

In [129...]

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

import statsmodels
import statsmodels.api as sm
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler
import statsmodels.api as sm
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
from sklearn.metrics import accuracy_score
from sklearn import preprocessing
import warnings
warnings.filterwarnings('ignore')
```

In []:

In []:

In []:

In [130...]

```
Delivery_time_Data=pd.read_csv('porter_data_1.csv')
Delivery_time_Data.head()
```

Out[130...]

	market_id	created_at	actual_delivery_time	store_primary_category	order_protocol	total_weight
0	1.0	2015-02-06 22:24:17	2015-02-06 23:11:17	4	1.0	
1	2.0	2015-02-10 21:49:25	2015-02-10 22:33:25	46	2.0	
2	2.0	2015-02-16 00:11:35	2015-02-16 01:06:35	36	3.0	
3	1.0	2015-02-12 03:36:46	2015-02-12 04:35:46	38	1.0	
4	1.0	2015-01-27 02:12:36	2015-01-27 02:58:36	38	1.0	

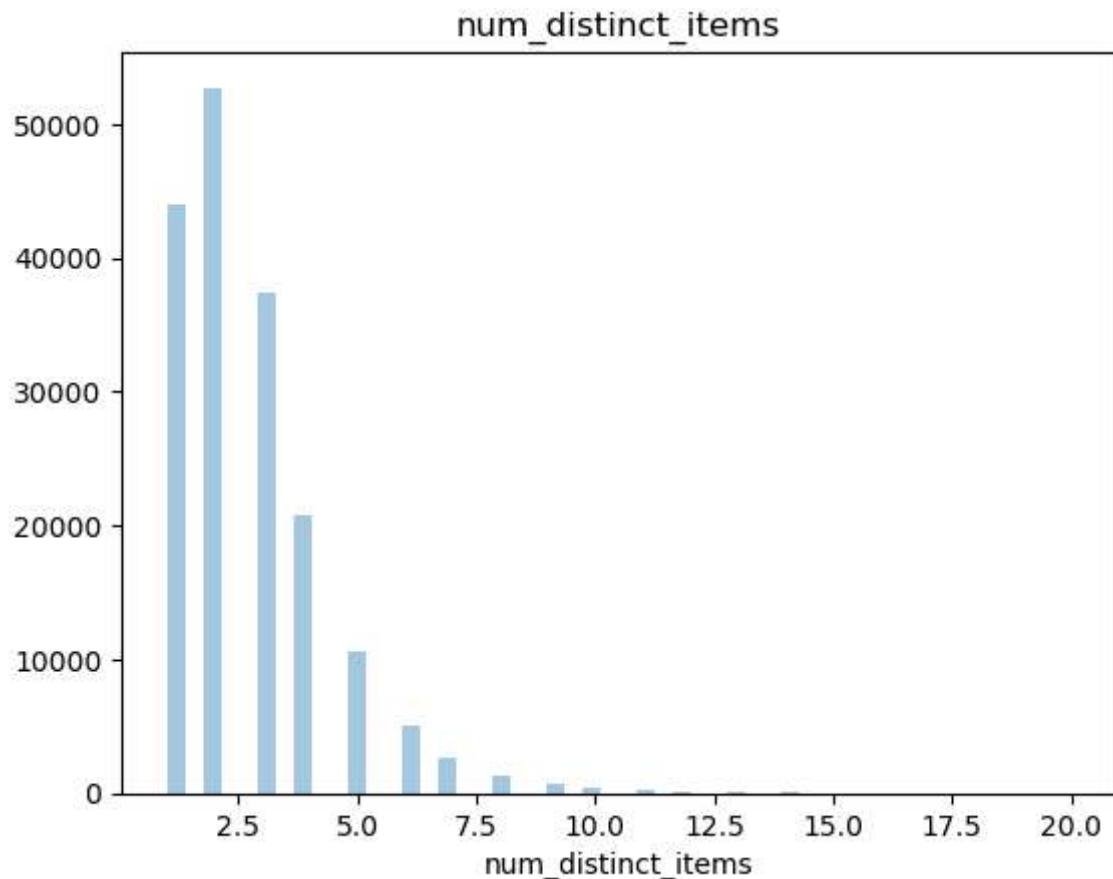
In [131...]

```
Delivery_time_Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175777 entries, 0 to 175776
Data columns (total 14 columns):
 #   Column           Non-Null Count   Dtype  
--- 
 0   market_id        175777 non-null    float64
 1   created_at       175777 non-null    object 
 2   actual_delivery_time 175777 non-null    object 
 3   store_primary_category 175777 non-null    int64  
 4   order_protocol    175777 non-null    float64
 5   total_items       175777 non-null    int64  
 6   subtotal          175777 non-null    int64  
 7   num_distinct_items 175777 non-null    int64  
 8   min_item_price    175777 non-null    int64  
 9   max_item_price    175777 non-null    int64  
 10  total_onshift_dashers 175777 non-null    float64
 11  total_busy_dashers 175777 non-null    float64
 12  total_outstanding_orders 175777 non-null    float64
 13  distance          175777 non-null    float64
dtypes: float64(6), int64(6), object(2)
memory usage: 18.8+ MB
```

In [132]:

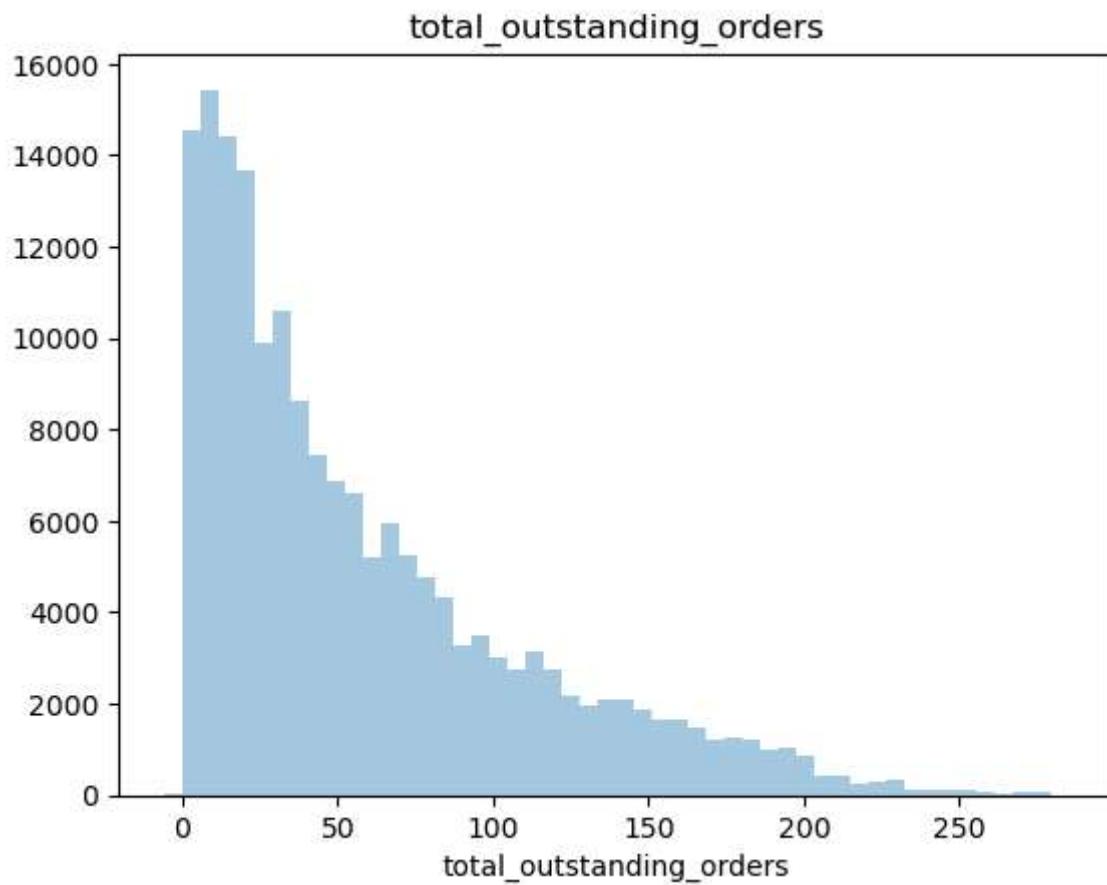
```
sns.distplot(Delivery_time_Data['num_distinct_items'], kde=False)
plt.title('num_distinct_items')
plt.show()
```



In [7]:

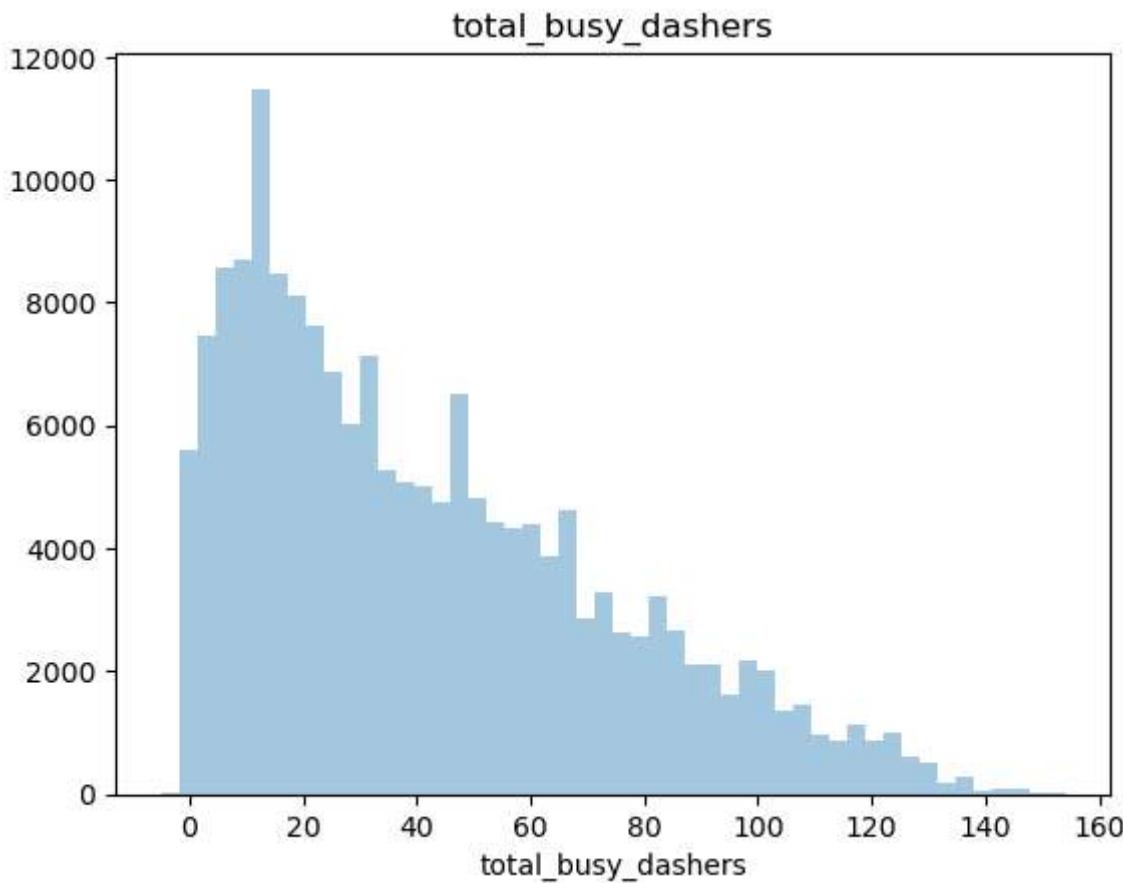
```
sns.distplot(Delivery_time_Data['total_outstanding_orders'], kde=False)
plt.title('total_outstanding_orders')
```

```
plt.show()
```



In []:

```
In [8]: sns.distplot(Delivery_time_Data['total_busy_dashers'],kde=False)
plt.title('total_busy_dashers')
plt.show()
```



```
In [9]: Delivery_time_Data.describe()
```

Out[9]:

	market_id	store_primary_category	order_protocol	total_items	subtot
count	175777.000000	175777.000000	175777.000000	175777.000000	175777.000000
mean	2.743726	35.887949	2.911752	3.204976	2697.111147
std	1.330963	20.728254	1.513128	2.674055	1828.554893
min	1.000000	0.000000	1.000000	1.000000	0.000000
25%	2.000000	18.000000	1.000000	2.000000	1412.000000
50%	2.000000	38.000000	3.000000	3.000000	2224.000000
75%	4.000000	55.000000	4.000000	4.000000	3410.000000
max	6.000000	72.000000	7.000000	411.000000	26800.000000



```
In [10]: Delivery_time_Data.columns
```

```
Out[10]: Index(['market_id', 'created_at', 'actual_delivery_time',
       'store_primary_category', 'order_protocol', 'total_items', 'subtotal',
       'num_distinct_items', 'min_item_price', 'max_item_price',
       'total_onshift_dashers', 'total_busy_dashers',
       'total_outstanding_orders', 'distance'],
      dtype='object')
```

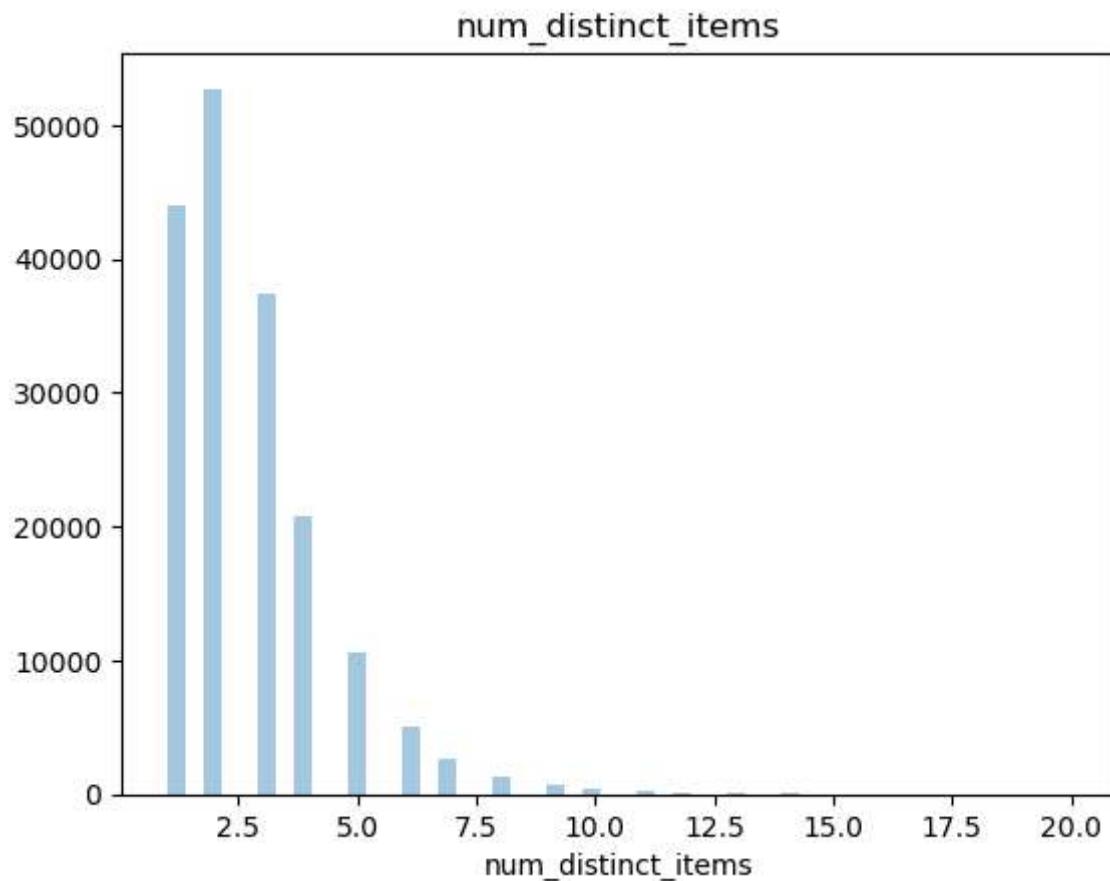
```
In [11]: Delivery_time_Data.describe()
```

```
Out[11]:
```

	market_id	store_primary_category	order_protocol	total_items	subtotal
count	175777.000000	175777.000000	175777.000000	175777.000000	175777.000000
mean	2.743726	35.887949	2.911752	3.204976	2697.111147
std	1.330963	20.728254	1.513128	2.674055	1828.554893
min	1.000000	0.000000	1.000000	1.000000	0.000000
25%	2.000000	18.000000	1.000000	2.000000	1412.000000
50%	2.000000	38.000000	3.000000	3.000000	2224.000000
75%	4.000000	55.000000	4.000000	4.000000	3410.000000
max	6.000000	72.000000	7.000000	411.000000	26800.000000



```
In [10]: sns.distplot(Delivery_time_Data['num_distinct_items'], kde=False)
plt.title('num_distinct_items')
plt.show()
```

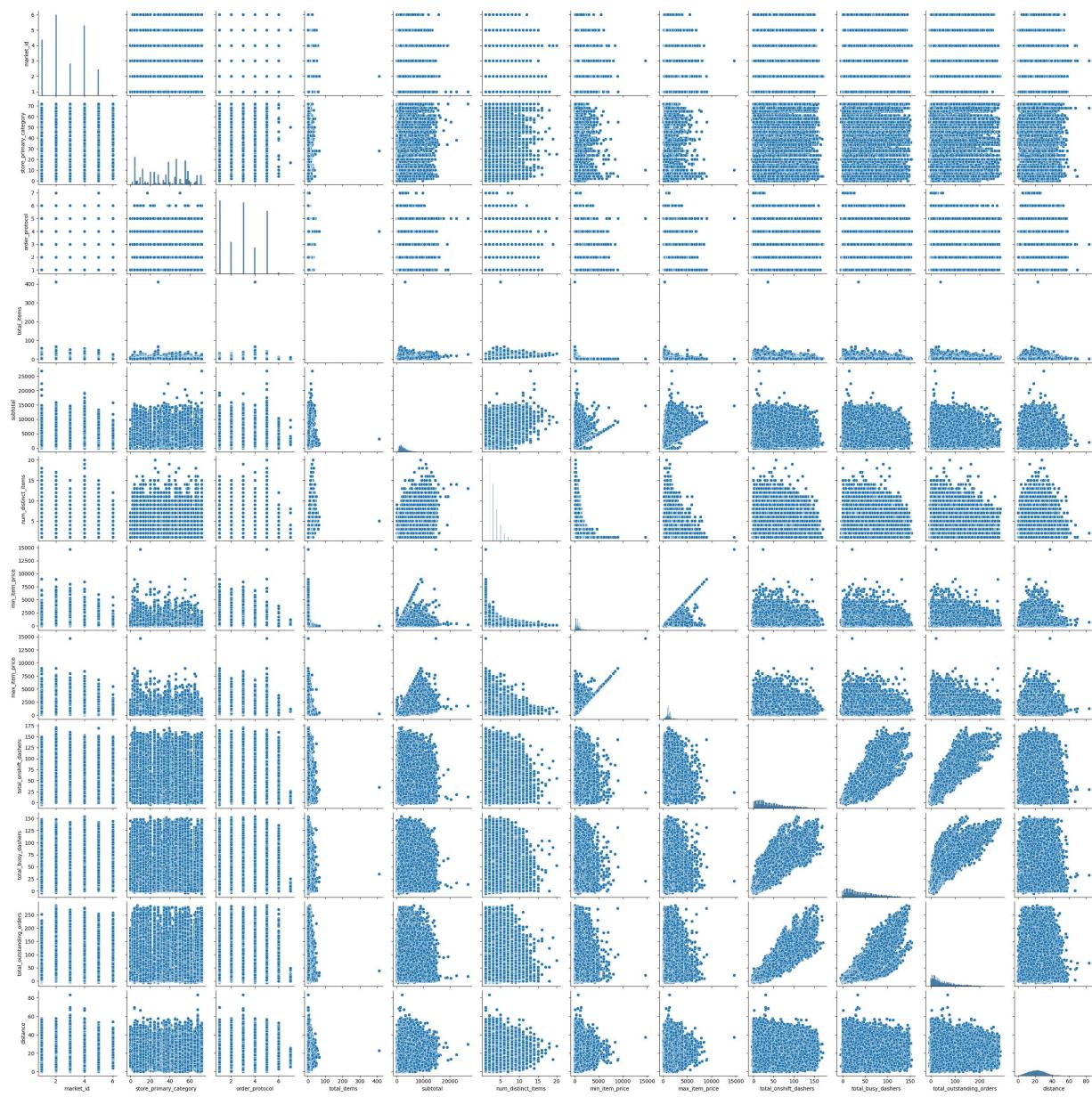


```
In [11]: Delivery_time_Data.describe(exclude=np.number).T
```

Out[11]:

	count	unique	top	freq
created_at	175777	162649	2015-02-11 19:50:43	6
actual_delivery_time	175777	160344	2015-02-14 03:01:28	5

```
In [12]: sns.pairplot(Delivery_time_Data)
plt.show()
```

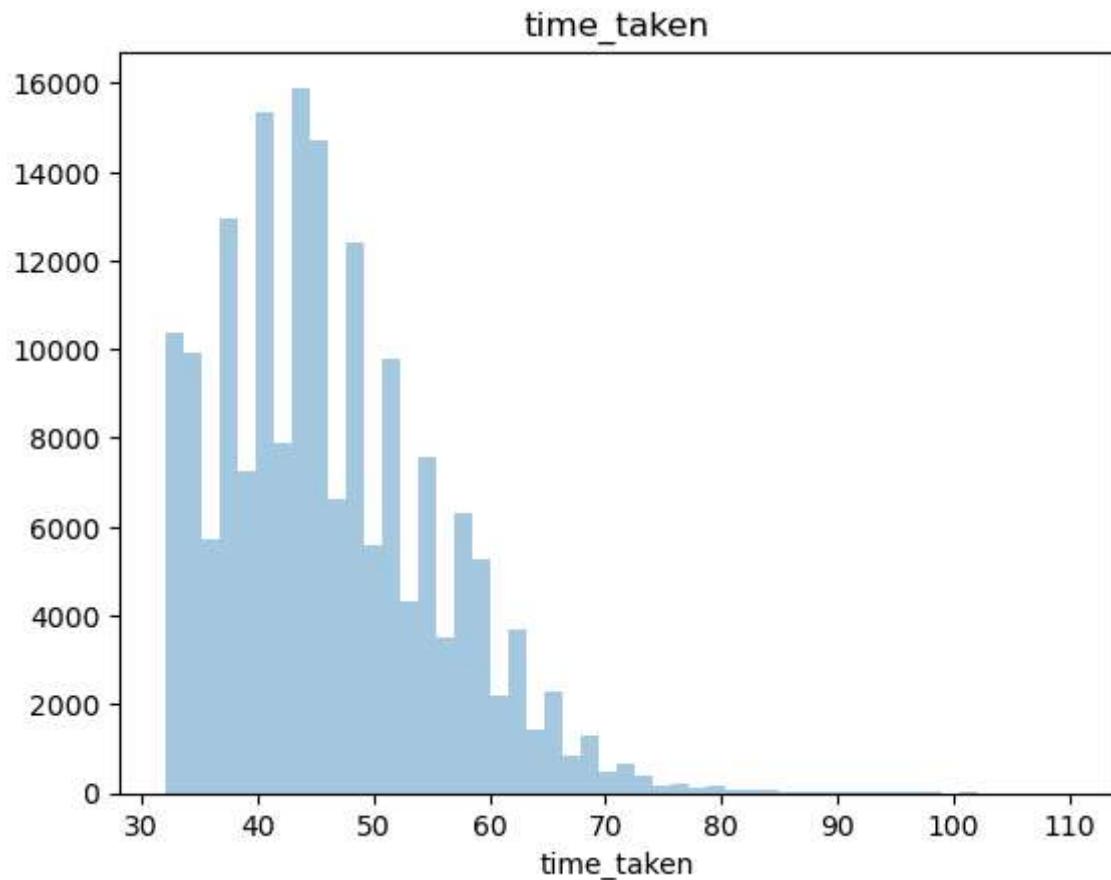


In [133...]

```
Delivery_time_Data['created_at_time_format'] = pd.to_datetime(Delivery_time_Data['  
Delivery_time_Data['actual_delivery_time_format'] = pd.to_datetime(Delivery_time_D  
Delivery_time_Data['time_taken'] = (Delivery_time_Data['actual_delivery_time_for  
Delivery_time_Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175777 entries, 0 to 175776
Data columns (total 17 columns):
 #   Column           Non-Null Count   Dtype  
--- 
 0   market_id        175777 non-null    float64
 1   created_at       175777 non-null    object 
 2   actual_delivery_time 175777 non-null    object 
 3   store_primary_category 175777 non-null    int64  
 4   order_protocol    175777 non-null    float64
 5   total_items       175777 non-null    int64  
 6   subtotal          175777 non-null    int64  
 7   num_distinct_items 175777 non-null    int64  
 8   min_item_price    175777 non-null    int64  
 9   max_item_price    175777 non-null    int64  
 10  total_onshift_dashers 175777 non-null    float64
 11  total_busy_dashers 175777 non-null    float64
 12  total_outstanding_orders 175777 non-null    float64
 13  distance          175777 non-null    float64
 14  created_at_time_format 175777 non-null    datetime64[ns]
 15  actual_delivery_time_format 175777 non-null    datetime64[ns]
 16  time_taken        175777 non-null    float64
dtypes: datetime64[ns](2), float64(7), int64(6), object(2)
memory usage: 22.8+ MB
```

```
In [134]: sns.distplot(Delivery_time_Data['time_taken'], kde=False)
plt.title('time_taken')
plt.show()
```



```
In [ ]: plt.figure(figsize=(20, 12))
plt.subplot(2,3,1)
sns.boxplot(x = 'actual_delivery_time', y = 'time_taken', data = Delivery_time_Data)
plt.subplot(2,3,2)
sns.boxplot(x = 'created_at', y = 'time_taken', data = Delivery_time_Data)
plt.show()
```

```
In [12]: Delivery_time_Data.describe().T
```

Out[12]:

	count	mean	std	min	25%	50%	75
market_id	175777.0	2.743726	1.330963	1.0	2.00	2.00	4.
store_primary_category	175777.0	35.887949	20.728254	0.0	18.00	38.00	55.
order_protocol	175777.0	2.911752	1.513128	1.0	1.00	3.00	4.
total_items	175777.0	3.204976	2.674055	1.0	2.00	3.00	4.
subtotal	175777.0	2697.111147	1828.554893	0.0	1412.00	2224.00	3410.
num_distinct_items	175777.0	2.675060	1.625681	1.0	1.00	2.00	3.
min_item_price	175777.0	684.965433	519.882924	-86.0	299.00	595.00	942.
max_item_price	175777.0	1160.158616	560.828571	0.0	799.00	1095.00	1395.
total_onshift_dashers	175777.0	44.918664	34.544724	-4.0	17.00	37.00	66.
total_busy_dashers	175777.0	41.861381	32.168505	-5.0	15.00	35.00	63.
total_outstanding_orders	175777.0	58.230115	52.731043	-6.0	17.00	41.00	85.
distance	175777.0	21.843090	8.748712	0.0	15.36	21.76	28.



In [135...]

```
Delivery_time_Data['created_at_time_format'] = pd.to_datetime(Delivery_time_Data['created_at'])
Delivery_time_Data['actual_delivery_time_format'] = pd.to_datetime(Delivery_time_Data['actual_delivery_time'])
Delivery_time_Data['time_taken'] = (Delivery_time_Data['actual_delivery_time_format'] - Delivery_time_Data['created_at_time_format']).dt.total_seconds()
Delivery_time_Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175777 entries, 0 to 175776
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   market_id        175777 non-null   float64
 1   created_at       175777 non-null   object 
 2   actual_delivery_time 175777 non-null   object 
 3   store_primary_category 175777 non-null   int64  
 4   order_protocol    175777 non-null   float64
 5   total_items       175777 non-null   int64  
 6   subtotal          175777 non-null   int64  
 7   num_distinct_items 175777 non-null   int64  
 8   min_item_price    175777 non-null   int64  
 9   max_item_price    175777 non-null   int64  
 10  total_onshift_dashers 175777 non-null   float64
 11  total_busy_dashers 175777 non-null   float64
 12  total_outstanding_orders 175777 non-null   float64
 13  distance          175777 non-null   float64
 14  created_at_time_format 175777 non-null   datetime64[ns]
 15  actual_delivery_time_format 175777 non-null   datetime64[ns]
 16  time_taken        175777 non-null   float64
dtypes: datetime64[ns](2), float64(7), int64(6), object(2)
memory usage: 22.8+ MB
```

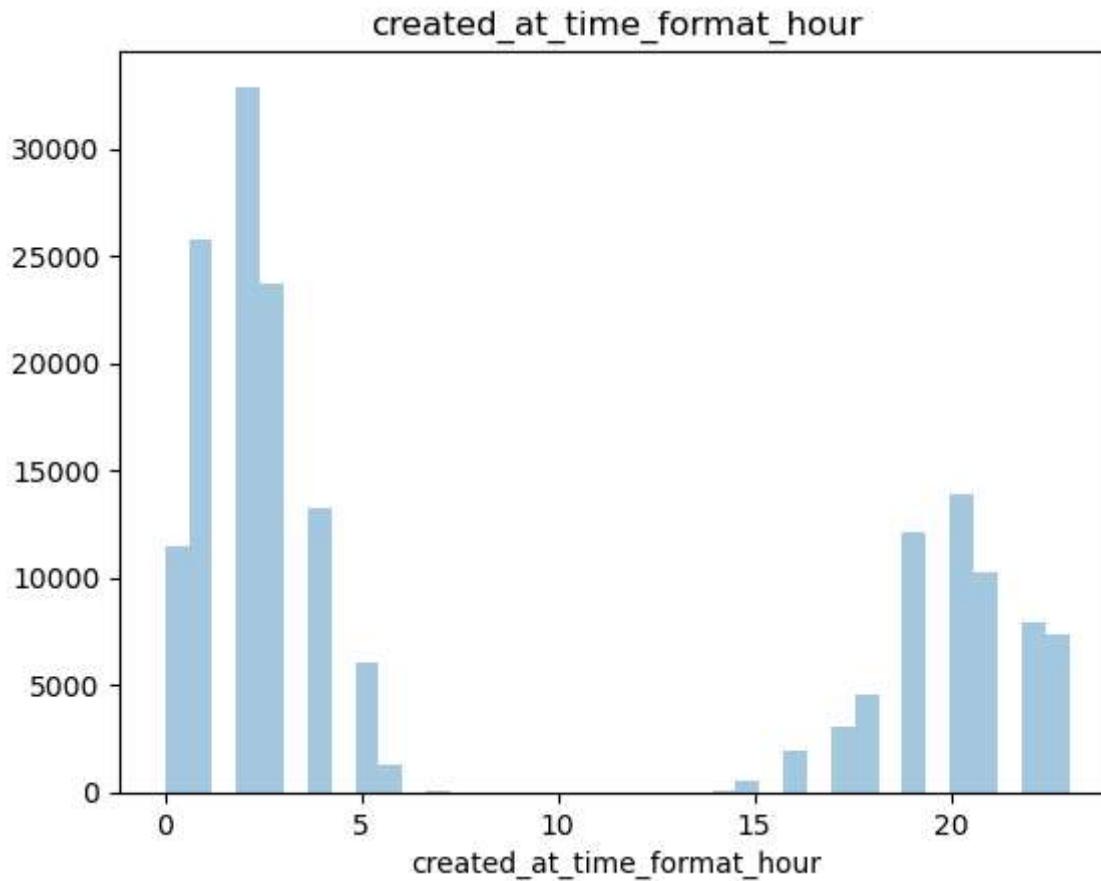
In []:

```
In [136...]: def extract_date_features(data):
    data["created_at_time_format_day"] = data.created_at_time_format.dt.day
    data["created_at_time_format_hour"] = data.created_at_time_format.dt.hour
    data["created_at_time_format_day_name"] = data.created_at_time_format.dt.day_name
    data["created_at_time_format_weekday"] = data.created_at_time_format.dt.weekday
    data['created_at_time_format_day_of_week'] = data.created_at_time_format.dt.day
    data['created_at_time_format_is_weekend'] = np.where(data['created_at_time_form

extract_date_features(Delivery_time_Data)
Delivery_time_Data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175777 entries, 0 to 175776
Data columns (total 23 columns):
 #   Column           Non-Null Count  Dtype  
---  -- 
 0   market_id        175777 non-null   float64 
 1   created_at       175777 non-null   object  
 2   actual_delivery_time 175777 non-null   object  
 3   store_primary_category 175777 non-null   int64  
 4   order_protocol    175777 non-null   float64 
 5   total_items       175777 non-null   int64  
 6   subtotal          175777 non-null   int64  
 7   num_distinct_items 175777 non-null   int64  
 8   min_item_price    175777 non-null   int64  
 9   max_item_price    175777 non-null   int64  
 10  total_onshift_dashers 175777 non-null   float64 
 11  total_busy_dashers 175777 non-null   float64 
 12  total_outstanding_orders 175777 non-null   float64 
 13  distance          175777 non-null   float64 
 14  created_at_time_format 175777 non-null   datetime64[ns] 
 15  actual_delivery_time_format 175777 non-null   datetime64[ns] 
 16  time_taken         175777 non-null   float64 
 17  created_at_time_format_day 175777 non-null   int32  
 18  created_at_time_format_hour 175777 non-null   int32  
 19  created_at_time_format_day_name 175777 non-null   object  
 20  created_at_time_format_weekday 175777 non-null   int32  
 21  created_at_time_format_day_of_week 175777 non-null   int32  
 22  created_at_time_format_is_weekend 175777 non-null   int32  
dtypes: datetime64[ns](2), float64(7), int32(5), int64(6), object(3)
memory usage: 27.5+ MB
```

```
In [137...]: sns.distplot(Delivery_time_Data['created_at_time_format_hour'], kde=False)
plt.title('created_at_time_format_hour')
plt.show()
```



In [138]:

```
def extract_date_features(data):
    data["actual_delivery_time_format_day"] = data.actual_delivery_time_format.dt.day
    data["actual_delivery_time_format_hour"] = data.actual_delivery_time_format.dt.hour
    data["actual_delivery_time_format_day_name"] = data.actual_delivery_time_format.dt.day_name
    data["actual_delivery_time_format_weekday"] = data.actual_delivery_time_format.dt.weekday
    data['actual_delivery_time_format_day_of_week'] = data.actual_delivery_time_format.dt.isocalendar().week
    data['actual_delivery_time_format_is_weekend'] = np.where(data['actual_delivery_time_format_weekday'] > 4, 1, 0)

extract_date_features(Delivery_time_Data)
Delivery_time_Data.info()
```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175777 entries, 0 to 175776
Data columns (total 29 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   market_id        175777 non-null   float64
 1   created_at       175777 non-null   object 
 2   actual_delivery_time 175777 non-null   object 
 3   store_primary_category 175777 non-null   int64  
 4   order_protocol    175777 non-null   float64
 5   total_items       175777 non-null   int64  
 6   subtotal          175777 non-null   int64  
 7   num_distinct_items 175777 non-null   int64  
 8   min_item_price    175777 non-null   int64  
 9   max_item_price    175777 non-null   int64  
 10  total_onshift_dashers 175777 non-null   float64
 11  total_busy_dashers 175777 non-null   float64
 12  total_outstanding_orders 175777 non-null   float64
 13  distance          175777 non-null   float64
 14  created_at_time_format 175777 non-null   datetime64[ns]
 15  actual_delivery_time_format 175777 non-null   datetime64[ns]
 16  time_taken        175777 non-null   float64
 17  created_at_time_format_day 175777 non-null   int32  
 18  created_at_time_format_hour 175777 non-null   int32  
 19  created_at_time_format_day_name 175777 non-null   object 
 20  created_at_time_format_weekday 175777 non-null   int32  
 21  created_at_time_format_day_of_week 175777 non-null   int32  
 22  created_at_time_format_is_weekend 175777 non-null   int32  
 23  actual_delivery_time_format_day 175777 non-null   int32  
 24  actual_delivery_time_format_hour 175777 non-null   int32  
 25  actual_delivery_time_format_day_name 175777 non-null   object 
 26  actual_delivery_time_format_weekday 175777 non-null   int32  
 27  actual_delivery_time_format_day_of_week 175777 non-null   int32  
 28  actual_delivery_time_format_is_weekend 175777 non-null   int32  
dtypes: datetime64[ns](2), float64(7), int32(10), int64(6), object(4)
memory usage: 32.2+ MB

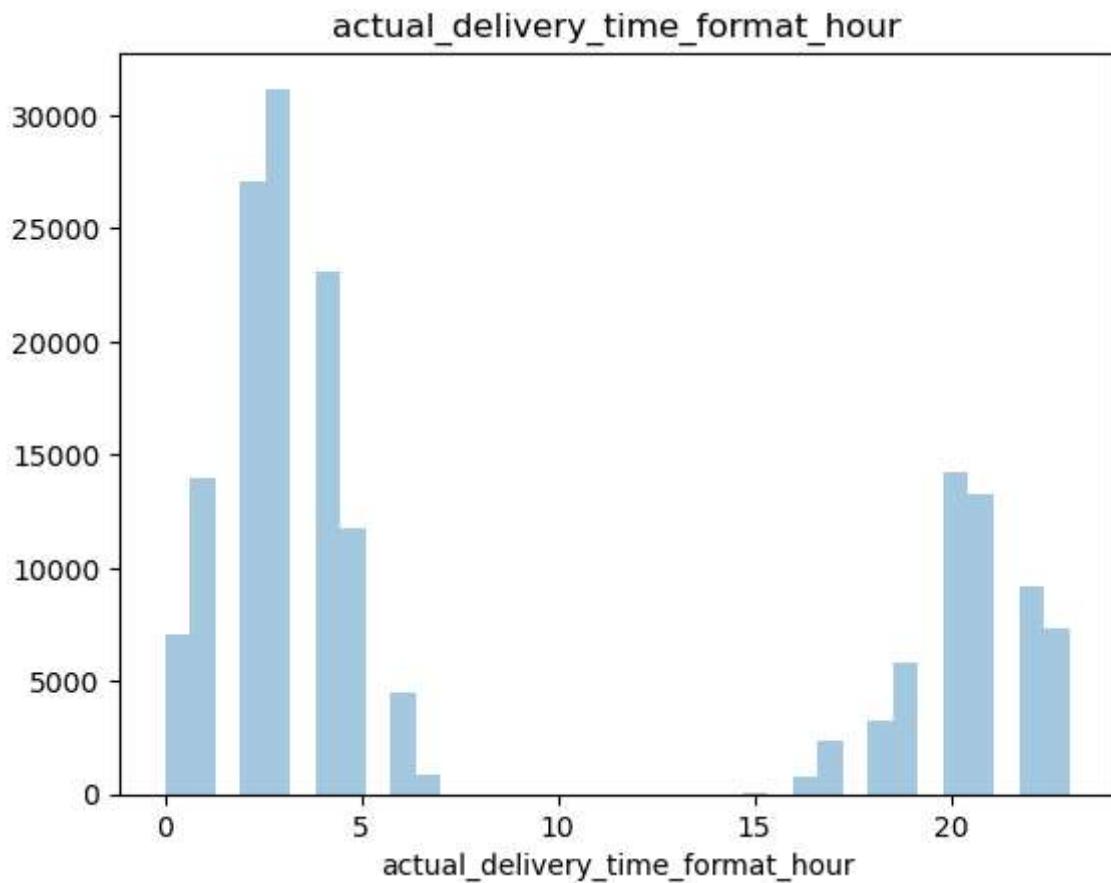
```

In [139...]

```

sns.distplot(Delivery_time_Data['actual_delivery_time_format_hour'], kde=False)
plt.title('actual_delivery_time_format_hour')
plt.show()

```



```
In [17]: # Convert 'Category' column to categorical type
Delivery_time_Data['actual_delivery_time_format_is_weekend'] = Delivery_time_Data['Category']
Delivery_time_Data.info()
```

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175777 entries, 0 to 175776
Data columns (total 29 columns):
 #   Column           Non-Null Count Dtype
 ---  -- 
 0   market_id        175777 non-null float64
 1   created_at       175777 non-null object
 2   actual_delivery_time 175777 non-null object
 3   store_primary_category 175777 non-null int64
 4   order_protocol    175777 non-null float64
 5   total_items       175777 non-null int64
 6   subtotal          175777 non-null int64
 7   num_distinct_items 175777 non-null int64
 8   min_item_price    175777 non-null int64
 9   max_item_price    175777 non-null int64
 10  total_onshift_dashers 175777 non-null float64
 11  total_busy_dashers 175777 non-null float64
 12  total_outstanding_orders 175777 non-null float64
 13  distance          175777 non-null float64
 14  created_at_time_format 175777 non-null datetime64[ns]
 15  actual_delivery_time_format 175777 non-null datetime64[ns]
 16  time_taken         175777 non-null float64
 17  created_at_time_format_day 175777 non-null int32
 18  created_at_time_format_hour 175777 non-null int32
 19  created_at_time_format_day_name 175777 non-null object
 20  created_at_time_format_weekday 175777 non-null int32
 21  created_at_time_format_day_of_week 175777 non-null int32
 22  created_at_time_format_is_weekend 175777 non-null int32
 23  actual_delivery_time_format_day 175777 non-null int32
 24  actual_delivery_time_format_hour 175777 non-null int32
 25  actual_delivery_time_format_day_name 175777 non-null object
 26  actual_delivery_time_format_weekday 175777 non-null int32
 27  actual_delivery_time_format_day_of_week 175777 non-null int32
 28  actual_delivery_time_format_is_weekend 175777 non-null category
dtypes: category(1), datetime64[ns](2), float64(7), int32(9), int64(6), object(4)
memory usage: 31.7+ MB

```

In []:

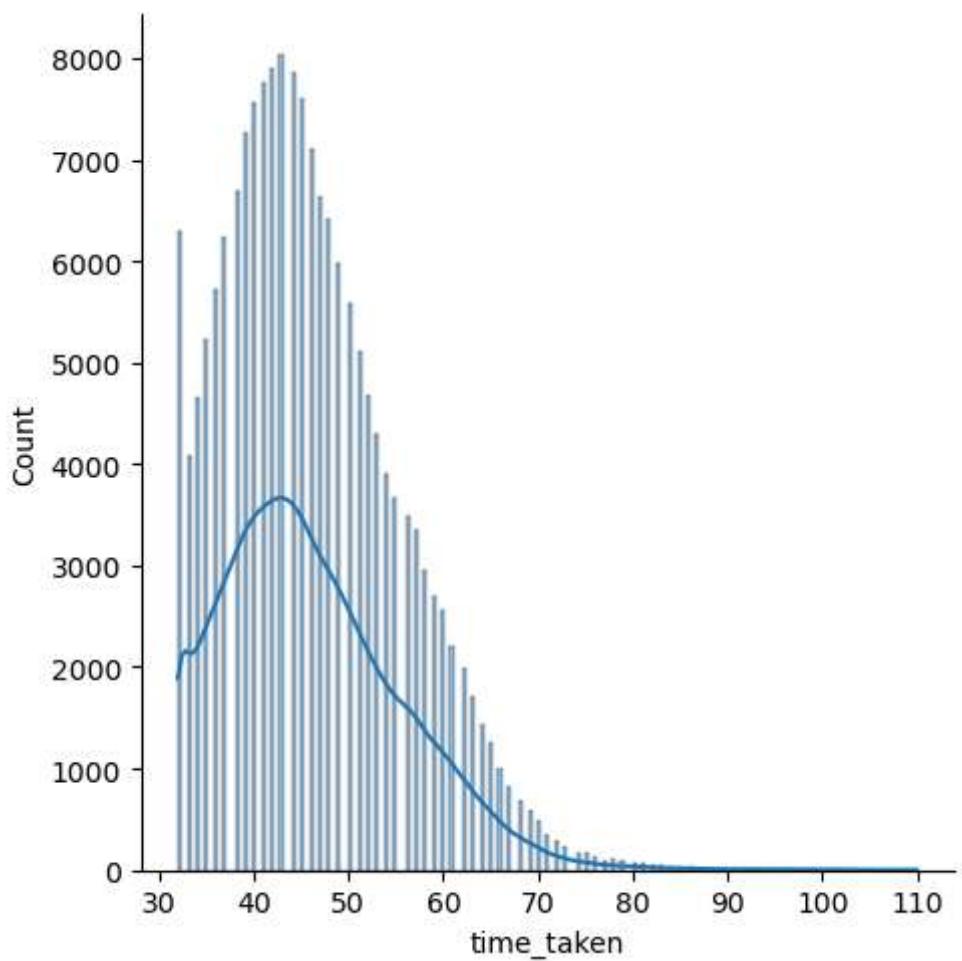
In [141...]

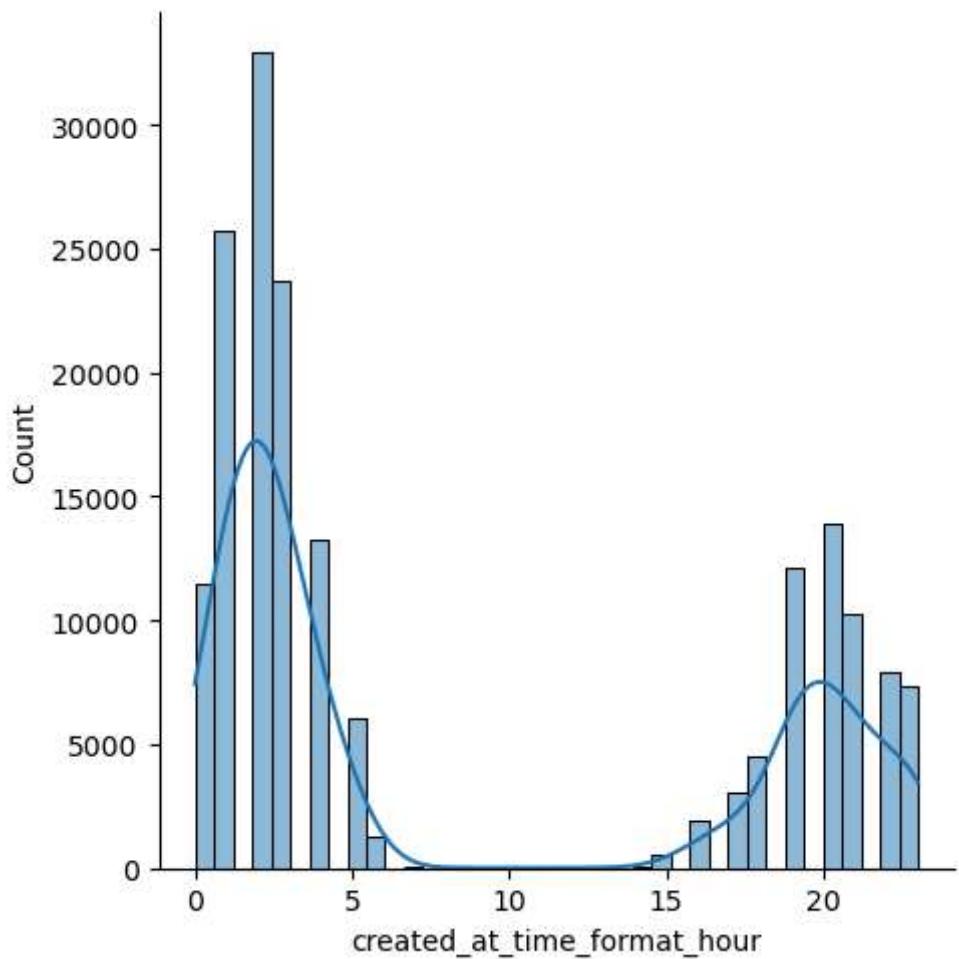
```

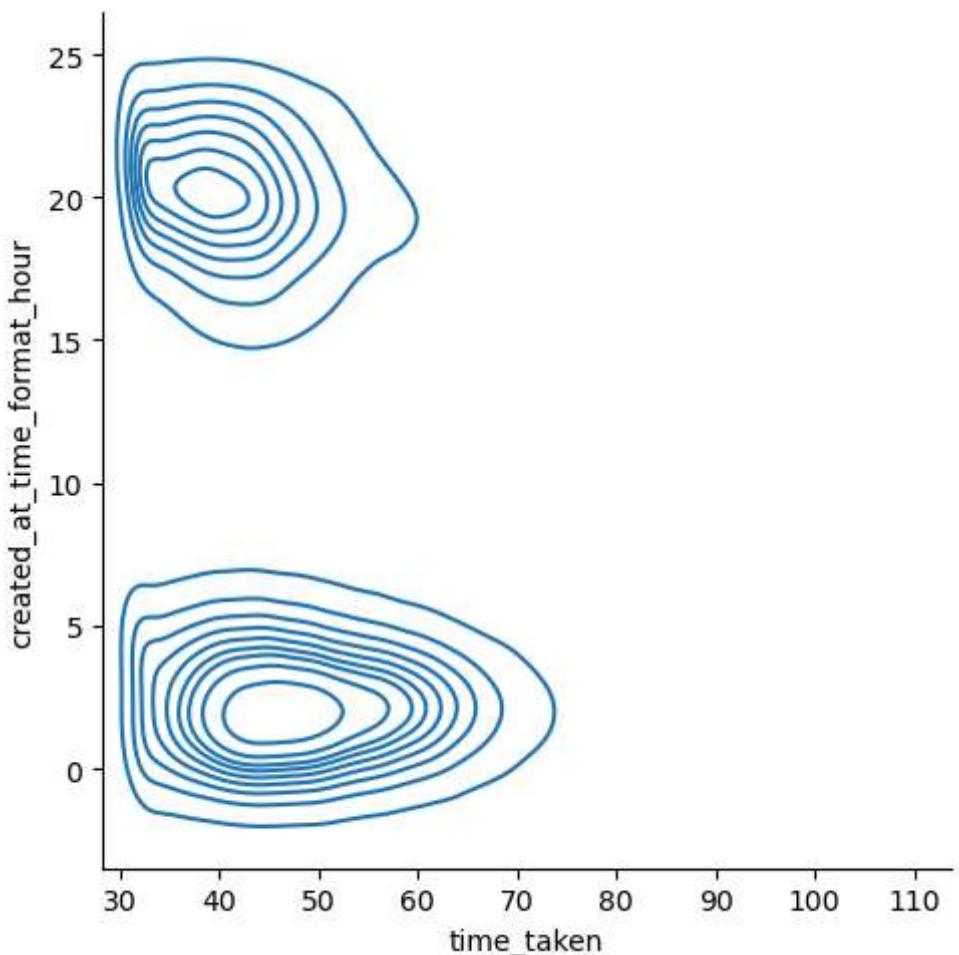
# Using displot for individual distributions
sns.displot(Delivery_time_Data, x='time_taken', kde=True)
sns.displot(Delivery_time_Data, x='created_at_time_format_hour', kde=True)
plt.show()

# Using displot for joint distribution
sns.displot(Delivery_time_Data, x='time_taken', y='created_at_time_format_hour',
plt.show()

```







In []:

In [25]: `print(Delivery_time_Data.columns.tolist())`

```
['market_id', 'created_at', 'actual_delivery_time', 'store_primary_category', 'order_protocol', 'total_items', 'subtotal', 'num_distinct_items', 'min_item_price', 'max_item_price', 'total_onshift_dashers', 'total_busy_dashers', 'total_outstanding_orders', 'distance', 'created_at_time_format', 'actual_delivery_time_format', 'time_taken', 'actual_delivery_time_format_day', 'actual_delivery_time_format_hour', 'actual_delivery_time_format_day_name', 'actual_delivery_time_format_weekday', 'actual_delivery_time_format_day_of_week', 'actual_delivery_time_format_is_weekend']
```

In []:

In [26]: `Delivery_time_Data.drop(columns=['actual_delivery_time_format_day', 'actual_delivery_time_format_hour'])`In [27]: `Delivery_time_Data.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 175777 entries, 0 to 175776
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   market_id        175777 non-null   float64
 1   created_at       175777 non-null   object 
 2   actual_delivery_time 175777 non-null   object 
 3   store_primary_category 175777 non-null   int64  
 4   order_protocol    175777 non-null   float64
 5   total_items       175777 non-null   int64  
 6   subtotal          175777 non-null   int64  
 7   num_distinct_items 175777 non-null   int64  
 8   min_item_price    175777 non-null   int64  
 9   max_item_price    175777 non-null   int64  
 10  total_onshift_dashers 175777 non-null   float64
 11  total_busy_dashers 175777 non-null   float64
 12  total_outstanding_orders 175777 non-null   float64
 13  distance          175777 non-null   float64
 14  created_at_time_format 175777 non-null   datetime64[ns]
 15  actual_delivery_time_format 175777 non-null   datetime64[ns]
 16  time_taken        175777 non-null   float64
dtypes: datetime64[ns](2), float64(7), int64(6), object(2)
memory usage: 22.8+ MB
```

In [34]: `print(Delivery_time_Data.columns.tolist())`

```
['market_id', 'created_at', 'actual_delivery_time', 'store_primary_category', 'order_protocol', 'total_items', 'subtotal', 'num_distinct_items', 'min_item_price', 'max_item_price', 'total_onshift_dashers', 'total_busy_dashers', 'total_outstanding_orders', 'distance', 'created_at_time_format', 'actual_delivery_time_format', 'time_taken']
```

In [35]: `Delivery_time_Data.drop(columns=['created_at', 'actual_delivery_time', 'created_at'])`

In []:

In [36]: `cat_cols=Delivery_time_Data.select_dtypes(include=['object']).columns
num_cols = Delivery_time_Data.select_dtypes(include=np.number).columns.tolist()
print("Categorical Variables:")
print(cat_cols)
print("Numerical Variables:")
print(num_cols)`

```
Categorical Variables:  
Index([], dtype='object')  
Numerical Variables:  
['market_id', 'store_primary_category', 'order_protocol', 'total_items', 'subtotal',  
'num_distinct_items', 'min_item_price', 'max_item_price', 'total_onshift_dashers',  
'total_busy_dashers', 'total_outstanding_orders', 'distance', 'time_taken']
```

In [158...]: `Delivery_time_Data = Delivery_time_Data[Delivery_time_Data.total_items <=100]
Delivery_time_Data.describe()`

Out[158...]

	market_id	store_primary_category	order_protocol	total_items	subtot
count	175776.000000	175776.000000	175776.000000	175776.000000	175776.000000
mean	2.743731	35.887994	2.911746	3.202656	2697.108765
min	1.000000	0.000000	1.000000	1.000000	0.000000
25%	2.000000	18.000000	1.000000	2.000000	1412.000000
50%	2.000000	38.000000	3.000000	3.000000	2224.000000
75%	4.000000	55.000000	4.000000	4.000000	3410.000000
max	6.000000	72.000000	7.000000	66.000000	26800.000000
std	1.330966	20.728305	1.513130	2.490889	1828.559823

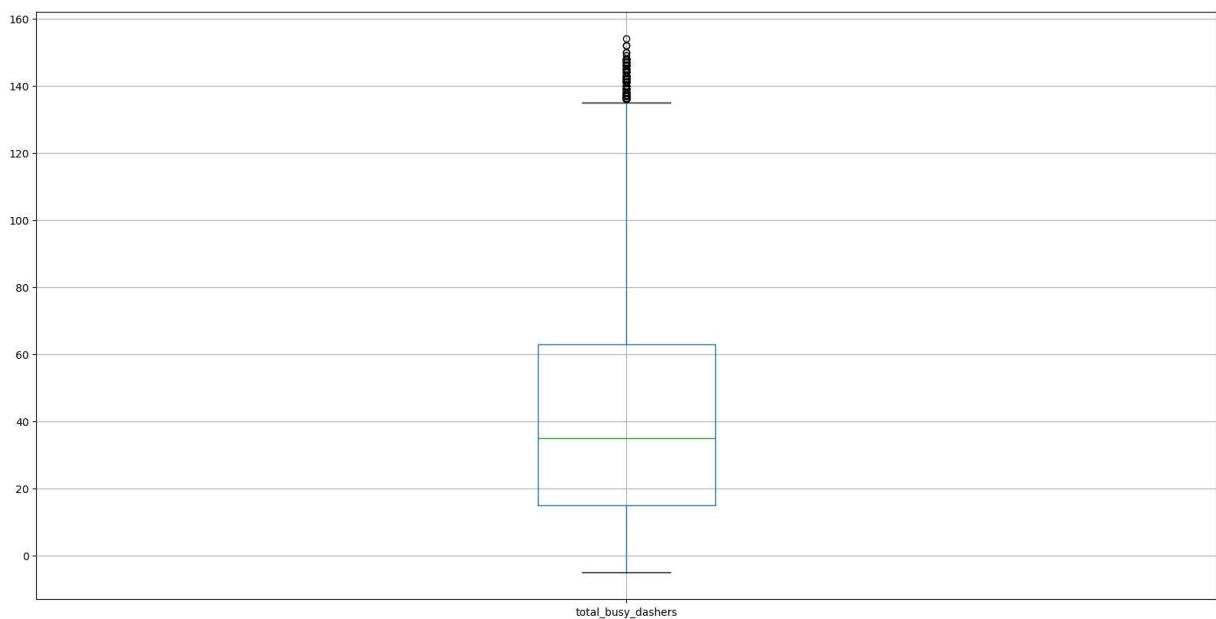
8 rows × 25 columns



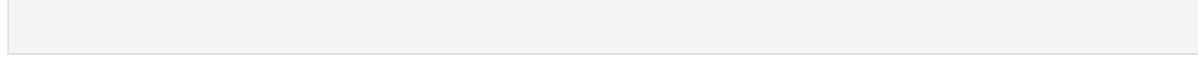
In [166...]

```
import pandas as pd
import matplotlib.pyplot as plt

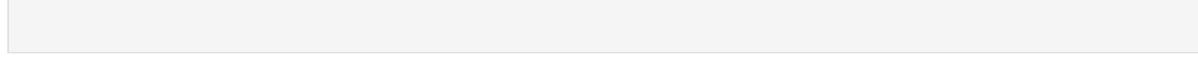
# Assuming df is your DataFrame and 'column_name' is the column you want to plot
Delivery_time_Data.boxplot(column=['total_busy_dashers'], figsize=(20,10))
plt.show()
```



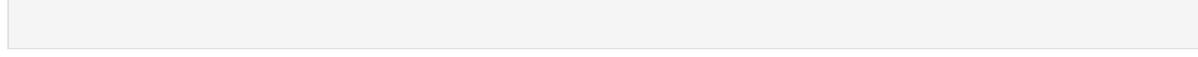
In []:



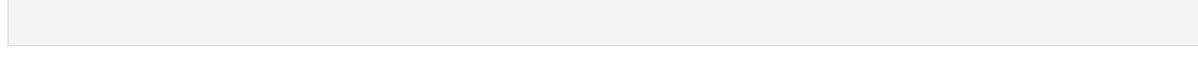
In []:



In []:



In []:

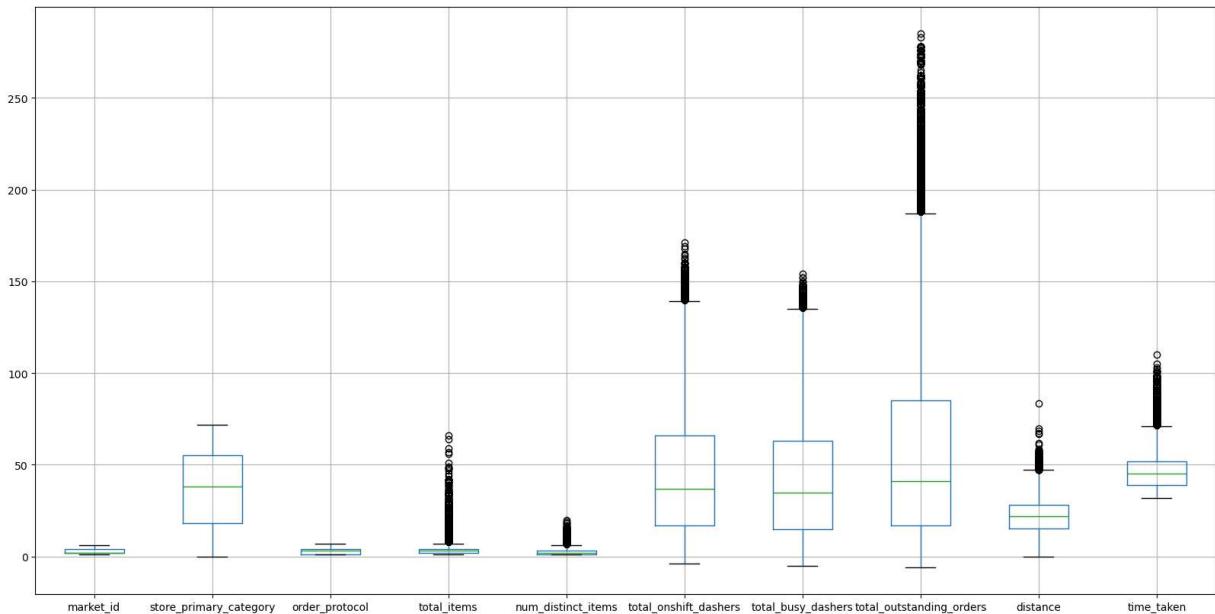


In []:

In [159]:

```
import pandas as pd
import matplotlib.pyplot as plt

# Assuming df is your DataFrame and 'column_name' is the column you want to plot
Delivery_time_Data.boxplot(column=['market_id', 'store_primary_category', 'order_protocol', 'total_items', 'num_distinct_items', 'total_onshift_dashers', 'total_busy_dashers', 'total_outstanding_orders', 'distance', 'time_taken'])
plt.show()
```



In [37]:

```
from sklearn.model_selection import train_test_split

# We specify this so that the train and test data set always have the same rows, re
np.random.seed(0)
df_train, df_test = train_test_split(Delivery_time_Data, train_size = 0.7, test_siz
```

In [38]:

```
from sklearn.preprocessing import MinMaxScaler
```

In [39]:

```
scaler = MinMaxScaler()
```

In [40]:

```
df_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 123043 entries, 94746 to 38408
Data columns (total 13 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   market_id        123043 non-null   float64
 1   store_primary_category 123043 non-null   int64  
 2   order_protocol    123043 non-null   float64
 3   total_items       123043 non-null   int64  
 4   subtotal          123043 non-null   int64  
 5   num_distinct_items 123043 non-null   int64  
 6   min_item_price    123043 non-null   int64  
 7   max_item_price    123043 non-null   int64  
 8   total_onshift_dashers 123043 non-null   float64
 9   total_busy_dashers 123043 non-null   float64
 10  total_outstanding_orders 123043 non-null   float64
 11  distance          123043 non-null   float64
 12  time_taken        123043 non-null   float64
dtypes: float64(7), int64(6)
memory usage: 13.1 MB
```

In []:

```
In [43]: # Apply scaler() to all the columns except the 'yes-no' and 'dummy' variables
num_vars = ['market_id', 'store_primary_category', 'order_protocol', 'total_items',
            df_train[num_vars] = scaler.fit_transform(df_train[num_vars])
```

In [44]: df_train.head()

	market_id	store_primary_category	order_protocol	total_items	subtotal	num_dist
94746	0.6	0.333333	0.666667	0.002439	0.066791	
173338	0.6	1.000000	0.666667	0.000000	0.031530	
37592	0.6	0.763889	0.666667	0.000000	0.070896	
42763	0.2	0.388889	0.500000	0.012195	0.017276	
27506	0.2	1.000000	0.000000	0.004878	0.130597	



In [45]: df_train.describe()

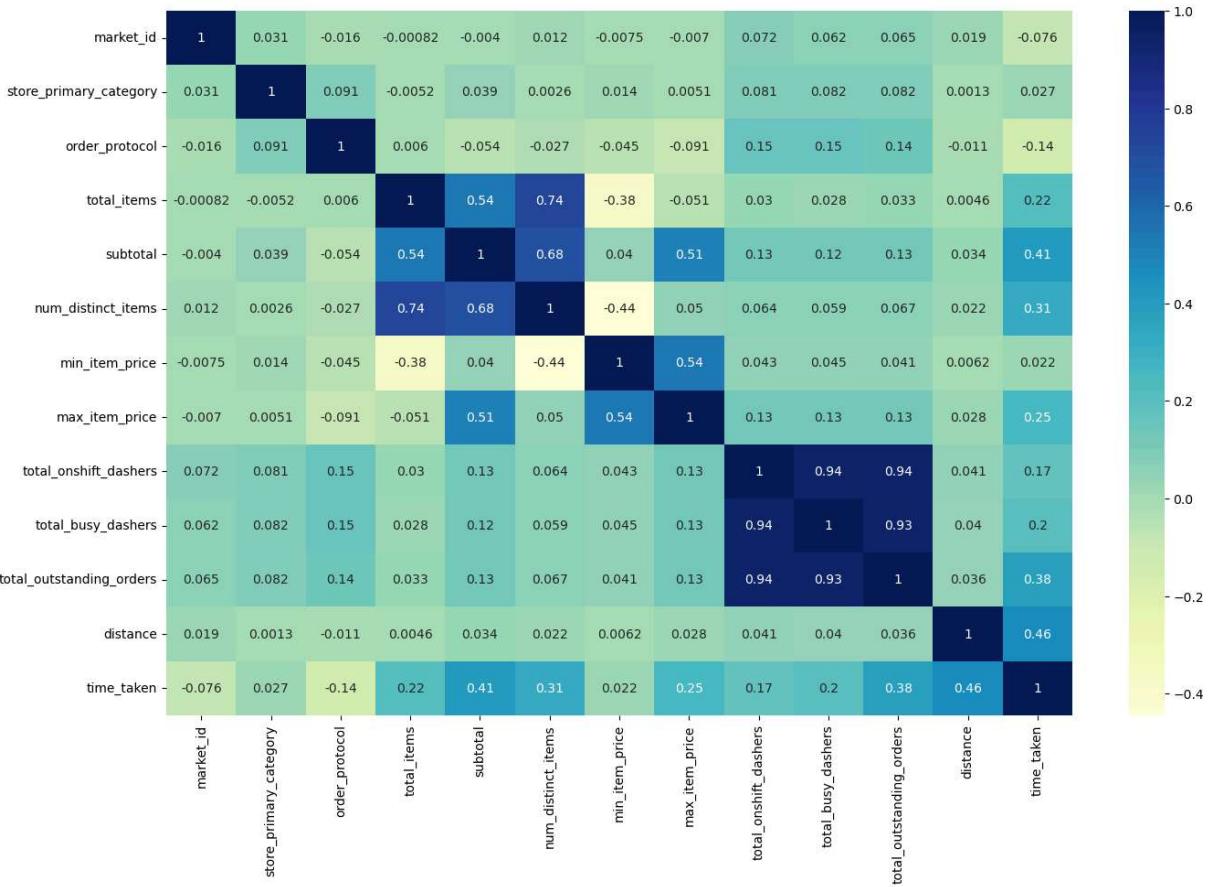
Out[45]:

	market_id	store_primary_category	order_protocol	total_items	subtotal
count	123043.000000	123043.000000	123043.000000	123043.000000	123043.000000
mean	0.348849	0.499179	0.318796	0.005381	0.100667
std	0.265612	0.288032	0.252108	0.006695	0.068296
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.200000	0.250000	0.000000	0.002439	0.052873
50%	0.200000	0.527778	0.333333	0.004878	0.082836
75%	0.600000	0.763889	0.500000	0.007317	0.127052
max	1.000000	1.000000	1.000000	1.000000	1.000000



In []:

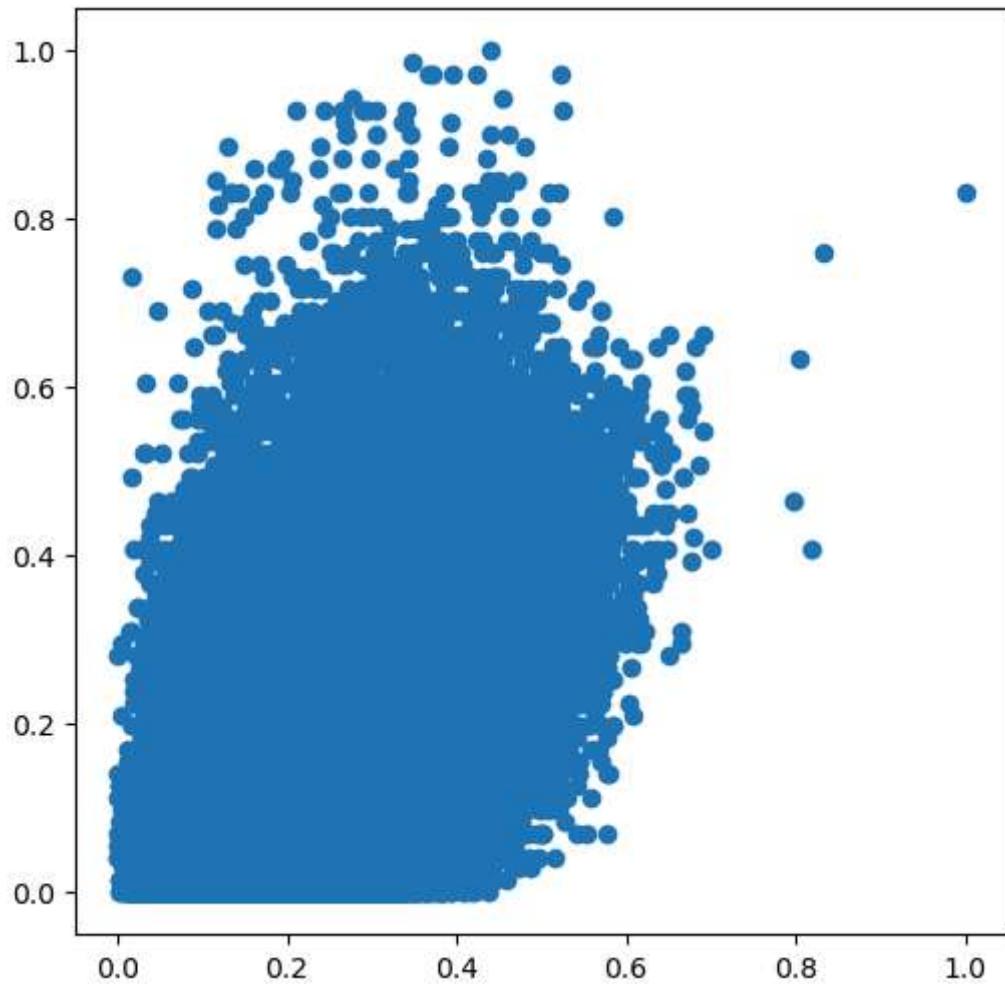
```
In [46]: plt.figure(figsize = (16, 10))
sns.heatmap(df_train.corr(), annot = True, cmap="YlGnBu")
plt.show()
```



In []:

```
In [47]: plt.figure(figsize=[6,6])
plt.scatter(df_train.distance, df_train.time_taken)
```

```
plt.show()
```



```
In [ ]:
```

```
In [48]: y_train = df_train.pop('time_taken')
X_train = df_train
```

```
In [ ]:
```

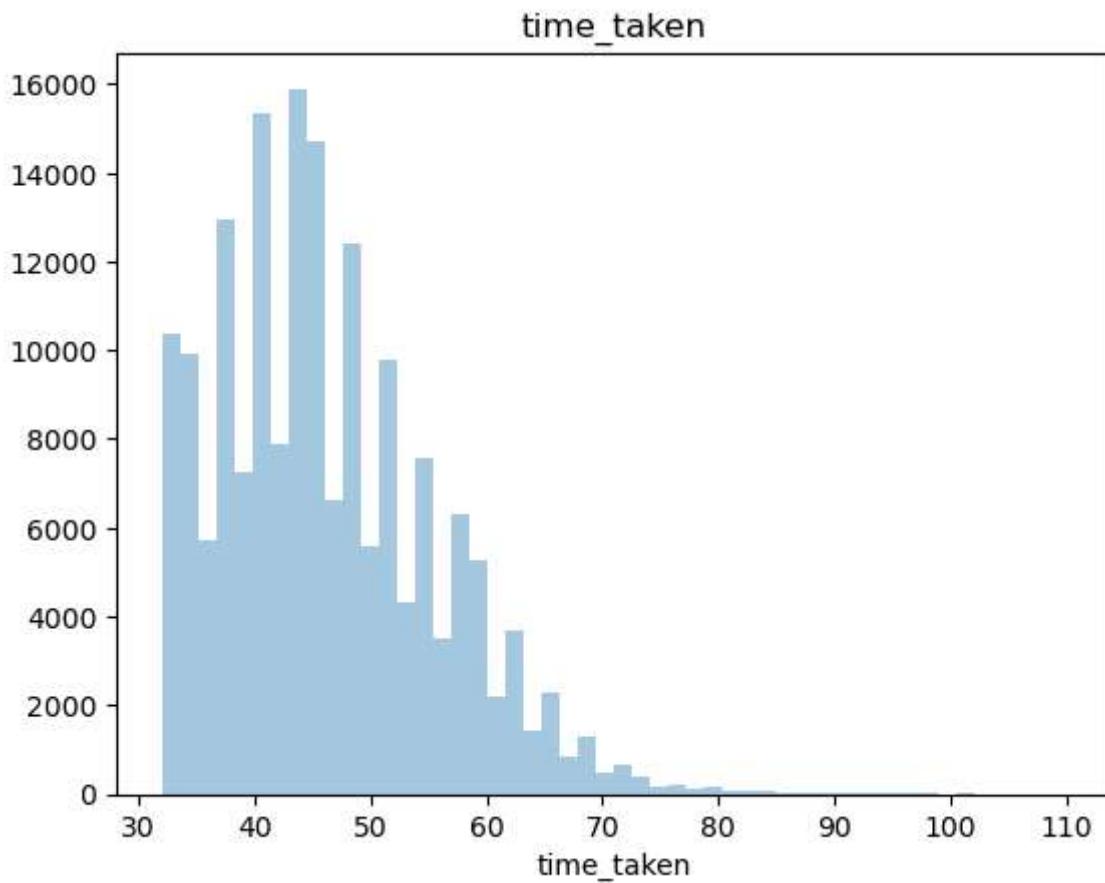
```
In [49]: import statsmodels.api as sm

# Add a constant
X_train_lm = sm.add_constant(X_train[['distance']])

# Create a first fitted model
lr = sm.OLS(y_train, X_train_lm).fit()
```

```
In [ ]:
```

```
In [126... sns.distplot(Delivery_time_Data['time_taken'], kde=False)
plt.title('time_taken')
plt.show()
```



In []:

In []:

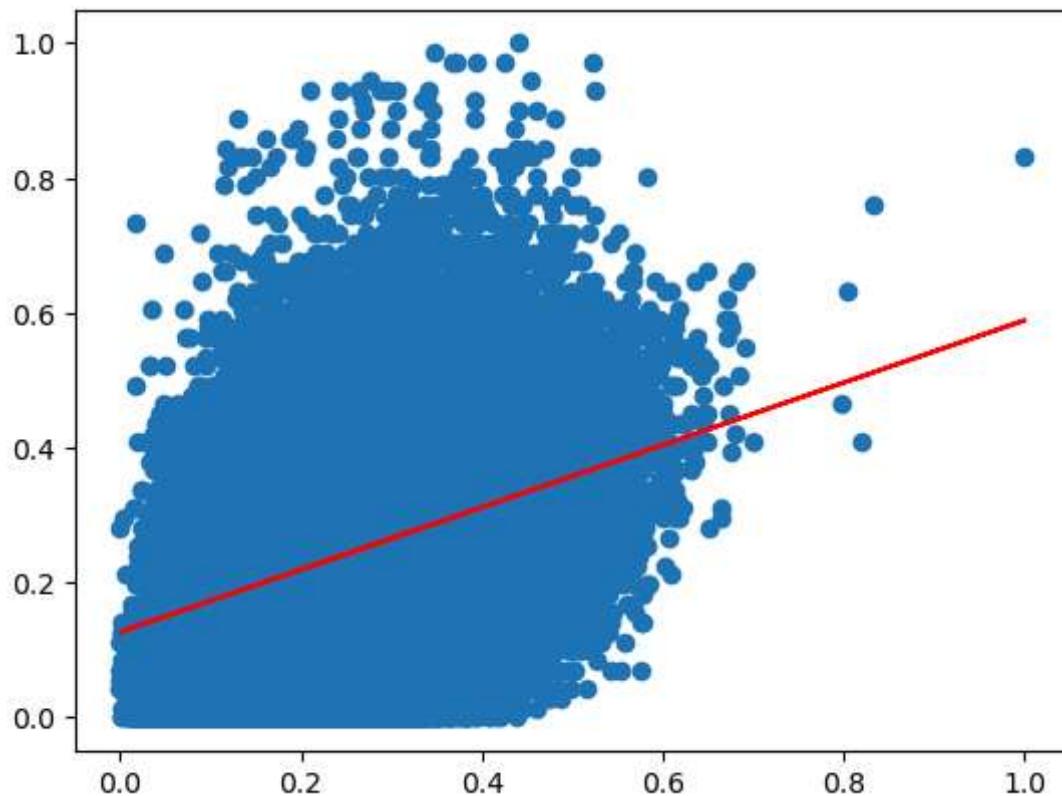
In []:

In [50]: lr.params

Out[50]: const 0.049405
distance 0.575767
dtype: float64

In []:

In [51]: *# Let's visualise the data with a scatter plot and the fitted regression line*
plt.scatter(X_train_lm.iloc[:, 1], y_train)
plt.plot(X_train_lm.iloc[:, 1], 0.127 + 0.462*X_train_lm.iloc[:, 1], 'r')
plt.show()



In [52]: `print(lr.summary())`

OLS Regression Results

Dep. Variable:	time_taken	R-squared:	0.211
Model:	OLS	Adj. R-squared:	0.211
Method:	Least Squares	F-statistic:	3.297e+04
Date:	Wed, 28 May 2025	Prob (F-statistic):	0.00
Time:	08:18:57	Log-Likelihood:	89696.
No. Observations:	123043	AIC:	-1.794e+05
Df Residuals:	123041	BIC:	-1.794e+05
Df Model:	1		
Covariance Type:	nonrobust		
=====			
	coef	std err	t
=====			
const	0.0494	0.001	55.318
distance	0.5758	0.003	181.578
=====			
Omnibus:	15005.725	Durbin-Watson:	2.012
Prob(Omnibus):	0.000	Jarque-Bera (JB):	23781.191
Skew:	0.868	Prob(JB):	0.00
Kurtosis:	4.274	Cond. No.	10.2
=====			

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

```
In [53]: # Assign all the feature variables to X
X_train_lm = X_train[['distance', 'total_outstanding_orders']]
```

```
In [54]: # Build a Linear model

import statsmodels.api as sm
X_train_lm = sm.add_constant(X_train_lm)

lr = sm.OLS(y_train, X_train_lm).fit()

lr.params
```

```
Out[54]: const           -0.004939
          distance        0.559368
          total_outstanding_orders 0.263610
          dtype: float64
```

```
In [55]: # Check the summary
print(lr.summary())
```

OLS Regression Results					
Dep. Variable:	time_taken	R-squared:	0.345		
Model:	OLS	Adj. R-squared:	0.345		
Method:	Least Squares	F-statistic:	3.238e+04		
Date:	Wed, 28 May 2025	Prob (F-statistic):	0.00		
Time:	08:19:54	Log-Likelihood:	1.0111e+05		
No. Observations:	123043	AIC:	-2.022e+05		
Df Residuals:	123040	BIC:	-2.022e+05		
Df Model:	2				
Covariance Type:	nonrobust				
	coef	std err	t	P> t	[0.025
0.975]					

const	-0.0049	0.001	-5.591	0.000	-0.007
-0.003					
distance	0.5594	0.003	193.426	0.000	0.554
0.565					
total_outstanding_orders	0.2636	0.002	158.359	0.000	0.260
0.267					

Omnibus:	10843.309	Durbin-Watson:	2.008		
Prob(Omnibus):	0.000	Jarque-Bera (JB):	16325.192		
Skew:	0.688	Prob(JB):	0.00		
Kurtosis:	4.137	Cond. No.	10.4		

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

```
In [ ]:
```

```
In [56]: # Assign all the feature variables to X  
X_train_lm = X_train[['distance', 'total_outstanding_orders', 'total_items', 'subtotal']]
```

```
In [57]: # Build a Linear model  
  
import statsmodels.api as sm  
X_train_lm = sm.add_constant(X_train_lm)  
  
lr = sm.OLS(y_train, X_train_lm).fit()  
  
lr.params
```

```
Out[57]: const           -0.063621  
distance          0.545611  
total_outstanding_orders 0.231538  
total_items        -0.588802  
subtotal           0.595075  
num_distinct_items 0.143784  
dtype: float64
```

```
In [ ]:
```

```
In [58]: # Print the summary of the model  
  
print(lr.summary())
```

OLS Regression Results

Dep. Variable:	time_taken	R-squared:	0.473		
Model:	OLS	Adj. R-squared:	0.473		
Method:	Least Squares	F-statistic:	2.211e+04		
Date:	Wed, 28 May 2025	Prob (F-statistic):	0.00		
Time:	08:20:21	Log-Likelihood:	1.1453e+05		
No. Observations:	123043	AIC:	-2.290e+05		
Df Residuals:	123037	BIC:	-2.290e+05		
Df Model:	5				
Covariance Type:	nonrobust				
	coef	std err	t	P> t	
0.975]				[0.025	
-----	-----	-----	-----	-----	
const	-0.0636	0.001	-73.726	0.000	-0.065
-0.062					
distance	0.5456	0.003	210.276	0.000	0.541
0.551					
total_outstanding_orders	0.2315	0.002	153.721	0.000	0.229
0.234					
total_items	-0.5888	0.061	-9.727	0.000	-0.707
-0.470					
subtotal	0.5951	0.005	108.290	0.000	0.584
0.606					
num_distinct_items	0.1438	0.005	26.423	0.000	0.133
0.154					
=====	=====	=====	=====	=====	
Omnibus:	11022.401	Durbin-Watson:	2.001		
Prob(Omnibus):	0.000	Jarque-Bera (JB):	19751.134		
Skew:	0.634	Prob(JB):	0.00		
Kurtosis:	4.498	Cond. No.	238.		
=====	=====	=====	=====	=====	

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In []:

```
In [59]: # Assign all the feature variables to X
X_train_lm = X_train[ ['market_id', 'store_primary_category', 'order_protocol', 'to
```

```
In [60]: # Build a Linear model
```

```
import statsmodels.api as sm
X_train_lm = sm.add_constant(X_train_lm)

lr = sm.OLS(y_train, X_train_lm).fit()

lr.params
```

```
Out[60]: const          0.001893
         market_id      -0.045745
         store_primary_category 0.004910
         order_protocol     -0.066977
         total_items        -0.248659
         subtotal           0.482451
         num_distinct_items 0.187595
         min_item_price     0.048234
         max_item_price     0.206059
         total_onshift_dashers -1.030789
         total_outstanding_orders 1.296662
         distance            0.559631
         dtype: float64
```

```
In [61]: # Print the summary of the model
print(lr.summary())
```

OLS Regression Results

Dep. Variable:	time_taken	R-squared:	0.812		
Model:	OLS	Adj. R-squared:	0.811		
Method:	Least Squares	F-statistic:	4.815e+04		
Date:	Wed, 28 May 2025	Prob (F-statistic):	0.00		
Time:	08:21:23	Log-Likelihood:	1.7775e+05		
No. Observations:	123043	AIC:	-3.555e+05		
Df Residuals:	123031	BIC:	-3.554e+05		
Df Model:	11				
Covariance Type:	nonrobust				
	coef	std err	t		
0.975]			P> t	[0.025	
-----	-----	-----	-----	-----	
const	0.0019	0.001	2.571	0.010	0.000
0.003					
market_id	-0.0457	0.001	-74.389	0.000	-0.047
-0.045					
store_primary_category	0.0049	0.001	8.613	0.000	0.004
0.006					
order_protocol	-0.0670	0.001	-101.503	0.000	-0.068
-0.066					
total_items	-0.2487	0.037	-6.661	0.000	-0.322
-0.175					
subtotal	0.4825	0.005	107.191	0.000	0.474
0.491					
num_distinct_items	0.1876	0.004	49.450	0.000	0.180
0.195					
min_item_price	0.0482	0.007	7.127	0.000	0.035
0.061					
max_item_price	0.2061	0.006	31.955	0.000	0.193
0.219					
total_onshift_dashers	-1.0308	0.002	-441.856	0.000	-1.035
-1.026					
total_outstanding_orders	1.2967	0.003	511.403	0.000	1.292
1.302					
distance	0.5596	0.002	360.369	0.000	0.557
0.563					
=====	=====	=====	=====	=====	
Omnibus:	22808.828	Durbin-Watson:	1.994		
Prob(Omnibus):	0.000	Jarque-Bera (JB):	59497.669		
Skew:	1.015	Prob(JB):	0.00		
Kurtosis:	5.735	Cond. No.	302.		
=====	=====	=====	=====	=====	

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In []:

In [62]: # Check for the VIF values of the feature variables.

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [68]:

```
# Create a dataframe that will contain the names of all the feature variables and their VIF values
vif = pd.DataFrame()
vif['Features'] = X_train_lm.columns
vif['VIF'] = [variance_inflation_factor(X_train_lm.values, i) for i in range(X_train_lm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

Out[68]:

	Features	VIF
0	const	20.33
9	total_busy_dashers	7.79
10	total_outstanding_orders	7.76
6	num_distinct_items	3.98
5	subtotal	3.57
4	total_items	2.36
8	max_item_price	2.30
7	min_item_price	2.16
3	order_protocol	1.05
2	store_primary_category	1.02
1	market_id	1.01
11	distance	1.00

In [71]:

```
# Dropping highly correlated variables and insignificant variables
```

```
X = X_train.drop('total_busy_dashers', axis=1)
```

In [72]:

```
# Build a second fitted model
X_train_lm = sm.add_constant(X)

lr_3 = sm.OLS(y_train, X_train_lm).fit()
```

In [73]:

```
# Print the summary of the model
```

```
print(lr_3.summary())
```

OLS Regression Results

Dep. Variable:	time_taken	R-squared:	0.812		
Model:	OLS	Adj. R-squared:	0.811		
Method:	Least Squares	F-statistic:	4.815e+04		
Date:	Wed, 28 May 2025	Prob (F-statistic):	0.00		
Time:	08:25:54	Log-Likelihood:	1.7775e+05		
No. Observations:	123043	AIC:	-3.555e+05		
Df Residuals:	123031	BIC:	-3.554e+05		
Df Model:	11				
Covariance Type:	nonrobust				
<hr/>					
<hr/>					
	coef	std err	t	P> t	[0.025
0.975]					
<hr/>					
const	0.0019	0.001	2.571	0.010	0.000
0.003					
market_id	-0.0457	0.001	-74.389	0.000	-0.047
-0.045					
store_primary_category	0.0049	0.001	8.613	0.000	0.004
0.006					
order_protocol	-0.0670	0.001	-101.503	0.000	-0.068
-0.066					
total_items	-0.2487	0.037	-6.661	0.000	-0.322
-0.175					
subtotal	0.4825	0.005	107.191	0.000	0.474
0.491					
num_distinct_items	0.1876	0.004	49.450	0.000	0.180
0.195					
min_item_price	0.0482	0.007	7.127	0.000	0.035
0.061					
max_item_price	0.2061	0.006	31.955	0.000	0.193
0.219					
total_onshift_dashers	-1.0308	0.002	-441.856	0.000	-1.035
-1.026					
total_outstanding_orders	1.2967	0.003	511.403	0.000	1.292
1.302					
distance	0.5596	0.002	360.369	0.000	0.557
0.563					
<hr/>					
Omnibus:	22808.828	Durbin-Watson:	1.994		
Prob(Omnibus):	0.000	Jarque-Bera (JB):	59497.669		
Skew:	1.015	Prob(JB):	0.00		
Kurtosis:	5.735	Cond. No.	302.		
<hr/>					

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [74]: # Calculate the VIFs again for the new model

```
vif = pd.DataFrame()
vif['Features'] = X_train_lm.columns
vif['VIF'] = [variance_inflation_factor(X_train_lm.values, i) for i in range(X_train_lm.shape[1])]
```

```
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

Out[74]:

	Features	VIF
0	const	20.47
9	total_onshift_dashers	8.12
10	total_outstanding_orders	8.07
6	num_distinct_items	3.98
5	subtotal	3.57
4	total_items	2.36
8	max_item_price	2.30
7	min_item_price	2.16
3	order_protocol	1.05
2	store_primary_category	1.02
1	market_id	1.01
11	distance	1.00

In []:

```
X = X_train.drop(['total_items'],axis=1)
```

In [123...]

```
# Build a second fitted model
X_train_lm = sm.add_constant(X)

lr_3 = sm.OLS(y_train, X_train_lm).fit()
```

In [125...]

```
# Print the summary of the model

print(lr_3.summary())
```

OLS Regression Results

Dep. Variable:	time_taken	R-squared:	0.834		
Model:	OLS	Adj. R-squared:	0.834		
Method:	Least Squares	F-statistic:	5.620e+04		
Date:	Wed, 28 May 2025	Prob (F-statistic):	0.00		
Time:	08:52:15	Log-Likelihood:	1.8557e+05		
No. Observations:	123043	AIC:	-3.711e+05		
Df Residuals:	123031	BIC:	-3.710e+05		
Df Model:	11				
Covariance Type:	nonrobust				
	coef	std err	t		
0.975]			[0.025		
-----	-----	-----	-----		
const	0.0015	0.001	2.128	0.033	0.000
0.003					
market_id	-0.0468	0.001	-81.124	0.000	-0.048
-0.046					
store_primary_category	0.0054	0.001	10.089	0.000	0.004
0.006					
order_protocol	-0.0635	0.001	-102.564	0.000	-0.065
-0.062					
subtotal	0.4784	0.004	116.090	0.000	0.470
0.486					
num_distinct_items	0.1723	0.003	53.402	0.000	0.166
0.179					
min_item_price	0.0502	0.006	7.912	0.000	0.038
0.063					
max_item_price	0.2208	0.006	37.187	0.000	0.209
0.232					
total_onshift_dashers	-0.8421	0.003	-320.203	0.000	-0.847
-0.837					
total_busy_dashers	-0.0021	1.59e-05	-129.354	0.000	-0.002
-0.002					
total_outstanding_orders	1.4425	0.003	547.851	0.000	1.437
1.448					
distance	0.5613	0.001	385.174	0.000	0.558
0.564					
-----	-----	-----	-----	-----	-----
Omnibus:	23126.342	Durbin-Watson:	1.997		
Prob(Omnibus):	0.000	Jarque-Bera (JB):	59158.221		
Skew:	1.036	Prob(JB):	0.00		
Kurtosis:	5.693	Cond. No.	2.58e+03		
-----	-----	-----	-----	-----	-----

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

[2] The condition number is large, 2.58e+03. This might indicate that there are strong multicollinearity or other numerical problems.

In [118]: # Calculate the VIFs again for the new model
vif = pd.DataFrame()

```
vif['Features'] = X_train_lm.columns
vif['VIF'] = [variance_inflation_factor(X_train_lm.values, i) for i in range(X_train_lm.shape[1])]
vif['VIF'] = round(vif['VIF'], 2)
vif = vif.sort_values(by = "VIF", ascending = False)
vif
```

Out[118...]

	Features	VIF
0	const	20.19
8	total_onshift_dashers	11.71
9	total_busy_dashers	11.25
10	total_outstanding_orders	9.89
4	subtotal	3.40
5	num_distinct_items	3.27
7	max_item_price	2.22
6	min_item_price	2.15
3	order_protocol	1.05
2	store_primary_category	1.02
1	market_id	1.01
11	distance	1.00

In []:

In [75]:

```
# Import LabelEncoder from sklearn
from sklearn.preprocessing import LabelEncoder

def label_encoding(df):
    categorical_columns = df.select_dtypes(include='object').columns
    label_encoder = LabelEncoder()
    df[categorical_columns] = df[categorical_columns].apply(lambda col: label_encode

label_encoding(Delivery_time_Data)
Delivery_time_Data.head()
```

	market_id	store_primary_category	order_protocol	total_items	subtotal	num_distinct_it
0	1.0		4	1.0	4	3441
1	2.0		46	2.0	1	1900
2	2.0		36	3.0	4	4771
3	1.0		38	1.0	1	1525
4	1.0		38	1.0	2	3620



In []:

```
In [76]: df_train, df_test = train_test_split(Delivery_time_Data, train_size=0.7, random_state=42)
print(df_train.shape)
print(df_test.shape)

(123043, 13)
(52734, 13)
```

In []:

```
In [77]: scaler= MinMaxScaler()
num_vars =['market_id','store_primary_category','order_protocol','total_items','sub
```

```
In [78]: df_test.info()
```

<class 'pandas.core.frame.DataFrame'>
Index: 52734 entries, 139667 to 3735
Data columns (total 13 columns):

#	Column	Non-Null Count	Dtype
0	market_id	52734	non-null float64
1	store_primary_category	52734	non-null int64
2	order_protocol	52734	non-null float64
3	total_items	52734	non-null int64
4	subtotal	52734	non-null int64
5	num_distinct_items	52734	non-null int64
6	min_item_price	52734	non-null int64
7	max_item_price	52734	non-null int64
8	total_onshift_dashers	52734	non-null float64
9	total_busy_dashers	52734	non-null float64
10	total_outstanding_orders	52734	non-null float64
11	distance	52734	non-null float64
12	time_taken	52734	non-null float64

dtypes: float64(7), int64(6)
memory usage: 5.6 MB

In []:

```
In [79]: df_train[num_vars]=scaler.fit_transform(df_train[num_vars])
```

```
In [80]: df_train.head()
```

Out[80]:

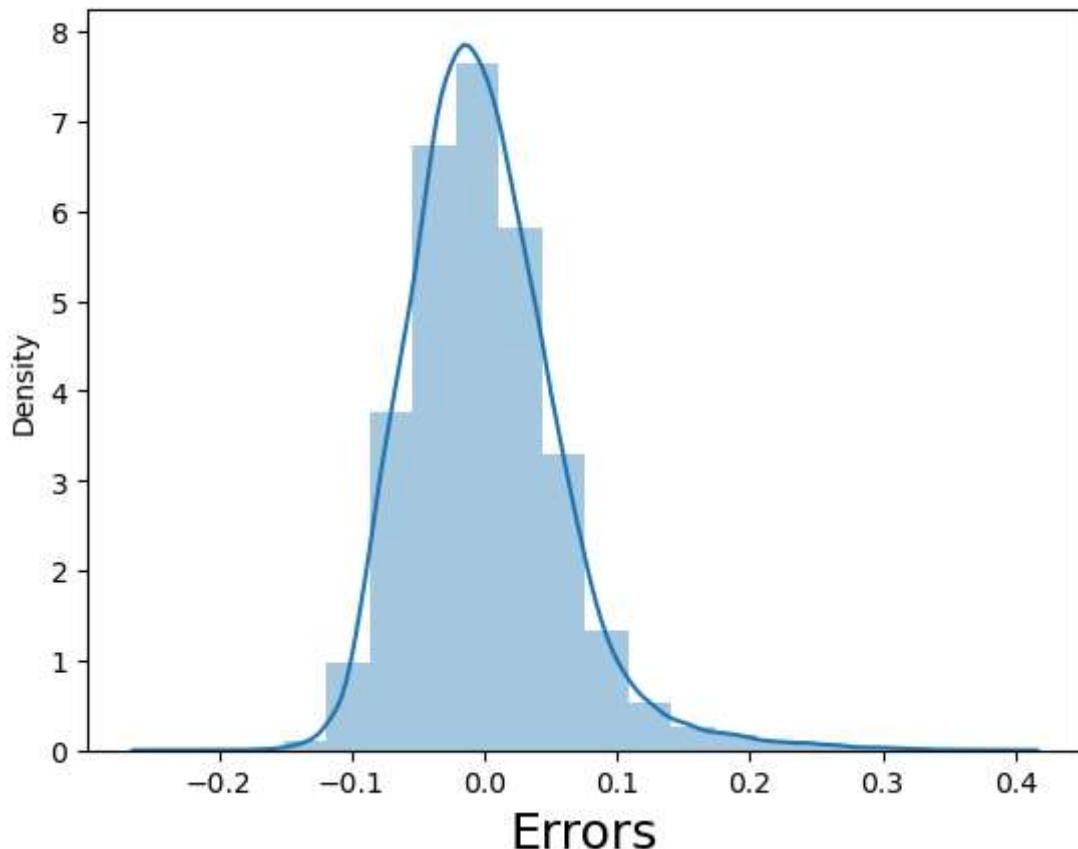
	market_id	store_primary_category	order_protocol	total_items	subtotal	num_dist
94746	0.6	0.333333	0.666667	0.002439	0.066791	
173338	0.6	1.000000	0.666667	0.000000	0.031530	
37592	0.6	0.763889	0.666667	0.000000	0.070896	
42763	0.2	0.388889	0.500000	0.012195	0.017276	
27506	0.2	1.000000	0.000000	0.004878	0.130597	



In []:

In [87]: `y_train_time_taken = lr_3.predict(X_train_lm)`In [88]: `# Plot the histogram of the error terms
fig = plt.figure()
sns.distplot((y_train - y_train_time_taken), bins = 20)
fig.suptitle('Error Terms', fontsize = 20) # Plot heading
plt.xlabel('Errors', fontsize = 18) # X-Label`Out[88]: `Text(0.5, 0, 'Errors')`

Error Terms



In []:

```
In [91]: num_vars =['market_id','store_primary_category','order_protocol','total_items','sub
df_test[num_vars] = scaler.transform(df_test[num_vars])
```

In [92]: df_test.describe()

Out[92]:

	market_id	store_primary_category	order_protocol	total_items	subtotal
count	52734.000000	52734.000000	52734.000000	52734.000000	52734.000000
mean	0.348504	0.496728	0.318226	0.005372	0.100573
std	0.267545	0.287561	0.252375	0.006099	0.068075
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.200000	0.236111	0.000000	0.002439	0.052239
50%	0.200000	0.527778	0.333333	0.004878	0.083022
75%	0.600000	0.763889	0.500000	0.007317	0.127425
max	1.000000	1.000000	1.000000	0.141463	0.839552



In [93]: y_test = df_test.pop('time_taken')
X_test = df_test

In [94]: # Adding constant variable to test dataframe
X_test_m4 = sm.add_constant(X_test)

In [98]: # Creating X_test_m4 dataframe by dropping variables from X_test_m4
X_test_m4 = X_test_m4.drop('total_busy_dashers', axis = 1)

In []:

In []:

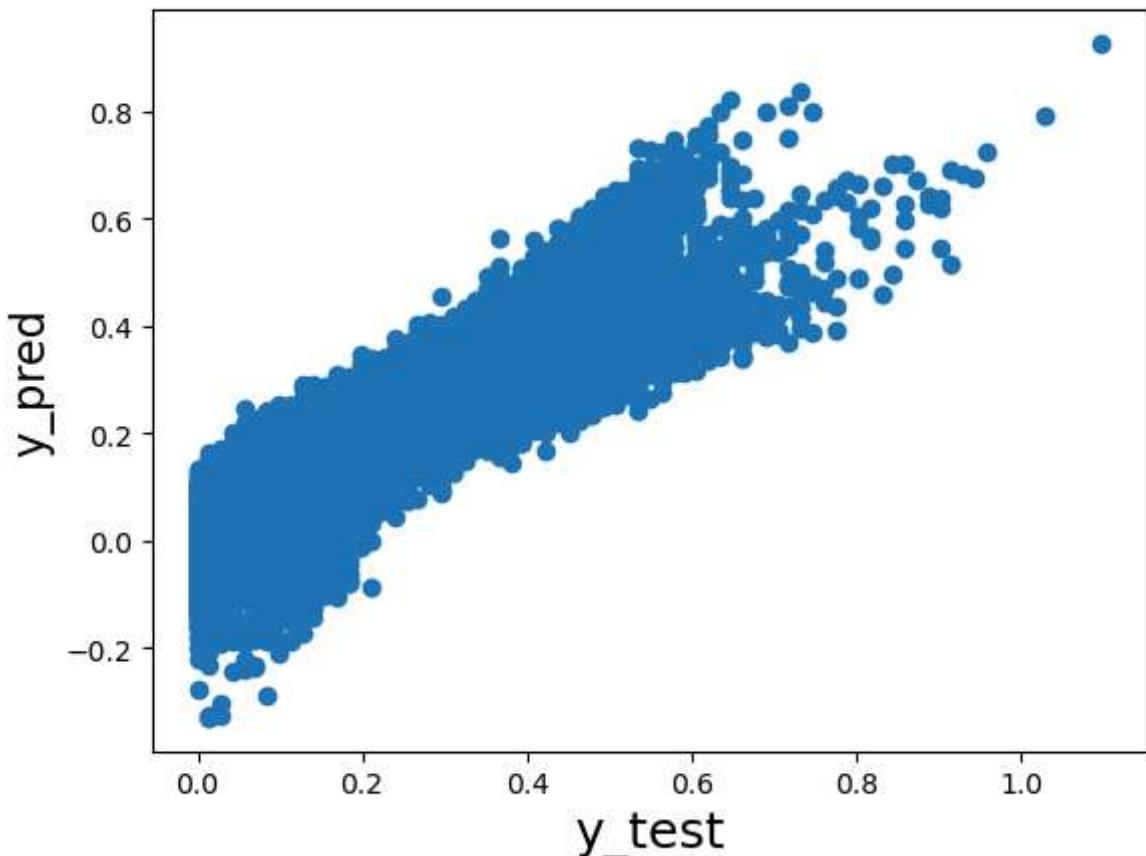
In [99]: # Making predictions using the fourth model
y_pred_m4 = lr_3.predict(X_test_m4)

In [100...]: # Plotting y_test and y_pred to understand the spread

```
fig = plt.figure()
plt.scatter(y_test, y_pred_m4)
fig.suptitle('y_test vs y_pred', fontsize = 20) # Plot heading
plt.xlabel('y_test', fontsize = 18) # X-Label
plt.ylabel('y_pred', fontsize = 16)
```

Out[100...]: Text(0, 0.5, 'y_pred')

y_test vs y_pred



```
In [101]: r2_score(y_train,y_train_pred)
```

```
NameError                                                 Traceback (most recent call last)
Cell In[101], line 1
----> 1 r2_score(y_train,y_train_pred)

NameError: name 'y_train_pred' is not defined
```

```
In [ ]:
```