Delhivery Casestudy

Problem statement

Delhivery, India's leading integrated commmerce player in Fiscal in 2021, aims to increase the efficiency and profitability of the business based on data insights. The challenge includes cleaning and manipulating raw data, extracting meaningful features, and supporting the data science team in developing forecasting models.

Installing Packages

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import seaborn as sns

from scipy.stats import norm
from scipy.stats import ttest_1samp, ttest_ind
```

```
pd.set_option("display.max_columns",None)
```

Loading Dataset

```
data1 = pd.read_csv("/content/drive/MyDrive/Scaler_case_study/delhivery_data.csv")
```

```
data = data1.copy()
```

```
data.head()
```

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	source_center	source_name	destination_cente
0	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA
1	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA
2	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA
3	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA
4	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA

Analysing shape and structure of data

```
data.shape
```

```
(144867, 24)
```

- There are 144867 rows and 24 columns

```
data.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
```

- Time columns are in the type object. It need to be converted into type datetime

```
data["trip_creation_time"] = pd.to_datetime(data["trip_creation_time"])
data["od_start_time"] = pd.to_datetime(data["od_start_time"])
data["od_end_time"] = pd.to_datetime(data["od_end_time"])
data["cutoff_timestamp"] = pd.to_datetime(data["cutoff_timestamp"])
```

Missing value Detection

```
data.isnull().sum()
```

```
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
```

- There are 293 and 261 missing values in source_name and destination_name respectively

Handling the missing values

- Drop the null values as it has only very less data

```
data.dropna(inplace = True)
```

```
data.isnull().sum()
```

```
data                              0
trip_creation_time                0
route_schedule_uuid               0
route_type                        0
trip_uuid                         0
source_center                     0
source_name                       0
destination_center                0
destination_name                  0
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
```

What are range of dates available in the dataset for the delivery

```
print(data["trip_creation_time"].max()-data["trip_creation_time"].min())
print(data["trip_creation_time"].max())
print(data["trip_creation_time"].min())
```

```
21 days 23:59:26.165951
2018-10-03 23:59:42.701692
2018-09-12 00:00:16.535741
```

- There is 21 days of data available in the dataset from 12th september 2018 to 3rd Oct 2018

```
data["trip_creation_time"].dt.month_name().value_counts()
```

```
September    126932
October       17384
Name: trip_creation_time, dtype: int64
```

Unique trip in the dataset

```
data.nunique()
```

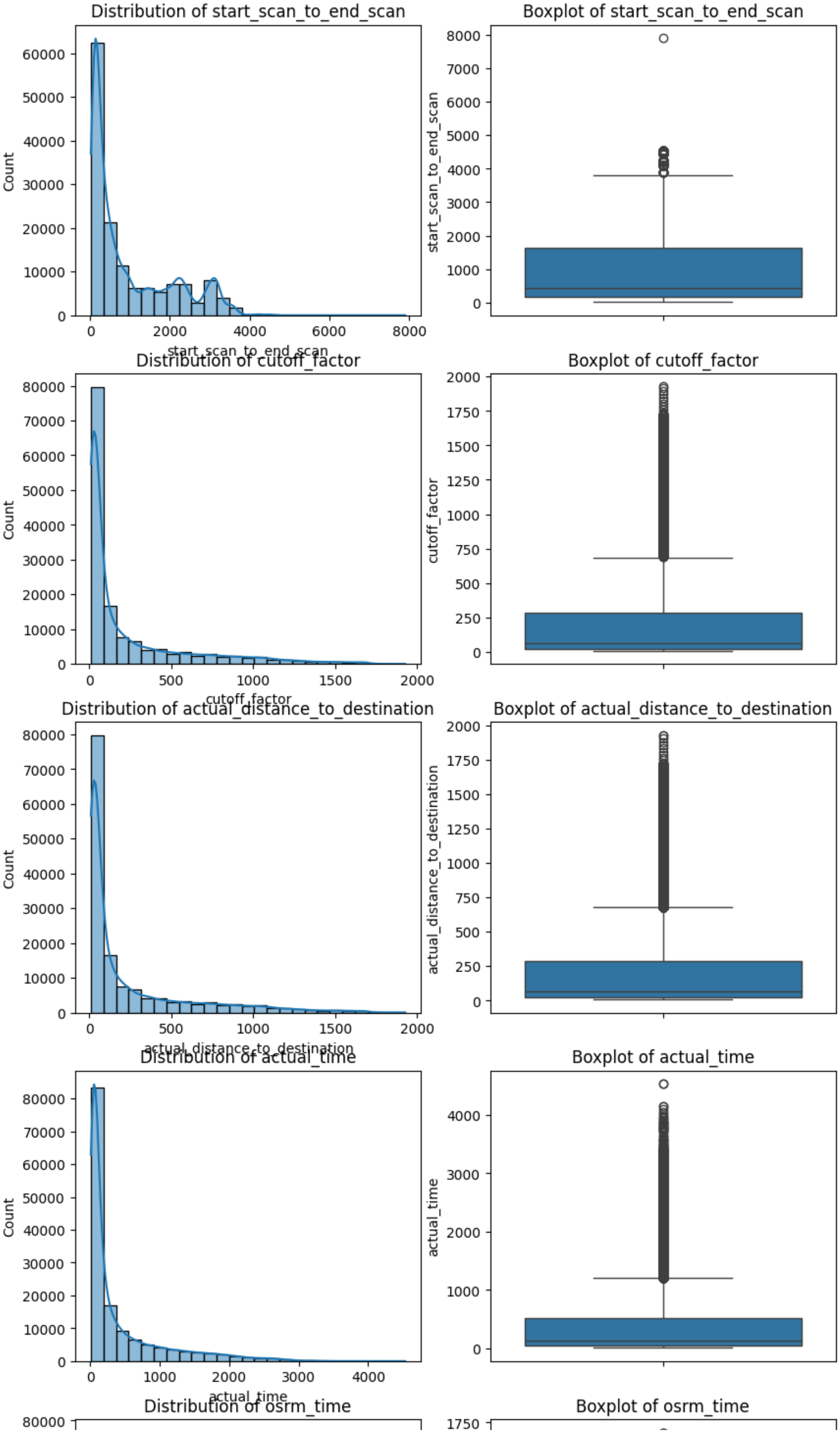
```
data                                   2
trip_creation_time                 14787
route_schedule_uuid                 1497
route_type                             2
trip_uuid                          14787
source_center                       1496
source_name                         1496
destination_center                  1466
destination_name                    1466
od_start_time                      26223
od_end_time                        26223
start_scan_to_end_scan              1914
is_cutoff                              2
cutoff_factor                        501
cutoff_timestamp                   92894
actual_distance_to_destination    143965
actual_time                         3182
```

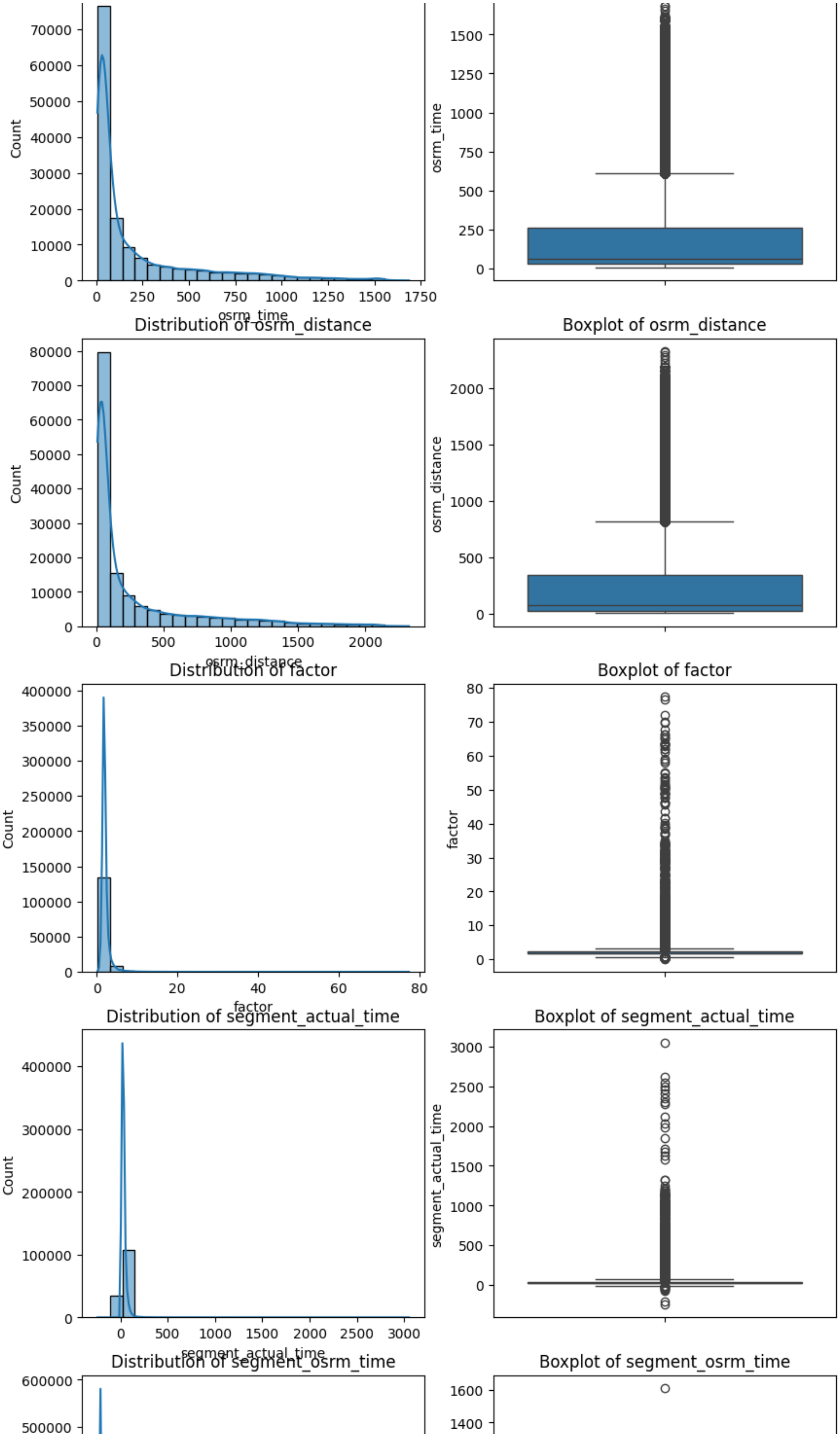
```
osrm_time                          1531
osrm_distance                    137544
factor                            45588
segment_actual_time                 746
segment_osrm_time                   214
segment_osrm_distance            113497
segment_factor                     5663
dtype: int64
```

- There are 14787 unique trip are available
- No of source center and destination center are 1496 and 1466 respectively
- There are 1497 unique routes are available

Visual Analysis

```
num_col = data.select_dtypes(include=np.number).columns.tolist()
fig, ax = plt.subplots(11,2,figsize = (10,50))

for i in range(len(num_col)):
    sns.histplot(data[num_col[i]], kde=True, ax=ax[i, 0],bins=25)
    ax[i,0].set_title(f"Distribution of {num_col[i]}")
    sns.boxplot(data[num_col[i]],ax = ax[i,1])
    ax[i,1].set_title(f"Boxplot of {num_col[i]}")
plt.show()
```


Distribution of start_scan_to_end_scan
Count
60000
50000
40000
30000
20000
10000
0
0
2000
4000
6000
8000
start_scan_to_end_scan
Boxplot of start_scan_to_end_scan
start_scan_to_end_scan
8000
7000
6000
5000
4000
3000
2000
1000
0
Distribution of cutoff_factor
Count
80000
70000
60000
50000
40000
30000
20000
10000
0
0
500
1000
1500
2000
cutoff_factor
Boxplot of cutoff_factor
cutoff_factor
2000
1750
1500
1250
1000
750
500
250
0
Distribution of actual_distance_to_destination
Count
80000
70000
60000
50000
40000
30000
20000
10000
0
0
500
1000
1500
2000
actual_distance_to_destination
Boxplot of actual_distance_to_destination
actual_distance_to_destination
2000
1750
1500
1250
1000
750
500
250
0
Distribution of actual_time
Count
80000
70000
60000
50000
40000
30000
20000
10000
0
0
1000
2000
3000
4000
actual_time
Boxplot of actual_time
actual_time
4000
3000
2000
1000
0
Distribution of osrm_time
80000
Boxplot of osrm_time
1750



Distribution of osrm_distance

Boxplot of osrm_distance

Distribution of factor

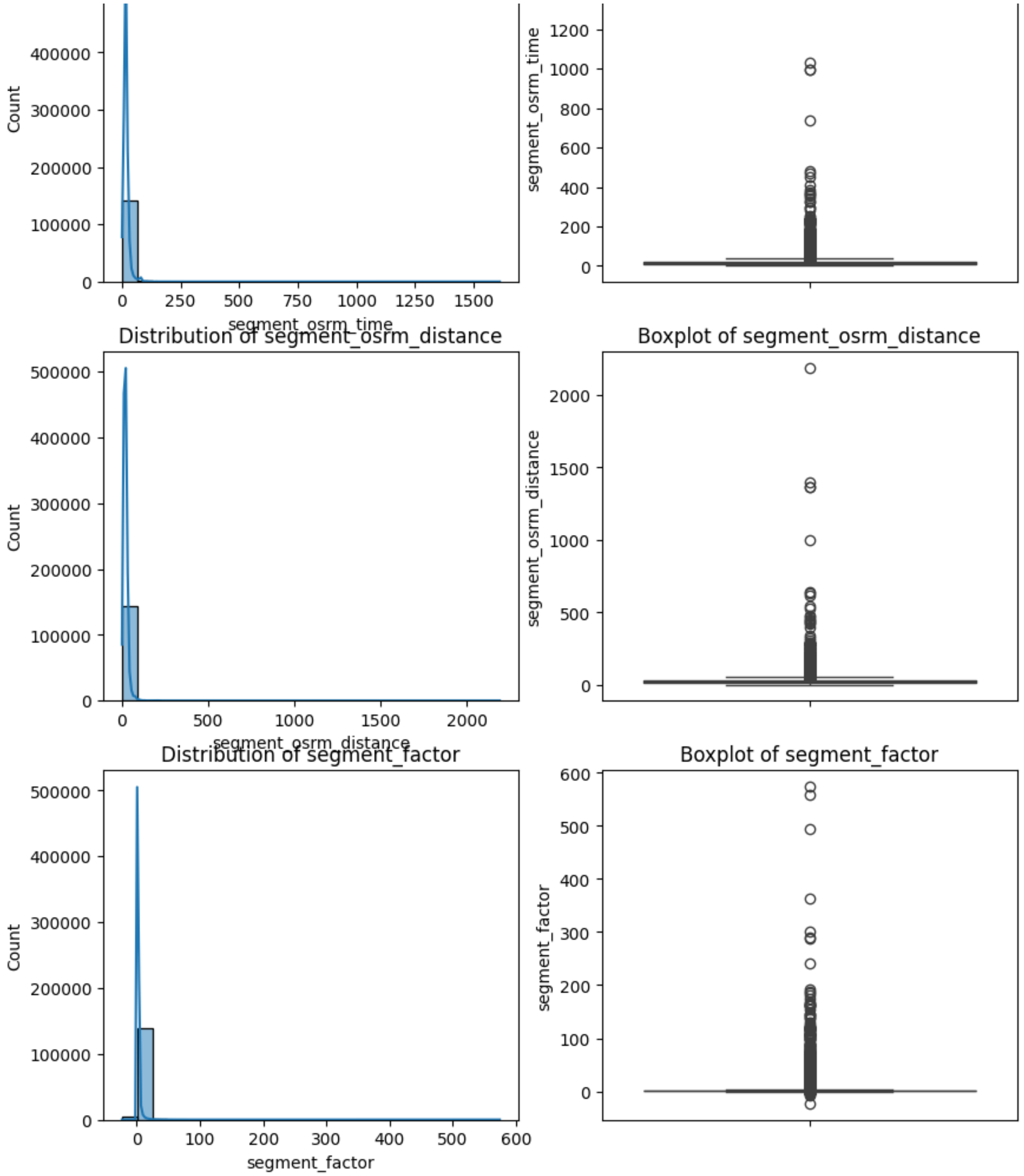
Boxplot of factor

Distribution of segment_actual_time

Boxplot of segment_actual_time

Distribution of segment_osrm_time

Boxplot of segment_osrm_time



Distribution of segment_osrm_distance

Boxplot of segment_osrm_distance

Distribution of segment_factor

Boxplot of segment_factor

Feature Creation

Extract the Source and Destination city name, state name and pincode from source and destination column

```
# Retrieving city name
data["source_city"] = data["source_name"].str.split(" ",expand = True,n=1)[0].str.split("_",n=1,expand=True)[0]
data["destination_city"] = data["destination_name"].str.split(" ",expand = True,n=1)[0].str.split("_",n=1,expand=True)[0]

# Retrieving state name
data["source_state"] = data["source_name"].str.split(" ",expand = True,n=1)[1].str.replace("(","").str.replace(")","")
data["destination_state"] = data["destination_name"].str.split(" ",expand = True,n=1)[1].str.replace("(","").str.replace(")","")

# Retrieving pincode
data["source_pin"] = data["source_center"].apply(lambda x:x[3:9])
data["destination_pin"] = data["destination_center"].apply(lambda x:x[3:9])
```

```
<ipython-input-16-9ec46a8e132d>:6: FutureWarning: The default value of regex will change from True to False in a future version. In addi
  data["source_state"] = data["source_name"].str.split(" ",expand = True,n=1)[1].str.replace("(","").str.replace(")","")
<ipython-input-16-9ec46a8e132d>:7: FutureWarning: The default value of regex will change from True to False in a future version. In addi
  data["destination_state"] = data["destination_name"].str.split(" ",expand = True,n=1)[1].str.replace("(","").str.replace(")","")
```

Convert time from minutes to hours

```
data["start_scan_to_end_scan"] = data["start_scan_to_end_scan"]/60
data["actual_time"] = data["actual_time"]/60
data["osrm_time"] = data["osrm_time"]/60
data["segment_actual_time"] = data["segment_actual_time"]/60
data["segment_osrm_time"] = data["segment_osrm_time"]/60
```

```
data.head()
```

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

Indepth Analysis and Feature Engineering

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 144316 entries, 0 to 144866
Data columns (total 30 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   data                            144316 non-null  object
 1   trip_creation_time              144316 non-null  datetime64[ns]
 2   route_schedule_uuid             144316 non-null  object
 3   route_type                      144316 non-null  object
 4   trip_uuid                       144316 non-null  object
 5   source_center                   144316 non-null  object
 6   source_name                     144316 non-null  object
 7   destination_center              144316 non-null  object
 8   destination_name                144316 non-null  object
 9   od_start_time                   144316 non-null  datetime64[ns]
 10  od_end_time                     144316 non-null  datetime64[ns]
 11  start_scan_to_end_scan          144316 non-null  float64
 12  is_cutoff                       144316 non-null  bool
 13  cutoff_factor                   144316 non-null  int64
 14  cutoff_timestamp                144316 non-null  datetime64[ns]
 15  actual_distance_to_destination  144316 non-null  float64
 16  actual_time                     144316 non-null  float64
 17  osrm_time                       144316 non-null  float64
 18  osrm_distance                   144316 non-null  float64
 19  factor                          144316 non-null  float64
 20  segment_actual_time             144316 non-null  float64
 21  segment_osrm_time               144316 non-null  float64
 22  segment_osrm_distance           144316 non-null  float64
 23  segment_factor                  144316 non-null  float64
 24  source_city                     144316 non-null  object
 25  destination_city                144316 non-null  object
 26  source_state                    144316 non-null  object
 27  destination_state               144316 non-null  object
 28  source_pin                      144316 non-null  object
 29  destination_pin                 144316 non-null  object
dtypes: bool(1), datetime64[ns](4), float64(10), int64(1), object(14)
memory usage: 33.2+ MB
```

```
data["source_state"] = data["source_state"].replace({"Road Punjab":"Punjab","Hub Maharashtra":"Maharashtra","Layout PC Karnataka":"Karnataka
```

```
data["destination_state"] = data["destination_state"].replace({"Delhi Delhi":"Delhi","West_Dc Maharashtra":"Maharashtra","Road Punjab":"Punj
```

Creating a feature with city and State

```
data["source_city_state"] = data["source_city"]+" "+data["source_state"]
data["destination_city_state"] = data["destination_city"]+" "+data["destination_state"]
```

```
data.head()
```

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	source_center	source_name	destination_cent
0	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA
1	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA
2	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA
3	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA
4	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IND388121AAA	Anand_VUNagar_DC (Gujarat)	IND388620AA

```
data["source_city_state"].nunique()
```

```
1248
```

Check any difference between start_scan_to_end_scan and Time taken from od_start_time to od_end_time

```
data["diff_od_start_end_time"] = (data["od_end_time"] - data["od_start_time"]).dt.total_seconds()/3600
```

Dropping Unnecessary columns

```
df = data.copy()
```

```
df.shape
```

```
(144316, 33)
```

```
df.drop(['source_center',"source_name","destination_center","destination_name","cutoff_timestamp", "od_end_time","od_start_time"],axis=1,inp
```

```
df.shape
```

```
(144316, 26)
```

Merging of rows and aggregation of fields

- Since the trip may include different source and destination, we need to aggregate to finds the feature of unique trip id

```
start_scan_to_end_scan = df.groupby("trip_uuid")["start_scan_to_end_scan"].agg(lambda x: sum(set(x))).reset_index()
```

```
diff_od_start_end_time = df.groupby("trip_uuid")["diff_od_start_end_time"].agg(lambda x: sum(set(x))).reset_index()
```

```
actual_time = df.groupby(["trip_uuid","start_scan_to_end_scan"])["actual_time"].max().reset_index().groupby("trip_uuid")["actual_time"].sum(
```

```
osrm_time = df.groupby(["trip_uuid","start_scan_to_end_scan"])["osrm_time"].max().reset_index().groupby("trip_uuid")["osrm_time"].sum().rese
```

```
osrm_distance = df.groupby(["trip_uuid","start_scan_to_end_scan"])["osrm_distance"].max().reset_index().groupby("trip_uuid")["osrm_distance"]
```

```
actual_distance_to_destination = df.groupby(["trip_uuid","start_scan_to_end_scan"])["actual_distance_to_destination"].max().reset_index().gr
```

```
segment_osrm_time = df.groupby("trip_uuid")["segment_osrm_time"].sum().reset_index()
```

```
segment_actual_time = df.groupby("trip_uuid")["segment_actual_time"].sum().reset_index()
```

```
segment_osrm_distance = df.groupby("trip_uuid")["segment_osrm_distance"].sum().reset_index()
```

Merging time and distance

```
merged_time = start_scan_to_end_scan.merge(diff_od_start_end_time.merge(diff_od_start_end_time.merge(actual_time.merge(osrm_time.merge(segmen
```

```
merged_distance = osrm_distance.merge(actual_distance_to_destination.merge(segment_osrm_distance),on="trip_uuid")
```

Merging Location

```
df.head()
```

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	start_scan_to_end_scan	is_cutoff	cutoff_factor	a
0	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	1.433333	True	9	
1	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	1.433333	True	18	
2	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	1.433333	True	27	
3	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	1.433333	True	36	
4	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	1.433333	False	39	

```
route_type = df.groupby("trip_uuid")["route_type"].unique().reset_index()
```

```
df2 = merged_time.merge(merged_distance.merge(route_type),on="trip_uuid")
df2
```

	trip_uuid	start_scan_to_end_scan	diff_od_start_end_time	actual_time
0	trip-153671041653548748	37.650000	37.668497	26.033333
1	trip-153671042288605164	3.000000	3.026865	2.383333
2	trip-153671043369099517	65.550000	65.572709	55.783333
3	trip-153671046011330457	1.666667	1.674916	0.983333
4	trip-153671052974046625	11.950000	11.972484	5.683333
...	...	...	...	...
14782	trip-153861095625827784	4.283333	4.300482	1.383333
14783	trip-153861104386292051	1.000000	1.009842	0.350000
14784	trip-153861106442901555	7.016667	7.035331	4.700000
14785	trip-153861115439069069	5.783333	5.808548	4.400000
14786	trip-153861118270144424	5.883333	5.906793	4.583333

14787 rows × 11 columns

```
df3 = df[["trip_uuid","source_city","destination_city","source_state","destination_state"]]
```

```
df4 = df3.merge(df2,on="trip_uuid")
df4
```

	trip_uuid	source_city	destination_city	source_state	destination_sta
0	trip-153741093647649320	Anand	Khambhat	Gujarat	Guja
1	trip-153741093647649320	Anand	Khambhat	Gujarat	Guja
2	trip-153741093647649320	Anand	Khambhat	Gujarat	Guja
3	trip-153741093647649320	Anand	Khambhat	Gujarat	Guja
4	trip-153741093647649320	Anand	Khambhat	Gujarat	Guja
...	...	...	...	...	
144311	trip-153746066843555182	Sonipat	Gurgaon	Haryana	Hary
144312	trip-153746066843555182	Sonipat	Gurgaon	Haryana	Hary
144313	trip-153746066843555182	Sonipat	Gurgaon	Haryana	Hary
144314	trip-153746066843555182	Sonipat	Gurgaon	Haryana	Hary
144315	trip-153746066843555182	Sonipat	Gurgaon	Haryana	Hary

144316 rows × 15 columns

```
df4['route_type'] = df4['route_type'].apply(lambda x: str(x))
df4 = df4.drop_duplicates()
```

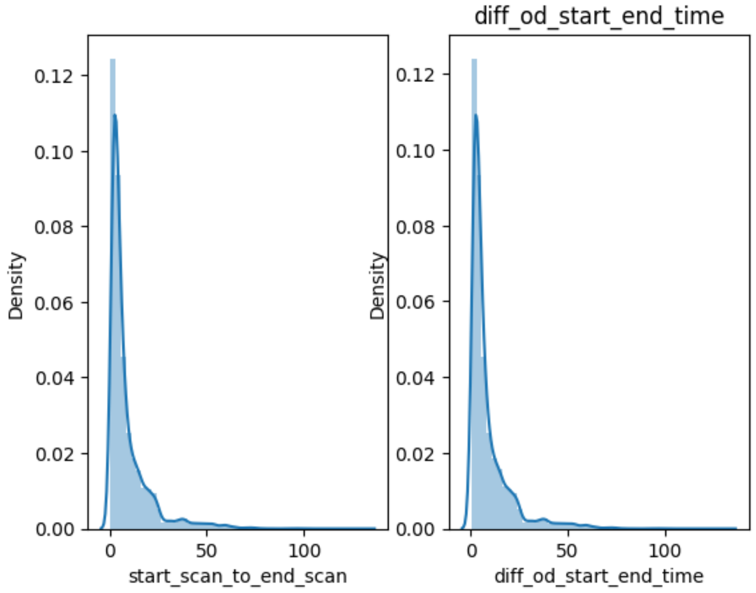
Hypothesis Test

1. Compare the difference between diff_od_start_end_time and start_scan_to_end_scan. Do hypothesis testing/ Visual analysis to check.

Visual Test

```
plt.subplot(1,2,1)
sns.distplot((start_scan_to_end_scan["start_scan_to_end_scan"]))
plt.subplot(1,2,2)
plt.title("start_scan_to_end_scan")
sns.distplot(diff_od_start_end_time["diff_od_start_end_time"],kde=True)
plt.title("diff_od_start_end_time")
plt.show()
```

```
<ipython-input-47-24f0fcc52033>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot((start_scan_to_end_scan["start_scan_to_end_scan"]))
<ipython-input-47-24f0fcc52033>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(diff_od_start_end_time["diff_od_start_end_time"],kde=True)
```



Hypothesis test

- H0: Mean of time taken from start to end time and start_scan_to_end_scan are equal
- Ha: Mean of time taken from start to end time and start_scan_to_end_scan are not equal

```
alpha = 0.05
t, p = ttest_ind(diff_od_start_end_time["diff_od_start_end_time"],start_scan_to_end_scan["start_scan_to_end_scan"])
t,p
```

```
(0.20369692885937926, 0.8385917439108145)
```

```
if p < alpha:
    print("Reject Ho: The distribution are different.")
else :
    print("Fail to reject Ho: The distribution is same.")
```

```
Fail to reject Ho: The distribution is same.
```

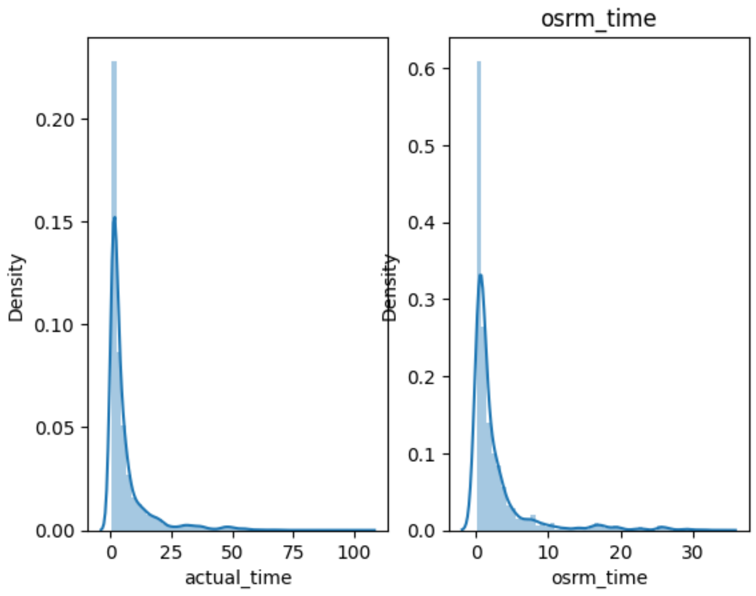
- **Conclusion:** Mean of time taken from start to end time and start_scan_to_end_scan are equal

2. 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)

Visual Test

```
plt.subplot(1,2,1)
sns.distplot((actual_time["actual_time"]))
plt.subplot(1,2,2)
plt.title("actual_time")
sns.distplot(osrm_time["osrm_time"],kde=True)
plt.title("osrm_time")
plt.show()
```

```
<ipython-input-50-27cfefb9b0f1>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot((actual_time["actual_time"]))
<ipython-input-50-27cfefb9b0f1>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(osrm_time["osrm_time"],kde=True)
```



Hypothesis test

- H0: Mean of time taken from actual_time and osrm_time are equal
- Ha: Mean of time taken from actual_time and osrm_time are not equal

```
alpha = 0.05
t, p = ttest_ind(actual_time["actual_time"],osrm_time["osrm_time"])
t,p
```

```
    (37.88902392536996, 8.547252231898765e-307)
```

```
if p < alpha:
    print("Reject Ho: The distribution are different.")
else :
    print("Fail to reject Ho: The distribution is same.")
```

```
    Reject Ho: The distribution are different.
```

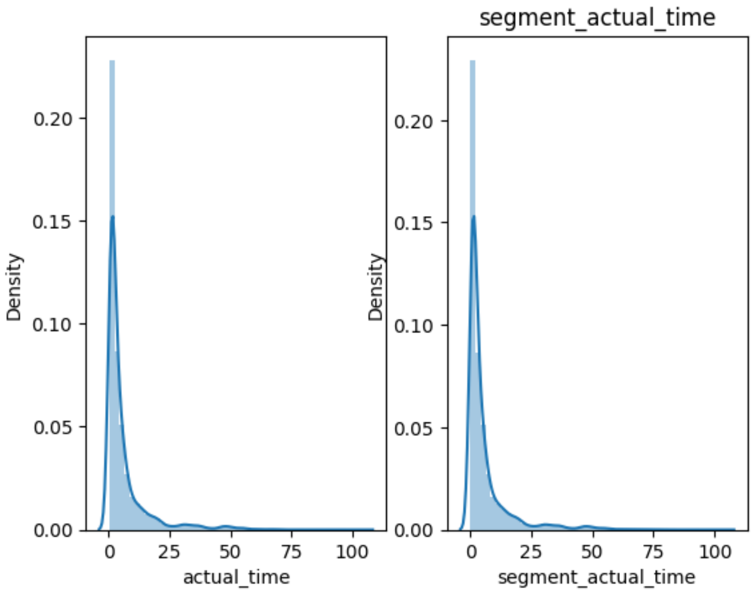
- **Conclusion:** Mean of time taken from actual_time and osrm_time are not equal

3. 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)

Visual Test

```
plt.subplot(1,2,1)
sns.distplot((actual_time["actual_time"]))
plt.subplot(1,2,2)
plt.title("actual_time")
sns.distplot(segment_actual_time["segment_actual_time"],kde=True)
plt.title("segment_actual_time")
plt.show()
```

```
<ipython-input-53-909ec77722de>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot((actual_time["actual_time"]))
<ipython-input-53-909ec77722de>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(segment_actual_time["segment_actual_time"],kde=True)
```



Hypothesis test

- H0: Mean of time taken from actual_time and segment_actual_time are equal
- Ha: Mean of time taken from actual_time and segment_actual_time are not equal

```
alpha = 0.05
t, p = ttest_ind(actual_time["actual_time"],segment_actual_time["segment_actual_time"])
t,p
```

```
(0.4330855966106441, 0.6649557448560977)
```

```
if p < alpha:
    print("Reject Ho: The distribution are different.")
else :
    print("Fail to reject Ho: The distribution is same.")
```

```
Fail to reject Ho: The distribution is same.
```

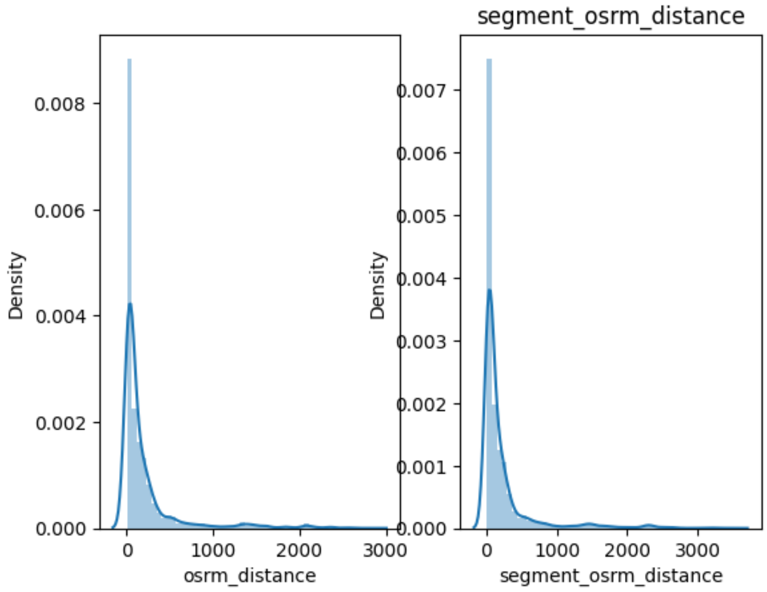
- **Conclusion:** Mean of time taken from actual_time and segment_actual_time are equal

4. Do hypothesis testing/ visual analysis between osrm distance aggregated value and segment osrm distance aggregated value (aggregated values are the values you'll get after merging the rows on the basis of trip_uuid)

Visual Test

```
plt.subplot(1,2,1)
sns.distplot((osrm_distance["osrm_distance"]))
plt.subplot(1,2,2)
plt.title("osrm_distance")
sns.distplot(segment_osrm_distance["segment_osrm_distance"],kde=True)
plt.title("segment_osrm_distance")
plt.show()
```

```
    <ipython-input-56-ae87b053458d>:2: UserWarning:

    `distplot` is a deprecated function and will be removed in seaborn v0.14.0.

    Please adapt your code to use either `displot` (a figure-level function with
    similar flexibility) or `histplot` (an axes-level function for histograms).

    For a guide to updating your code to use the new functions, please see
    https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

      sns.distplot((osrm_distance["osrm_distance"]))
    <ipython-input-56-ae87b053458d>:5: UserWarning:

    `distplot` is a deprecated function and will be removed in seaborn v0.14.0.

    Please adapt your code to use either `displot` (a figure-level function with
    similar flexibility) or `histplot` (an axes-level function for histograms).

    For a guide to updating your code to use the new functions, please see
    https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

      sns.distplot(segment_osrm_distance["segment_osrm_distance"],kde=True)
```



Hypothesis test

- H0: Mean of time taken from osrm_distance and segment_osrm_distance are equal
- Ha: Mean of time taken from osrm_distance and segment_osrm_distance are not equal

```
alpha = 0.05
t, p = ttest_ind(osrm_distance["osrm_distance"],segment_osrm_distance["segment_osrm_distance"])
t,p
```

```
(-3.994945182743642, 6.486579656555584e-05)
```

```
if p < alpha:
    print("Reject Ho: The distribution are different.")
else :
    print("Fail to reject Ho: The distribution is same.")
```

```
Reject Ho: The distribution are different.
```

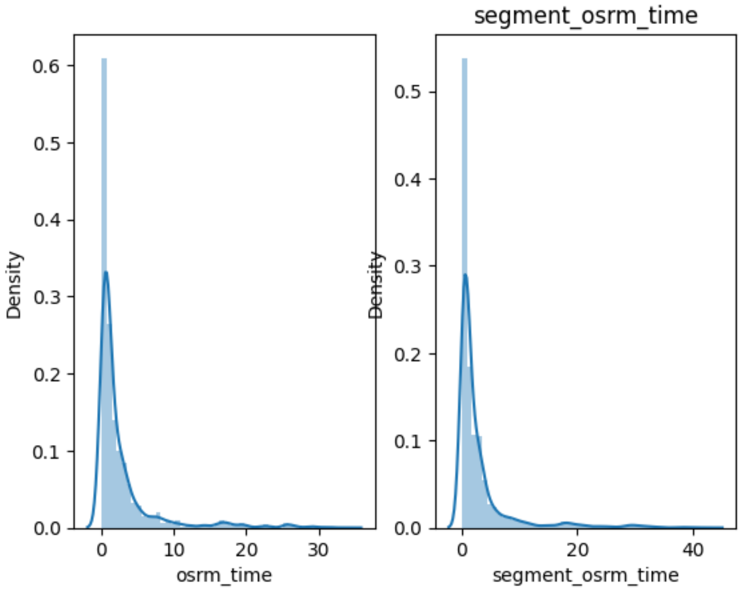
- **Conclusion:** Mean of time taken from osrm_distance and segment_osrm_distance are not equal

5. Do hypothesis testing/ visual analysis between osrm time aggregated value and segment osrm time aggregated value (aggregated values are the values you’ll get after merging the rows on the basis of trip_uuid)

Visual Test

```
plt.subplot(1,2,1)
sns.distplot((osrm_time["osrm_time"]))
plt.subplot(1,2,2)
plt.title("osrm_time")
sns.distplot(segment_osrm_time["segment_osrm_time"],kde=True)
plt.title("segment_osrm_time")
plt.show()
```

```
<ipython-input-59-9659f7003fab>:2: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot((osrm_time["osrm_time"]))
<ipython-input-59-9659f7003fab>:5: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(segment_osrm_time["segment_osrm_time"],kde=True)
```



Hypothesis test

- H0: Mean of time taken from osrm_time and segment_osrm_time are equal
- Ha: Mean of time taken from osrm_time and segment_osrm_time are not equal

```
alpha = 0.05
t, p = ttest_ind(osrm_time["osrm_time"],segment_osrm_time["segment_osrm_time"])
t,p
```

```
(-5.57306901255405, 2.5246923223978116e-08)
```

```
if p < alpha:
    print("Reject Ho: The distribution are different.")
else :
    print("Fail to reject Ho: The distribution is same.")
```

```
Reject Ho: The distribution are different.
```

- **Conclusion:** Mean of time taken from osrm_time and segment_osrm_time are not equal

Cleaned Data

```
data_merged = df4.copy()
```

```
data_merged.head()
```

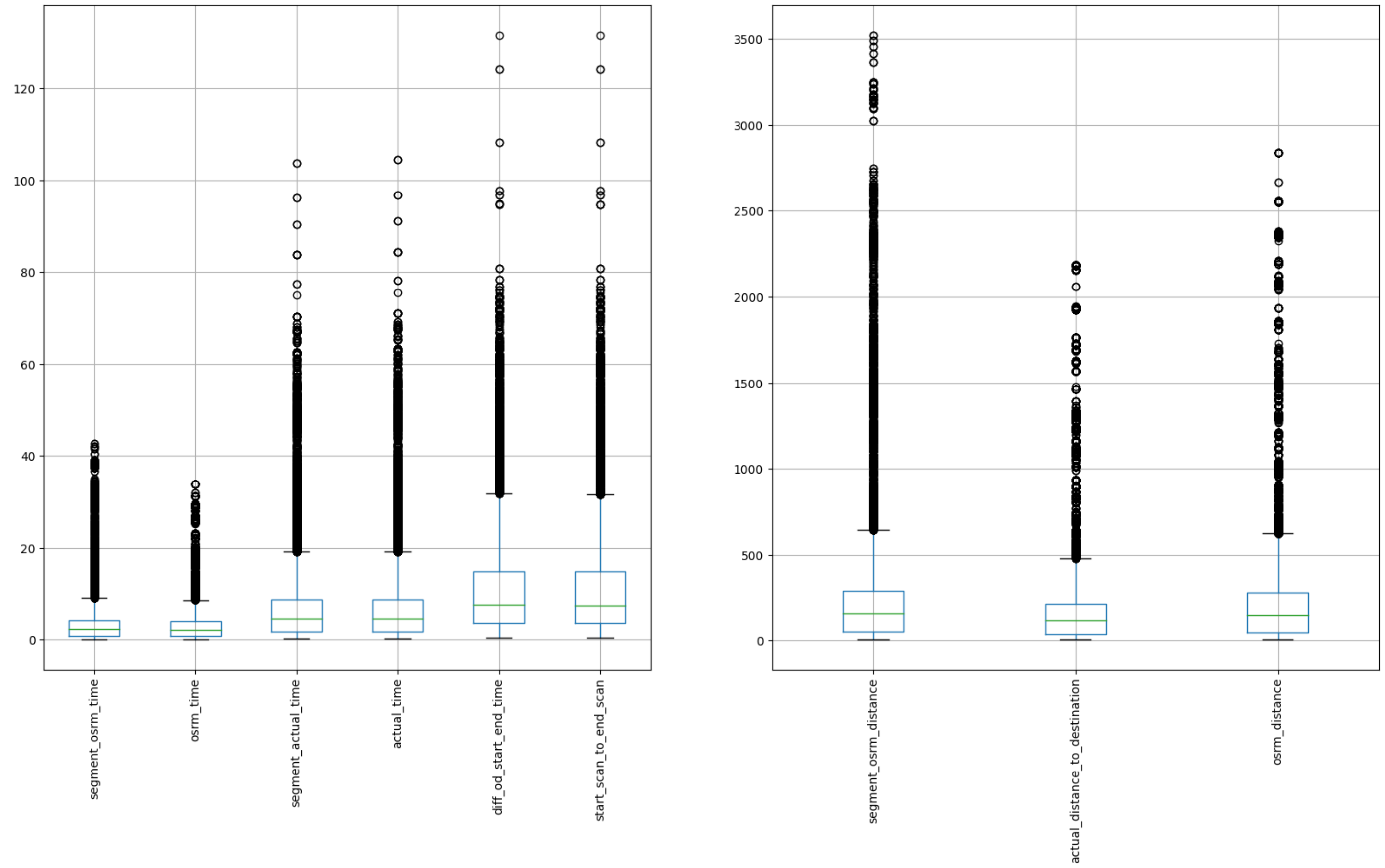
	trip_uuid	source_city	destination_city	source_state	destination_state
0	trip-153741093647649320	Anand	Khambhat	Gujarat	Gujarat
5	trip-153741093647649320	Khambhat	Anand	Gujarat	Gujarat
10	trip-153768492602129387	Bhiwandi	Pune	Maharashtra	Maharashtra
15	trip-153693976643699843	LowerParel	Mumbai	Maharashtra	Maharashtra
17	trip-153687145942424248	Bangalore	Bengaluru	Karnataka	Karnataka

```
data_merged.shape
```

```
(25927, 15)
```

Outlier Detection

```
plt.figure(figsize = (20,10))
plt.subplot(121)
data_merged[['segment_osrm_time', 'osrm_time','segment_actual_time', 'actual_time','diff_od_start_end_time', 'start_scan_to_end_scan']].boxpl
plt.xticks(rotation =90)
plt.subplot(122)
data_merged[['segment_osrm_distance', 'actual_distance_to_destination','osrm_distance']].boxplot()
plt.xticks(rotation =90)
plt.show()
```



Outlier treatment

```
data_merged_otl = data_merged.copy()
column_names = data_merged_otl[['segment_osrm_time', 'osrm_time','segment_actual_time', 'actual_time','diff_od_start_end_time', 'start_scan_
from scipy import stats

for column_name in column_names:
    z_scores = stats.zscore(data_merged_otl[column_name])
    threshold = 3
    outliers = data_merged_otl[abs(z_scores) > threshold]
    data_merged_otl[column_name] = np.where(abs(z_scores) > threshold, data_merged_otl[column_name].mean(), data_merged_otl[column_name])


data_merged_otl.head()
```

	trip_uuid	source_city	destination_city	source_state	destination_state
0	trip-153741093647649320	Anand	Khambhat	Gujarat	Gujarat
5	trip-153741093647649320	Khambhat	Anand	Gujarat	Gujarat
10	trip-153768492602129387	Bhiwandi	Pune	Maharashtra	Maharashtra
15	trip-153693976643699843	LowerParel	Mumbai	Maharashtra	Maharashtra
17	trip-153687145942424248	Bangalore	Bengaluru	Karnataka	Karnataka

```
data_merged_otl["route_type"].value_counts()
```

```
['FTL']        13797
['Carting']    12130
Name: route_type, dtype: int64
```

```
data_merged_otl["destination_state"].value_counts()
```

```
Karnataka                 3460
Maharashtra               3421
Tamil Nadu                2067
Haryana                   2014
Uttar Pradesh             1787
Telangana                 1531
West Bengal               1399
Gujarat                   1381
Andhra Pradesh            1308
Rajasthan                 1166
Punjab                    1132
Bihar                     1053
Kerala                     746
Madhya Pradesh             713
Delhi                      693
Assam                      455
Uttarakhand                367
Jharkhand                  306
Orissa                     259
Himachal Pradesh           241
Chandigarh                 123
Goa                         78
Arunachal Pradesh           55
Chhattisgarh                52
Jammu & Kashmir             45
Pondicherry                 31
Dadra and Nagar Haveli      17
Meghalaya                   13
Mizoram                     10
Tripura                      2
Daman & Diu                  1
Nagaland                     1
Name: destination_state, dtype: int64
```

Handling categorical values

```
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()

# Fit and transform the categorical column
data_merged["route_type-1"] = label_encoder.fit_transform(data_merged["route_type"])
data_merged
```

f_od_start_end_time	actual_time	osrm_time	segment_osrm_time	segment_actual_time	osrm_distance	actual_distance_to_destination	segme
3.256447	2.833333	1.483333	1.466667	2.783333	107.4515	82.981842	
3.256447	2.833333	1.483333	1.466667	2.783333	107.4515	82.981842	
5.039540	3.050000	1.583333	1.766667	3.000000	129.3519	100.708423	
1.816370	1.000000	0.266667	0.266667	1.000000	18.7941	16.431273	
17.194261	13.416667	8.100000	8.366667	13.316667	524.7155	371.458435	
...	...	...	...	...	...	...	
14.505827	10.416667	7.050000	8.450000	10.283333	518.8885	377.878467	
6.564614	5.733333	0.516667	0.600000	5.733333	33.7957	23.034042	
10.239925	4.816667	1.583333	1.850000	4.800000	129.1588	100.562078	
1.947858	1.400000	0.550000	0.516667	1.366667	36.7672	31.698687	
7.128106	7.100000	1.633333	3.083333	7.050000	111.2709	73.680667	

One hot encoder for state column

```
from sklearn.preprocessing import OneHotEncoder
enc = OneHotEncoder()
enc.fit(data_merged[["source_state"]])
```

```
▾ OneHotEncoder
OneHotEncoder()
```

```
feature_names = enc.get_feature_names_out(['source_state'])
encoded_data = enc.transform(data_merged[["source_state"]]).toarray()
encoded_df = pd.DataFrame(encoded_data, columns=feature_names)
encoded_df
```

ram	source_state_Nagaland	source_state_Orissa	source_state_Pondicherry	source_state_Punjab	source_state_Rajasthan	source_state_Tamil Nadu
0.0	0.0	0.0	0.0	0.0	0.0	0.(
0.0	0.0	0.0	0.0	0.0	0.0	0.(
0.0	0.0	0.0	0.0	0.0	0.0	0.(
0.0	0.0	0.0	0.0	0.0	0.0	0.(
0.0	0.0	0.0	0.0	0.0	0.0	0.(
...	...	...	...	...	...	..
0.0	0.0	0.0	0.0	0.0	0.0	0.(
0.0	0.0	0.0	0.0	0.0	0.0	0.(
0.0	0.0	0.0	0.0	0.0	0.0	0.(
0.0	0.0	0.0	0.0	0.0	0.0	1.(
0.0	0.0	0.0	0.0	0.0	0.0	0.(