In [47]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import ttest_ind
```

In [2]:
```
df = pd.read_csv('delhivery_data.csv')
print("Shape:",df.shape)
df.head(3)
```

```
Shape: (144867, 24)
```

Out[2]:

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

3 rows × 24 columns

In [3]:
```
#- Removing null values
df = df.dropna(how='any')
df = df.reset_index(drop=True)
```

In [5]:
```
# - Converting time columns into pandas datatime
df['od_start_time'] =  pd.to_datetime(df['od_start_time'])
df['od_end_time']  =  pd.to_datetime(df['od_end_time'])
```

In [6]:
```
df.head(20)
```

Out[6]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	s
0	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
1	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
2	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
3	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
4	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
5	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
6	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
7	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
8	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
9	training	2018-09-20 02:35:36.476840	thanos::sroute:eb7bfc78-b351-4c0e-a951-fa3d5c3...	Carting	trip-153741093647649320	IN
10	training	2018-09-23 06:42:06.021680	thanos::sroute:ff52ef7a-4d0d-4063-9bfe-cc21172...	FTL	trip-153768492602129387	IN
11	training	2018-09-23 06:42:06.021680	thanos::sroute:ff52ef7a-4d0d-4063-9bfe-cc21172...	FTL	trip-153768492602129387	IN
12	training	2018-09-23 06:42:06.021680	thanos::sroute:ff52ef7a-4d0d-4063-9bfe-cc21172...	FTL	trip-153768492602129387	IN
13	training	2018-09-23 06:42:06.021680	thanos::sroute:ff52ef7a-4d0d-4063-9bfe-cc21172...	FTL	trip-153768492602129387	IN

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	s
14	training	2018-09-23 06:42:06.021680	thanos::sroute:ff52ef7a-4d0d-4063-9bfe-cc21172...	FTL	trip-153768492602129387	IN
15	training	2018-09-14 15:42:46.437249	thanos::sroute:a16bfa03-3462-4bce-9c82-5784c7d...	Carting	trip-153693976643699843	IN
16	training	2018-09-14 15:42:46.437249	thanos::sroute:a16bfa03-3462-4bce-9c82-5784c7d...	Carting	trip-153693976643699843	IN
17	training	2018-09-13 20:44:19.424489	thanos::sroute:76951383-1608-44e4-a284-46d92e8...	FTL	trip-153687145942424248	IN
18	training	2018-09-13 20:44:19.424489	thanos::sroute:76951383-1608-44e4-a284-46d92e8...	FTL	trip-153687145942424248	IN
19	training	2018-09-13 20:44:19.424489	thanos::sroute:76951383-1608-44e4-a284-46d92e8...	FTL	trip-153687145942424248	IN

20 rows × 24 columns

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 144316 entries, 0 to 144315
Data columns (total 24 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   data                            144316 non-null  object
 1   trip_creation_time              144316 non-null  object
 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  object
 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
dtypes: bool(1), datetime64[ns](2), float64(10), int64(1), object(10)
memory usage: 25.5+ MB
```

In [8]:

```python
# - Grouping by sub-journey in the trip
df['segment_key'] = df['trip_uuid'] + df['source_center'] + df['destination_center'

segment_cols = ['segment_actual_time', 'segment_osrm_distance', 'segment_osrm_time'

for col in segment_cols:
    df[col + '_sum'] = df.groupby('segment_key')[col].cumsum()


df[[col + '_sum' for col in segment_cols]]
```

Out[8]:

	segment_actual_time_sum	segment_osrm_distance_sum	segment_osrm_time_sum
0	14.0	11.9653	11.0
1	24.0	21.7243	20.0
2	40.0	32.5395	27.0
3	61.0	45.5619	39.0
4	67.0	49.4772	44.0
...	...	...	...
144311	92.0	65.3487	94.0
144312	118.0	82.7212	115.0
144313	138.0	103.4265	149.0
144314	155.0	122.3150	176.0
144315	423.0	131.1238	185.0

144316 rows × 3 columns

In [9]:

```
# aggregating at sub-journey level
create_segment_dict = {

    'data' : 'first',
    'trip_creation_time': 'first',
    'route_schedule_uuid' : 'first',
    'route_type' : 'first',
    'trip_uuid' : 'first',
    'source_center' : 'first',
    'source_name' : 'first',

    'destination_center' : 'last',
    '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' : 'last',
    'segment_osrm_distance_sum' : 'last',
    'segment_osrm_time_sum' : 'last',

    }
```

In [10]:
```
# - Groupby mini-trips, sorting by time
segment = df.groupby('segment_key').agg(create_segment_dict).reset_index()
segment = segment.sort_values(by=['segment_key','od_end_time'], ascending=True).res
```

In [11]:
```
segment
```

Out[11]:

	index	segment_key	data	trip_creation_time
0	0	trip-153671041653548748IND209304AAAIND000000ACB	training	2018-09-12 00:00:16.535741
1	1	trip-153671041653548748IND462022AAAIND209304AAA	training	2018-09-12 00:00:16.535741
2	2	trip-153671042288605164IND561203AABIND562101AAA	training	2018-09-12 00:00:22.886430
3	3	trip-153671042288605164IND572101AAAIND561203AAB	training	2018-09-12 00:00:22.886430
4	4	trip-153671043369099517IND000000ACBIND160002AAC	training	2018-09-12 00:00:33.691250
...	...	...	...	...
26217	26217	trip-153861115439069069IND628204AAAIND627657AAA	test	2018-10-03 23:59:14.390954
26218	26218	trip-153861115439069069IND628613AAAIND627005AAA	test	2018-10-03 23:59:14.390954
26219	26219	trip-153861115439069069IND628801AAAIND628204AAA	test	2018-10-03 23:59:14.390954
26220	26220	trip-153861118270144424IND583119AAAIND583101AAA	test	2018-10-03 23:59:42.701692
26221	26221	trip-153861118270144424IND583201AAAIND583119AAA	test	2018-10-03 23:59:42.701692

26222 rows × 21 columns

In [12]:
```
segment.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 26222 entries, 0 to 26221
Data columns (total 21 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   index                           26222 non-null  int64
 1   segment_key                     26222 non-null  object
 2   data                            26222 non-null  object
 3   trip_creation_time              26222 non-null  object
 4   route_schedule_uuid             26222 non-null  object
 5   route_type                      26222 non-null  object
 6   trip_uuid                       26222 non-null  object
 7   source_center                   26222 non-null  object
 8   source_name                     26222 non-null  object
 9   destination_center              26222 non-null  object
 10  destination_name                26222 non-null  object
 11  od_start_time                   26222 non-null  datetime64[ns]
 12  od_end_time                     26222 non-null  datetime64[ns]
 13  start_scan_to_end_scan          26222 non-null  float64
 14  actual_distance_to_destination  26222 non-null  float64
 15  actual_time                     26222 non-null  float64
 16  osrm_time                       26222 non-null  float64
 17  osrm_distance                   26222 non-null  float64
 18  segment_actual_time_sum         26222 non-null  float64
 19  segment_osrm_distance_sum       26222 non-null  float64
 20  segment_osrm_time_sum           26222 non-null  float64
dtypes: datetime64[ns](2), float64(8), int64(1), object(10)
memory usage: 4.2+ MB
```

Calculate time taken between od_start_time and od_end_time and keep it as a feature. `

- od_time_diff_hour is matching with start_scan_to_end_scan

In [13]:
```
segment['od_time_diff_hour'] = (segment['od_end_time'] - segment['od_start_time']).
segment['od_time_diff_hour']
```

Out[13]:
```
0        1260.604421
1         999.505379
2          58.832388
3         122.779486
4         834.638929
             ...
26217      62.115193
26218      91.087797
26219      44.174403
26220     287.474007
26221      66.933565
Name: od_time_diff_hour, Length: 26222, dtype: float64
```

In [14]:
```
segment
```

Out[14]:

	index	segment_key	data	trip_creation_time
0	0	trip-153671041653548748IND209304AAAIND000000ACB	training	2018-09-12 00:00:16.535741
1	1	trip-153671041653548748IND462022AAAIND209304AAA	training	2018-09-12 00:00:16.535741
2	2	trip-153671042288605164IND561203AABIND562101AAA	training	2018-09-12 00:00:22.886430
3	3	trip-153671042288605164IND572101AAAIND561203AAB	training	2018-09-12 00:00:22.886430
4	4	trip-153671043369099517IND000000ACBIND160002AAC	training	2018-09-12 00:00:33.691250
...	...	...	...	...
26217	26217	trip-153861115439069069IND628204AAAIND627657AAA	test	2018-10-03 23:59:14.390954
26218	26218	trip-153861115439069069IND628613AAAIND627005AAA	test	2018-10-03 23:59:14.390954
26219	26219	trip-153861115439069069IND628801AAAIND628204AAA	test	2018-10-03 23:59:14.390954
26220	26220	trip-153861118270144424IND583119AAAIND583101AAA	test	2018-10-03 23:59:42.701692
26221	26221	trip-153861118270144424IND583201AAAIND583119AAA	test	2018-10-03 23:59:42.701692

26222 rows × 22 columns

In [15]:
```
create_trip_dict = {

    'data' : 'first',
    'trip_creation_time': 'first',
    'route_schedule_uuid' : 'first',
    'route_type' : 'first',
    'trip_uuid' : 'first',

    'source_center' : 'first',
    'source_name' : 'first',
```

```
    'destination_center' : 'last',
    'destination_name' : 'last',

    'start_scan_to_end_scan' : 'sum',
    'od_time_diff_hour' : 'sum',

    'actual_distance_to_destination' : 'sum',
    'actual_time' : 'sum',
    'osrm_time' : 'sum',
    'osrm_distance' : 'sum',

    'segment_actual_time_sum' : 'sum',
    'segment_osrm_distance_sum' : 'sum',
    'segment_osrm_time_sum' : 'sum',

    }
```

In [16]:
```
trip = segment.groupby('trip_uuid').agg(create_trip_dict).reset_index(drop = True)
```

In [17]:
```
trip
```

Out[17]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid
0	training	2018-09-12 00:00:16.535741	thanos::sroute:d7c989ba-a29b-4a0b-b2f4-288cdc6...	FTL	trip-153671041653548748
1	training	2018-09-12 00:00:22.886430	thanos::sroute:3a1b0ab2-bb0b-4c53-8c59-eb2a2c0...	Carting	trip-153671042288605164
2	training	2018-09-12 00:00:33.691250	thanos::sroute:de5e208e-7641-45e6-8100-4d9fb1e...	FTL	trip-153671043369099517
3	training	2018-09-12 00:01:00.113710	thanos::sroute:f0176492-a679-4597-8332-bbd1c7f...	Carting	trip-153671046011330457
4	training	2018-09-12 00:02:09.740725	thanos::sroute:d9f07b12-65e0-4f3b-bec8-df06134...	FTL	trip-153671052974046625
...	...	...	...	...	..
14782	test	2018-10-03 23:55:56.258533	thanos::sroute:8a120994-f577-4491-9e4b-b7e4a14...	Carting	trip-153861095625827784
14783	test	2018-10-03 23:57:23.863155	thanos::sroute:b30e1ec3-3bfa-4bd2-a7fb-3b75769...	Carting	trip-153861104386292051
14784	test	2018-10-03 23:57:44.429324	thanos::sroute:5609c268-e436-4e0a-8180-3db4a74...	Carting	trip-153861106442901555
14785	test	2018-10-03 23:59:14.390954	thanos::sroute:c5f2ba2c-8486-4940-8af6-d1d2a6a...	Carting	trip-153861115439069069
14786	test	2018-10-03 23:59:42.701692	thanos::sroute:412fea14-6d1f-4222-8a5f-a517042...	FTL	trip-153861118270144424

14787 rows × 18 columns

In [18]:
```
trip[['actual_time', 'segment_actual_time_sum']]
```

Out[18]:

	actual_time	segment_actual_time_sum
0	1562.0	1548.0
1	143.0	141.0
2	3347.0	3308.0
3	59.0	59.0
4	341.0	340.0
...	...	...
14782	83.0	82.0
14783	21.0	21.0
14784	282.0	281.0
14785	264.0	258.0
14786	275.0	274.0

14787 rows × 2 columns

In [19]:
```
trip
```

Out[19]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid
0	training	2018-09-12 00:00:16.535741	thanos::sroute:d7c989ba-a29b-4a0b-b2f4-288cdc6...	FTL	trip-153671041653548748
1	training	2018-09-12 00:00:22.886430	thanos::sroute:3a1b0ab2-bb0b-4c53-8c59-eb2a2c0...	Carting	trip-153671042288605164
2	training	2018-09-12 00:00:33.691250	thanos::sroute:de5e208e-7641-45e6-8100-4d9fb1e...	FTL	trip-153671043369099517
3	training	2018-09-12 00:01:00.113710	thanos::sroute:f0176492-a679-4597-8332-bbd1c7f...	Carting	trip-153671046011330457
4	training	2018-09-12 00:02:09.740725	thanos::sroute:d9f07b12-65e0-4f3b-bec8-df06134...	FTL	trip-153671052974046625
...	...	...	...	...	..
14782	test	2018-10-03 23:55:56.258533	thanos::sroute:8a120994-f577-4491-9e4b-b7e4a14...	Carting	trip-153861095625827784
14783	test	2018-10-03 23:57:23.863155	thanos::sroute:b30e1ec3-3bfa-4bd2-a7fb-3b75769...	Carting	trip-153861104386292051
14784	test	2018-10-03 23:57:44.429324	thanos::sroute:5609c268-e436-4e0a-8180-3db4a74...	Carting	trip-153861106442901555
14785	test	2018-10-03 23:59:14.390954	thanos::sroute:c5f2ba2c-8486-4940-8af6-d1d2a6a...	Carting	trip-153861115439069069
14786	test	2018-10-03 23:59:42.701692	thanos::sroute:412fea14-6d1f-4222-8a5f-a517042...	FTL	trip-153861118270144424

14787 rows × 18 columns

In [20]:
```
trip[['actual_distance_to_destination','osrm_distance']]
```

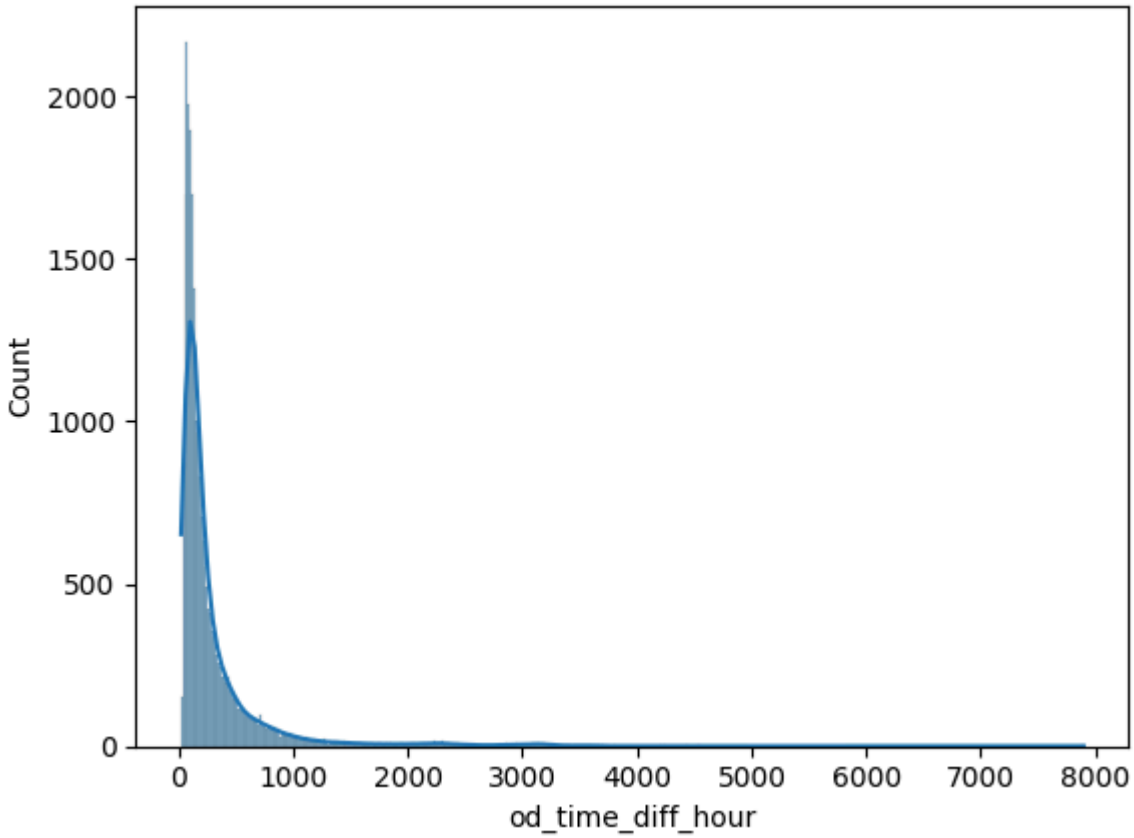
Out[20]:

	actual_distance_to_destination	osrm_distance
0	824.732854	991.3523
1	73.186911	85.1110
2	1927.404273	2354.0665
3	17.175274	19.6800
4	127.448500	146.7918
...	...	...
14782	57.762332	73.4630
14783	15.513784	16.0882
14784	38.684839	58.9037
14785	134.723836	171.1103
14786	66.081533	80.5787

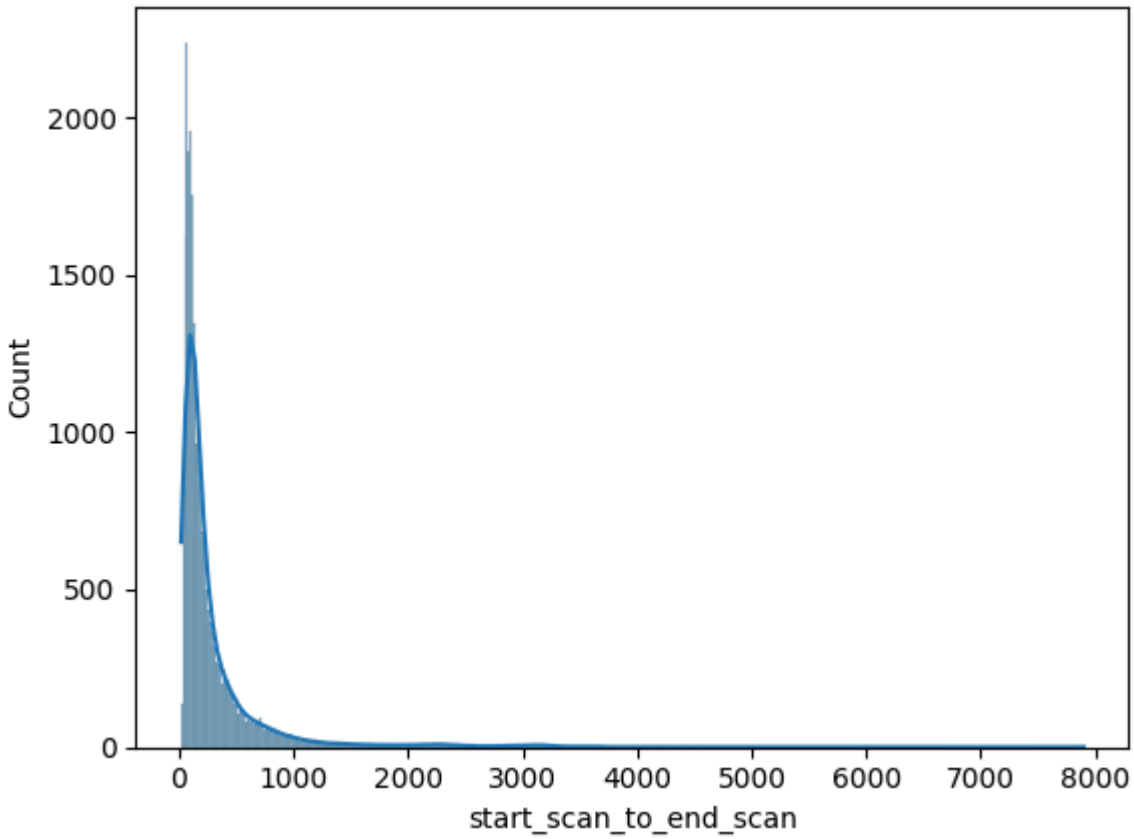
14787 rows × 2 columns

hypothesis testing --> recommendation to the business

In [51]:
```
# Visual analysis
sns.histplot(segment['od_time_diff_hour'], kde=True, label='Time Taken')
plt.show()
```



In [52]:
```
sns.histplot(segment['start_scan_to_end_scan'], kde=True, label='Start Scan to End
plt.show()
```



In [54]:
```
# Hypothesis testing
t_stat, p_val = ttest_ind(segment['od_time_diff_hour'], segment['start_scan_to_end_
print(f'T-statistic: {t_stat}, P-value: {p_val}')
```

```
T-statistic: 0.12947099971100354, P-value: 0.896985467204072
```

In [56]:
```
alpha=0.05
if p_val<alpha:
    print("Reject Ho")
else:
    print("Fail to Reject Ho")
```

```
Fail to Reject Ho
```

In [21]:
```
trip['destination_name'] = trip['destination_name'].str.lower() #lowering all colum
trip['source_name'] = trip['source_name'].str.lower()
```

In [22]:
```
def place2state(x):
    # transform  "gurgaon_bilaspur_hb (haryana)" into "haryana)""
    state = x.split('(')[1] # haryana)

    return state[:-1] #removing ')' from ending # haryana

def place2city(x):
    # We will remove state #"gurgaon_bilaspur_hb (haryana)"
    city = x.split(' (')[0]#"gurgaon_bilaspur_hb

    city = city.split('_')[0]#"gurgaon

    #Now dealing with edge cases

    if city == 'pnq vadgaon sheri dpc':
      return 'vadgaonsheri'

    # ['PNQ Pashan DPC', 'Bhopal MP Nagar', 'HBR Layout PC',
    #  'PNQ Rahatani DPC', 'Pune Balaji Nagar', 'Mumbai Antop Hill']

    if city in ['pnq pashan dpc','pnq rahatani dpc', 'pune balaji nagar']:
        return 'pune'

    if city == 'hbr layout pc' : return 'bengaluru'
    if city == 'bhopal mp nagar' : return 'bhopal'
    if city == 'mumbai antop hill' : return 'mumbai'


    return city

def place2city_place(x):

    # We will remove state
    x = x.split(' (')[0]

    len_ = len(x.split('_'))

    if len_ >= 3:
        return x.split('_')[1]
```

```
    # Small cities have same city and place name
    if len_ == 2:
        return x.split('_')[0]


    # Now we need to deal with edge cases or imporper name convention

    #if len(x.split(' ')) == 2:
    #

    return x.split(' ')[0]


def place2code(x):
    # We will remove state
    x = x.split(' (')[0]

    if len(x.split('_')) >= 3 :
        return x.split('_')[-1]

    return 'none'
```

In [23]:
```
trip['destination_state'] = trip['destination_name'].apply(lambda x: place2state(x)
trip['destination_city']  = trip['destination_name'].apply(lambda x: place2city(x))
trip['destination_place'] = trip['destination_name'].apply(lambda x: place2city_pla
trip['destination_code']  = trip['destination_name'].apply(lambda x: place2code(x))
```

In [24]:
```
trip[['destination_state', 'destination_city', 'destination_place', 'destination_co
```

Out[24]:

	destination_state	destination_city	destination_place	destination_code
0	uttar pradesh	kanpur	central	6
1	karnataka	doddablpur	chikadpp	d
2	haryana	gurgaon	bilaspur	hb
3	maharashtra	mumbai	mirard	ip
4	karnataka	sandur	wrdn1dpp	d
...	...	...	...	...
14782	punjab	chandigarh	mehmdpur	h
14783	haryana	faridabad	blbgarh	dc
14784	uttar pradesh	kanpur	govndngr	dc
14785	tamil nadu	tirchchndr	shnmgprm	d
14786	karnataka	sandur	wrdn1dpp	d

14787 rows × 4 columns

In [25]:
```
trip['source_state'] = trip['source_name'].apply(lambda x: place2state(x))
trip['source_city']  = trip['source_name'].apply(lambda x: place2city(x))
trip['source_place'] = trip['source_name'].apply(lambda x: place2city_place(x))
trip['source_code']  = trip['source_name'].apply(lambda x: place2code(x))
```

In [26]:
```
trip[['source_state', 'source_city', 'source_place', 'source_code']]
```

Out[26]:

	source_state	source_city	source_place	source_code
0	uttar pradesh	kanpur	central	6
1	karnataka	doddablpur	chikadpp	d
2	haryana	gurgaon	bilaspur	hb
3	maharashtra	mumbai hub	mumbai	none
4	karnataka	bellary	bellary	none
...	...	...	...	...
14782	punjab	chandigarh	mehmdpur	h
14783	haryana	fbd	balabhgarh	dpc
14784	uttar pradesh	kanpur	govndngr	dc
14785	tamil nadu	tirunelveli	vdkkusrt	i
14786	karnataka	sandur	wrdn1dpp	d

14787 rows × 4 columns

In [27]:
```
trip['trip_creation_time'] =  pd.to_datetime(trip['trip_creation_time'])

trip['trip_year'] = trip['trip_creation_time'].dt.year
trip['trip_month'] = trip['trip_creation_time'].dt.month
trip['trip_hour'] = trip['trip_creation_time'].dt.hour
trip['trip_day'] = trip['trip_creation_time'].dt.day
trip['trip_week'] = trip['trip_creation_time'].dt.isocalendar().week
trip['trip_dayofweek'] = trip['trip_creation_time'].dt.dayofweek
```

In [28]:
```
trip[['trip_year', 'trip_month', 'trip_hour', 'trip_day', 'trip_week', 'trip_dayofw
```

Out[28]:

	trip_year	trip_month	trip_hour	trip_day	trip_week	trip_dayofweek
0	2018	9	0	12	37	2
1	2018	9	0	12	37	2
2	2018	9	0	12	37	2
3	2018	9	0	12	37	2
4	2018	9	0	12	37	2
...	...	...	...	...	...	...
14782	2018	10	23	3	40	2
14783	2018	10	23	3	40	2
14784	2018	10	23	3	40	2
14785	2018	10	23	3	40	2
14786	2018	10	23	3	40	2

14787 rows × 6 columns

In [29]:
```
trip.head(5)
```

Out[29]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid	sc
0	training	2018-09-12 00:00:16.535741	thanos::sroute:d7c989ba-a29b-4a0b-b2f4-288cdc6...	FTL	trip-153671041653548748	IN
1	training	2018-09-12 00:00:22.886430	thanos::sroute:3a1b0ab2-bb0b-4c53-8c59-eb2a2c0...	Carting	trip-153671042288605164	IN
2	training	2018-09-12 00:00:33.691250	thanos::sroute:de5e208e-7641-45e6-8100-4d9fb1e...	FTL	trip-153671043369099517	IN
3	training	2018-09-12 00:01:00.113710	thanos::sroute:f0176492-a679-4597-8332-bbd1c7f...	Carting	trip-153671046011330457	IN
4	training	2018-09-12 00:02:09.740725	thanos::sroute:d9f07b12-65e0-4f3b-bec8-df06134...	FTL	trip-153671052974046625	IN

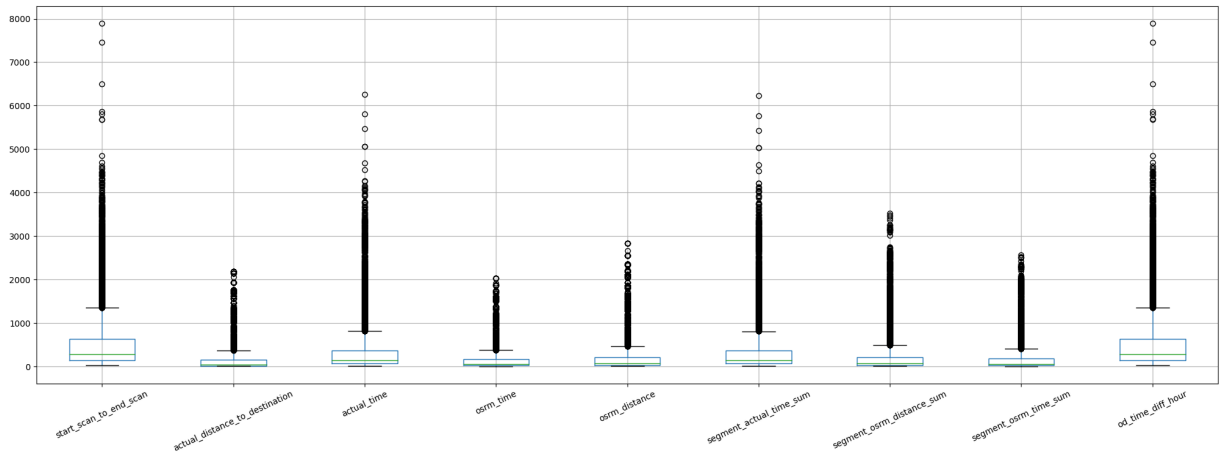
5 rows × 32 columns

In [30]:
```
num_cols = ['start_scan_to_end_scan', 'actual_distance_to_destination', 'actual_tim
            'osrm_distance', 'segment_actual_time_sum', 'segment_osrm_distance_sum'
            'segment_osrm_time_sum', 'od_time_diff_hour']
```

Find outliers in numerical variable (you might find outliers in almost all the variables), and visualize it using visual analysis

In [32]:
```
trip[num_cols].boxplot(rot=25, figsize=(25,8))
```

Out[32]: <Axes: >



Handle the outliers using IQR method

In [33]:
```
Q1 = trip[num_cols].quantile(0.25)
Q3 = trip[num_cols].quantile(0.75)

IQR = Q3 - Q1
```

In [34]:
```
trip = trip[~((trip[num_cols] < (Q1 - 1.5 * IQR)) | (trip[num_cols] > (Q3 + 1.5 * I
trip = trip.reset_index(drop=True)  #---- and or or
```

In [35]:
```
trip
```

Out[35]:

	data	trip_creation_time	route_schedule_uuid	route_type	trip_uuid
0	training	2018-09-12 00:00:22.886430	thanos::sroute:3a1b0ab2-bb0b-4c53-8c59-eb2a2c0...	Carting	trip-153671042288605164
1	training	2018-09-12 00:01:00.113710	thanos::sroute:f0176492-a679-4597-8332-bbd1c7f...	Carting	trip-153671046011330457
2	training	2018-09-12 00:02:09.740725	thanos::sroute:d9f07b12-65e0-4f3b-bec8-df06134...	FTL	trip-153671052974046625
3	training	2018-09-12 00:02:34.161600	thanos::sroute:9bf03170-d0a2-4a3f-aa4d-9aaab3d...	Carting	trip-153671055416136166
4	training	2018-09-12 00:04:22.011653	thanos::sroute:a97698cc-846e-41a7-916b-88b1741...	Carting	trip-153671066201138152
...	...	...	...	...	..
12718	test	2018-10-03 23:55:56.258533	thanos::sroute:8a120994-f577-4491-9e4b-b7e4a14...	Carting	trip-153861095625827784
12719	test	2018-10-03 23:57:23.863155	thanos::sroute:b30e1ec3-3bfa-4bd2-a7fb-3b75769...	Carting	trip-153861104386292051
12720	test	2018-10-03 23:57:44.429324	thanos::sroute:5609c268-e436-4e0a-8180-3db4a74...	Carting	trip-153861106442901555
12721	test	2018-10-03 23:59:14.390954	thanos::sroute:c5f2ba2c-8486-4940-8af6-d1d2a6a...	Carting	trip-153861115439069069
12722	test	2018-10-03 23:59:42.701692	thanos::sroute:412fea14-6d1f-4222-8a5f-a517042...	FTL	trip-153861118270144424

12723 rows × 32 columns

In [37]:
```
trip[num_cols].boxplot(rot=25, figsize=(25,8))
```

Out[37]:
```
<Axes: >
```

Handling Categorical Variables

Only two route_type – Do one hot encoding

In [38]:
```
trip['route_type'].value_counts()
```

Out[38]:
```
route_type
Carting    8812
FTL        3911
Name: count, dtype: int64
```

In [39]:
```
trip['route_type'] = trip['route_type'].map({'FTL':0, 'Carting':1})
```

Normalize/ Standardize the numerical features using MinMaxScaler or StandardScaler

In [41]:
```
from sklearn.preprocessing import StandardScaler
```

In [42]:
```
scaler = StandardScaler()
scaler.fit(trip[num_cols])
```

Out[42]:
```
▾ StandardScaler
StandardScaler()
```

In [43]:
```
trip[num_cols] = scaler.transform(trip[num_cols])
```

In [44]:
```
trip[num_cols]
```

Out[44]:

	start_scan_to_end_scan	actual_distance_to_destination	actual_time	osrm_time	osrm
0	-0.548546	0.012060	-0.217856	-0.144341	
1	-0.861602	-0.765152	-0.749015	-0.877085	
2	1.552838	0.764988	1.034163	0.533102	
3	-0.513328	-0.662169	-0.736369	-0.766482	
4	-0.869428	-0.877197	-0.970332	-0.904736	
...	...	...	...	...	...
12718	-0.247231	-0.201970	-0.597255	-0.227293	
12719	-1.018130	-0.788207	-0.989302	-0.918561	
12720	0.394533	-0.466688	0.661086	-0.420848	
12721	0.104957	0.865940	0.547267	1.390274	
12722	0.128436	-0.086534	0.616823	-0.144341	

12723 rows × 9 columns

In [45]:
```
trip[num_cols].describe()
```

Out[45]:

	start_scan_to_end_scan	actual_distance_to_destination	actual_time	osrm_time
count	1.272300e+04	1.272300e+04	1.272300e+04	1.272300e+04
mean	-1.619566e-17	-7.371818e-17	-8.041983e-17	4.467769e-17
std	1.000039e+00	1.000039e+00	1.000039e+00	1.000039e+00
min	-1.162918e+00	-8.785574e-01	-1.065181e+00	-1.001514e+00
25%	-7.207269e-01	-7.065920e-01	-7.363685e-01	-7.111809e-01
50%	-3.411472e-01	-4.689012e-01	-4.012322e-01	-3.931975e-01
75%	4.023595e-01	4.073375e-01	4.650634e-01	4.224989e-01
max	4.049455e+00	4.178358e+00	4.031419e+00	4.113871e+00

Recomendation examples:

There is a significant difference between OSRM and actual parameters.

There is a need to:

Revisit information fed to routing engine for trip planning. Check for discrepancies with transporters, if the routing engine is configured for optimum results.

North, South and West Zones corridors have significant traffic of orders. But, we have a smaller presence in Central, Eastern and North-Eastern zone. However it would be difficult to conclude this, by looking at just 2 months data. It is worth investigating and increasing our presence in these regions.

From state point of view, we have heavy traffic in Mahrashtra followed by Karnataka. This is a good indicator that we need to plan for resources on ground in these 2 states on priority. Especially, during festive seasons.

In [ ]: