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

```
data= pd.read_csv("https://d2beiqkhq929f0.cloudfront.net/public_assets/assets/000/001/551/original/delhivery_data.csv?1642751181")
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

5 rows × 24 columns

```
data.shape
```

```
(144867, 24)
```

```
data.size
```

```
3476808
```

```
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
```

```
data.isna().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
```

```
missing_source_name = data.loc[data["source_name"].isnull(), "source_center"].unique()
print(missing_source_name)

missing_destination_name = data.loc[data["destination_name"].isnull(), "destination_center"].unique()
print(missing_destination_name)
```

```
['IND342902A1B' 'IND577116AAA' 'IND282002AAD' 'IND465333A1B'
 'IND841301AAC' 'IND509103AAC' 'IND126116AAA' 'IND331022A1B'
 'IND505326AAB' 'IND852118A1B']
['IND342902A1B' 'IND577116AAA' 'IND282002AAD' 'IND465333A1B'
 'IND841301AAC' 'IND505326AAB' 'IND852118A1B' 'IND126116AAA'
 'IND509103AAC' 'IND221005A1A' 'IND250002AAC' 'IND331001A1C'
 'IND122015AAC']
```

```
data.describe().T
```

	count	mean	std	min	25%	
start_scan_to_end_scan	144867.0	961.262986	1037.012769	20.000000	161.000000	449
cutoff_factor	144867.0	232.926567	344.755577	9.000000	22.000000	66
actual_distance_to_destination	144867.0	234.073372	344.990009	9.000045	23.355874	66
actual_time	144867.0	416.927527	598.103621	9.000000	51.000000	132
osrm_time	144867.0	213.868272	308.011085	6.000000	27.000000	64
osrm_distance	144867.0	284.771297	421.119294	9.008200	29.914700	78
factor	144867.0	2.120107	1.715421	0.144000	1.604264	1
segment_actual_time	144867.0	36.196111	53.571158	-244.000000	20.000000	29
segment_osrm_time	144867.0	18.507548	14.775960	0.000000	11.000000	17
segment_osrm_distance	144867.0	22.829020	17.860660	0.000000	12.070100	23
segment_factor	144867.0	2.218368	4.847530	-23.444444	1.347826	1

```
data.describe(include="object").T
```

	count	unique	top	freq
data	144867	2	training	104858
trip_creation_time	144867	14817	2018-09-28 05:23:15.359220	101
route_schedule_uuid	144867	1504	thanos::sroute:4029a8a2-6c74-4b7e-a6d8-f9e069f...	1812
route_type	144867	2	FTL	99660
trip_uuid	144867	14817	trip-153811219535896559	101
source_center	144867	1508	IND000000ACB	23347
source_name	144574	1498	Gurgaon_Bilaspur_HB (Haryana)	23347
destination_center	144867	1481	IND000000ACB	15192
destination_name	144606	1468	Gurgaon_Bilaspur_HB (Haryana)	15192
od_start_time	144867	26369	2018-09-21 18:37:09.322207	81
od_end_time	144867	26369	2018-09-24 09:59:15.691618	81
cutoff_timestamp	144867	93180	2018-09-24 05:19:20	40

```
data["trip_creation_time"].min(), data["od_end_time"].max()
```

```
('2018-09-12 00:00:16.535741', '2018-10-08 03:00:24.353479')
```

```
#Removing null values as most of it is training data

data= data.dropna()
```

```
# Converting some columns from object datatype to required datatype

data["od_start_time"]= pd.to_datetime(data["od_start_time"])
data["od_end_time"]= pd.to_datetime(data["od_end_time"])
data["data"]= data["data"].astype("category")
data["route_type"]= data["route_type"].astype("category")
```

```
<ipython-input-141-f71bd55fa438>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data["od_start_time"]= pd.to_datetime(data["od_start_time"])
<ipython-input-141-f71bd55fa438>:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data["od_end_time"]= pd.to_datetime(data["od_end_time"])
<ipython-input-141-f71bd55fa438>:5: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data["data"]= data["data"].astype("category")
<ipython-input-141-f71bd55fa438>:6: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-cc
  data["route_type"]= data["route_type"].astype("category")
```

```
# Removing unknown fields

unknown_fields = ["is_cutoff", "cutoff_factor", "cutoff_timestamp", "factor", "segment_factor"]
data = data.drop(columns = unknown_fields)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 144316 entries, 0 to 144866
Data columns (total 19 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
```

```
 0   data                            144316 non-null  category
 1   trip_creation_time              144316 non-null  object
 2   route_schedule_uuid             144316 non-null  object
 3   route_type                      144316 non-null  category
 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  actual_distance_to_destination  144316 non-null  float64
 13  actual_time                     144316 non-null  float64
 14  osrm_time                       144316 non-null  float64
 15  osrm_distance                   144316 non-null  float64
 16  segment_actual_time             144316 non-null  float64
 17  segment_osrm_time               144316 non-null  float64
 18  segment_osrm_distance           144316 non-null  float64
dtypes: category(2), datetime64[ns](2), float64(8), object(7)
memory usage: 20.1+ MB
```

```
data["segment_key"]= data["trip_uuid"]+"_"+data["source_center"]+"_"+data["destination_center"]

data["segment_actual_time_sum"]= data.groupby("segment_key")["segment_actual_time"].cumsum()
data["segment_osrm_distance_sum"]= data.groupby("segment_key")["segment_osrm_distance"].cumsum()
data["segment_osrm_time_sum"]= data.groupby("segment_key")["segment_osrm_time"].cumsum()

data[["segment_key", "segment_actual_time_sum", "segment_osrm_distance_sum", "segment_osrm_time_sum"]]
```

	segment_key	segment_actual_time_sum	segment
0	trip-153741093647649320_IND388121AAA_IND388620AAB	14.0	
1	trip-153741093647649320_IND388121AAA_IND388620AAB	24.0	
2	trip-153741093647649320_IND388121AAA_IND388620AAB	40.0	
3	trip-153741093647649320_IND388121AAA_IND388620AAB	61.0	
4	trip-153741093647649320_IND388121AAA_IND388620AAB	67.0	
...	...	...	
144862	trip-153746066843555182_IND131028AAB_IND000000ACB	92.0	
144863	trip-153746066843555182_IND131028AAB_IND000000ACB	118.0	

```
# Merging rows

create_segment_dict={ "data" : "first",
                      "route_type" : "first",
                      "trip_creation_time" : "first",
                      "trip_uuid" : "first",
                      "source_center" : "first",
                      "source_name" : "first",
                      "destination_center" : "first",
                      "destination_name" : "last",
                      "od_start_time" : "first",
                      "od_end_time" : "first",
                      "start_scan_to_end_scan" : "first",
                      "actual_distance_to_destination" : "last",
                      "actual_time" : "last",
                      "osrm_time" : "last",
                      "osrm_distance" : "last",
                      "segment_actual_time" : "sum",
                      "segment_osrm_time" : "sum",
                      "segment_osrm_distance" : "sum"}

segmented_data = data.groupby(by= "segment_key", as_index = False).agg(create_segment_dict)
segmented_data = segmented_data.sort_values(by=["segment_key","od_end_time"], ascending=True)
segmented_data.head()
```

	segment_key	data	route_type	trip_creation_tim
0	trip-153671041653548748_IND209304AAA_IND000000ACB	training	FTL	2018-09-1 00:00:16.53574
1	trip-153671041653548748_IND462022AAA_IND209304AAA	training	FTL	2018-09-1 00:00:16.53574
2	trip-153671042288605164_IND561203AAB_IND562101AAA	training	Carting	2018-09-1 00:00:22.88643
3	trip-153671042288605164_IND572101AAA_IND561203AAB	training	Carting	2018-09-1 00:00:22.88643
4	trip-153671043369099517_IND000000ACB_IND160002AAC	training	FTL	2018-09-1 00:00:33.69125

```
# FeatureEngineering
# Calculate time taken between od_start_time and od_end_time

segmented_data["od_time_diff_hour"]= round(((segmented_data["od_end_time"] - segmented_data["od_start_time"]).dt.total_seconds()/3600),2)
segmented_data.head()
```

	segment_key	data	route_type	trip_creation_tim
0	trip-153671041653548748_IND209304AAA_IND000000ACB	training	FTL	2018-09-1 00:00:16.53574
1	trip-153671041653548748_IND462022AAA_IND209304AAA	training	FTL	2018-09-1 00:00:16.53574
2	trip-153671042288605164_IND561203AAB_IND562101AAA	training	Carting	2018-09-1 00:00:22.88643
3	trip-153671042288605164_IND572101AAA_IND561203AAB	training	Carting	2018-09-1 00:00:22.88643
4	trip-153671043369099517_IND000000ACB_IND160002AAC	training	FTL	2018-09-1 00:00:33.69125

```
# Splitting and extracting features out of destination

segmented_data[["Destination_city", "Destination_place_code_state"]] = segmented_data["destination_name"].str.split("_", 1, expand=True)
segmented_data[["Destination_place", "Destination_code_state"]] = segmented_data["Destination_place_code_state"].str.rstrip(")").str.split("_
segmented_data[["Destination_code", "Destination_state"]]= segmented_data["Destination_code_state"].str.split(" ", 1, expand= True)
segmented_data["Destination_state"]= segmented_data["Destination_state"].str[1:]
segmented_data.drop(["Destination_place_code_state", "Destination_code_state"], axis=1, inplace=True)
```

```
<ipython-input-147-fcffba9b37fc>:3: FutureWarning: In a future version of pandas all arguments of StringMethods.split except for the arg
  segmented_data[['Destination_city', 'Destination_place_code_state']] = segmented_data['destination_name'].str.split('_', 1, expand=Tru
<ipython-input-147-fcffba9b37fc>:4: FutureWarning: In a future version of pandas all arguments of StringMethods.split except for the arg
  segmented_data[['Destination_place', 'Destination_code_state']] = segmented_data['Destination_place_code_state'].str.rstrip(')').str.s
<ipython-input-147-fcffba9b37fc>:5: FutureWarning: In a future version of pandas all arguments of StringMethods.split except for the arg
  segmented_data[["Destination_code", "Destination_state"]]= segmented_data["Destination_code_state"].str.split(" ", 1, expand= True)
```

```
# Splitting and extracting features out of Source

segmented_data[["Source_city", "Source_place_code_state"]] = segmented_data["source_name"].str.split("_", 1, expand=True)
segmented_data[["Source_place", "Source_code_state"]] = segmented_data["Source_place_code_state"].str.rstrip(")").str.split("_", 1, expand=T
segmented_data[["Source_code", "Source_state"]]= segmented_data["Source_code_state"].str.split(" ", 1, expand= True)
segmented_data["Source_state"]= segmented_data["Source_state"].str[1:]
segmented_data.drop(["Source_place_code_state", "Source_code_state"], axis=1, inplace=True)
segmented_data.head()
```

```
<ipython-input-148-64f434af2676>:3: FutureWarning: In a future version of pandas all arg
  segmented_data[['Source_city', 'Source_place_code_state']] = segmented_data['source_na
<ipython-input-148-64f434af2676>:4: FutureWarning: In a future version of pandas all arg
  segmented_data[['Source_place', 'Source_code_state']] = segmented_data['Source_place_c
<ipython-input-148-64f434af2676>:5: FutureWarning: In a future version of pandas all arg
  segmented_data[["Source_code", "Source_state"]]= segmented_data["Source_code_state"].s
```

	segment_key	data	route_type	trip_creation_tim
0	trip- 153671041653548748_IND209304AAA_IND000000ACB	training	FTL	2018-09-1 00:00:16.53574
1	trip- 153671041653548748_IND462022AAA_IND209304AAA	training	FTL	2018-09-1 00:00:16.53574
2	trip- 153671042288605164_IND561203AAB_IND562101AAA	training	Carting	2018-09-1 00:00:22.88643
3	trip- 153671042288605164_IND572101AAA_IND561203AAB	training	Carting	2018-09-1 00:00:22.88643
4	trip- 153671043369099517_IND000000ACB_IND160002AAC	training	FTL	2018-09-1 00:00:33.69125

5 rows × 28 columns

```
segmented_data.head()
```

	segment_key	data	route_type	trip_creation_tim
0	trip- 153671041653548748_IND209304AAA_IND000000ACB	training	FTL	2018-09-1 00:00:16.53574
1	trip- 153671041653548748_IND462022AAA_IND209304AAA	training	FTL	2018-09-1 00:00:16.53574
2	trip- 153671042288605164_IND561203AAB_IND562101AAA	training	Carting	2018-09-1 00:00:22.88643
3	trip- 153671042288605164_IND572101AAA_IND561203AAB	training	Carting	2018-09-1 00:00:22.88643
4	trip- 153671043369099517_IND000000ACB_IND160002AAC	training	FTL	2018-09-1 00:00:33.69125

5 rows × 28 columns

```
#  Extracting features like month, year, day from trip_creation_time

segmented_data["trip_creation_day"] = pd.to_datetime(segmented_data["trip_creation_time"]).dt.day
segmented_data["trip_creation_month"] = pd.to_datetime(segmented_data["trip_creation_time"]).dt.month
segmented_data["trip_creation_year"] = pd.to_datetime(segmented_data["trip_creation_time"]).dt.year
```

```
segmented_data.head()
```

	segment_key	data	route_type	trip_creation_tim
0	trip- 153671041653548748_IND209304AAA_IND000000ACB	training	FTL	2018-09-1 00:00:16.53574
1	trip- 153671041653548748_IND462022AAA_IND209304AAA	training	FTL	2018-09-1 00:00:16.53574
2	trip- 153671042288605164_IND561203AAB_IND562101AAA	training	Carting	2018-09-1 00:00:22.88643
3	trip- 153671042288605164_IND572101AAA_IND561203AAB	training	Carting	2018-09-1 00:00:22.88643
4	trip- 153671043369099517_IND000000ACB_IND160002AAC	training	FTL	2018-09-1 00:00:33.69125

5 rows × 31 columns

```
segmented_data.describe(include= "object").T
```

	count	unique	top	freq
segment_key	26222	26222	trip-153671041653548748_IND209304AAA_IND000000ACB	1
trip_creation_time	26222	14787	2018-09-17 08:30:59.260046	8
trip_uuid	26222	14787	trip-153717306559016761	8
source_center	26222	1496	IND000000ACB	1052
source_name	26222	1496	Gurgaon_Bilaspur_HB (Haryana)	1052
destination_center	26222	1466	IND000000ACB	928
destination_name	26222	1466	Gurgaon_Bilaspur_HB (Haryana)	928
Destination_city	26222	1256	Bengaluru	1180
Destination_place	25238	1154	Central	1860
Destination_code	23208	48	D	9156
Destination_state	23208	31	Karnataka	3198
Source_city	26222	1260	Bengaluru	1136
Source_place	25399	1177	Central	1976
Source_code	23252	48	D	9139

```
create_trip_dic= {"data" : "first",
                  "route_type" : "first",
                  "trip_creation_time" : "first",
                  "trip_creation_day" : "first",
                  "trip_creation_month" : "first",
                  "trip_creation_year" : "first",
                  "source_center" : "first",
                  "source_name" : "first",
                  "Source_state" : "first",
                  "Source_city" : "first",
                  "Source_place" : "first",
                  "Source_code" : "first",
                  "destination_center" : "last",
                  "destination_name" : 'last',
                  "Destination_state" : "last",
                  "Destination_city" : "last",
                  "Destination_place" : "last",
                  "Destination_code" : "last",
                  "od_start_time" : "first",
                  "od_end_time" : "first",
                  "start_scan_to_end_scan" : "first",
                  "actual_distance_to_destination" : "last",
                  "actual_time" : "last",
                  "osrm_time" : "last",
                  "osrm_distance" : "last",
                  "segment_actual_time" : "sum",
                  "segment_osrm_time" : "sum",
                  "segment_osrm_distance" : "sum"
                  }
trip_data= segmented_data.groupby("trip_uuid", as_index= False).agg(create_trip_dic)
trip_data.head()
```

	trip_uuid	data	route_type	trip_creation_time	trip_creation_day	trip_
0	trip-153671041653548748	training	FTL	2018-09-12 00:00:16.535741	12	
1	trip-153671042288605164	training	Carting	2018-09-12 00:00:22.886430	12	
2	trip-153671043369099517	training	FTL	2018-09-12 00:00:33.691250	12	
3	trip-153671046011330457	training	Carting	2018-09-12 00:01:00.113710	12	
4	trip-153671052974046625	training	FTL	2018-09-12 00:02:09.740725	12	

5 rows × 29 columns

```
trip_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14787 entries, 0 to 14786
Data columns (total 29 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   trip_uuid                       14787 non-null  object
 1   data                            14787 non-null  category
 2   route_type                      14787 non-null  category
 3   trip_creation_time              14787 non-null  object
 4   trip_creation_day               14787 non-null  int64
 5   trip_creation_month             14787 non-null  int64
 6   trip_creation_year              14787 non-null  int64
 7   source_center                   14787 non-null  object
 8   source_name                     14787 non-null  object
 9   Source_state                    13564 non-null  object
 10  Source_city                     14787 non-null  object
 11  Source_place                    14277 non-null  object
 12  Source_code                     13564 non-null  object
 13  destination_center              14787 non-null  object
 14  destination_name                14787 non-null  object
 15  Destination_state               13508 non-null  object
 16  Destination_city                14787 non-null  object
 17  Destination_place               14165 non-null  object
 18  Destination_code                13508 non-null  object
 19  od_start_time                   14787 non-null  datetime64[ns]
 20  od_end_time                     14787 non-null  datetime64[ns]
 21  start_scan_to_end_scan          14787 non-null  float64
 22  actual_distance_to_destination  14787 non-null  float64
 23  actual_time                     14787 non-null  float64
 24  osrm_time                       14787 non-null  float64
 25  osrm_distance                   14787 non-null  float64
 26  segment_actual_time             14787 non-null  float64
 27  segment_osrm_time               14787 non-null  float64
 28  segment_osrm_distance           14787 non-null  float64
dtypes: category(2), datetime64[ns](2), float64(8), int64(3), object(14)
memory usage: 3.1+ MB
```

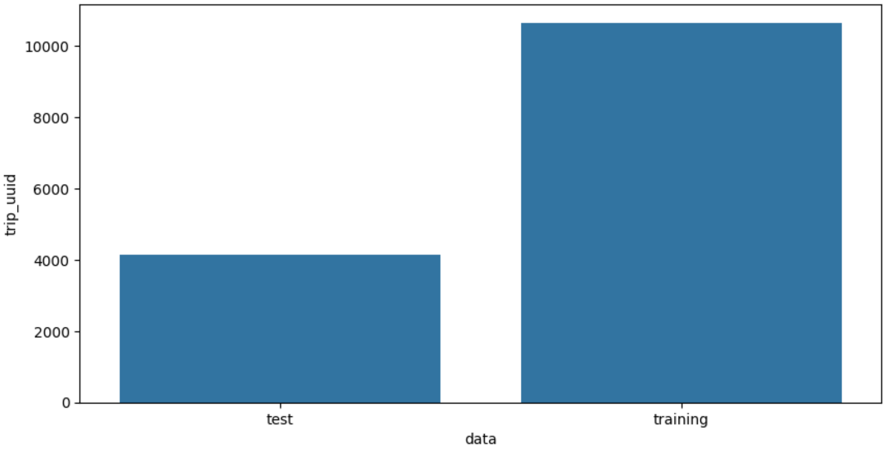
```
trip_data.describe().T
```

	count	mean	std	min	25%	
trip_creation_day	14787.0	18.375127	7.882198	1.000000	14.000000	
trip_creation_month	14787.0	9.120105	0.325096	9.000000	9.000000	
trip_creation_year	14787.0	2018.000000	0.000000	2018.000000	2018.000000	20
start_scan_to_end_scan	14787.0	339.769730	505.407155	22.000000	104.000000	1
actual_distance_to_destination	14787.0	104.005219	242.069053	9.002461	20.086307	
actual_time	14787.0	227.443836	443.875166	9.000000	51.000000	
osrm_time	14787.0	101.437817	213.971631	6.000000	23.000000	
osrm_distance	14787.0	129.210983	293.953554	9.072900	26.018550	
segment_actual_time	14787.0	353.059174	556.365911	9.000000	66.000000	1
segment_osrm_time	14787.0	180.511598	314.679279	6.000000	30.000000	
segment_osrm_distance	14787.0	222.705466	416.846279	9.072900	32.578850	

```
data_type= trip_data.groupby("data")["trip_uuid"].count().reset_index()
data_type
```

	data	trip_uuid
0	test	4142
1	training	10645

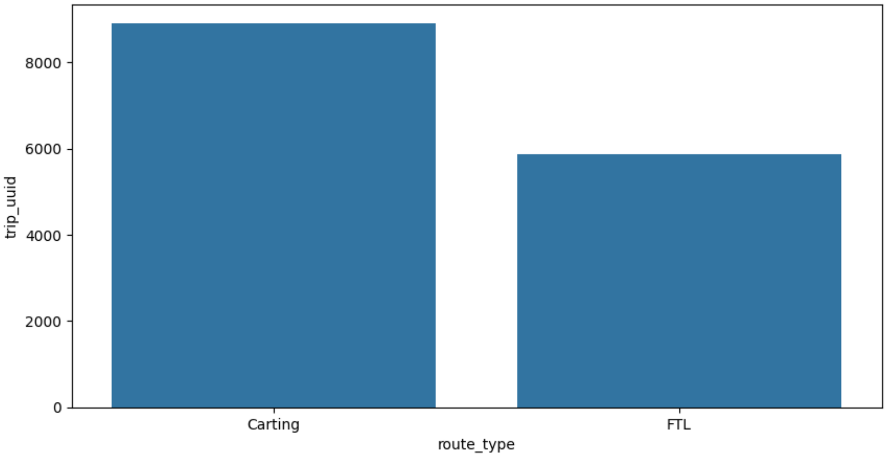
```
plt.figure(figsize = (10, 5))
sns.barplot(data= data_type, x="data", y="trip_uuid")
plt.show()
```



```
data_route= trip_data.groupby("route_type")["trip_uuid"].count().reset_index()
data_route
```

	route_type	trip_uuid
0	Carting	8906
1	FTL	5881

```
plt.figure(figsize = (10, 5))
sns.barplot(data= data_route, x="route_type", y="trip_uuid")
plt.show()
```

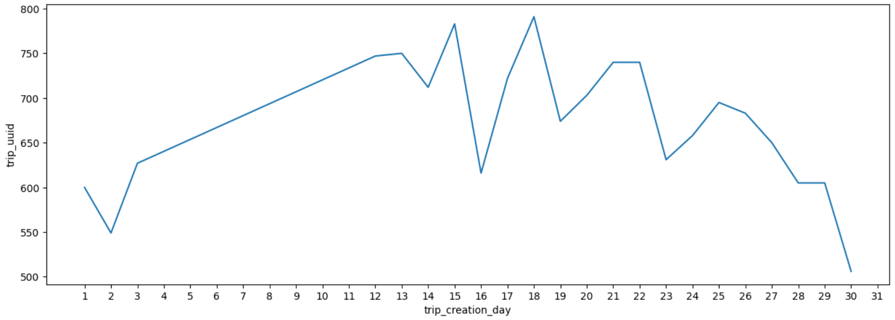


```
data_day= trip_data.groupby("trip_creation_day")["trip_uuid"].count().reset_index()
data_day= data_day.sort_values(by= "trip_uuid", ascending= False)
data_day
```

	trip_creation_day	trip_uuid
9	18	791
6	15	783
4	13	750
3	12	747
13	22	740
12	21	740
8	17	722
5	14	712
11	20	703
16	25	695
17	26	683
10	19	674
15	24	658
18	27	650
14	23	631
2	3	627
7	16	616
19	28	605
20	29	605
0	1	600
1	2	549
21	30	506

```
plt.figure(figsize = (15, 5))
sns.lineplot(data = data_day,
             x = data_day['trip_creation_day'],
             y = data_day['trip_uuid'])
plt.xticks(np.arange(1, 32))
plt.plot()
```

```
[]
```

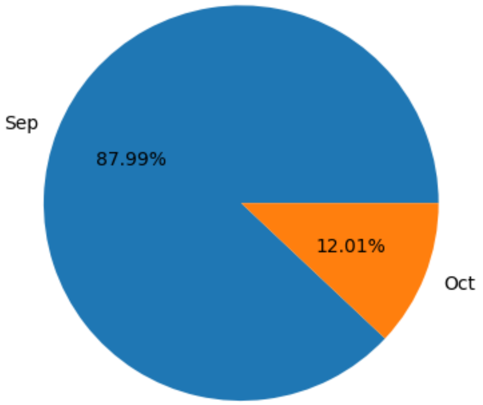


```
data_month= trip_data.groupby("trip_creation_month")["trip_uuid"].count().reset_index()
data_month= data_month.sort_values(by= "trip_uuid", ascending= False)
data_month
```

	trip_creation_month	trip_uuid
0	9	13011
1	10	1776

```
plt.pie(x= data_month["trip_uuid"], labels= ["Sep", "Oct"], autopct= "%.2f%%")
plt.plot()
```

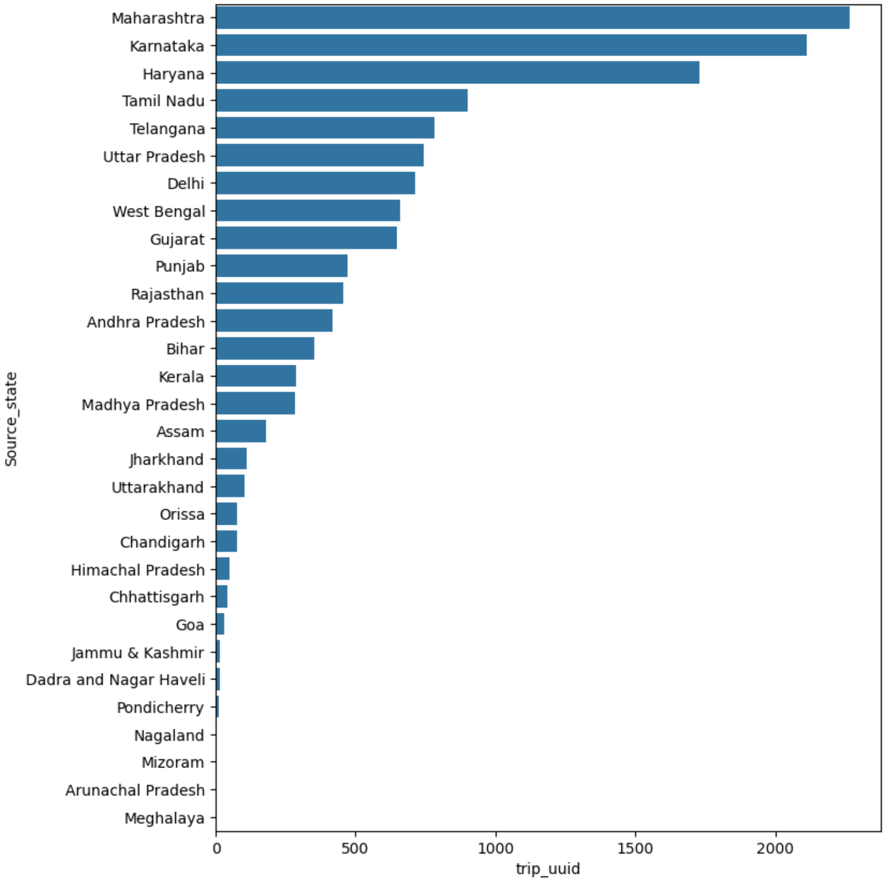
```
[]
```



```
data_source_state = trip_data.groupby(by = 'Source_state')['trip_uuid'].count().reset_index()
data_source_state = data_source_state.sort_values(by = 'trip_uuid', ascending = False)
data_source_state.head()
```

	Source_state	trip_uuid
17	Maharashtra	2264
14	Karnataka	2113
10	Haryana	1730
25	Tamil Nadu	902
26	Telangana	783

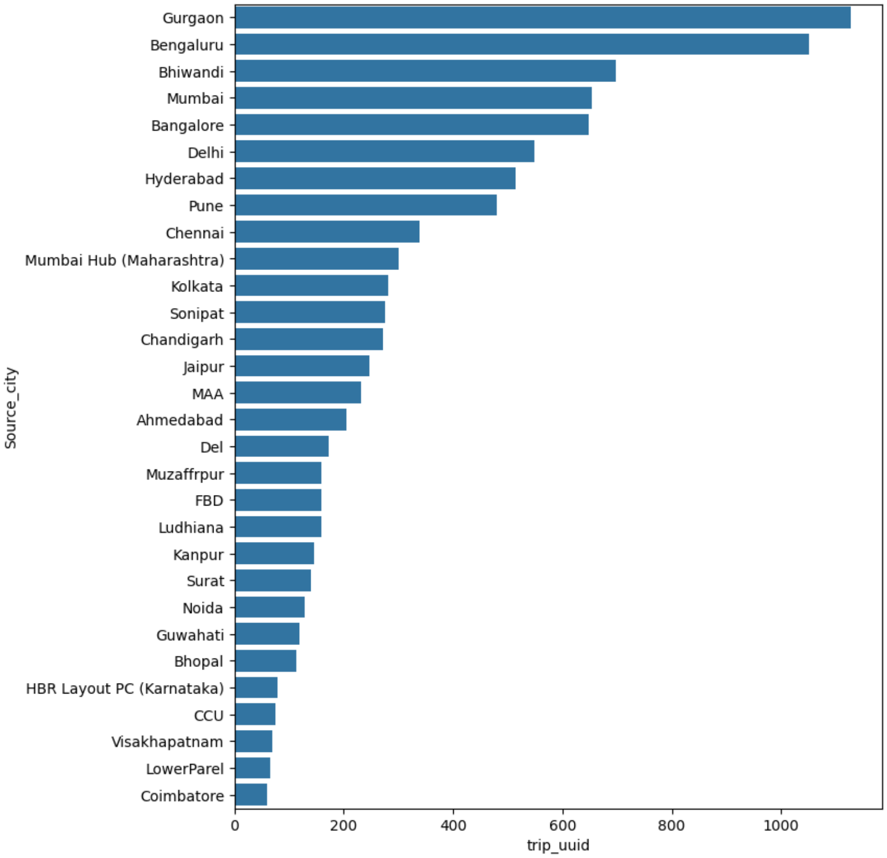
```
plt.figure(figsize = (8, 10))
sns.barplot(data = data_source_state,
            x = data_source_state['trip_uuid'],
            y = data_source_state['Source_state'])
plt.show()
```



```
data_source_city = trip_data.groupby(by = "Source_city")["trip_uuid"].count().reset_index()
data_source_city = data_source_city.sort_values(by= "trip_uuid", ascending = False)[:30]
data_source_city.head()
```

	Source_city	trip_uuid
256	Gurgaon	1128
84	Bengaluru	1052
105	Bhiwandi	697
466	Mumbai	654
62	Bangalore	648

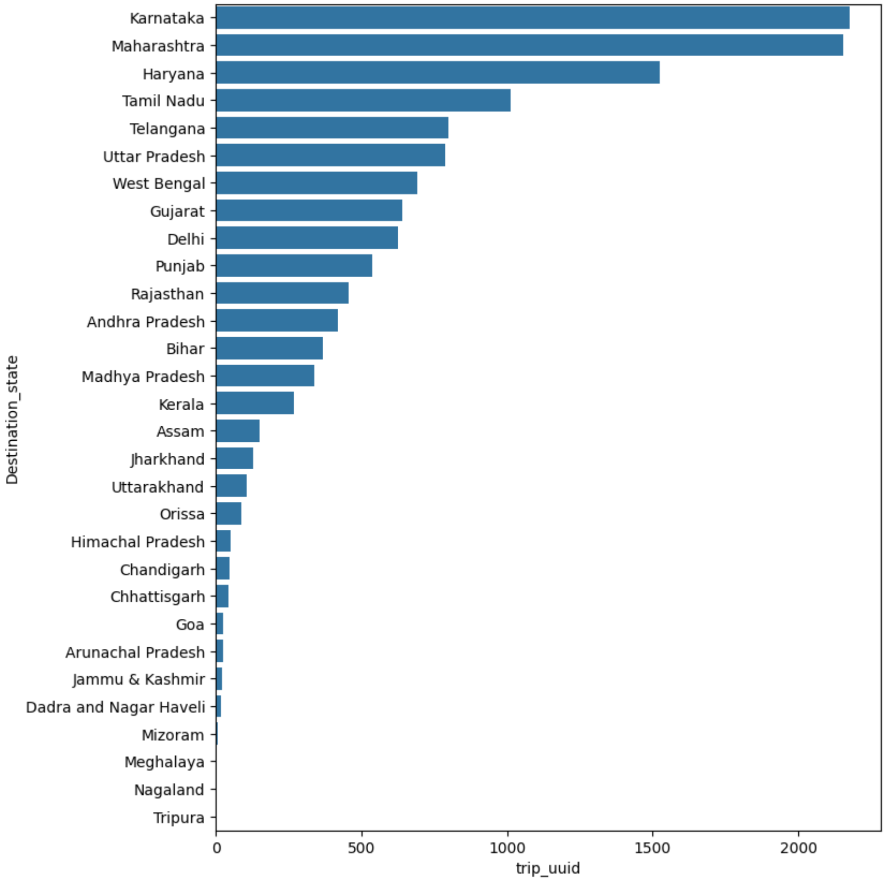
```
plt.figure(figsize = (8, 10))
sns.barplot(data = data_source_city,
            x = data_source_city["trip_uuid"],
            y = data_source_city['Source_city'])
plt.show()
```



```
data_destination_state = trip_data.groupby("Destination_state")["trip_uuid"].count().reset_index()
data_destination_state = data_destination_state.sort_values(by= "trip_uuid", ascending = False)
data_destination_state.head()
```

	Destination_state	trip_uuid
14	Karnataka	2175
17	Maharashtra	2154
10	Haryana	1524
24	Tamil Nadu	1014
25	Telangana	797

```
plt.figure(figsize = (8, 10))
sns.barplot(data = data_destination_state,
            x = data_destination_state["trip_uuid"],
            y = data_destination_state["Destination_state"])
plt.show()
```

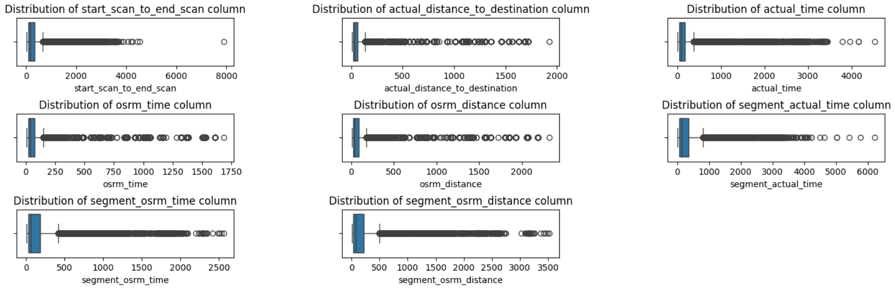


```
data_destination_city = trip_data.groupby("Destination_city")["trip_uuid"].count().reset_index()
data_destination_city = data_destination_city.sort_values(by= "trip_uuid", ascending = False)
data_destination_city.head()
```

	Destination_city	trip_uuid
103	Bengaluru	1088
548	Mumbai	966
301	Gurgaon	877
214	Delhi	554
79	Bangalore	551

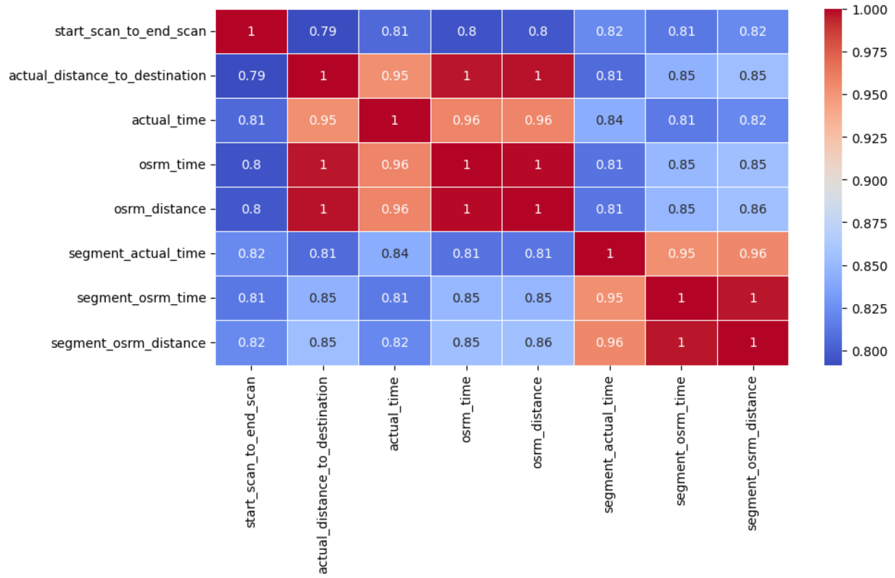
```
num_cols = ["start_scan_to_end_scan","actual_distance_to_destination","actual_time","osrm_time", "osrm_distance", "segment_actual_time", "se
data_corr= trip_data[num_cols].corr()
```

```
plt.figure(figsize= (18,5))
for i in range(len(num_cols)):
  plt.subplot(3, 3, i+1)
  sns.boxplot(x= trip_data[num_cols[i]])
  plt.title(f"Distribution of {num_cols[i]} column")

plt.subplots_adjust(hspace=1, wspace=0.5)
plt.show()
```



```
plt.figure(figsize= (10, 5))
sns.heatmap(data= data_corr, annot = True, cmap='coolwarm', linewidths=0.5)
plt.plot()
```

[]



	start_scan_to_end_scan	actual_distance_to_destination	actual_time	osrm_time	osrm_distance	segment_actual_time	segment_osrm_time	segment_osrm_distance
start_scan_to_end_scan	1	0.79	0.81	0.8	0.8	0.82	0.81	0.82
actual_distance_to_destination	0.79	1	0.95	1	1	0.81	0.85	0.85
actual_time	0.81	0.95	1	0.96	0.96	0.84	0.81	0.82
osrm_time	0.8	1	0.96	1	1	0.81	0.85	0.85
osrm_distance	0.8	1	0.96	1	1	0.81	0.85	0.86
segment_actual_time	0.82	0.81	0.84	0.81	0.81	1	0.95	0.96
segment_osrm_time	0.81	0.85	0.81	0.85	0.85	0.95	1	1
segment_osrm_distance	0.82	0.85	0.82	0.85	0.86	0.96	1	1

```
# hot encoding on categorical features

trip_data["route_type"]= trip_data["route_type"].map({"FTL": 0, "Carting": 1})
trip_data.head()
```

	trip_uuid	data	route_type	trip_creation_time	trip_creation_day	trip_
0	trip-153671041653548748	training	0	2018-09-12 00:00:16.535741	12	
1	trip-153671042288605164	training	1	2018-09-12 00:00:22.886430	12	
2	trip-153671043369099517	training	0	2018-09-12 00:00:33.691250	12	
3	trip-153671046011330457	training	1	2018-09-12 00:01:00.113710	12	
4	trip-153671052974046625	training	0	2018-09-12 00:02:09.740725	12	

5 rows × 29 columns

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaler.fit(trip_data[num_cols])
```

```
▾ StandardScaler
StandardScaler()
```

```
trip_data[num_cols] = scaler.transform(trip_data[num_cols])
trip_data[num_cols]
```

	start_scan_to_end_scan	actual_distance_to_destination	actual_time	osrm_time
0	1.820832	1.392082	1.357536	1.339298
1	-0.557529	-0.229126	-0.296138	-0.277793
2	0.977918	6.551904	5.651681	6.667288
3	-0.474425	-0.358711	-0.379498	-0.403982
4	-0.371534	-0.319774	-0.370486	-0.347898
...	...	...	...	...
14782	-0.371534	-0.300518	-0.402027	-0.315182
14783	-0.553572	-0.365575	-0.465110	-0.418003
14784	-0.181582	-0.349786	-0.305150	-0.361919
14785	-0.464532	-0.377357	-0.444833	-0.408656
14786	-0.104414	-0.324176	-0.417798	-0.352572

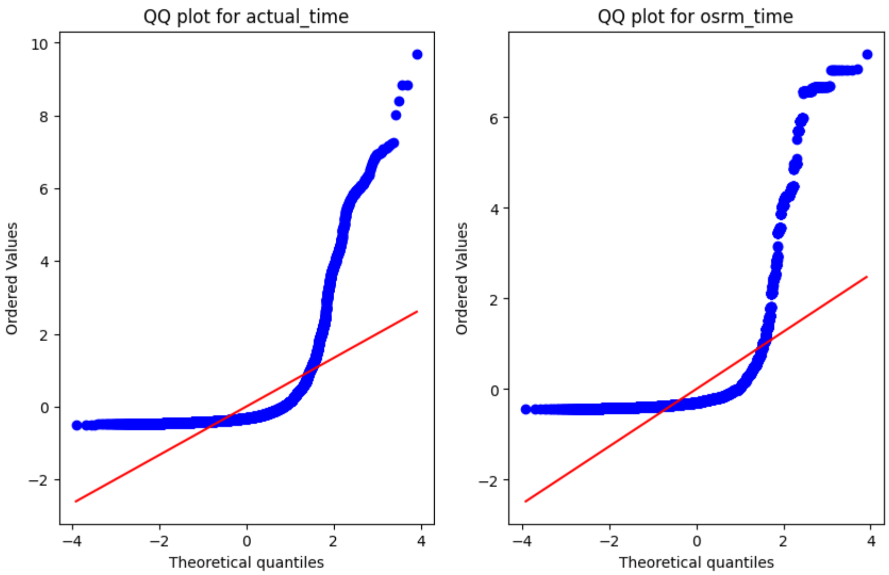
14787 rows × 8 columns

```
# Hypothesis Testing:
# 1. actual_time aggregated value and OSRM time aggregated value

trip_data[["actual_time", "osrm_time"]].describe()
```

	actual_time	osrm_time
count	1.478700e+04	1.478700e+04
mean	3.003241e-17	-2.594800e-17
std	1.000034e+00	1.000034e+00
min	-4.921456e-01	-4.460453e-01
25%	-3.975212e-01	-3.665928e-01
50%	-3.186676e-01	-2.964877e-01
75%	-1.023832e-01	-1.329091e-01
max	9.698001e+00	7.405728e+00

```
# Distribution check using QQ plot

plt.figure(figsize = (10, 6))
plt.subplot(1, 2, 1)
spy.probplot(trip_data['actual_time'], plot = plt, dist = 'norm')
plt.title('QQ plot for actual_time')
plt.subplot(1, 2, 2)
spy.probplot(trip_data['osrm_time'], plot = plt, dist = 'norm')
plt.title('QQ plot for osrm_time')
plt.plot()
```

[]



```
# Homogeneity of Variances using Lavene's test

# H0- Variance are significantly different
# HA- Variance are not significantly different

test_stat, p_value = spy.levene(trip_data["actual_time"], trip_data["osrm_time"])
print('p-value', p_value)
if p_value < 0.05:
    print("Variance are significantly different")
else:
    print("Variance are not significantly different")
```

```
p-value 0.004369154964102629
Variance are significantly different
```

```
# Since the samples do not follow any of the assumptions T-Test cannot be applied here.
# we can perform its non parametric equivalent test i.e., Mann-Whitney U rank test for two independent samples.

t_stat, p_value = spy.mannwhitneyu(trip_data["actual_time"], trip_data["osrm_time"])
print("p-value", p_value)
if p_value < 0.05:
    print("The samples are not similar")
else:
    print("The samples are similar")
```

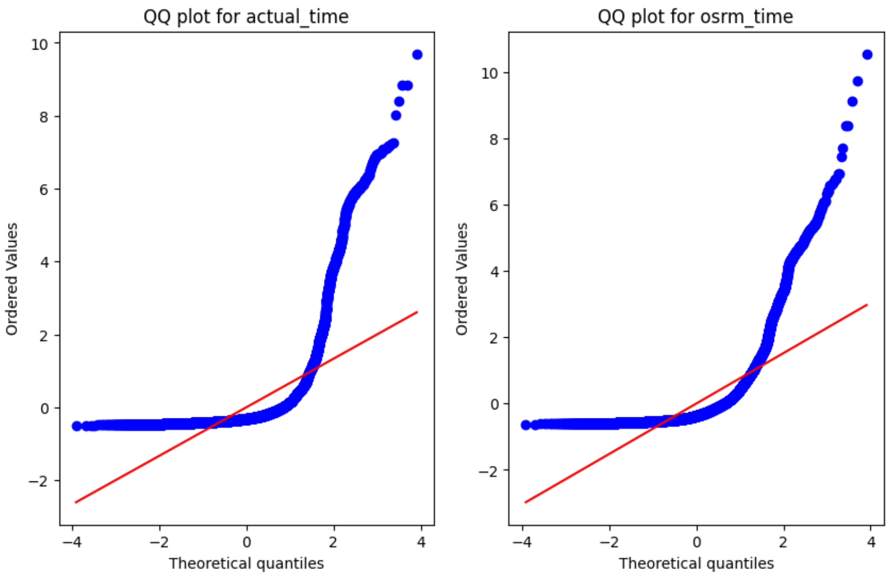
```
p-value 9.509176874996746e-61
The samples are not similar
```

```
# 2. actual_time aggregated value and segment actual time aggregated value.

trip_data[['actual_time', 'segment_actual_time']].describe()
```

	actual_time	segment_actual_time
count	1.478700e+04	1.478700e+04
mean	3.003241e-17	-2.979215e-17
std	1.000034e+00	1.000034e+00
min	-4.921456e-01	-6.184254e-01
25%	-3.975212e-01	-5.159714e-01
50%	-3.186676e-01	-3.703788e-01
75%	-1.023832e-01	1.966547e-02
max	9.698001e+00	1.056344e+01

```
# Distribution check using QQ plot

plt.figure(figsize = (10, 6))
plt.subplot(1, 2, 1)
spy.probplot(trip_data["actual_time"], plot = plt, dist = "norm")
plt.title("QQ plot for actual_time")
plt.subplot(1, 2, 2)
spy.probplot(trip_data["segment_actual_time"], plot = plt, dist = "norm")
plt.title("QQ plot for osrm_time")
plt.plot()
```

```
[]
```



QQ plot for actual_time

QQ plot for osrm_time

```
# Homogeneity of Variances using Lavene's test

# H0- Variance are significantly different
# HA- Variance are not significantly different

test_stat, p_value = spy.levene(trip_data["actual_time"], trip_data["segment_actual_time"])
print("p-value", p_value)
if p_value < 0.05:
    print("Variance are significantly different")
else:
    print("Variance are not significantly different")
```

```
p-value 2.1119134589517006e-23
Variance are significantly different
```

```
# Since the samples do not follow any of the assumptions T-Test cannot be applied here.
# we can perform its non parametric equivalent test i.e., Mann-Whitney U rank test for two independent samples.

t_stat, p_value = spy.mannwhitneyu(trip_data["actual_time"], trip_data["segment_actual_time"])
print("p-value", p_value)
if p_value < 0.05:
    print("The samples are not similar")
else:
    print("The samples are similar")
```

```
p-value 3.3269870340576406e-184
```

```
# 3.OSRM distance aggregated value and segment OSRM distance aggregated value.

trip_data[['osrm_distance', 'segment_osrm_distance']].describe()
```

	osrm_distance	segment_osrm_distance
count	1.478700e+04	1.478700e+04
mean	4.180511e-17	-7.399985e-17
std	1.000034e+00	1.000034e+00
min	-4.087113e-01	-5.125146e-01
25%	-3.510620e-01	-4.561227e-01
50%	-2.997272e-01	-3.668653e-01
75%	-1.469189e-01	-1.474182e-02
max	7.474182e+00	7.919079e+00

```
# Distribution check using QQ plot

plt.figure(figsize = (10, 6))
plt.subplot(1, 2, 1)
spy.probplot(trip_data["osrm_distance"], plot = plt, dist = "norm")
plt.title("QQ plot for actual_time")
plt.subplot(1, 2, 2)
spy.probplot(trip_data["segment_osrm_distance"], plot = plt, dist = "norm")
plt.title("QQ plot for osrm_time")
plt.plot()
```

Could not connect to the reCAPTCHA service. Please check your internet connection and reload to get a reCAPTCHA challenge.