000007
thanos::sroute:d29fd731-9f1f-490c-922e-6d79d166db24    0.000007
thanos::sroute:238d7712-f567-405b-bad5-874ba36fb3c4    0.000007
thanos::sroute:036f372d-28d8-4d19-877c-6277077ad09e    0.000007
Name: route_schedule_uuid, Length: 1504, dtype: float64
******************************************************************************
**************
route_type
FTL        0.687941
Carting    0.312059
Name: route_type, dtype: float64
******************************************************************************
**************
trip_uuid
trip-153802363942560700    0.000697
trip-153793758186488532    0.000697
trip-153819749763881430    0.000697
trip-153854305492910872    0.000697
trip-153741795740530104    0.000697
                             ...
trip-153779228813052637    0.000007
trip-153762302742584394    0.000007
```

```
trip-153759657429809563    0.000007
trip-153813029204130085    0.000007
trip-153809265997799146    0.000007
Name: trip_uuid, Length: 14817, dtype: float64
****************************************************************************
**************
source_center
IND000000ACB    0.161162
IND562132AAA    0.068856
IND421302AAG    0.062733
IND411033AAA    0.028033
IND501359AAE    0.023056
                  ...
IND400072AAI    0.000007
IND733202AAC    0.000007
IND733202AAB    0.000007
IND741101AAB    0.000007
IND493445AAB    0.000007
Name: source_center, Length: 1508, dtype: float64
****************************************************************************
**************
source_name
Gurgaon_Bilaspur_HB (Haryana)            0.161488
Bangalore_Nelmngla_H (Karnataka)         0.068996
Bhiwandi_Mankoli_HB (Maharashtra)        0.062861
Pune_Tathawde_H (Maharashtra)            0.028089
Hyderabad_Shamshbd_H (Telangana)         0.023102
                                           ...
Chikhli_KKndrDPP_D (Maharashtra)         0.000007
Kasganj_BnkrGate_D (Uttar Pradesh)       0.000007
Jetpur_DC (Gujarat)                      0.000007
Islampure_ShbdnDPP_D (West Bengal)       0.000007
Islampure_Central_DPP_2 (West Bengal)    0.000007
Name: source_name, Length: 1498, dtype: float64
****************************************************************************
**************
destination_center
IND000000ACB    0.104869
IND562132AAA    0.076063
IND421302AAG    0.037911
IND501359AAE    0.035495
IND712311AAA    0.033769
                  ...
IND686141AAA    0.000007
IND396210AAA    0.000007
IND421302AAF    0.000007
IND221401AAA    0.000007
IND761020AAA    0.000007
Name: destination_center, Length: 1481, dtype: float64
****************************************************************************
**************
destination_name
Gurgaon_Bilaspur_HB (Haryana)          0.105058
Bangalore_Nelmngla_H (Karnataka)       0.076200
Bhiwandi_Mankoli_HB (Maharashtra)      0.037979
Hyderabad_Shamshbd_H (Telangana)       0.035559
Kolkata_Dankuni_HB (West Bengal)       0.033830
                                         ...
Bhadohi_Rajpura_D (Uttar Pradesh)      0.000007
Ranaghat_ArickDPP_D (West Bengal)      0.000007
Vaikom_KotyamRD_D (Kerala)             0.000007
```

```
Khatauli_TilakNgr_D (Uttar Pradesh)    0.000007
Mumbai_Skynet_INT (Maharashtra)        0.000007
Name: destination_name, Length: 1468, dtype: float64
****************************************************************************
**************
od_start_time
2018-09-21 18:37:09.322207    0.000559
2018-09-13 04:57:25.708746    0.000545
2018-09-15 06:03:01.496238    0.000545
2018-09-21 06:51:50.532257    0.000545
2018-09-26 05:33:10.899941    0.000545
                                ...
2018-10-03 23:10:20.756105    0.000007
2018-09-22 02:24:22.946198    0.000007
2018-09-13 23:38:08.229837    0.000007
2018-09-23 03:28:58.460080    0.000007
2018-09-30 03:42:39.055097    0.000007
Name: od_start_time, Length: 26369, dtype: float64
****************************************************************************
**************
od_end_time
2018-09-24 09:59:15.691618    0.000559
2018-09-29 12:11:18.125562    0.000545
2018-09-28 12:13:41.675546    0.000545
2018-10-02 10:36:25.970169    0.000545
2018-09-15 10:44:35.527660    0.000545
                                ...
2018-09-26 08:02:10.953476    0.000007
2018-09-13 05:49:21.813943    0.000007
2018-09-13 03:40:53.355885    0.000007
2018-09-18 08:48:09.325396    0.000007
2018-09-15 10:52:18.794809    0.000007
Name: od_end_time, Length: 26369, dtype: float64
****************************************************************************
**************
start_scan_to_end_scan
110.0     0.003168
72.0      0.002927
99.0      0.002837
95.0      0.002796
86.0      0.002754
            ...
1336.0    0.000007
1296.0    0.000007
2045.0    0.000007
1167.0    0.000007
2701.0    0.000007
Name: start_scan_to_end_scan, Length: 1915, dtype: float64
****************************************************************************
**************
is_cutoff
True     0.81971
False    0.18029
Name: is_cutoff, dtype: float64
****************************************************************************
**************
cutoff_factor
22      0.090821
9       0.085444
44      0.057529
18      0.057039
```

```
66      0.040002
          ...
368     0.000007
240     0.000007
310     0.000007
1461    0.000007
239     0.000007
Name: cutoff_factor, Length: 501, dtype: float64
***************************************************************************
**************
cutoff_timestamp
2018-09-24 05:19:20    0.000276
2018-09-24 05:19:21    0.000228
2018-09-14 05:29:26    0.000131
2018-09-24 07:21:24    0.000124
2018-09-18 05:19:27    0.000117
                         ...
2018-10-04 04:33:53    0.000007
2018-09-19 21:40:11    0.000007
2018-09-20 16:48:28    0.000007
2018-09-17 08:10:30    0.000007
2018-09-28 09:59:34    0.000007
Name: cutoff_timestamp, Length: 93180, dtype: float64
***************************************************************************
**************
actual_distance_to_destination
18.036366      0.000014
195.585266     0.000014
19.574937      0.000014
100.282892     0.000014
25.757877      0.000014
                 ...
27.530666      0.000007
9.606108       0.000007
1540.042272    0.000007
11.365110      0.000007
903.265473     0.000007
Name: actual_distance_to_destination, Length: 144515, dtype: float64
***************************************************************************
**************
actual_time
32.0      0.009961
36.0      0.009802
30.0      0.009319
38.0      0.009174
42.0      0.008566
            ...
2788.0    0.000007
3031.0    0.000007
3498.0    0.000007
2896.0    0.000007
3860.0    0.000007
Name: actual_time, Length: 3182, dtype: float64
***************************************************************************
**************
osrm_time
21.0      0.016664
20.0      0.016298
18.0      0.015552
22.0      0.014820
17.0      0.014482
```

```
           ...
1284.0    0.000007
1293.0    0.000007
1505.0    0.000007
1491.0    0.000007
1080.0    0.000007
Name: osrm_time, Length: 1531, dtype: float64
***************************************************************************
**************
osrm_distance
48.0394      0.000076
11.2300      0.000035
15.6814      0.000028
24.3558      0.000028
23.5248      0.000028
               ...
163.8151     0.000007
114.7974     0.000007
1258.7837    0.000007
63.8134      0.000007
32.2584      0.000007
Name: osrm_distance, Length: 138046, dtype: float64
***************************************************************************
**************
factor


2.000000     0.016229
1.500000     0.008822
1.666667     0.005729
1.750000     0.004604
1.333333     0.004135
               ...
1.794562     0.000007
2.087010     0.000007
12.882353    0.000007
1.765258     0.000007
1.875758     0.000007
Name: factor, Length: 45641, dtype: float64
***************************************************************************
**************
segment_actual_time
24.0     0.042715
26.0     0.037821
30.0     0.033845
27.0     0.030642
23.0     0.030380
           ...
479.0    0.000007
345.0    0.000007
736.0    0.000007
546.0    0.000007
718.0    0.000007
Name: segment_actual_time, Length: 747, dtype: float64
***************************************************************************
**************
segment_osrm_time
16.0     0.079266
17.0     0.074938
18.0     0.060290
19.0     0.047802
```

```
15.0      0.047257
            ...
211.0     0.000007
254.0     0.000007
997.0     0.000007
370.0     0.000007
294.0     0.000007
Name: segment_osrm_time, Length: 214, dtype: float64
****************************************************************************
**************
segment_osrm_distance
0.0000     0.010603
22.6267    0.000055
25.6081    0.000055
26.5134    0.000048
26.6974    0.000048
             ...
18.2517    0.000007
27.6473    0.000007
45.7671    0.000007
22.1294    0.000007
6.3429     0.000007
Name: segment_osrm_distance, Length: 113799, dtype: float64
****************************************************************************
**************
segment_factor
 2.000000     0.041424
 1.500000     0.032009
 1.000000     0.016367
 1.666667     0.016360
-1.000000     0.016201
                ...
 0.520833     0.000007
 1.051724     0.000007
 1.963415     0.000007
 1.306122     0.000007
 10.416667    0.000007
Name: segment_factor, Length: 5675, dtype: float64
```

In [18]:

```
for col in df.columns:
#     print("***"*30)
#     print(col)
    print(col,': ',df[col].nunique())
```

```
data :  2
trip_creation_time :  14817
route_schedule_uuid :  1504
route_type :  2
trip_uuid :  14817
source_center :  1508
source_name :  1498
destination_center :  1481
destination_name :  1468
od_start_time :  26369
od_end_time :  26369
start_scan_to_end_scan :  1915
is_cutoff :  2
cutoff_factor :  501
cutoff_timestamp :  93180
actual_distance_to_destination :  144515
actual_time :  3182
osrm_time :  1531
osrm_distance :  138046
factor :  45641
segment_actual_time :  747
segment_osrm_time :  214
segment_osrm_distance :  113799
segment_factor :  5675
```

In [19]:

```
df['trip_creation_time'].max()- df['trip_creation_time'].min(), #df['trip_creation_time'].m
```

Out[19]:

```
(Timedelta('21 days 23:59:26.165951'),)
```

In [20]:

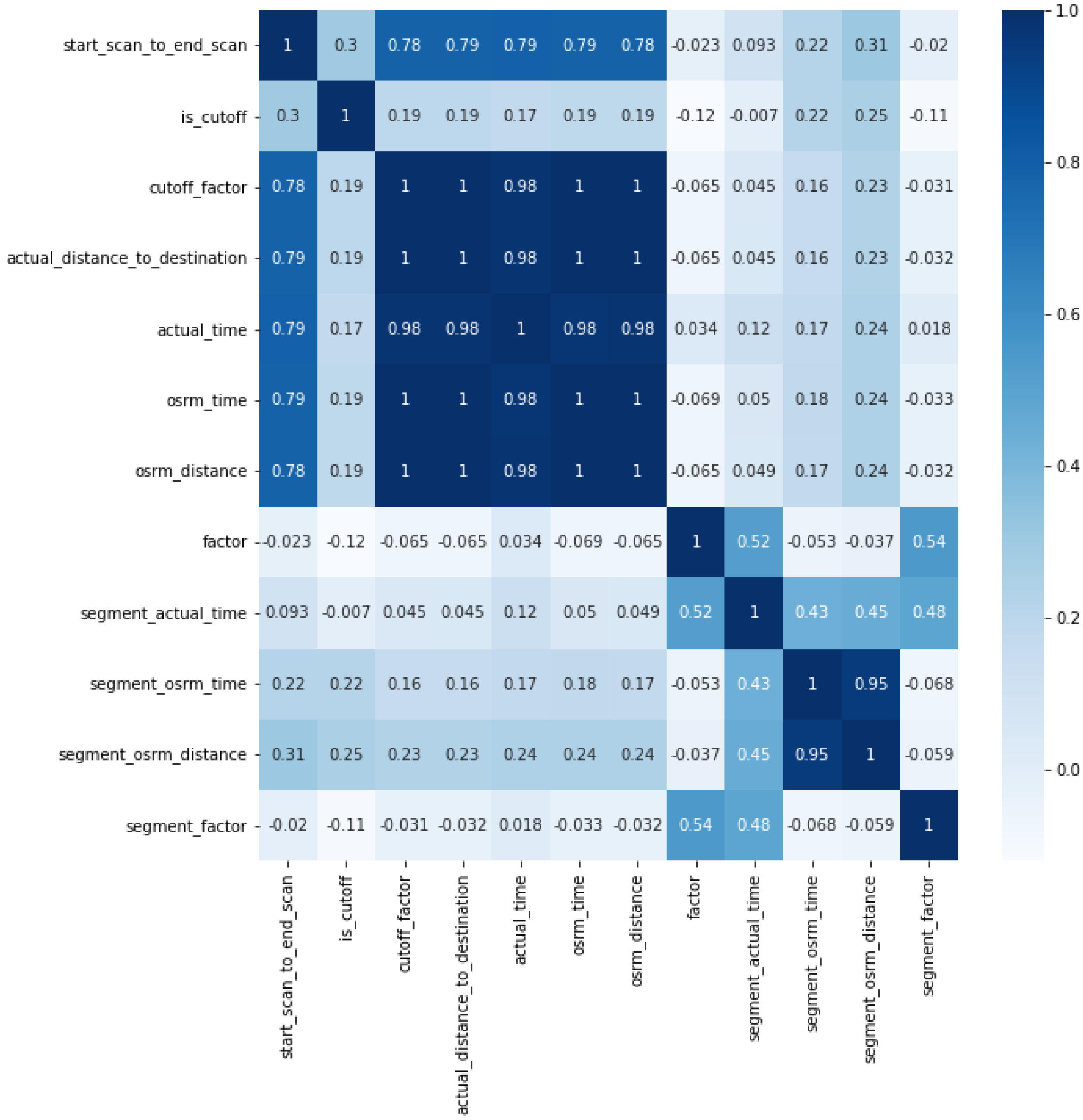
```
df.isna().sum()
```

Out[20]:

```
data                               0
trip_creation_time                 0
route_schedule_uuid                0
route_type                         0
trip_uuid                          0
source_center                      0
source_name                      293
destination_center                 0
destination_name                 261
od_start_time                      0
od_end_time                        0
start_scan_to_end_scan             0
is_cutoff                          0
cutoff_factor                      0
cutoff_timestamp                   0
actual_distance_to_destination     0
actual_time                        0
osrm_time                          0
osrm_distance                      0
factor                             0
segment_actual_time                0
segment_osrm_time                  0
segment_osrm_distance              0
segment_factor                     0
dtype: int64
```

In [22]:

```
plt.figure(figsize=(10,10))
sns.heatmap(df.corr(), cmap= "Blues", annot=True)
plt.show()
```



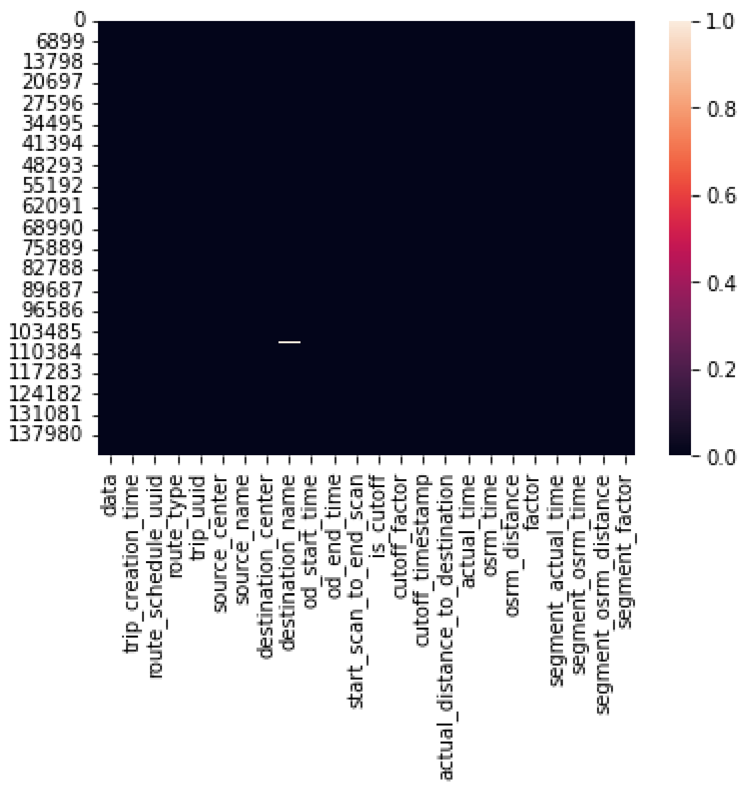
1. start_scan_to_end_scan
    - It has high correlation with cutoff_factor, actual_distance_to_destination, actual_time, osrm_time, osrm_distance
2. is_cutoff
    - It does not have high correlation with any feature
3. cutoff_factor
    - It has perfect correlation with actual_distance_to_destination, osrm_time, osrm_distance. High correlation with actual_time, good correlation with start_scan_to_end_scan
4. actual_distance_to_destination
    - It has perfect correlation with cutoff_factor, osrm_time, osrm_distance. High correlation with actual_time, good correlation with start_scan_to_end_scan
5. actual_time
    - It has high correlation with cutoff_factor,actual_distance_to_destination, osrm_time, osrm_distance. good correlation with start_scan_to_end_scan
6. osrm_time
    - It has perfect correlation with actual_distance_to_destination,cutoff_factor, osrm_distance. High correlation with actual_time, good correlation with start_scan_to_end_scan
7. osrm_distance
    - It has perfect correlation with actual_distance_to_destination,cutoff_factor, osrm_time. High correlation with actual_time, good correlation with start_scan_to_end_scan
8. factor
    - It has some correlation with segment_actual_time and segment_factor
9. segment_actual_time
    - It has some correlation with segment_osrm_time, segment_osrm_distance, segment_factor
10. Segment_osrm_time
    - It has high correlation with segment_osrm_distance. Some correlation with segment_actual_time
11. Segment_osrm_distance
    - It has high correlation with segment_osrm_time. Some correlation with segment_actual_time
12. segment_factor
    - It has some correlation with factor, segment_actual_time

In [23]:

```
sns.heatmap(df.isnull())
```

Out[23]:

```
<AxesSubplot:>
```



Observations

1. 1.4 lakhs rows, 24 cols
2. 12 object, 1 bool, 11 numeric datatypes
3. some missing values in source_name and destination_name
4. 22 days data : 2018-09-12 to 2018-10-03
5. Outliers possible: difference in mean and median start_scan_to_end_scan, cutoff_factor, actual_distance_to_destination, actual_time, osrm_time, osrm_distance, segment_actual_time

Univariate Analysis

In [24]:

```
df.head()
```

Out[24]:

ıce	factor	segment_actual_time	segment_osrm_time	segment_osrm_distance	segment_factor
;53	1.272727	14.0	11.0	11.9653	1.272727
!43	1.200000	10.0	9.0	9.7590	1.111111
;95	1.428571	16.0	7.0	10.8152	2.285714
;20	1.550000	21.0	12.0	13.0224	1.750000
81	1.545455	6.0	5.0	3.9153	1.200000

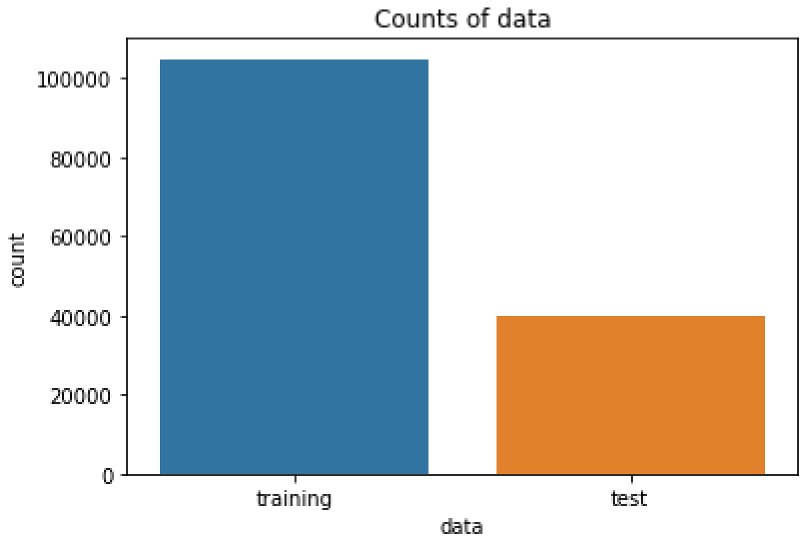
In [26]:

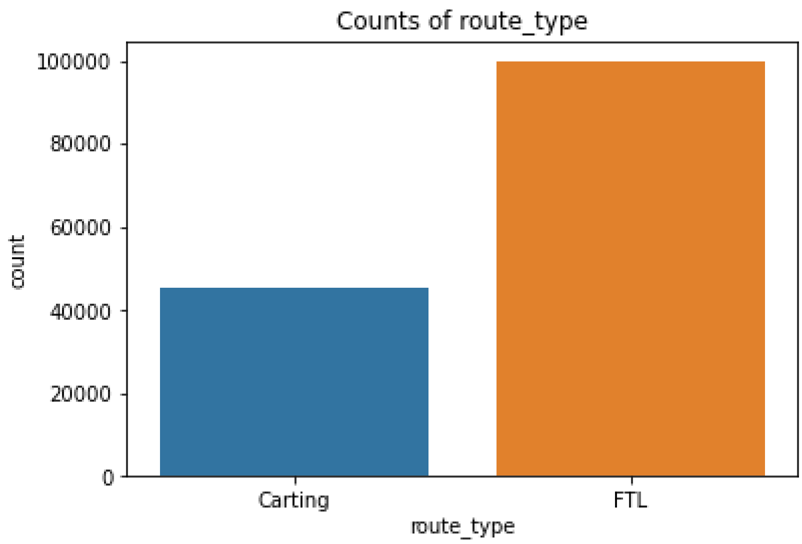
```
count_plot_cols = ['data','route_type','is_cutoff']
hist_plot_cols = ['start_scan_to_end_scan','cutoff_factor','actual_distance_to_destination'
                  'osrm_time','osrm_distance','factor','segment_actual_time','segment_osrm_
                 'segment_factor']
```

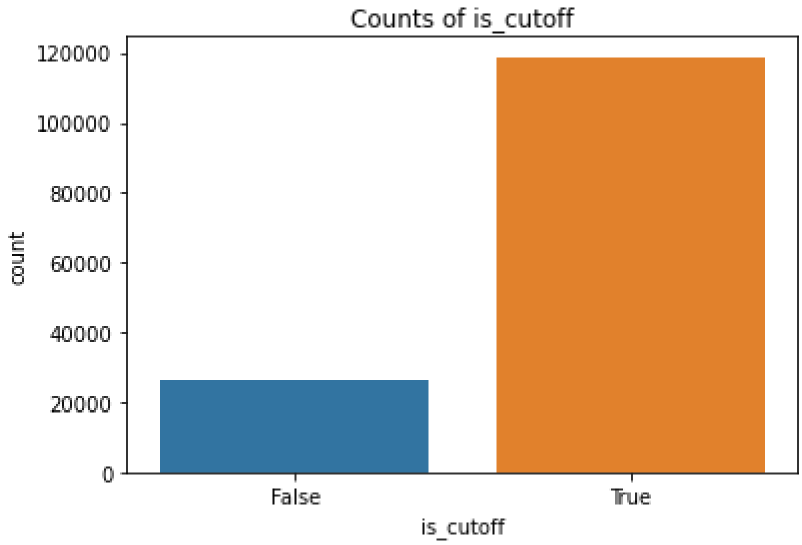
In [27]:

```
print("Univariate Analysis")
print("Count plots")
for col in count_plot_cols:
    sns.countplot(x=col, data=df)
    plt.title(f"Counts of {col}")
    plt.show()
```

```
Univariate Analysis
Count plots
```



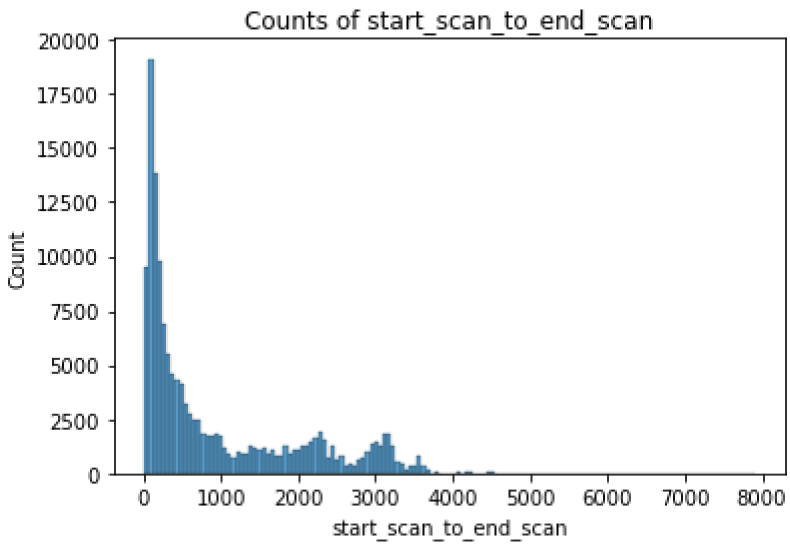


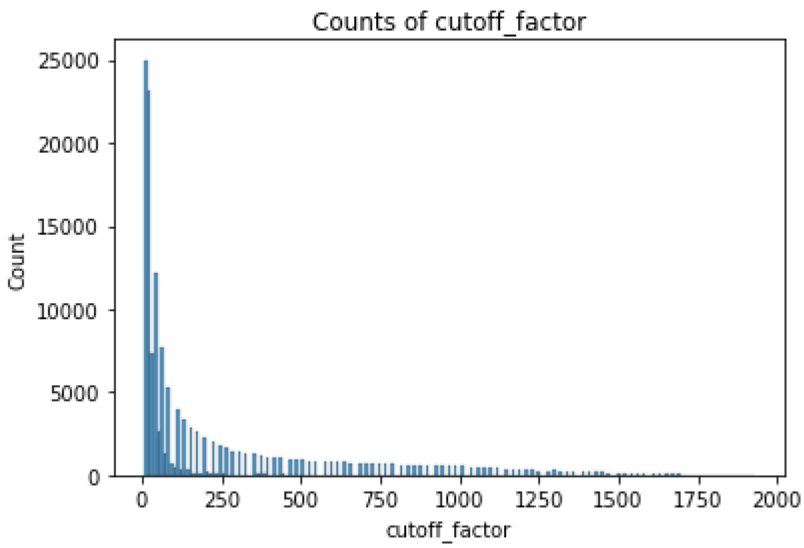

Counts of is_cutoff
120000
100000
80000
60000
40000
20000
0
count
False
True
is_cutoff

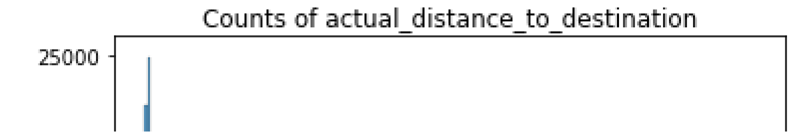
In [28]:

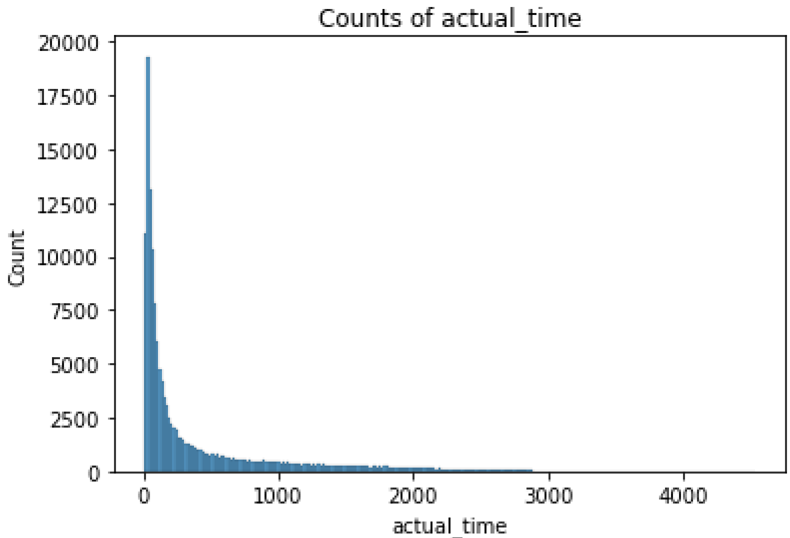
```
print("Univariate Analysis")
print("Hist plots")
for col in hist_plot_cols:
    sns.histplot(x=col, data=df)
    plt.title(f"Counts of {col}")
    plt.show()
```

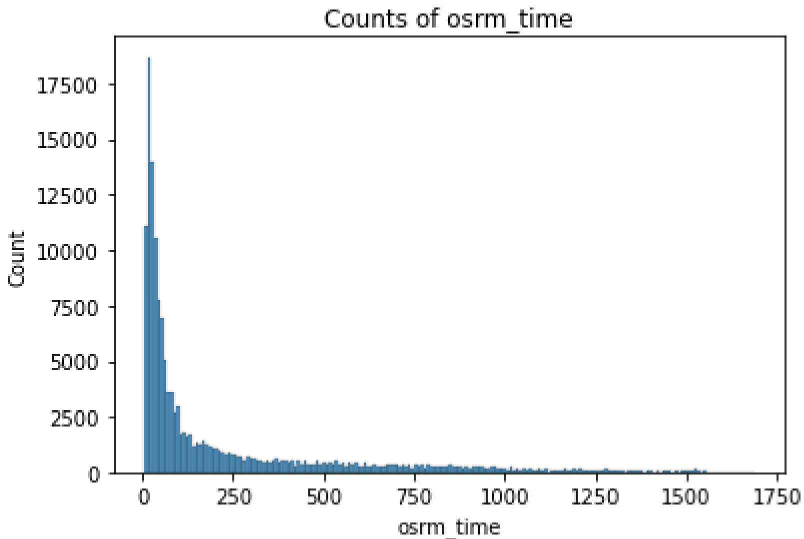
```
Univariate Analysis
Hist plots
```

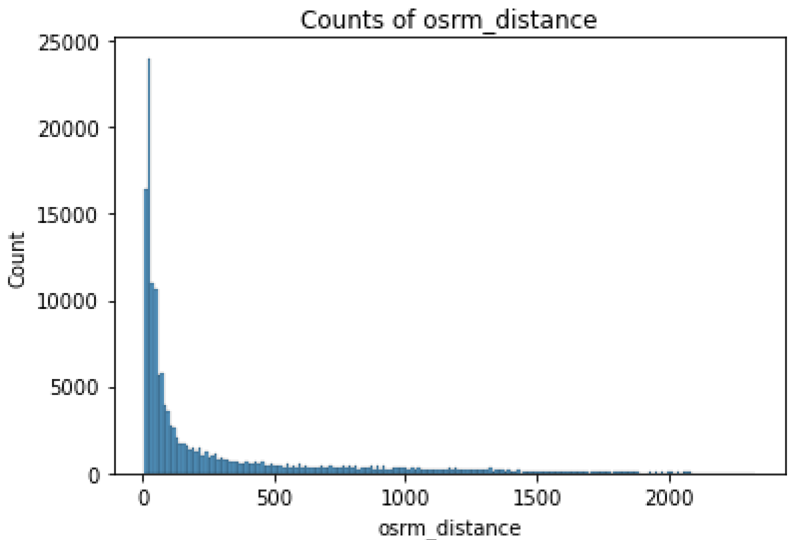




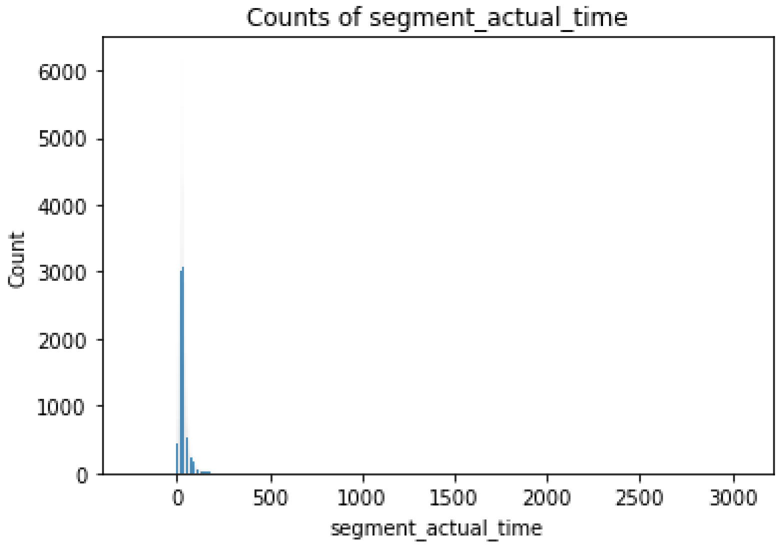

Counts of actual_distance_to_destination
25000

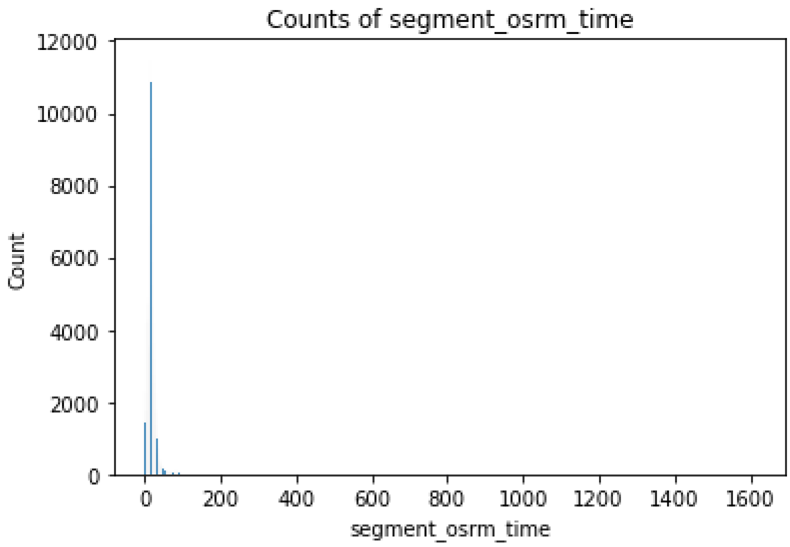

Counts of actual_time
Count
20000
17500
15000
12500
10000
7500
5000
2500
0
0
1000
2000
3000
4000
actual_time

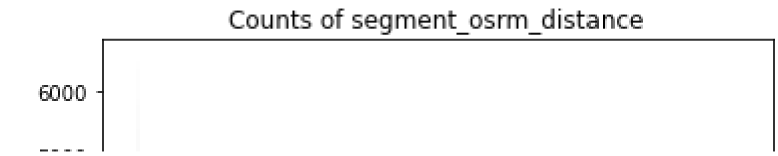

Counts of osrm_time
Count
17500
15000
12500
10000
7500
5000
2500
0
0
250
500
750
1000
1250
1500
1750
osrm_time

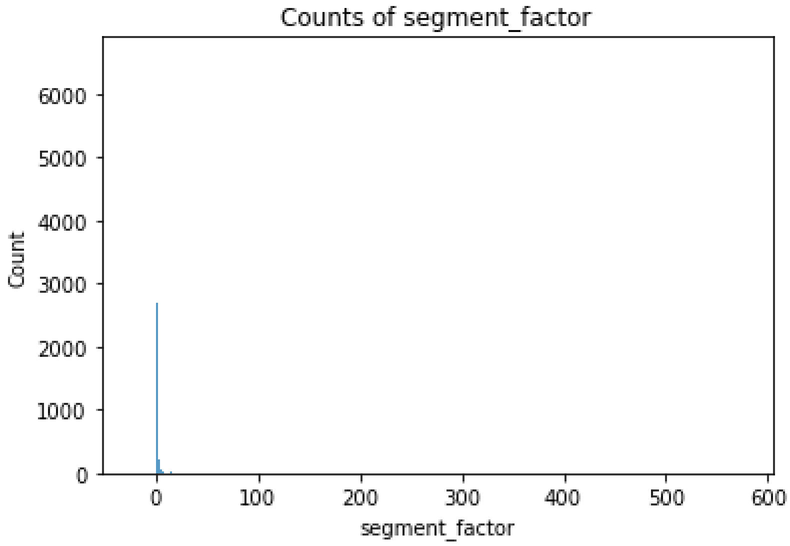

Counts of osrm_distance
Count
25000
20000
15000
10000
5000
0
0
500
1000
1500
2000
osrm_distance


Counts of factor
Count
4000
3500
3000
2500
2000
1500
1000
500


Counts of segment_actual_time
Count
6000
5000
4000
3000
2000
1000
0
500
1000
1500
2000
2500
3000
segment_actual_time


Counts of segment_osrm_time
Count
12000
10000
8000
6000
4000
2000
0
200
400
600
800
1000
1200
1400
1600
segment_osrm_time

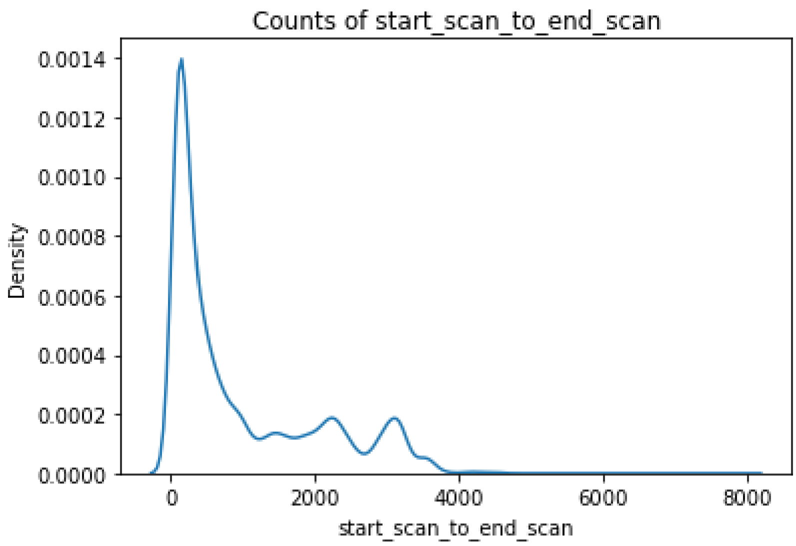

Counts of segment_osrm_distance
6000

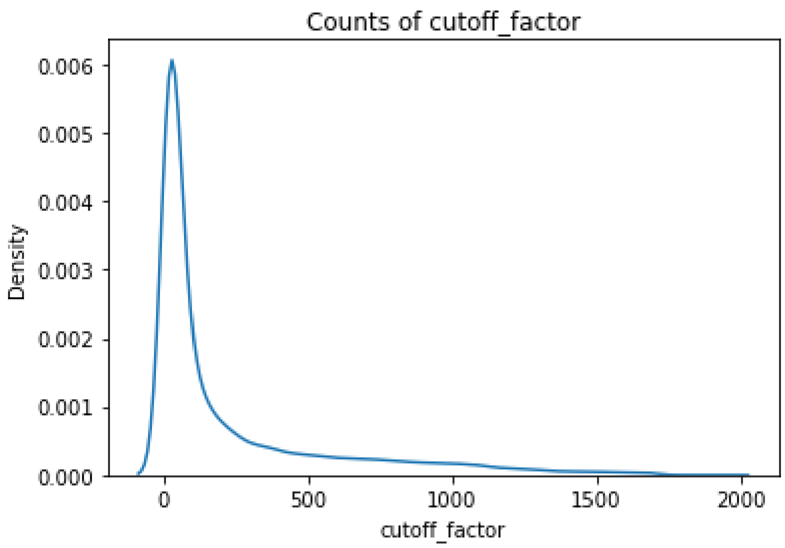

Counts of segment_factor
6000
5000
4000
3000
2000
1000
0
Count
0
100
200
300
400
500
600
segment_factor

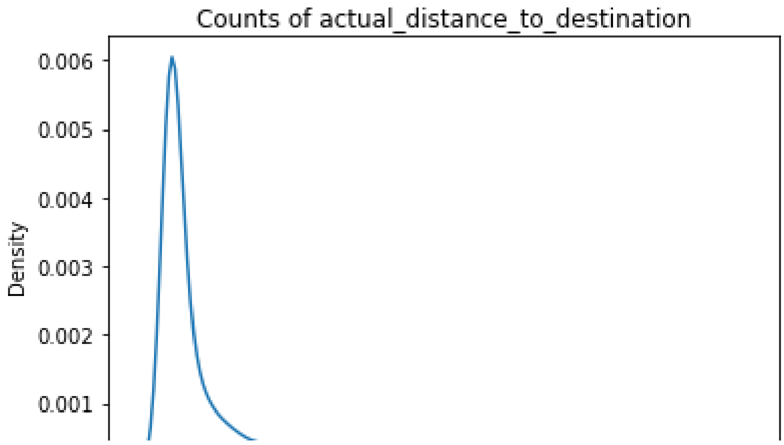
In [29]:

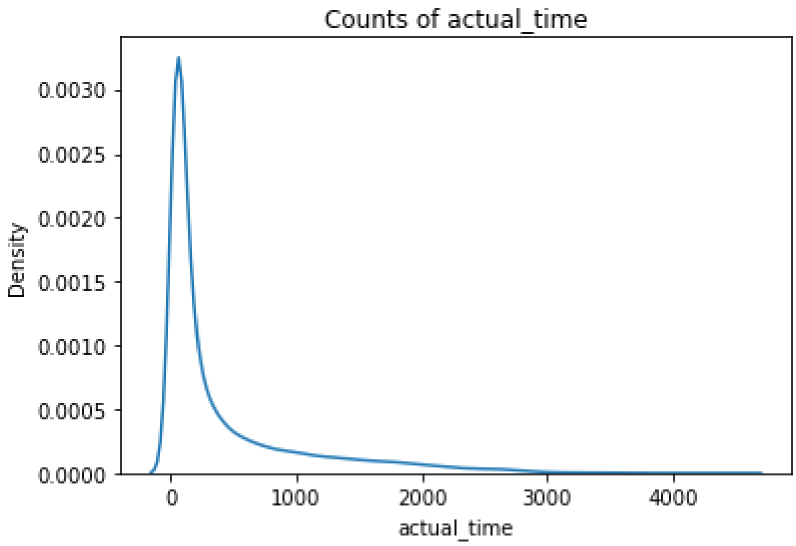
```
print("Univariate Analysis")
print("Hist plots")
for col in hist_plot_cols:
    sns.kdeplot(x=col, data=df)
    plt.title(f"Counts of {col}")
    plt.show()
```

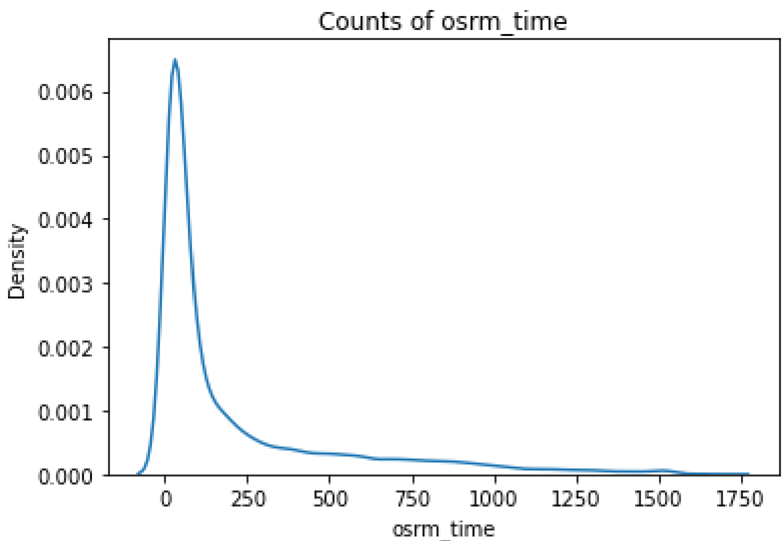
```
Univariate Analysis
Hist plots
```

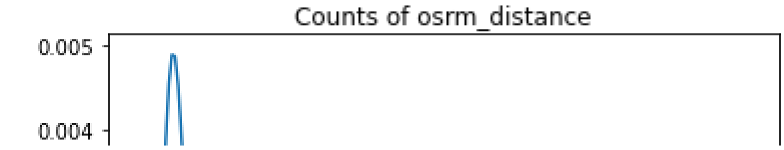


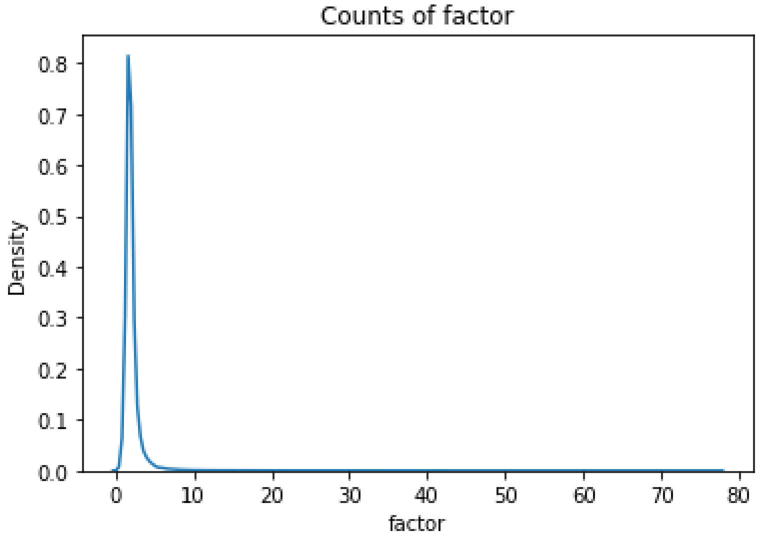


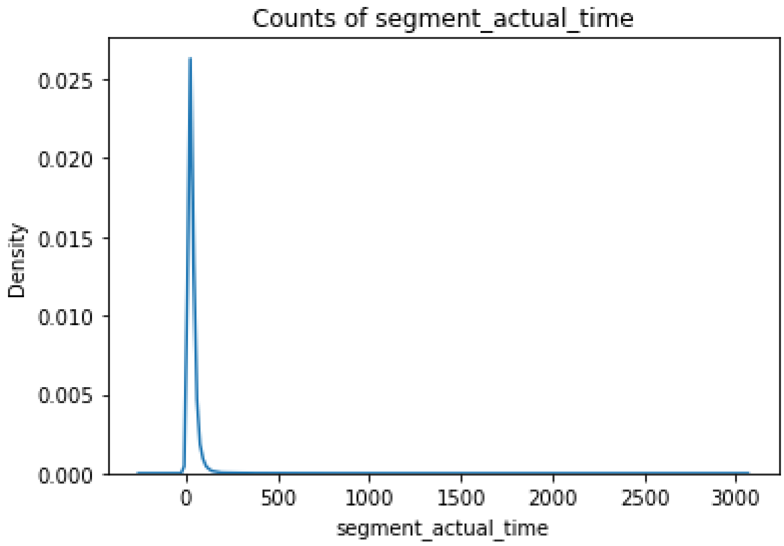

Counts of actual_distance_to_destination
Density
0.006
0.005
0.004
0.003
0.002
0.001

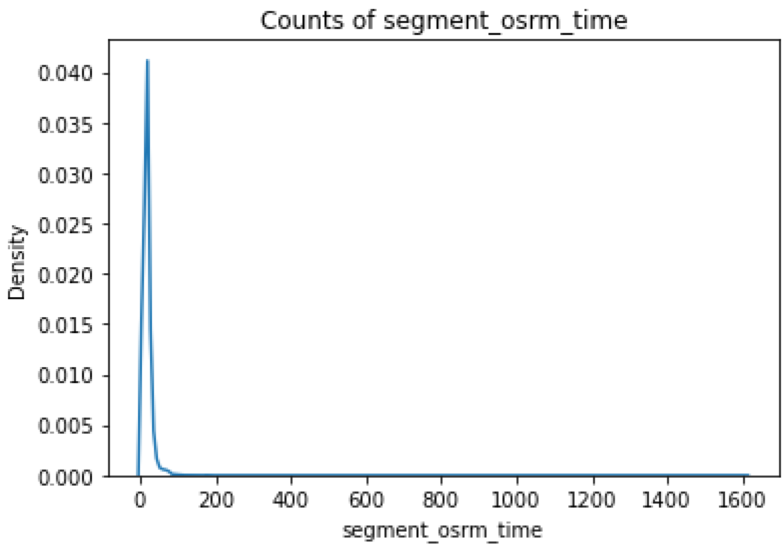

Counts of actual_time
Density
0.0030
0.0025
0.0020
0.0015
0.0010
0.0005
0.0000
0
1000
2000
3000
4000
actual_time

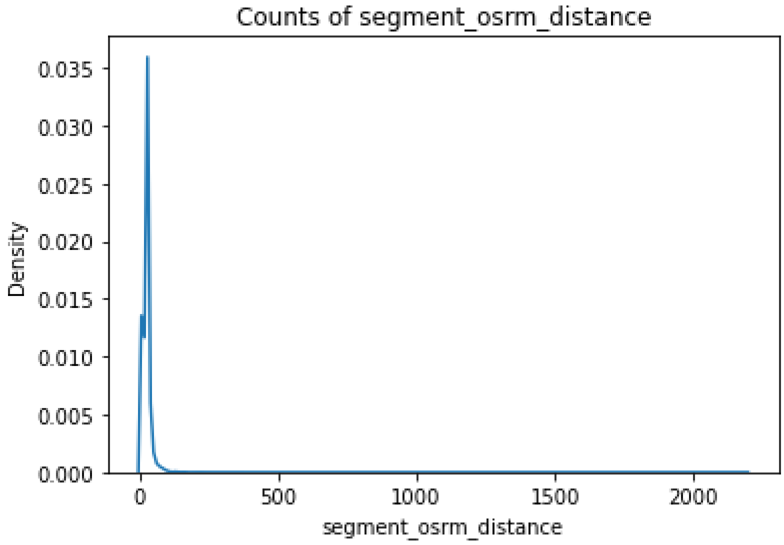

Counts of osrm_time
Density
0.006
0.005
0.004
0.003
0.002
0.001
0.000
0
250
500
750
1000
1250
1500
1750
osrm_time

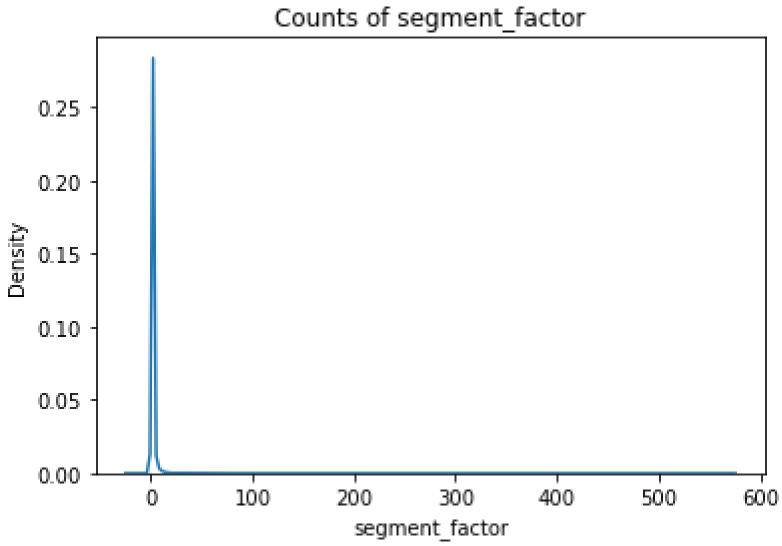

Counts of osrm_distance
0.005
0.004


Counts of factor
Density
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0 10 20 30 40 50 60 70 80
factor


Counts of segment_actual_time
Density
0.025
0.020
0.015
0.010
0.005
0.000
0 500 1000 1500 2000 2500 3000
segment_actual_time


Counts of segment_osrm_time
Density
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
0 200 400 600 800 1000 1200 1400 1600
segment_osrm_time



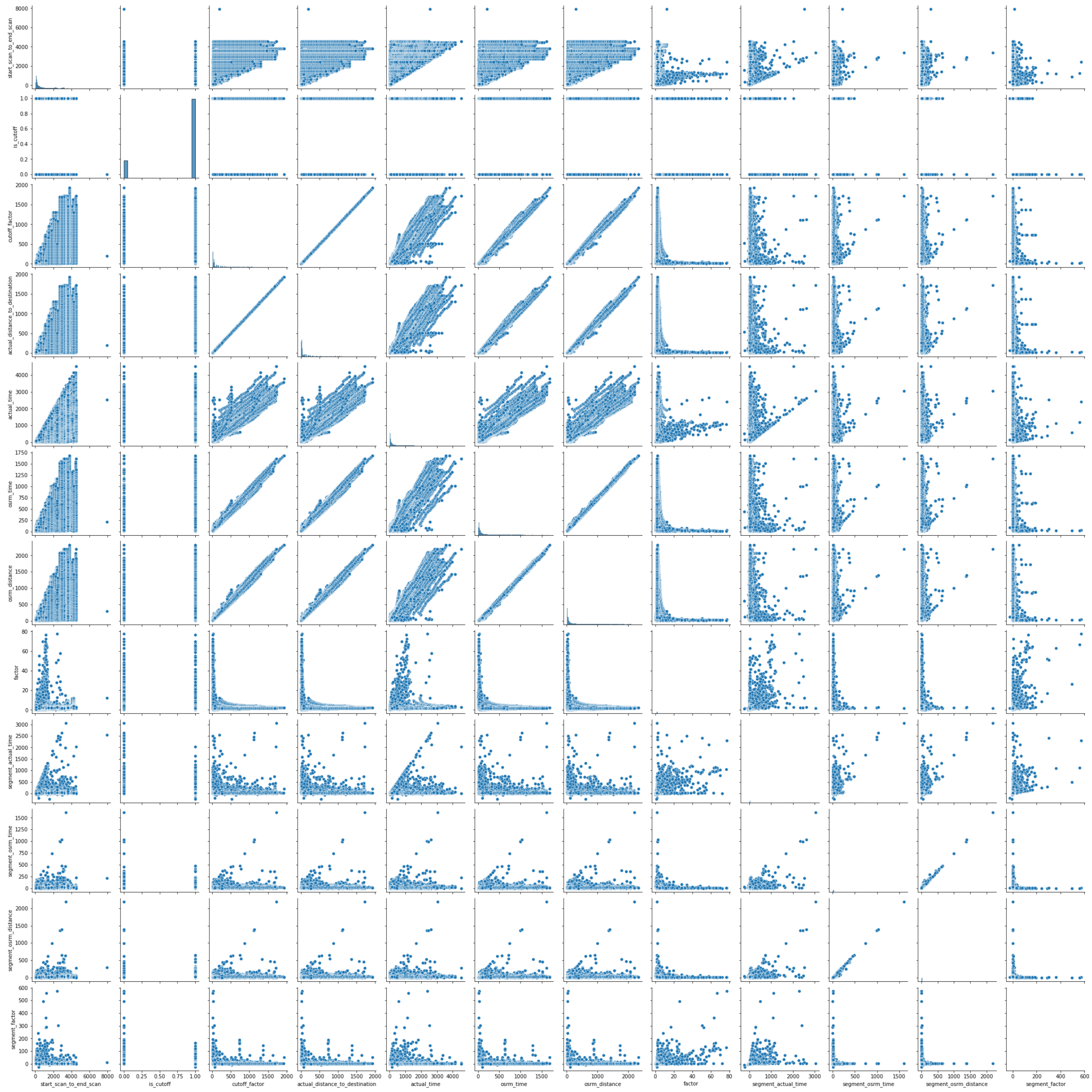


Bivariate Analysis

In [30]:

```
sns.pairplot(df)
plt.show()
```

```
<__array_function__ internals>:5: RuntimeWarning: Converting input from bool
to <class 'numpy.uint8'> for compatibility.
<__array_function__ internals>:5: RuntimeWarning: Converting input from bool
to <class 'numpy.uint8'> for compatibility.
```



2. Aggregation

In [82]:

```
df.head()
```

Out[82]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	source_(
0	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
1	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
2	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
3	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812
4	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND38812

Merge-1

In [70]:

```
drop_tripid = 'trip-153784572117438961'
df = df[df['trip_uuid']!=drop_tripid]
```

In [98]:

```
stat_cols = ['data', 'trip_creation_time', 'route_schedule_uuid', 'route_type',
        'source_name', 'destination_name', 'od_start_time', 'od_end_time',
       'start_scan_to_end_scan',]
agg_cols = ['actual_distance_to_destination', 'actual_time',
       'osrm_time', 'osrm_distance', 'segment_actual_time',
       'segment_osrm_time', 'segment_osrm_distance']
agg_col_map = {'actual_distance_to_destination':'max', 'actual_time':'max',
       'osrm_time':'max', 'osrm_distance':'max', 'segment_actual_time':'sum',
       'segment_osrm_time':'sum', 'segment_osrm_distance':'sum'}
```

In [99]:

```
static_df = df.groupby(['trip_uuid','source_center','destination_center'])[stat_cols].agg(p
agg_df = df.groupby(['trip_uuid','source_center','destination_center'])[agg_cols].agg(agg_c
```

In [101]:

```
agg_df.shape, static_df.shape
```

Out[101]:

```
((26364, 10), (26364, 12))
```

In [102]:

```
merged_df = static_df.merge(agg_df, on=['trip_uuid','source_center','destination_center'])
```

In [104]:

```
merged_df.head()
```

Out[104]:

	trip_uuid	source_center	destination_center	data	trip_creation_time	route_
0	trip-153671041653548748	IND209304AAA	IND000000ACB	training	2018-09-12 00:00:16.535741	thanos::sr a
1	trip-153671041653548748	IND462022AAA	IND209304AAA	training	2018-09-12 00:00:16.535741	thanos::sr a
2	trip-153671042288605164	IND561203AAB	IND562101AAA	training	2018-09-12 00:00:22.886430	thanos::sr b
3	trip-153671042288605164	IND572101AAA	IND561203AAB	training	2018-09-12 00:00:22.886430	thanos::sr b
4	trip-153671043369099517	IND000000ACB	IND160002AAC	training	2018-09-12 00:00:33.691250	thanos::sr 7(

Merge-2

In [105]:

```
df.shape, merged_df.shape
```

Out[105]:

```
((144856, 24), (26364, 19))
```

In [125]:

```
m_cols = list(merged_df.columns)

m_cols_map = { 'source_center':'first', 'destination_center':'last', 'data':pd.Series.mode,
              'trip_creation_time':pd.Series.mode, 'route_schedule_uuid':pd.Series.mode, 'r
              'source_name':'first', 'destination_name':'last', 'od_start_time':'first', 'o
              'start_scan_to_end_scan':'sum',
              'actual_distance_to_destination':'sum', 'actual_time':'sum', 'osrm_time':'sum
              'segment_actual_time':'sum', 'segment_osrm_time':'sum', 'segment_osrm_distanc
```

In [126]:

```
merged_df = merged_df.sort_values(['od_start_time'])
```

In [352]:

```
final_df = merged_df.groupby(['trip_uuid']).agg(m_cols_map).reset_index()
```

In [353]:

```
final_df.shape, merged_df.shape
```

Out[353]:

```
((14816, 19), (26364, 19))
```

In [354]:

```
final_df.head()
```

Out[354]:

	trip_uuid	source_center	destination_center	data	trip_creation_time	route_
0	trip-153671041653548748	IND462022AAA	IND000000ACB	training	2018-09-12 00:00:16.535741	thanos::sr a
1	trip-153671042288605164	IND572101AAA	IND562101AAA	training	2018-09-12 00:00:22.886430	thanos::sr bl
2	trip-153671043369099517	IND562132AAA	IND160002AAC	training	2018-09-12 00:00:33.691250	thanos::sr 76
3	trip-153671046011330457	IND400072AAB	IND401104AAA	training	2018-09-12 00:01:00.113710	thanos::sr a6
4	trip-153671052974046625	IND583101AAA	IND583101AAA	training	2018-09-12 00:02:09.740725	thanos::sr 6

2. Feature engineering

feat engg

1. source_center, destination_center: IND, NUMBER, 3CHAR STRING--x
2. time: year,month,day, weekday(sunday),
3. source_name, destination_name: last(state), first(city). City-place-code (State)
4. diff_time = od_end-od_start

In [355]:

```
def center_split(df, col):
    df[col+'_first'] = df[col].apply(lambda x:x[:3])
    df[col+'_second'] = df[col].apply(lambda x:x[3:-3])
    df[col+'_third'] = df[col].apply(lambda x:x[-3:])
    return df.drop([col], axis=1)

final_df = center_split(final_df, 'source_center')
final_df = center_split(final_df, 'destination_center')
```

In [356]:

```
def place_name(mystr, ind):
    try:
        out = str(mystr).split()[0].split("_")[ind]
        return out
    except:
        return ""

def name_features(df, col):
    df[col+'_state'] = df[col].apply(lambda x:str(x).split('(')[-1][:-1])
    df[col+'_city'] = df[col].apply(lambda x:place_name(x, 0))
    df[col+'_place'] = df[col].apply(lambda x:place_name(x, 1))
    df[col+'_code'] = df[col].apply(lambda x:place_name(x, 2))
    return df.drop([col], axis=1)

final_df['source_name'] = final_df['source_name'].fillna("")
final_df['destination_name'] = final_df['destination_name'].fillna("")

final_df = name_features(final_df, 'source_name')
final_df = name_features(final_df, 'destination_name')
```

In [357]:

```
def time_feats(df, col):
    df[col] = pd.to_datetime(df[col])
    df[col+'_YEAR'] = df[col].dt.year
    df[col+'_MONTH'] = df[col].dt.month
    df[col+'_DATE'] = df[col].dt.day
    df[col+'_DAYOFWEEK'] = df[col].dt.dayofweek
    return df.drop([col], axis=1)

final_df = time_feats(final_df, 'trip_creation_time')
```

In [358]:

```
final_df.head()
```

Out[358]:

	trip_uuid	data	route_schedule_uuid	route_type	od_start_time	od_end_
0	trip-153671041653548748	training	thanos::sroute:d7c989ba-a29b-4a0b-b2f4-288cdc6...	FTL	2018-09-12 00:00:16.535741	2018-(13:40:23.12
1	trip-153671042288605164	training	thanos::sroute:3a1b0ab2-bb0b-4c53-8c59-eb2a2c0...	Carting	2018-09-12 00:00:22.886430	2018-(03:01:59.59
2	trip-153671043369099517	training	thanos::sroute:de5e208e-7641-45e6-8100-4d9fb1e...	FTL	2018-09-12 00:00:33.691250	2018-(17:34:55.44
3	trip-153671046011330457	training	thanos::sroute:f0176492-a679-4597-8332-bbd1c7f...	Carting	2018-09-12 00:01:00.113710	2018-(01:41:29.80
4	trip-153671052974046625	training	thanos::sroute:d9f07b12-65e0-4f3b-bec8-df06134...	FTL	2018-09-12 00:02:09.740725	2018-(12:00:30.68

3. In-Depth analysis

In [359]:

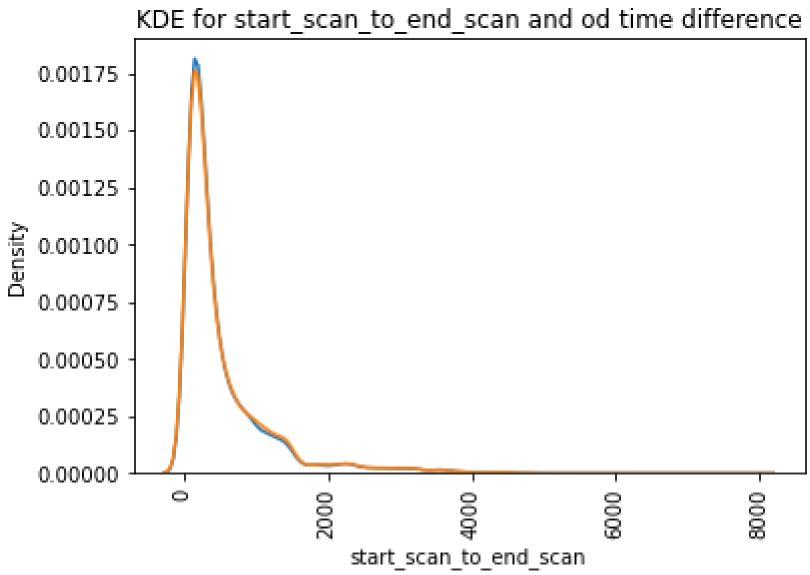
```
final_df['od_diff'] = final_df['od_end_time']-final_df['od_start_time']

final_df['od_diff_in_hours'] = final_df['od_diff'].astype('timedelta64[h]')
final_df['od_diff_in_mins'] = final_df['od_diff'].astype('timedelta64[m]')
final_df = final_df.drop(['od_diff', 'od_start_time','od_end_time'], axis=1)
```

1. od_end_time-od_start_time and start_scan_to_end_scan

In [238]:

```
sns.kdeplot(final_df['start_scan_to_end_scan'])
sns.kdeplot(final_df['od_diff_in_mins'])
plt.xticks(rotation=90)
plt.title(f'KDE for start_scan_to_end_scan and od time difference')
plt.show()
```



In [247]:

```
print(final_df['start_scan_to_end_scan'].mean(), final_df['od_diff_in_mins'].mean())
print(final_df['start_scan_to_end_scan'].std(), final_df['od_diff_in_mins'].std())
```

```
530.7146328293736 546.8666306695465
658.6258521033716 668.5807270807622
```

2-sample T-test

1. H0: mu1 = mu2 (mu1: population mean of start_scan_to_end_scan, mu2: population mean of od_diff_in_mins)
   HA: mu1 != mu1
2. Test-Statistic: Tobs = (m1 - m2)/sqrt(s1^2/n1 + s2^2/n2)
3. 2-sided
4. T_obs from data
5. Calculate p-val
6. significance value
7. Compare p-val and significance value

Assumptions of T-test:

1. sample mean m1 and m2 follows CLT
   It has finite mean and variance

1. Both the samples have finite mean and variance, so this assumption is satisfied

In [244]:

```
alpha =0.05

w_stats, p_value =stats.ttest_ind(a=final_df['start_scan_to_end_scan'], b=final_df['od_diff

if p_value > alpha :
  print("We do not reject the null hypothesis: means are similar")
else:
  print("Reject the Null Hypothesis: means are different")
```

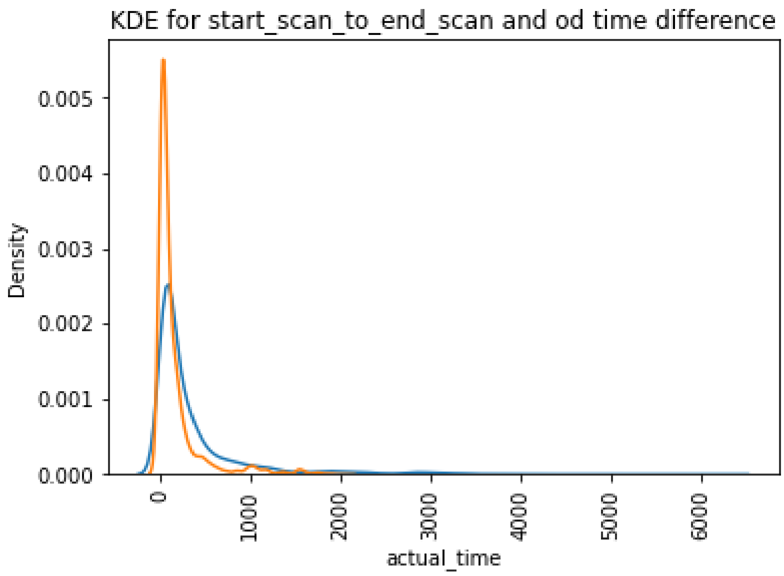
```
Reject the Null Hypothesis: means are different
```

2. actual_time vs OSRM_time

Do hypothesis testing/ visual analysis between actual_time aggregated value and OSRM time aggregated value (aggregated values are the values you'll get after merging the rows on the basis of trip_uuid)

In [246]:

```
sns.kdeplot(final_df['actual_time'])
sns.kdeplot(final_df['osrm_time'])
plt.xticks(rotation=90)
plt.title(f'KDE for start_scan_to_end_scan and od time difference')
plt.show()
```



In [248]:

```
print(final_df['actual_time'].mean(), final_df['osrm_time'].mean())
print(final_df['actual_time'].std(), final_df['osrm_time'].std())
```

```
357.1114335853132 162.0707343412527
561.4013174958384 272.3138170862162
```

In [249]:

```
alpha =0.05

w_stats, p_value =stats.ttest_ind(a=final_df['actual_time'], b=final_df['osrm_time'], equal

if p_value > alpha :
  print("We do not reject the null hypothesis: means are similar")
else:
  print("Reject the Null Hypothesis: means are different")
```

```
Reject the Null Hypothesis: means are different
```

3. actual_time vs segment_actual

Do hypothesis testing/ visual analysis between actual_time aggregated value and segment actual time aggregated value (aggregated values are the values you'll get after merging the rows on the basis of trip_uuid)

In [251]:

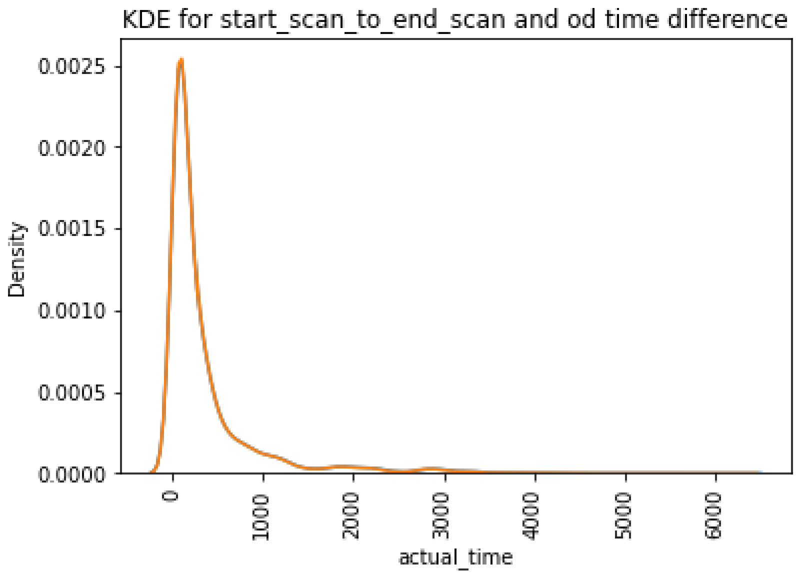
```
final_df.head()
```

Out[251]:

	trip_uuid	data	route_schedule_uuid	route_type	start_scan_to_end_scan	ac
0	trip-153671041653548748	training	thanos::sroute:d7c989ba-a29b-4a0b-b2f4-288cdc6...	FTL	2259.0	
1	trip-153671042288605164	training	thanos::sroute:3a1b0ab2-bb0b-4c53-8c59-eb2a2c0...	Carting	180.0	
2	trip-153671043369099517	training	thanos::sroute:de5e208e-7641-45e6-8100-4d9fb1e...	FTL	3933.0	
3	trip-153671046011330457	training	thanos::sroute:f0176492-a679-4597-8332-bbd1c7f...	Carting	100.0	
4	trip-153671052974046625	training	thanos::sroute:d9f07b12-65e0-4f3b-bec8-df06134...	FTL	717.0	

In [252]:

```
sns.kdeplot(final_df['actual_time'])
sns.kdeplot(final_df['segment_actual_time'])
plt.xticks(rotation=90)
plt.title(f'KDE for start_scan_to_end_scan and od time difference')
plt.show()
```



In [253]:

```
print(final_df['actual_time'].mean(), final_df['segment_actual_time'].mean())
print(final_df['actual_time'].std(), final_df['segment_actual_time'].std())
```

```
357.1114335853132 353.8496220302376
561.4013174958384 556.2424934983527
```

In [254]:

```
alpha =0.05

w_stats, p_value =stats.ttest_ind(a=final_df['actual_time'], b=final_df['segment_actual_tim

if p_value > alpha :
  print("We do not reject the null hypothesis: means are similar")
else:
  print("Reject the Null Hypothesis: means are different")
```

```
We do not reject the null hypothesis: means are similar
```

4. osrm_distance vs segment_osrm_distance

Do hypothesis testing/ visual analysis between osrm_distance aggregated value and segment_osrm_distance value (aggregated values are the values you'll get after merging the rows on the basis of trip_uuid)

In [255]:

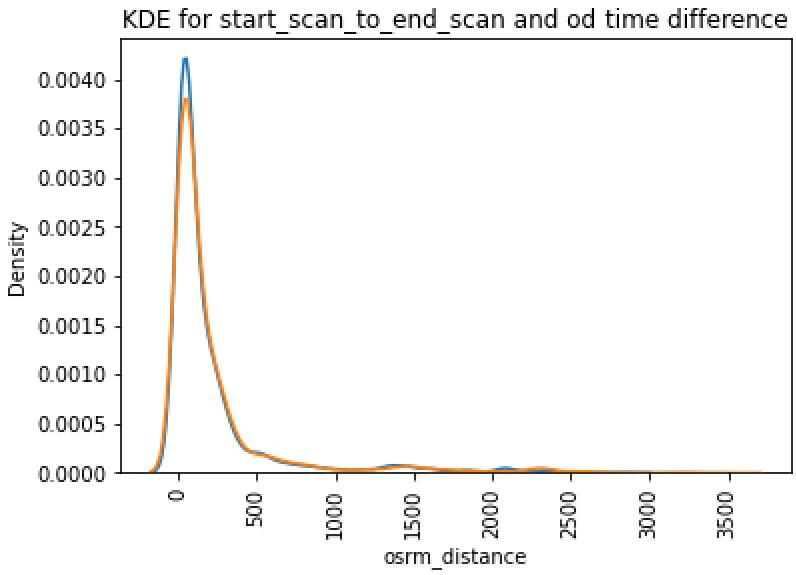
```
final_df.head()
```

Out[255]:

actual_time	osrm_time	osrm_distance	segment_actual_time	segment_osrm_time	segment_osrm_
1562.0	743.0	991.3523	1548.0	1008.0	1
143.0	68.0	85.1110	141.0	65.0	
3347.0	1741.0	2372.0852	3308.0	1941.0	2
59.0	15.0	19.6800	59.0	16.0	
341.0	117.0	146.7918	340.0	115.0	

In [256]:

```
sns.kdeplot(final_df['osrm_distance'])
sns.kdeplot(final_df['segment_osrm_distance'])
plt.xticks(rotation=90)
plt.title(f'KDE for start_scan_to_end_scan and od time difference')
plt.show()
```



In [257]:

```
print(final_df['osrm_distance'].mean(), final_df['segment_osrm_distance'].mean())
print(final_df['osrm_distance'].std(), final_df['segment_osrm_distance'].std())
```

```
205.10097050486002 223.19943614335884
370.7925205105662 416.6423821620311
```

In [258]:

```
alpha =0.05

w_stats, p_value =stats.ttest_ind(a=final_df['osrm_distance'], b=final_df['segment_osrm_dis

if p_value > alpha :
  print("We do not reject the null hypothesis: means are similar")
else:
  print("Reject the Null Hypothesis: means are different")
```

```
Reject the Null Hypothesis: means are different
```

5. osrm_time vs segment_osrm_time

Do hypothesis testing/ visual analysis between osrm_distance aggregated value and segment_osrm_distance value (aggregated values are the values you'll get after merging the rows on the basis of trip_uuid)

In [255]:

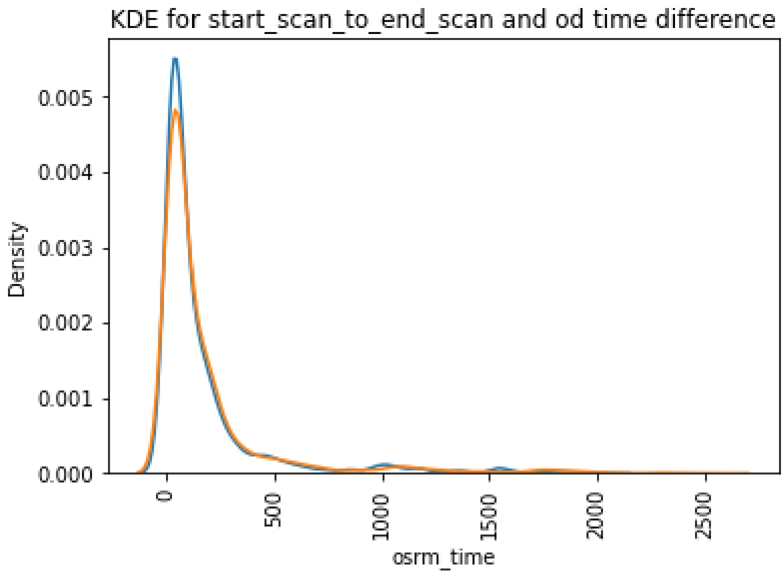
```
final_df.head()
```

Out[255]:

:tual_time	osrm_time	osrm_distance	segment_actual_time	segment_osrm_time	segment_osrm_di
1562.0	743.0	991.3523	1548.0	1008.0	132
143.0	68.0	85.1110	141.0	65.0	ε
3347.0	1741.0	2372.0852	3308.0	1941.0	254
59.0	15.0	19.6800	59.0	16.0	1
341.0	117.0	146.7918	340.0	115.0	14

In [259]:

```
sns.kdeplot(final_df['osrm_time'])
sns.kdeplot(final_df['segment_osrm_time'])
plt.xticks(rotation=90)
plt.title(f'KDE for start_scan_to_end_scan and od time difference')
plt.show()
```



In [260]:

```
print(final_df['osrm_time'].mean(), final_df['segment_osrm_time'].mean())
print(final_df['osrm_time'].std(), final_df['segment_osrm_time'].std())
```

```
162.0707343412527 180.94897408207345
272.3138170862162 314.5526464028807
```

In [261]:

```
alpha =0.05

w_stats, p_value =stats.ttest_ind(a=final_df['osrm_time'], b=final_df['segment_osrm_time'],

if p_value > alpha :
  print("We do not reject the null hypothesis: means are similar")
else:
  print("Reject the Null Hypothesis: means are different")
```

```
Reject the Null Hypothesis: means are different
```

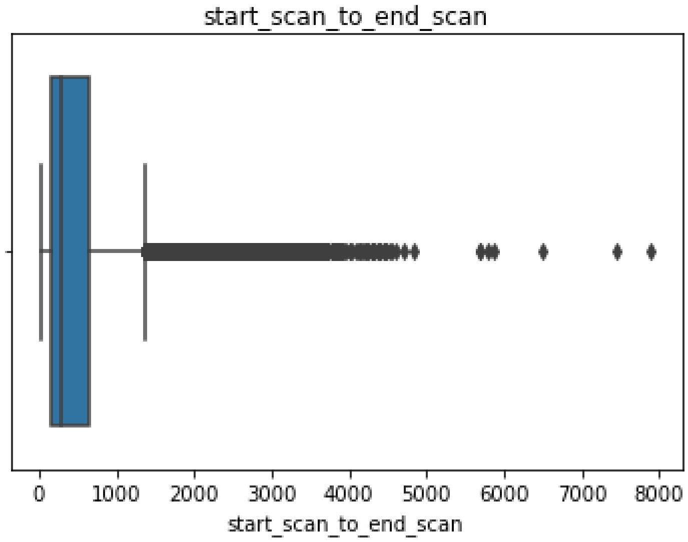
4. Outlier detection and Removal

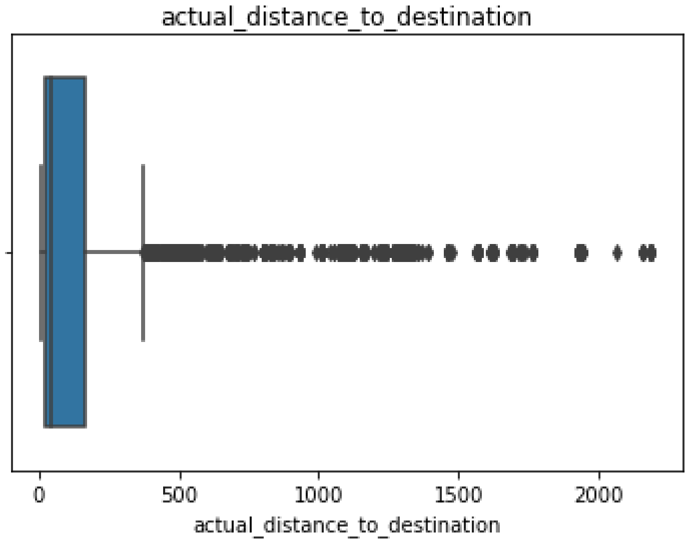
In [360]:

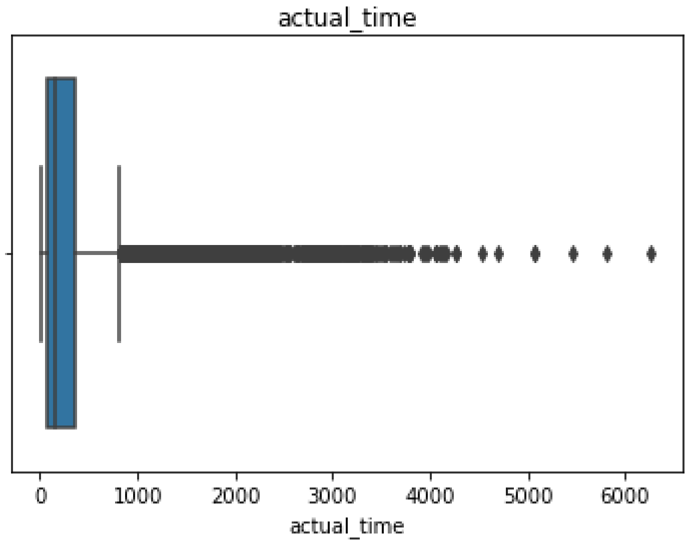
```
out_cols = final_df.select_dtypes(include=['int','float']).columns
```

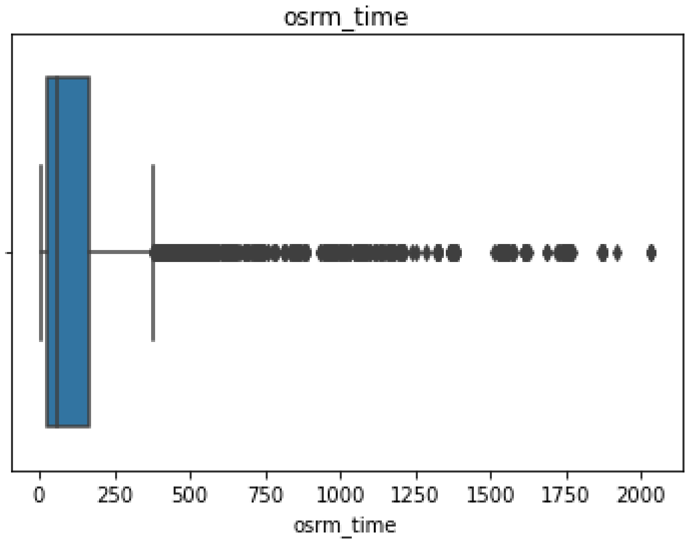
In [361]:

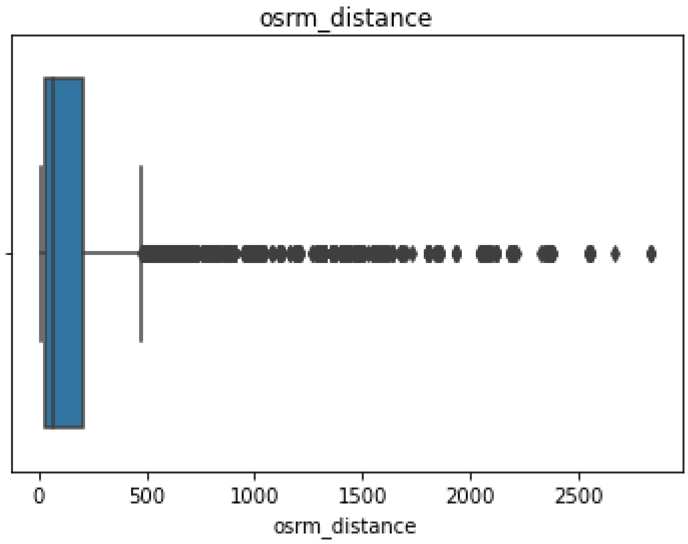
```
for col in out_cols:
    sns.boxplot(x= final_df[col])
    plt.title(col)
    plt.show()
```

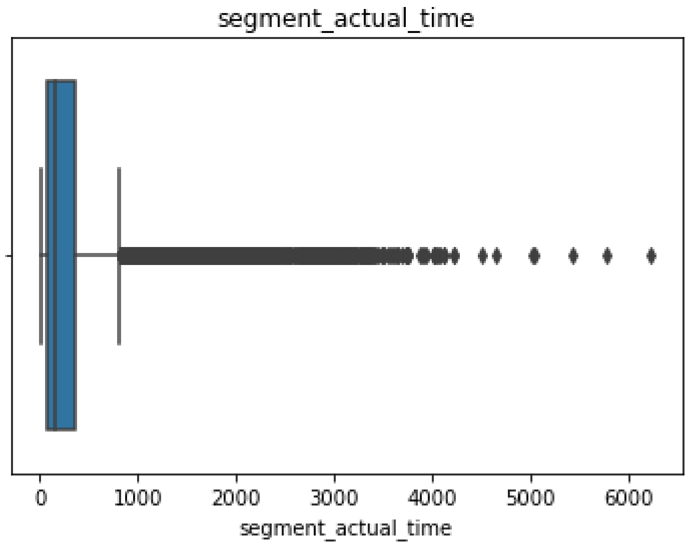


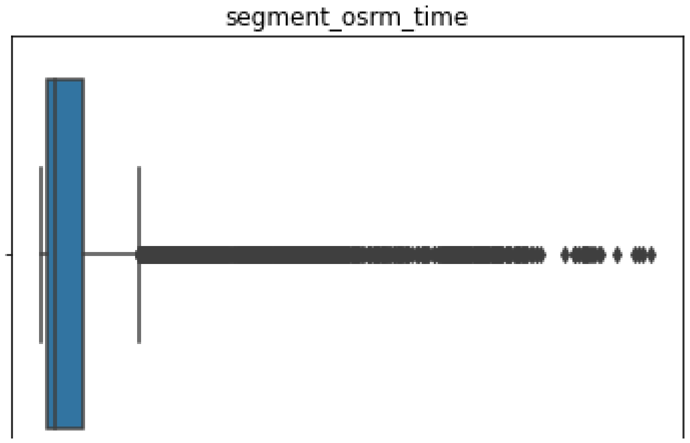


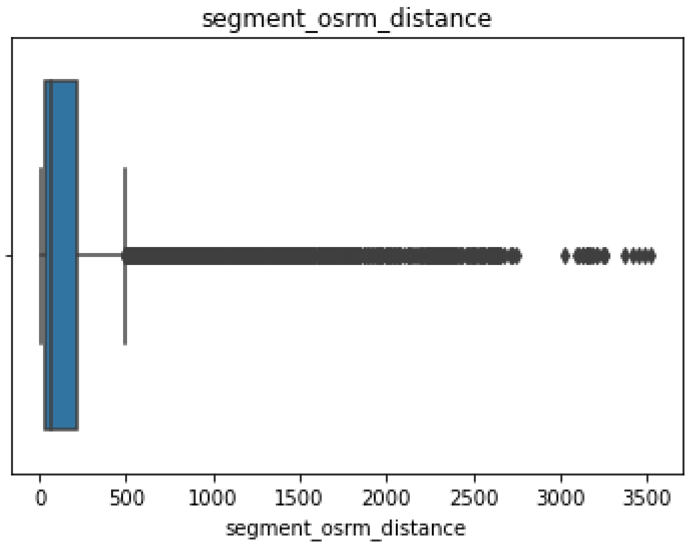


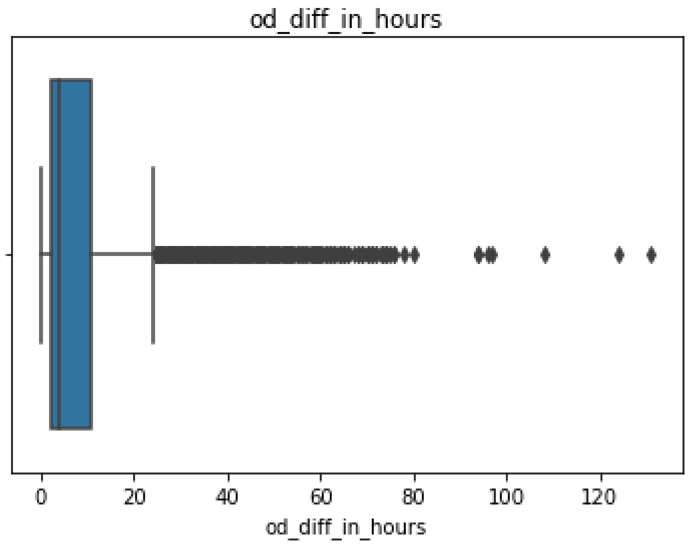

osrm_time
0
250
500
750
1000
1250
1500
1750
2000
osrm_time

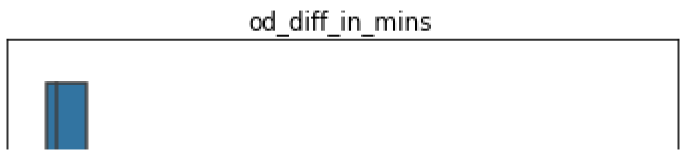

osrm_distance
0
500
1000
1500
2000
2500
osrm_distance


segment_actual_time
0
1000
2000
3000
4000
5000
6000
segment_actual_time


segment_osrm_time


segment_osrm_distance
0
500
1000
1500
2000
2500
3000
3500
segment_osrm_distance


od_diff_in_hours
0
20
40
60
80
100
120
od_diff_in_hours



In [362]:

```
def check_outlier(df, col):
    """find outliers from a list based on IQR method: outside of the range of 1.5IQR"""
    q1, q3 = np.percentile(df[col], [25,75])
    iqr = q3 - q1
    upper_val = q3 + 1.5*iqr
    lower_val = q1 - 1.5*iqr

    df.loc[df[col]<lower_val, col] = np.nan
    df.loc[df[col]>upper_val, col] = np.nan

    return df
```

In [363]:

```
final_df[out_cols].describe()
```

Out[363]:

	start_scan_to_end_scan	actual_distance_to_destination	actual_time	osrm_time	osrm_dis
count	14,816.00	14,816.00	14,816.00	14,816.00	14,[illegible]
mean	530.71	164.68	357.11	162.07	[illegible]
std	658.63	305.57	561.40	272.31	[illegible]
min	23.00	9.00	9.00	6.00	
25%	149.00	22.86	67.00	29.00	
50%	280.00	48.49	149.00	60.00	
75%	637.00	164.77	369.25	169.00	[illegible]
max	7,898.00	2,187.48	6,265.00	2,032.00	2,[illegible]

In [364]:

```
for col in out_cols:
    final_df = check_outlier(final_df, col)
```

In [365]:

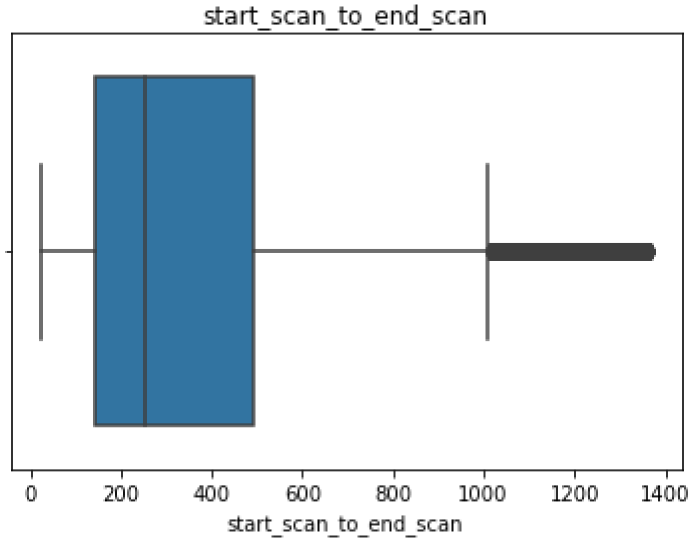
```
final_df[out_cols].describe()
```

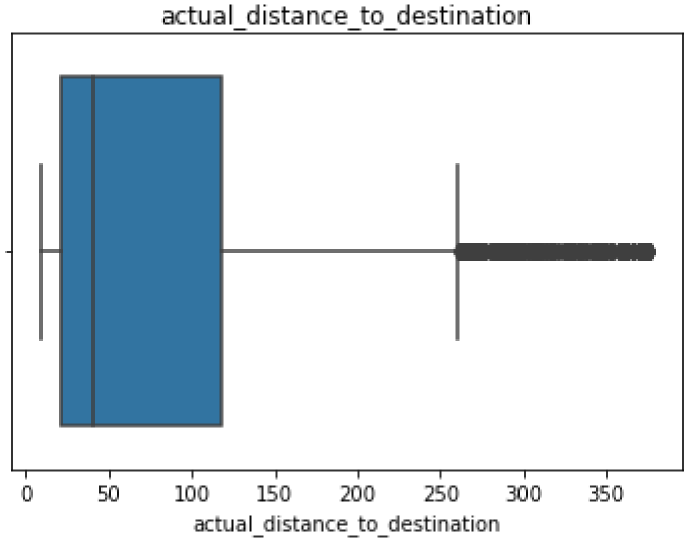
Out[365]:

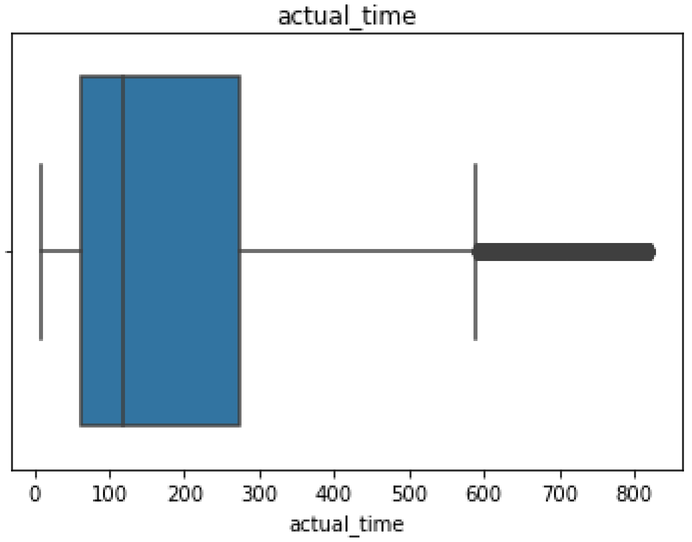
	start_scan_to_end_scan	actual_distance_to_destination	actual_time	osrm_time	osrm_dis
count	13,550.00	13,367.00	13,171.00	13,300.00	13,2
mean	367.79	80.15	193.78	85.55	1
std	313.03	81.57	180.02	80.28	
min	23.00	9.00	9.00	6.00	
25%	142.00	21.69	62.00	28.00	
50%	251.00	40.61	120.00	52.00	
75%	489.75	117.24	273.00	122.00	1
max	1,368.00	376.58	822.00	379.00	4

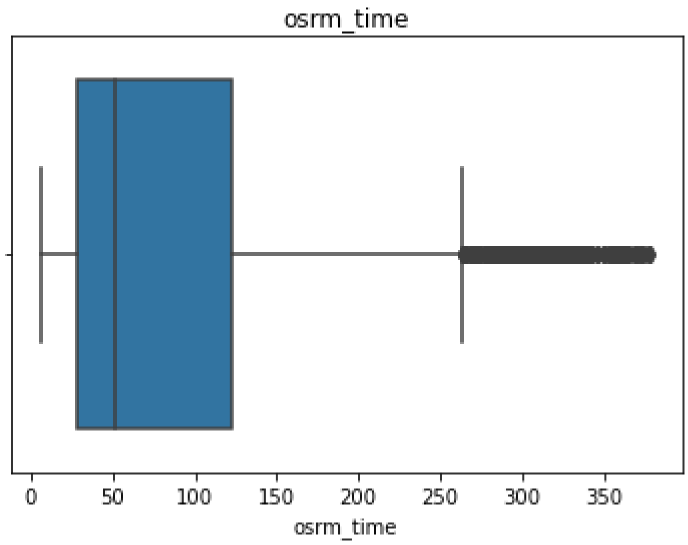
In [317]:

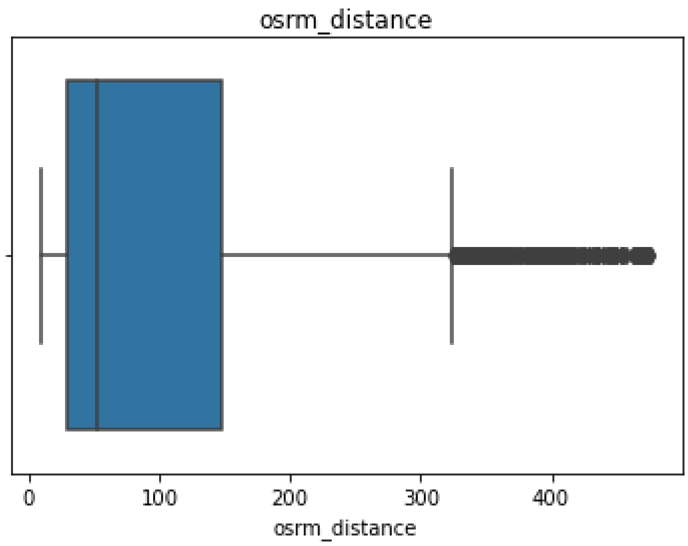
```
for col in out_cols:
    sns.boxplot(x= final_df[col])
    plt.title(col)
    plt.show()
```

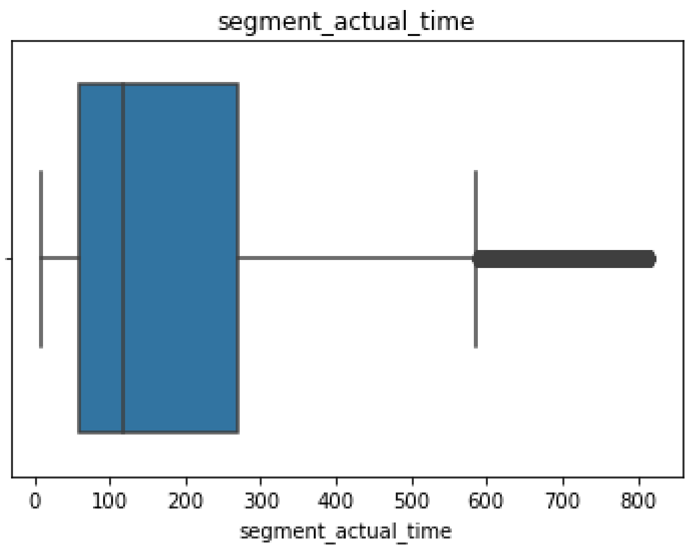


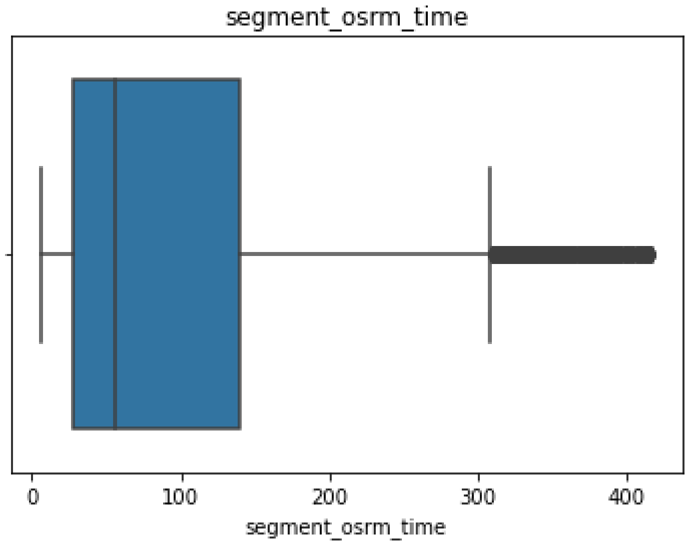


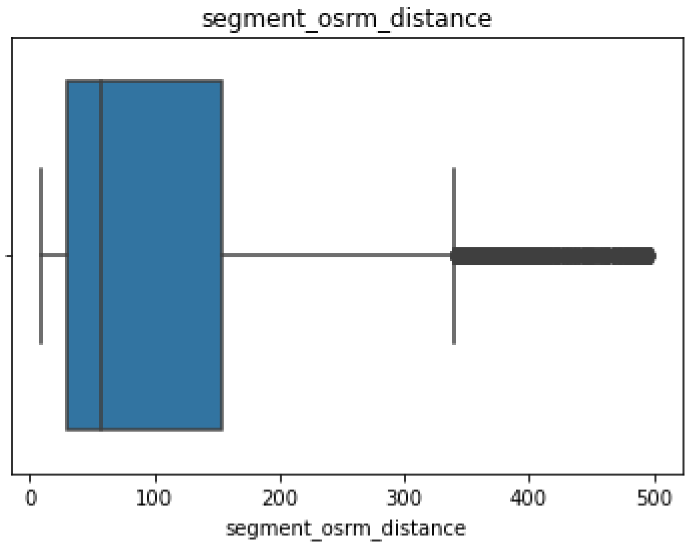


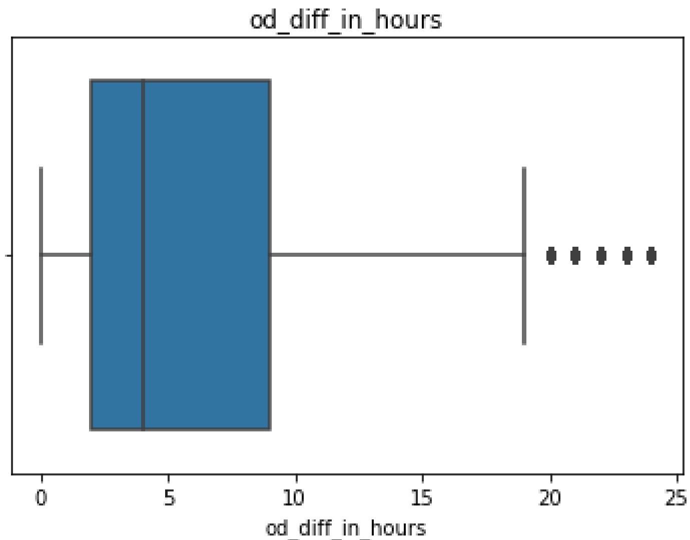

osrm_time
0
50
100
150
200
250
300
350
osrm_time

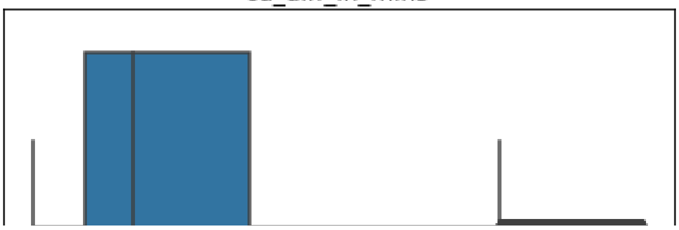

osrm_distance
0
100
200
300
400
osrm_distance


segment_actual_time
0
100
200
300
400
500
600
700
800
segment_actual_time


segment_osrm_time
0
100
200
300
400
segment_osrm_time


segment_osrm_distance
0
100
200
300
400
500
segment_osrm_distance


od_diff_in_hours
0
5
10
15
20
25
od_diff_in_hours



5. One-hot encoding

In [310]:

```
final_df
```

Out[310]:

	trip_uuid	data	route_schedule_uuid	route_type	start_scan_to_end_scar
0	trip-153671041653548748	training	thanos::sroute:d7c989ba-a29b-4a0b-b2f4-288cdc6...	FTL	2259.(
1	trip-153671042288605164	training	thanos::sroute:3a1b0ab2-bb0b-4c53-8c59-eb2a2c0...	Carting	180.(
2	trip-153671043369099517	training	thanos::sroute:de5e208e-7641-45e6-8100-4d9fb1e...	FTL	3933.(
3	trip-153671046011330457	training	thanos::sroute:f0176492-a679-4597-8332-bbd1c7f...	Carting	100.(
4	trip-153671052974046625	training	thanos::sroute:d9f07b12-65e0-4f3b-bec8-df06134...	FTL	717.(
...	...	...	...	...	..
14811	trip-153861095625827784	test	thanos::sroute:8a120994-f577-4491-9e4b-b7e4a14...	Carting	257.(
14812	trip-153861104386292051	test	thanos::sroute:b30e1ec3-3bfa-4bd2-a7fb-3b75769...	Carting	60.(
14813	trip-153861106442901555	test	thanos::sroute:5609c268-e436-4e0a-8180-3db4a74...	Carting	421.(
14814	trip-153861115439069069	test	thanos::sroute:c5f2ba2c-8486-4940-8af6-d1d2a6a...	Carting	347.(
14815	trip-153861118270144424	test	thanos::sroute:412fea14-6d1f-4222-8a5f-a517042...	FTL	353.(

14816 rows × 32 columns

In [320]:

```
ohe_cols = ['data','route_type']
```

In [323]:

```
def create_ohe(df, col):
    one_hot = pd.get_dummies(df[col])
    new_cols = {i:col+"_"+i for i in one_hot.columns}
    one_hot = one_hot.rename(columns=new_cols)
    # Drop column B as it is now encoded
    df = df.drop(col,axis = 1)
    # Join the encoded df
    df = df.join(one_hot)
    return df
```

In [325]:

```
for col in ohe_cols:
    final_df = create_ohe(final_df, col)
```

6. Standardization/Normalization

In [328]:

```
num_cols = final_df.select_dtypes(['int','float']).columns
```

In [329]:

```
num_cols
```

Out[329]:

```
Index(['start_scan_to_end_scan', 'actual_distance_to_destination',
       'actual_time', 'osrm_time', 'osrm_distance', 'segment_actual_time',
       'segment_osrm_time', 'segment_osrm_distance', 'od_diff_in_hours',
       'od_diff_in_mins'],
      dtype='object')
```

In [331]:

```
minmax_cols = [i for i in num_cols if 'time' in i]
std_cols = [i for i in num_cols if i not in minmax_cols]
```

In [334]:

```
final_df[minmax_cols]
```

Out[334]:

	actual_time	osrm_time	segment_actual_time	segment_osrm_time
0	NaN	NaN	NaN	NaN
1	143.0	68.0	141.0	65.0
2	NaN	NaN	NaN	NaN
3	59.0	15.0	59.0	16.0
4	341.0	117.0	340.0	115.0
...	...	...	...	...
14811	83.0	62.0	82.0	62.0
14812	21.0	12.0	21.0	11.0
14813	282.0	54.0	281.0	88.0
14814	264.0	184.0	258.0	221.0
14815	275.0	68.0	274.0	67.0

14816 rows × 4 columns

In [336]:

```
from sklearn.preprocessing import MinMaxScaler, StandardScaler
# define data

# define min max scaler
scaler = MinMaxScaler()
# transform data
scaled = scaler.fit_transform(final_df[minmax_cols])
final_df[minmax_cols] = scaled
```

In [339]:

```
final_df[minmax_cols]
```

Out[339]:

	actual_time	osrm_time	segment_actual_time	segment_osrm_time
0	NaN	NaN	NaN	NaN
1	0.164822	0.166220	0.163164	0.143902
2	NaN	NaN	NaN	NaN
3	0.061501	0.024129	0.061805	0.024390
4	0.408364	0.297587	0.409147	0.265854
...	...	...	...	...
14811	0.091021	0.150134	0.090235	0.136585
14812	0.014760	0.016086	0.014833	0.012195
14813	0.335793	0.128686	0.336218	0.200000
14814	0.313653	0.477212	0.307787	0.524390
14815	0.327183	0.166220	0.327565	0.148780

14816 rows × 4 columns

In [340]:

```
final_df[std_cols]
```

Out[340]:

	start_scan_to_end_scan	actual_distance_to_destination	osrm_distance	segment_osrm_di
0	NaN	NaN	NaN	
1	180.0	73.186911	85.1110	8
2	NaN	NaN	NaN	
3	100.0	17.175274	19.6800	1
4	717.0	127.448500	146.7918	14
...	...	...	...	
14811	257.0	57.762332	73.4630	6
14812	60.0	15.513784	16.0882	1
14813	421.0	38.684839	63.2841	10
14814	347.0	134.723836	177.6635	22
14815	353.0	66.081533	80.5787	8

14816 rows × 6 columns

In [341]:

```
scaler = StandardScaler()
# transform data
scaled = scaler.fit_transform(final_df[std_cols])
final_df[std_cols] = scaled
```

In [342]:

```
final_df[std_cols]
```

Out[342]:

	start_scan_to_end_scan	actual_distance_to_destination	osrm_distance	segment_osrm_di
0	NaN	NaN	NaN	
1	-0.599916	-0.085359	-0.155451	-0.2
2	NaN	NaN	NaN	
3	-0.855491	-0.772037	-0.812812	-0.8
4	1.115630	0.579864	0.464234	0.3
...	...	...	...	
14811	-0.353925	-0.274458	-0.272474	-0.3
14812	-0.983278	-0.792407	-0.848897	-0.8
14813	0.170003	-0.508340	-0.374737	-0.0
14814	-0.066403	0.669056	0.774390	1.1
14815	-0.047235	-0.172468	-0.200985	-0.2

14816 rows × 6 columns

In [ ]:

```
Check from where most orders are coming from (State, Corridor etc)
Busiest corridor, avg distance between them, avg time taken
```

Business Insights

- Most orders are coming from Maharastra, Karnataka, Haryana, Tamil Nadu
- Most orders source dest are intra-state. Within same state. Ex: maharastra to maharastra. Karnataka to karnataka
- Most orders are sent from Gurgaon, Bengaluru, Mumbai, Bhiwandi
- Most orders are sent to Mumbai, Bengaluru, Gurgaon, Hyderabad
- Most ordering city source dest pairs are: (mumbai, mumbai), (bengaluru, bengaluru), (bhiwandi, mumbai), (hyd, hyd)
- Median distance is 40km. Which means mostly couriers are sent in 40km range.
- Median time required to reach destination is 120 mins. 2 hours

Recommendations

- Mostly people are sending within city or state. Increase/Improve logistics within city/state.
- Increase transportation service in high frequency cities like mumbai, bengaluru, hyd

- Increase logistics in top states like Maharastra, Karnataka, haryana, tamil nadu

In [ ]: