#Delivery Analysis

##Business Understanding

In a world where anything short of one-day deliveries or same-day service calls seems unacceptable, logistics businesses of all sizes are flocking to route optimization solutions to increase the efficiency of their operations. Route optimization helps logistics transportation companies increase their top and bottom line by increasing operational efficiencies, minimizing expenses, and better serving customers, so it's well worth incorporating into the business. Delivery routing analysis is needed to find out the most important aspects for optimization of existing operational routes.

The purpose of this analysis is to find out the most important branch and what data most affects the time of delivery of goods.

##Analytic Approach

After reading the data, I tried to make an analysis of the delivery time for each branch. The data obtained at this time is in the form of branch origin, branch destination, and location of goods receipt. Based on this, this analysis is carried out by assuming that each task shows the following flow of delivery: branch origin -> branch destination -> task location done

Assumptions used:

1. The time for creating a task and completing a task is the time required for delivery from the destination branch to the receiver's location
2. The location of the branch destination is not known with certainty, so the latitude and longitude determination is only based on the available city/district after the 3-letter city code has been changed. The value given depends on the data obtained from the nominatim website

For this project I will be using the libraries for data manipulation (Pandas, Numpy), data visualization (Matplotlib, Seaborn), machine learning (Scikit-learn, XGBoost) and some statistics to get some insight and the trend of the data. For the data visualization, I will fetch all data to BigQuery and visualize it on Google Data Studio.

##Data Understanding

```
#Install the library
!pip install pyvis
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pyvis
  Downloading pyvis-0.3.2-py3-none-any.whl (756 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 756.0/756.0 kB 11.4 MB/s eta 0:00:00
ent already satisfied: ipython>=5.3.0 in /usr/local/lib/python3.10/dist-packages (from pyvis) (7.34.0)
```

```
Requirement already satisfied: jinja2>=2.9.6 in
/usr/local/lib/python3.10/dist-packages (from pyvis) (3.1.2)
Requirement already satisfied: jsonpickle>=1.4.1 in
/usr/local/lib/python3.10/dist-packages (from pyvis) (3.0.1)
Requirement already satisfied: networkx>=1.11 in
/usr/local/lib/python3.10/dist-packages (from pyvis) (3.1)
Requirement already satisfied: setuptools>=18.5 in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->pyvis)
(67.7.2)
Collecting jedi>=0.16 (from ipython>=5.3.0->pyvis)
  Downloading jedi-0.18.2-py2.py3-none-any.whl (1.6 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.6/1.6 MB 51.9 MB/s eta
0:00:00
ent already satisfied: decorator in /usr/local/lib/python3.10/dist-
packages (from ipython>=5.3.0->pyvis) (4.4.2)
Requirement already satisfied: pickleshare in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->pyvis)
(0.7.5)
Requirement already satisfied: traitlets>=4.2 in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->pyvis)
(5.7.1)
Requirement already satisfied: prompt-toolkit!=3.0.0,!
=3.0.1,<3.1.0,>=2.0.0 in /usr/local/lib/python3.10/dist-packages (from
ipython>=5.3.0->pyvis) (3.0.38)
Requirement already satisfied: pygments in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->pyvis)
(2.14.0)
Requirement already satisfied: backcall in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->pyvis)
(0.2.0)
Requirement already satisfied: matplotlib-inline in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->pyvis)
(0.1.6)
Requirement already satisfied: pexpect>4.3 in
/usr/local/lib/python3.10/dist-packages (from ipython>=5.3.0->pyvis)
(4.8.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.10/dist-packages (from jinja2>=2.9.6->pyvis)
(2.1.2)
Requirement already satisfied: parso<0.9.0,>=0.8.0 in
/usr/local/lib/python3.10/dist-packages (from jedi>=0.16-
>ipython>=5.3.0->pyvis) (0.8.3)
Requirement already satisfied: ptyprocess>=0.5 in
/usr/local/lib/python3.10/dist-packages (from pexpect>4.3-
>ipython>=5.3.0->pyvis) (0.7.0)
Requirement already satisfied: wcwidth in
/usr/local/lib/python3.10/dist-packages (from prompt-toolkit!=3.0.0,!
=3.0.1,<3.1.0,>=2.0.0->ipython>=5.3.0->pyvis) (0.2.6)
Installing collected packages: jedi, pyvis
Successfully installed jedi-0.18.2 pyvis-0.3.2
```

```python
#Import the libraries
import os
import numpy as np
import pandas as pd
import seaborn as sns
import xgboost
import matplotlib.pyplot as plt

from urllib.request import urlopen
import json
import requests
import urllib.parse
import geopy
import geopy.distance
import pyvis.network as net
import networkx as nx
from IPython.display import display, HTML
from geopy.geocoders import Nominatim

from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.pipeline import Pipeline
from xgboost import XGBRegressor
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import MinMaxScaler
from sklearn import metrics
from lightgbm import LGBMRegressor
from sklearn import tree, model_selection
from sklearn.metrics import mean_squared_error, r2_score,
mean_absolute_error

from google.cloud import bigquery
from google.oauth2 import service_account
import pandas_gbq

# store the URL in url to read the data
url = "https://raw.githubusercontent.com/indrasetiadhip/data-task-
sample/main/data-sample.json"

# store the response of URL
response = urlopen(url)

# storing the JSON response
data_json = json.loads(response.read())

#Store the JSON into dataframe format
df = pd.json_normalize(data_json, errors='ignore')
df.head(10)
```

```
             taskCreatedTime  taskAssignedTo           
taskCompletedTime  \
0  2022-11-01 20:17:26 +0700   pacifiedLion0  2022-11-01 20:46:30 
+0700   
1  2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 12:33:48 
+0700   
2  2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 13:41:57 
+0700   
3  2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 18:18:19 
+0700   
4  2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 10:51:49 
+0700   
5  2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 19:34:44 
+0700   
6  2022-11-01 12:00:28 +0700   pacifiedLion0  2022-11-01 20:46:03 
+0700   
7  2022-11-01 14:23:20 +0700   pacifiedLion0  2022-11-01 15:45:13 
+0700   
8  2022-11-01 09:13:16 +0700  giddyCockatoo1  2022-11-01 15:39:01 
+0700   
9  2022-11-01 09:13:16 +0700  giddyCockatoo1  2022-11-01 15:36:44 
+0700   

  taskStatus      flow                taskId  taskLocationDone.lon  \
0       done  Delivery  4fe3b237c832ca4841a2            109.762910   
1       done  Delivery  08a4da25256affae8446            110.033986   
2       done  Delivery  2ff0dc469826158b7684            109.999733   
3       done  Delivery  331c172c2b383f774328            110.003708   
4       done  Delivery  a9d53fa96c80baee8b23            110.013887   
5       done  Delivery  67ec7d34b4f3adbf2895            110.023131   
6       done  Delivery  2079aa99bda230940785            109.762910   
7       done  Delivery  b3975d6adb8e802c749b            109.729141   
8       done  Delivery  ea26e88eaf27edd7885b            109.780323   
9       done  Delivery  f53a4daf67534816dbd9            109.780821   

   taskLocationDone.lat  cod.amount cod.received 
UserVar.branch_dest  \
0             -6.926608    685000.0         True                 SRG 
  
1             -7.876154     53500.0         True                 MGL 
  
2             -7.849777    179500.0         True                 MGL 
  
3             -7.710998     31815.0         True                 MGL 
  
4             -7.829742    144562.0         True                 MGL 
  
5             -7.706646    206610.0         True                 MGL 
```

```
6              -6.926608     38200.0        True                 SRG

7              -6.911588     33000.0        True                 SRG

8              -7.663731     65867.0        True                 MGL

9              -7.663288     26800.0        True                 MGL


  UserVar.taskStatusLabel UserVar.receiver_city
UserVar.taskDetailStatusLabel  \
0                 Success    BATANG ,KAB BATANG             YANG
BERSANGKUTAN
1                 Success   PURWODADI,PURWOREJO             YANG
BERSANGKUTAN
2                 Success   PURWODADI,PURWOREJO             YANG
BERSANGKUTAN
3                 Success   PURWODADI,PURWOREJO             YANG
BERSANGKUTAN
4                 Success     BAGELEN,PURWOREJO             YANG
BERSANGKUTAN
5                 Success   PURWODADI,PURWOREJO             YANG
BERSANGKUTAN
6                 Success       KANDEMAN,BATANG             YANG
BERSANGKUTAN
7                 Success    BATANG ,KAB BATANG             YANG
BERSANGKUTAN
8                 Success       BUTUH,PURWOREJO             YANG
BERSANGKUTAN
9                 Success       BUTUH,PURWOREJO             YANG
BERSANGKUTAN

  UserVar.taskDetailStatus UserVar.weight UserVar.branch_origin  \
0                      D01             13                   CGK
1                      D01            1.3                   CGK
2                      D01              3                   CGK
3                      D01          0.625                   CGK
4                      D01              3                   CGK
5                      D01            2.5                   CGK
6                      D01            0.7                   CGK
7                      D01           0.04                   CGK
8                      D01            0.8                   CGK
9                      D01            0.1                   CGK

  UserVar.taskStatus
0             COLF01
1             COLF01
2             COLF01
3             COLF01
```

```
4             COLF01
5             COLF01
6             COLF01
7             COLF01
8             COLF01
9             COLF01
```

```
df.shape
```

```
(8334, 18)
```

```
#Check the number of unique values
df.nunique()
```

```
taskCreatedTime                  4447
taskAssignedTo                   2787
taskCompletedTime                4051
taskStatus                          2
flow                                1
taskId                           8334
taskLocationDone.lon             3664
taskLocationDone.lat             3675
cod.amount                       1585
cod.received                        2
UserVar.branch_dest                62
UserVar.taskStatusLabel             2
UserVar.receiver_city            1830
UserVar.taskDetailStatusLabel      31
UserVar.taskDetailStatus           31
UserVar.weight                    686
UserVar.branch_origin              59
UserVar.taskStatus                  2
dtype: int64
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8334 entries, 0 to 8333
Data columns (total 18 columns):
 #   Column                         Non-Null Count  Dtype  
---  ------                         --------------  -----  
 0   taskCreatedTime                8334 non-null   object 
 1   taskAssignedTo                 8333 non-null   object 
 2   taskCompletedTime              7566 non-null   object 
 3   taskStatus                     8334 non-null   object 
 4   flow                           8334 non-null   object 
 5   taskId                         8334 non-null   object 
 6   taskLocationDone.lon           7566 non-null   float64
 7   taskLocationDone.lat           7566 non-null   float64
 8   cod.amount                     2358 non-null   float64
 9   cod.received                   2358 non-null   object 
```

```
 10  UserVar.branch_dest            8334 non-null   object
 11  UserVar.taskStatusLabel        7572 non-null   object
 12  UserVar.receiver_city          8282 non-null   object
 13  UserVar.taskDetailStatusLabel  7572 non-null   object
 14  UserVar.taskDetailStatus       7572 non-null   object
 15  UserVar.weight                 8334 non-null   object
 16  UserVar.branch_origin          8041 non-null   object
 17  UserVar.taskStatus             7572 non-null   object
dtypes: float64(3), object(15)
memory usage: 1.1+ MB
```

```python
#Check the unique number of each columns
for i in df.select_dtypes(include=['object']).columns:
    print("This is column {0}".format(i))
    print(df[i].value_counts())
    print("-------------------------------------------")
```

```
This is column taskCreatedTime
2022-11-05 07:45:30 +0700    50
2022-11-07 07:13:02 +0700    44
2022-11-07 07:10:40 +0700    42
2022-11-07 07:13:03 +0700    39
2022-11-01 08:16:31 +0700    37
                             ..
2022-11-04 07:18:13 +0700     1
2022-11-04 11:06:03 +0700     1
2022-11-04 11:35:04 +0700     1
2022-11-04 12:01:51 +0700     1
2022-11-10 07:25:40 +0700     1
Name: taskCreatedTime, Length: 4447, dtype: int64
-------------------------------------------
This is column taskAssignedTo
gutturalLion9     103
gloomyLlama0       83
zestyPear3         64
emptyIcecream6     57
artisticHyena7     56
                 ...
emptyAntelope3      1
finickyCoati6       1
thriftyLion5        1
somberHeron8        1
murkyThrushe3       1
Name: taskAssignedTo, Length: 2787, dtype: int64
-------------------------------------------
This is column taskCompletedTime
2022-11-07 07:14:54 +0700    83
2022-11-05 07:48:44 +0700    55
2022-11-05 07:20:19 +0700    49
2022-11-03 07:47:21 +0700    48
2022-11-08 08:41:43 +0800    42
```

```
                              ..
2022-11-03 08:21:07 +0700     1
2022-11-03 09:21:11 +0800     1
2022-11-03 08:21:10 +0700     1
2022-11-03 08:21:09 +0700     1
2022-11-10 09:38:03 +0700     1
Name: taskCompletedTime, Length: 4051, dtype: int64
------------------------------------------
This is column taskStatus
done       7572
ongoing     762
Name: taskStatus, dtype: int64
------------------------------------------
This is column flow
Delivery    8334
Name: flow, dtype: int64
------------------------------------------
This is column taskId
4fe3b237c832ca4841a2    1
7b0956716cff51ec034f    1
d1156143b6e4188e6834    1
a9839eb4ea1b792b89a3    1
0f9047af30ab4e9fd295    1
                       ..
696ef937df1d87ca94f0    1
246f4fa65c2ee858a6c0    1
a2a16244f782d1587adc    1
d92e51e159dd764e619a    1
cdb90c597655282306fd    1
Name: taskId, Length: 8334, dtype: int64
------------------------------------------
This is column cod.received
False    1663
True      695
Name: cod.received, dtype: int64
------------------------------------------
This is column UserVar.branch_dest
PLM    562
CGK    482
SRG    480
BDO    450
KOE    432
      ...
BTJ     26
DPK     23
TNJ     23
TJQ     22
DJB     19
Name: UserVar.branch_dest, Length: 62, dtype: int64
------------------------------------------
```

```
This is column UserVar.taskStatusLabel
Success    5427
Failed     2145
Name: UserVar.taskStatusLabel, dtype: int64
------------------------------------------
This is column UserVar.receiver_city
SEBERANG ULU I, PALE    82
DENPASAR SELATAN,DEN    79
CIDAUN, CIANJUR         75
SUNGAI RAYA,KUBU RAY    68
PONTIANAK KOTA , PON    63
                        ..
SINGKAWANG SELATAN       1
SINJAI TIMUR,SINJAI      1
KADUPANDAK, CIANJUR      1
METRO PUSAT, METRO       1
KOTA BANTUL              1
Name: UserVar.receiver_city, Length: 1830, dtype: int64
------------------------------------------
This is column UserVar.taskDetailStatusLabel
YANG BERSANGKUTAN                                         3109
KELUARGA/SAUDARA                                           774
MISROUTE                                                   763
ATASAN/STAFF/KARYAWAN/BAWAHAN                              634
SECURITY                                                   564
ALAMAT TIDAK LENGKAP service/ TIDAK DIKENAL                322
RUMAH service/ KANTOR KOSONG (MASIH DIHUNI)                304
NEW ADDRESS                                                247
DIAMBIL SENDIRI                                            100
SUAMI/ISTRI/ANAK                                            94
RECEPTIONIST                                                87
TUTUP PADA AKHIR PEKAN service/ HARI LIBUR                  70
PENERIMA TIDAK DIKENAL                                      64
MAILING ROOM                                                62
PEMBANTU                                                    61
DITOLAK OLEH PENERIMA                                       52
PENERIMA MENOLAK BAYAR (KIRIMAN COD)                        48
PENERIMA PINDAH ALAMAT                                      45
FORCE MAJEURE                                               42
MENUNGGU PEMBAYARAN COD                                     27
HOLD FOR FURTHER INSTRUCTI0N                                24
PENJAGA KOS                                                 21
PENERIMA MENOLAK MENERIMA KIRIMAN COD (TDK PESAN)           17
TUTUP/LIBUR CUTI/DINAS LUAR KOTA (KIRIMAN COD)              13
SUPIR                                                       11
OFFICE BOY                                                   6
SEKRETARIS                                                   4
RUMAH service/ KANTOR TIDAK DIHUNI                           3
MENUNGGU KONFIRMASI NILAI COD                                2
CRISS-CROSS                                                  1
```

```
DAMAGE CASE                                             1
Name: UserVar.taskDetailStatusLabel, dtype: int64
------------------------------------------
This is column UserVar.taskDetailStatus
D01    3109
D09     774
U12     763
D10     634
D04     564
U01     322
U05     304
CR6     247
CR3     100
D06      94
D02      87
U09      70
U02      64
D05      62
D07      61
U06      52
U08      48
U03      45
U10      42
U25      27
CR5      24
D08      21
U21      17
U22      13
D11      11
D12       6
D03       4
U07       3
U24       2
U13       1
U11       1
Name: UserVar.taskDetailStatus, dtype: int64
------------------------------------------
This is column UserVar.weight
1        4130
2         305
0.5       202
0.2       159
3         150
         ...
13.75       1
3.54        1
33.39       1
18.42       1
54.8        1
Name: UserVar.weight, Length: 686, dtype: int64
```

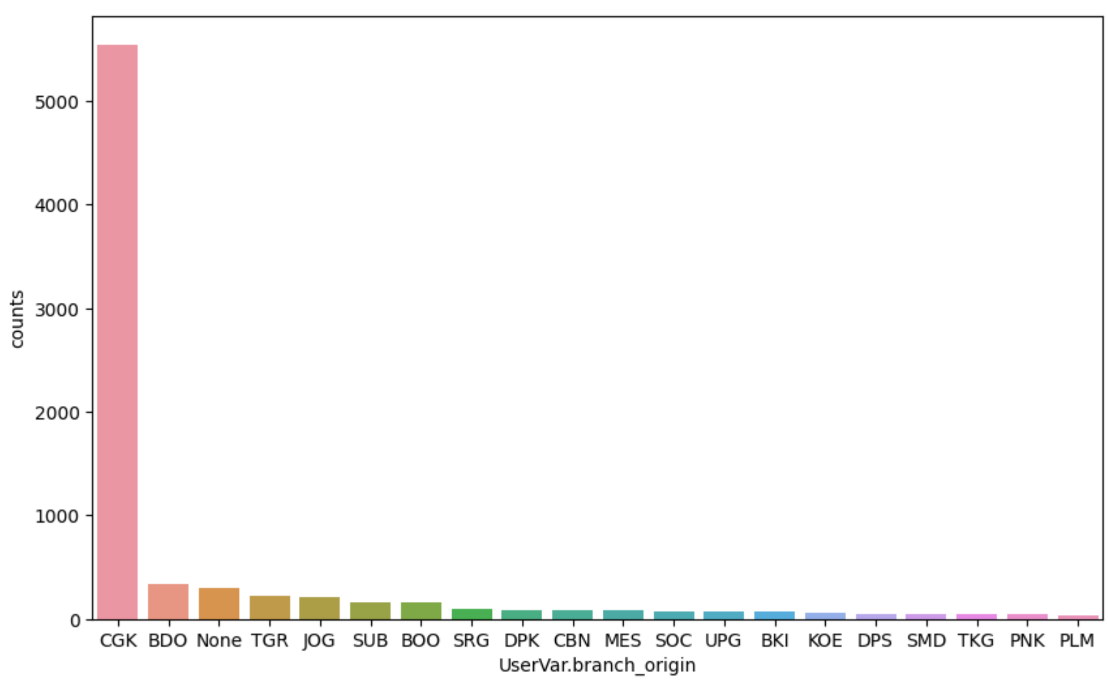
```
------------------------------------------
This is column UserVar.branch_origin
CGK    5550
BDO     341
TGR     226
JOG     206
SUB     164
BOO     158
SRG      95
DPK      89
CBN      85
MES      81
SOC      76
UPG      72
BKI      70
KOE      62
DPS      48
SMD      46
TKG      40
PNK      39
PLM      37
SMI      37
PKU      35
BPN      32
CLG      32
KRW      28
MDN      28
MJK      28
PGK      28
TSM      26
MXG      26
JBR      24
KDR      22
AMI      22
BDJ      22
PDG      19
PBL      18
BTH      13
KDI      12
MGL      12
PSR      11
CXP      10
TRK       8
CKR       6
DJJ       5
DJB       5
PLW       5
SOQ       5
MDC       5
PWT       5
```

```
TGL       5
TTE       4
AMQ       3
BTG       3
TJQ       3
GTO       2
BTJ       2
PKY       2
BKS       1
TNJ       1
DTB       1
Name: UserVar.branch_origin, dtype: int64
------------------------------------------
This is column UserVar.taskStatus
COLF01    5427
COLF02    2145
Name: UserVar.taskStatus, dtype: int64
------------------------------------------
```

##Exploratory Data Analysis

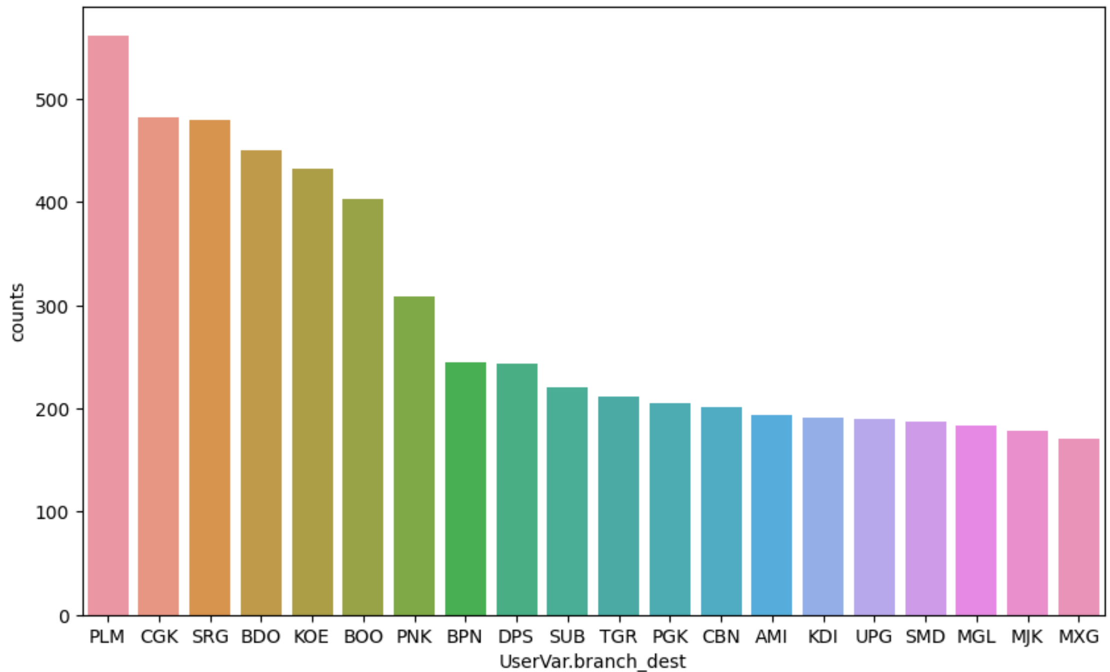
```
#Show the most used branch origin
plt.figure(figsize=(10, 6))
sns.barplot(data=df.groupby(by=['UserVar.branch_origin']).size().reset
_index(name='counts').sort_values(by='counts',ascending=False).head(20
),
              x='UserVar.branch_origin',
              y='counts')
```

```
<Axes: xlabel='UserVar.branch_origin', ylabel='counts'>
```



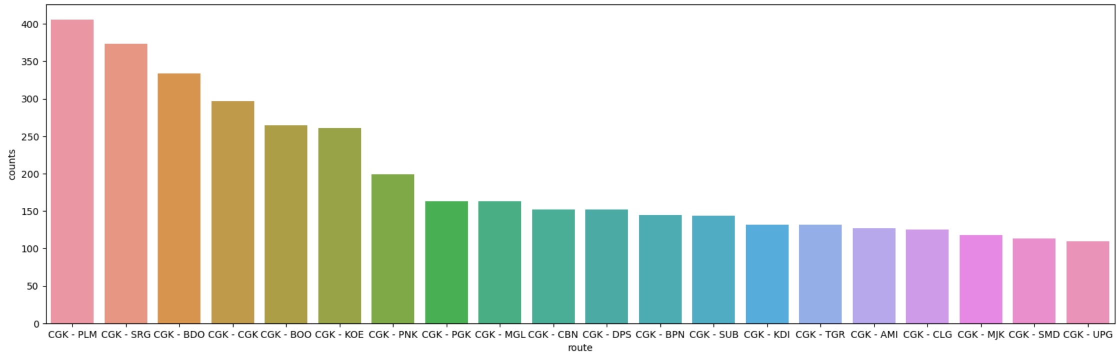
```python
#Show the most used branch destination
plt.figure(figsize=(10, 6))
sns.barplot(data=df.groupby(by=['UserVar.branch_dest']).size().reset_i
ndex(name='counts').sort_values(by='counts',ascending=False).head(20),
              x='UserVar.branch_dest',
              y='counts')
```

```
<Axes: xlabel='UserVar.branch_dest', ylabel='counts'>
```




```python
#Show the most used route for the delivery
df_grouped = df.groupby(by=['UserVar.branch_origin',
'UserVar.branch_dest']).size().reset_index(name='counts').sort_values(
by='counts',ascending=False)
df_grouped["route"] = df_grouped['UserVar.branch_origin'] + ' - ' +
df_grouped['UserVar.branch_dest']
plt.figure(figsize=(20, 6))
sns.barplot(data=df_grouped.head(20),
              x='route',
              y='counts')
```

```
<Axes: xlabel='route', ylabel='counts'>
```




```
#Combine unique value of branch origin and destination to get all
possible branch, the counting is based on the number of branch
destination (not origin)
init_list =list(df_grouped["UserVar.branch_origin"])
+list(df_grouped["UserVar.branch_dest"])
node_area = pd.Series([*set(init_list)], name='UserVar.branch_dest')
df_grouped_dest=df.groupby(by=['UserVar.branch_dest']).size().reset_in
dex(name='counts').sort_values(by='counts',ascending=False)
result_node = pd.merge(node_area, df_grouped_dest, how='left',
on='UserVar.branch_dest')
result_node.fillna(1, inplace=True)
result_node

   UserVar.branch_dest  counts
0                  DPS   244.0
1                  PKY    27.0
2                  PLW    89.0
3                  CKR   139.0
4                  PWT    33.0
..                 ...     ...
58                 KDR    87.0
59                 SRG   480.0
60                 TGL    81.0
61                 JOG    56.0
62                 BDO   450.0

[63 rows x 2 columns]

#Try to visualize the relationship for each branch using Graph Network
#You can filter the graph ifyou want to see the specific branch
source = df_grouped['UserVar.branch_origin']
target = df_grouped['UserVar.branch_dest']
weights = df_grouped['counts']

source_node = result_node['UserVar.branch_dest']
weights_node = result_node['counts']

g_from_data = net.Network(height='1000px',
                          width='1000px',
```

```
                                    # bgcolor='white',
                                    # font_color="black",
                                    directed=True,
                                    notebook=True,
                                    cdn_resources='in_line',
                                    filter_menu=True)


for (name,weight) in zip(source_node,weights_node):
  try:
    g_from_data.add_node(name,label=name,title=name,value=weight)
  except:
    pass

for (i,j,k) in zip(source,target,weights):
    try:
      g_from_data.add_edge(i,j,value=k)
    except:
      pass

g_from_data.show_buttons()
g_from_data.toggle_physics(False)

g_from_data.show('A_Complete_Networkx_Graph_From_DataFrame.html')
display(HTML('A_Complete_Networkx_Graph_From_DataFrame.html'))
```

```
A_Complete_Networkx_Graph_From_DataFrame.html
```

```
<IPython.core.display.HTML object>
```

##Data Cleansing

Before modelling the data, we need to clean and prepare the data first, it can be:

1. Removing or filling missing data
2. Removing unneded columns
3. Removing outliers
4. Standardizing value
5. Fixing error

```
#Check the number of null values
df.isnull().sum()
```

```
taskCreatedTime                  0
taskAssignedTo                   1
taskCompletedTime              768
taskStatus                       0
flow                             0
taskId                           0
taskLocationDone.lon           768
taskLocationDone.lat           768
```

```
cod.amount                          5976
cod.received                        5976
UserVar.branch_dest                    0
UserVar.taskStatusLabel              762
UserVar.receiver_city                 52
UserVar.taskDetailStatusLabel        762
UserVar.taskDetailStatus             762
UserVar.weight                         0
UserVar.branch_origin                293
UserVar.taskStatus                   762
dtype: int64
```

```
#Convert column types
df["UserVar.weight"]=df["UserVar.weight"].astype(np.float64)
df[['taskAssignedTo', 'cod.received', 'UserVar.branch_origin',
'UserVar.branch_dest', 'UserVar.taskDetailStatus',
'UserVar.taskStatus']] = df[['taskAssignedTo', 'cod.received',
'UserVar.branch_origin', 'UserVar.branch_dest',
'UserVar.taskDetailStatus', 'UserVar.taskStatus']].astype(str)
```

##Feature Engineering

```
#Standardize the value for receiver_city column
df['receiver_city_clean'] =
df['UserVar.receiver_city'].str.replace(r'\bKAB\b', '', regex=True)
df['receiver_city_clean'] = df['receiver_city_clean'].str.replace(r'\
bKOTA\b', '', regex=True)
df['receiver_city_clean'] =
df['receiver_city_clean'].str.replace(r'.', ' ', regex=True)
df['receiver_city_clean'] =
df['receiver_city_clean'].str.replace(r',', ' ', regex=True)
df['receiver_city_clean'] = df['receiver_city_clean'].str.strip(' ')
```

```
#Only get the unique value of cleaned receiver city column
df_city_unique = df['receiver_city_clean'].drop_duplicates()
df_city_unique.dropna(inplace=True)
df_city_unique
```

```
0            BATANG   BATANG
1        PURWODADI PURWOREJO
4          BAGELEN PURWOREJO
6            KANDEMAN BATANG
8            BUTUH PURWOREJO
                ...
8309          CISARUA  BOGOR
8321    UJUNGBERUNG  BANDUNG
8323            MEDAN  MEDAN
8325      DENDANG MUARASABAK
8330                  BANTUL
Name: receiver_city_clean, Length: 1815, dtype: object
```

```python
#Try the url to get the address based on latitude and longitude


# Latitude & Longitude input
lat = "-7.710998"
long = "110.003708"

url = "https://nominatim.openstreetmap.org/reverse.php?
lat={0}&lon={1}&zoom=18&format=jsonv2".format(lat, long)


response = requests.get(url).json()

# Display
print(response)
```

{'place_id': 366434330, 'licence': 'Data © OpenStreetMap contributors, ODbL 1.0. https://osm.org/copyright', 'osm_type': 'way', 'osm_id': 210736359, 'lat': '-7.710843649025564', 'lon': '110.00381692998457', 'place_rank': 26, 'category': 'highway', 'type': 'tertiary', 'importance': 0.10000999999999993, 'addresstype': 'road', 'name': 'Jalan Raden Ajeng Kartini', 'display_name': 'Jalan Raden Ajeng Kartini, Purworejo, Jawa Tengah, Jawa, 54113, Indonesia', 'address': {'road': 'Jalan Raden Ajeng Kartini', 'city': 'Purworejo', 'county': 'Purworejo', 'state': 'Jawa Tengah', 'ISO3166-2-lvl4': 'ID-JT', 'region': 'Jawa', 'ISO3166-2-lvl3': 'ID-JW', 'postcode': '54113', 'country': 'Indonesia', 'country_code': 'id'}, 'boundingbox': ['-7.7114963', '-7.7106112', '110.0031801', '110.0047245']}

```python
#This is the converted destination city I made based on the code name
df_city =
pd.read_csv('https://docs.google.com/spreadsheets/d/1TKP55H5wCEK5svEBo
_JRgDDBzzR_zw00RzYhZjeq2Ow'+'/export?gid=728410653&format=csv')
df_city
```

```
   UserVar.branch_dest           dest_city_name
0                  TGL                    Tegal
1                  DJJ                 Jayapura
2                  GTO      Gorontalo, Sulawesi
3                  UPG  Ujung Pandang, Sulawesi
4                  CKR                 Cikarang
..                 ...                      ...
57                 SUB                 Surabaya
58                 TNJ           Tanjung Pinang
59                 TGR                Tangerang
60                 PSR                 Pasuruan
61                 TIM                   Timika

[62 rows x 2 columns]
```

```
#Get the latitude and longitude of each destination ciy
#Actually, it might be not accurate because I dont have enough data to
collect the delivery service of each trip and each city
#So here I just calculate randomly only based on the name of the city,
and let the web find out the latitude and longitude of the city (not
exactly the branch)
lat_city = []
long_city = []

for address in df_city['dest_city_name'].values:
  try:
    address = address
    url = 'https://nominatim.openstreetmap.org/search/' +
urllib.parse.quote(address) +'?format=json'

    response = requests.get(url).json()
    if response != []:
      lat_city.append(response[0]["lat"])
      long_city.append(response[0]["lon"])
    else:
      lat_city.append(np.nan)
      long_city.append(np.nan)
  except:
    print(address)

print(len(lat_city))
print(len(long_city))
```

```
62
62
```

```
#Check the result
df_city['dest_lat_city'] = np.array(lat_city)
df_city['dest_lon_city'] = np.array(long_city)
df_city.head(10)
```

```
  UserVar.branch_dest           dest_city_name       dest_lat_city  \
0                 TGL                    Tegal         -7.05644335
1                 DJJ                 Jayapura          -2.5387539
2                 GTO      Gorontalo, Sulawesi          -0.8870281
3                 UPG  Ujung Pandang, Sulawesi  -5.141435550000001
4                 CKR                 Cikarang          -6.2587148
5                 TKG           Bandar Lampung          -5.4460713
6                 SDA                 Sidoarjo         -7.45597405
7                 MKQ                  Merauke          -7.7925193
8                 MDN                   Madiun         -7.61188765
9                 BOO                    Bogor          -6.5962986

        dest_lon_city
0  109.13157658590205
1         140.7037389
```

```
2         123.3838946
3   119.4136705800444
4          107.145742
5         105.2643742
6  112.66088771295344
7        140.01832515
8  111.67319262808837
9         106.7972421
```

```python
#Merge to all data
df_all = pd.merge(df, df_city, how='left', on='UserVar.branch_dest')
df_all
```

```
                taskCreatedTime  taskAssignedTo
taskCompletedTime  \
0     2022-11-01 20:17:26 +0700   pacifiedLion0  2022-11-01 20:46:30
+0700
1     2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 12:33:48
+0700
2     2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 13:41:57
+0700
3     2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 18:18:19
+0700
4     2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 10:51:49
+0700
...                         ...             ...
...
8329  2022-11-10 09:07:12 +0700  debonairPonie1  2022-11-10 09:38:04
+0700
8330  2022-11-10 09:21:42 +0700   murkyThrushe3  2022-11-10 09:37:52
+0700
8331  2022-11-10 09:36:44 +0700    enragedCake7  2022-11-10 09:37:55
+0700
8332  2022-11-10 07:25:40 +0700    lyingPaella2  2022-11-10 10:37:53
+0800
8333  2022-11-10 07:46:13 +0700  emptyPretzels3  2022-11-10 09:37:50
+0700

     taskStatus      flow                taskId  taskLocationDone.lon
\
0          done  Delivery  4fe3b237c832ca4841a2            109.762910

1          done  Delivery  08a4da25256affae8446            110.033986

2          done  Delivery  2ff0dc469826158b7684            109.999733

3          done  Delivery  331c172c2b383f774328            110.003708

4          done  Delivery  a9d53fa96c80baee8b23            110.013887
```

```
...          ...       ...                   ...                   ...

8329       done  Delivery  501af4e040a742e9e878              0.000000

8330       done  Delivery  5cc952d9e9f8066dbf24            110.352054

8331       done  Delivery  1b136b5a3c60749eb571            105.664897

8332       done  Delivery  e92e813c8539080c922e            119.877173

8333       done  Delivery  cdb90c597655282306fd              0.000000


      taskLocationDone.lat  cod.amount cod.received  ...  \
0                -6.926608    685000.0         True  ...
1                -7.876154     53500.0         True  ...
2                -7.849777    179500.0         True  ...
3                -7.710998     31815.0         True  ...
4                -7.829742    144562.0         True  ...
...                    ...         ...          ...  ...
8329              0.000000         NaN          nan  ...
8330             -7.892571         NaN          nan  ...
8331             -5.359063         NaN          nan  ...
8332             -8.513305    151000.0        False  ...
8333              0.000000         NaN          nan  ...

     UserVar.receiver_city
UserVar.taskDetailStatusLabel  \
0       BATANG ,KAB BATANG                            YANG
BERSANGKUTAN
1      PURWODADI,PURWOREJO                            YANG
BERSANGKUTAN
2      PURWODADI,PURWOREJO                            YANG
BERSANGKUTAN
3      PURWODADI,PURWOREJO                            YANG
BERSANGKUTAN
4        BAGELEN,PURWOREJO                            YANG
BERSANGKUTAN
...                    ...                                          ..
.
8329  PALMERAH ,JAKARTA BA
ATASAN/STAFF/KARYAWAN/BAWAHAN
8330           KOTA BANTUL  ALAMAT TIDAK LENGKAP service/ TIDAK
DIKENAL
8331  MARGA SEKAMPUNG KAB.                            YANG
BERSANGKUTAN
8332    KOMODO,LABUAN BAJO                       PENERIMA PINDAH
ALAMAT
8333         JAKARTA PUSAT
```

```
RECEPTIONIST

     UserVar.taskDetailStatus UserVar.weight UserVar.branch_origin  \
0                         D01         13.000                   CGK
1                         D01          1.300                   CGK
2                         D01          3.000                   CGK
3                         D01          0.625                   CGK
4                         D01          3.000                   CGK
...                       ...            ...                   ...
8329                      D10          1.000                   CGK
8330                      U01          1.000                   TGR
8331                      D01          1.440                   CGK
8332                      U03          0.600                   CGK
8333                      D02          1.000                   BPN

      UserVar.taskStatus   receiver_city_clean  dest_city_name  \
0                 COLF01       BATANG   BATANG        Semarang
1                 COLF01   PURWODADI PURWOREJO        Magelang
2                 COLF01   PURWODADI PURWOREJO        Magelang
3                 COLF01   PURWODADI PURWOREJO        Magelang
4                 COLF01     BAGELEN PURWOREJO        Magelang
...                  ...                   ...             ...
8329              COLF01  PALMERAH  JAKARTA BA         Jakarta
8330              COLF02                BANTUL      Yogyakarta
8331              COLF01       MARGA SEKAMPUNG  Bandar Lampung
8332              COLF02    KOMODO LABUAN BAJO   Kupang, Timor
8333              COLF01         JAKARTA PUSAT         Jakarta

            dest_lat_city       dest_lon_city
0              -6.9903988         110.4229104
1             -7.51361445   110.2145132553504
2             -7.51361445   110.2145132553504
3             -7.51361445   110.2145132553504
4             -7.51361445   110.2145132553504
...                   ...                 ...
8329            -6.175247         106.8270488
8330  -7.9778383999999996  110.36722565020224
8331           -5.4460713         105.2643742
8332          -10.1432432         123.6585378
8333            -6.175247         106.8270488

[8334 rows x 22 columns]
```

```
#Here, I calculate the distance between the branch destination and
task location done. It might be no accurate because of the reason
before
id_list = []
distance = []

for (id, lat1, lon1, lat2, lon2) in zip(df_all['taskId'].values,
```

```
df_all['taskLocationDone.lat'].values,

df_all['taskLocationDone.lon'].values,
                                    df_all['dest_lat_city'].values,
                                    df_all['dest_lon_city']):
  try:
    if (lat1!=0 or lon1!=0):
      coords_1 = (lat1, lon1)
      coords_2 = (lat2, lon2)
      distance.append(geopy.distance.geodesic(coords_1, coords_2).km)
      id_list.append(id)
    else:
      pass

  except:
    pass

df_distance = pd.DataFrame(list(zip(id_list, distance)),
columns=['taskId', 'distance(km)'])
df_distance
```

```
                    taskId  distance(km)
0     4fe3b237c832ca4841a2     73.273671
1     08a4da25256affae8446     44.768913
2     2ff0dc469826158b7684     44.087177
3     331c172c2b383f774328     31.900013
4     a9d53fa96c80baee8b23     41.379771
...                    ...           ...
5110  abb2cc73275d23947762      9.155145
5111  4df98016923e193d39ec     18.600634
5112  5cc952d9e9f8066dbf24      9.577383
5113  1b136b5a3c60749eb571     45.420134
5114  e92e813c8539080c922e    452.825279

[5115 rows x 2 columns]
```

```
#Merge all the data
df_all = pd.merge(df_all, df_distance, how='left', on='taskId')
df_all
```

```
                taskCreatedTime  taskAssignedTo
taskCompletedTime  \
0     2022-11-01 20:17:26 +0700   pacifiedLion0  2022-11-01 20:46:30
+0700
1     2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 12:33:48
+0700
2     2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 13:41:57
+0700
```

```
3     2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 18:18:19
+0700
4     2022-11-01 08:41:07 +0700  peacefulTacos6  2022-11-01 10:51:49
+0700
...                         ...             ...
...
8329  2022-11-10 09:07:12 +0700  debonairPonie1  2022-11-10 09:38:04
+0700
8330  2022-11-10 09:21:42 +0700   murkyThrushe3  2022-11-10 09:37:52
+0700
8331  2022-11-10 09:36:44 +0700    enragedCake7  2022-11-10 09:37:55
+0700
8332  2022-11-10 07:25:40 +0700    lyingPaella2  2022-11-10 10:37:53
+0800
8333  2022-11-10 07:46:13 +0700  emptyPretzels3  2022-11-10 09:37:50
+0700

     taskStatus      flow                taskId  taskLocationDone.lon
\
0          done  Delivery  4fe3b237c832ca4841a2            109.762910

1          done  Delivery  08a4da25256affae8446            110.033986

2          done  Delivery  2ff0dc469826158b7684            109.999733

3          done  Delivery  331c172c2b383f774328            110.003708

4          done  Delivery  a9d53fa96c80baee8b23            110.013887

...         ...       ...                   ...                   ...

8329       done  Delivery  501af4e040a742e9e878              0.000000

8330       done  Delivery  5cc952d9e9f8066dbf24            110.352054

8331       done  Delivery  1b136b5a3c60749eb571            105.664897

8332       done  Delivery  e92e813c8539080c922e            119.877173

8333       done  Delivery  cdb90c597655282306fd              0.000000


      taskLocationDone.lat  cod.amount cod.received  ...  \
0                -6.926608    685000.0         True  ...
1                -7.876154     53500.0         True  ...
2                -7.849777    179500.0         True  ...
3                -7.710998     31815.0         True  ...
4                -7.829742    144562.0         True  ...
...                    ...         ...          ...  ...
```

```
8329             0.000000        NaN         nan  ...
8330            -7.892571        NaN         nan  ...
8331            -5.359063        NaN         nan  ...
8332            -8.513305   151000.0       False  ...
8333             0.000000        NaN         nan  ...

                    UserVar.taskDetailStatusLabel
UserVar.taskDetailStatus  \
0                                 YANG BERSANGKUTAN
D01
1                                 YANG BERSANGKUTAN
D01
2                                 YANG BERSANGKUTAN
D01
3                                 YANG BERSANGKUTAN
D01
4                                 YANG BERSANGKUTAN
D01
...                                             ...
...
8329                ATASAN/STAFF/KARYAWAN/BAWAHAN
D10
8330  ALAMAT TIDAK LENGKAP service/ TIDAK DIKENAL
U01
8331                              YANG BERSANGKUTAN
D01
8332                          PENERIMA PINDAH ALAMAT
U03
8333                                   RECEPTIONIST
D02

     UserVar.weight UserVar.branch_origin UserVar.taskStatus  \
0            13.000                   CGK             COLF01
1             1.300                   CGK             COLF01
2             3.000                   CGK             COLF01
3             0.625                   CGK             COLF01
4             3.000                   CGK             COLF01
...             ...                   ...                ...
8329          1.000                   CGK             COLF01
8330          1.000                   TGR             COLF02
8331          1.440                   CGK             COLF01
8332          0.600                   CGK             COLF02
8333          1.000                   BPN             COLF01

       receiver_city_clean  dest_city_name        dest_lat_city  \
0          BATANG   BATANG        Semarang           -6.9903988
1       PURWODADI PURWOREJO       Magelang          -7.51361445
2       PURWODADI PURWOREJO       Magelang          -7.51361445
3       PURWODADI PURWOREJO       Magelang          -7.51361445
4         BAGELEN PURWOREJO       Magelang          -7.51361445
```

```
...                      ...             ...                 ...
8329  PALMERAH  JAKARTA BA         Jakarta           -6.175247
8330                BANTUL      Yogyakarta  -7.9778383999999996
8331       MARGA SEKAMPUNG  Bandar Lampung          -5.4460713
8332    KOMODO LABUAN BAJO   Kupang, Timor         -10.1432432
8333          JAKARTA PUSAT         Jakarta           -6.175247

           dest_lon_city distance(km)
0            110.4229104    73.273671
1      110.2145132553504    44.768913
2      110.2145132553504    44.087177
3      110.2145132553504    31.900013
4      110.2145132553504    41.379771
...                  ...          ...
8329         106.8270488          NaN
8330  110.36722565020224     9.577383
8331         105.2643742    45.420134
8332         123.6585378   452.825279
8333         106.8270488          NaN

[8334 rows x 23 columns]
```

```
#Convert the columns to datetime
df_all['taskCreatedTime'] = pd.to_datetime(df_all['taskCreatedTime'],
utc=True)
df_all['taskCompletedTime'] =
pd.to_datetime(df_all['taskCompletedTime'], utc=True)

#Calculate the time diff to check the time needed to complete a task
df_all['time_diff(s)'] = (df_all['taskCompletedTime'] -
df_all['taskCreatedTime']).dt.seconds

#Assume that the created time is the time of delivery of goods from
the destination branch
df_all['average_speed(kmph)'] =
df_all['distance(km)']/(df_all['time_diff(s)']/3600)

df_all
```

```
               taskCreatedTime  taskAssignedTo
taskCompletedTime  \
0    2022-11-01 13:17:26+00:00   pacifiedLion0 2022-11-01
13:46:30+00:00
1    2022-11-01 01:41:07+00:00  peacefulTacos6 2022-11-01
05:33:48+00:00
2    2022-11-01 01:41:07+00:00  peacefulTacos6 2022-11-01
06:41:57+00:00
3    2022-11-01 01:41:07+00:00  peacefulTacos6 2022-11-01
11:18:19+00:00
4    2022-11-01 01:41:07+00:00  peacefulTacos6 2022-11-01
03:51:49+00:00
```

```
...                          ...             ...                       .
..
8329 2022-11-10 02:07:12+00:00  debonairPonie1 2022-11-10
02:38:04+00:00
8330 2022-11-10 02:21:42+00:00   murkyThrushe3 2022-11-10
02:37:52+00:00
8331 2022-11-10 02:36:44+00:00    enragedCake7 2022-11-10
02:37:55+00:00
8332 2022-11-10 00:25:40+00:00    lyingPaella2 2022-11-10
02:37:53+00:00
8333 2022-11-10 00:46:13+00:00  emptyPretzels3 2022-11-10
02:37:50+00:00

     taskStatus      flow                taskId  taskLocationDone.lon
\
0          done  Delivery  4fe3b237c832ca4841a2            109.762910

1          done  Delivery  08a4da25256affae8446            110.033986

2          done  Delivery  2ff0dc469826158b7684            109.999733

3          done  Delivery  331c172c2b383f774328            110.003708

4          done  Delivery  a9d53fa96c80baee8b23            110.013887

...         ...       ...                   ...                   ...

8329       done  Delivery  501af4e040a742e9e878              0.000000

8330       done  Delivery  5cc952d9e9f8066dbf24            110.352054

8331       done  Delivery  1b136b5a3c60749eb571            105.664897

8332       done  Delivery  e92e813c8539080c922e            119.877173

8333       done  Delivery  cdb90c597655282306fd              0.000000


      taskLocationDone.lat  cod.amount cod.received  ...
UserVar.weight  \
0                -6.926608    685000.0         True  ...
13.000
1                -7.876154     53500.0         True  ...
1.300
2                -7.849777    179500.0         True  ...
3.000
3                -7.710998     31815.0         True  ...
0.625
4                -7.829742    144562.0         True  ...
```

```
3.000
...                    ...         ...          ...  ...            ..
.
8329              0.000000         NaN          nan  ...
1.000
8330             -7.892571         NaN          nan  ...
1.000
8331             -5.359063         NaN          nan  ...
1.440
8332             -8.513305    151000.0        False  ...
0.600
8333              0.000000         NaN          nan  ...
1.000

     UserVar.branch_origin UserVar.taskStatus   receiver_city_clean  \
0                      CGK             COLF01        BATANG   BATANG
1                      CGK             COLF01   PURWODADI PURWOREJO
2                      CGK             COLF01   PURWODADI PURWOREJO
3                      CGK             COLF01   PURWODADI PURWOREJO
4                      CGK             COLF01     BAGELEN PURWOREJO
...                    ...                ...                   ...
8329                   CGK             COLF01  PALMERAH  JAKARTA BA
8330                   TGR             COLF02                BANTUL
8331                   CGK             COLF01       MARGA SEKAMPUNG
8332                   CGK             COLF02    KOMODO LABUAN BAJO
8333                   BPN             COLF01         JAKARTA PUSAT

      dest_city_name        dest_lat_city       dest_lon_city
distance(km)  \
0           Semarang           -6.9903988         110.4229104
73.273671
1           Magelang          -7.51361445   110.2145132553504
44.768913
2           Magelang          -7.51361445   110.2145132553504
44.087177
3           Magelang          -7.51361445   110.2145132553504
31.900013
4           Magelang          -7.51361445   110.2145132553504
41.379771
...              ...                  ...                 ...
...
8329         Jakarta            -6.175247         106.8270488
NaN
8330      Yogyakarta  -7.9778383999999996  110.36722565020224
9.577383
8331  Bandar Lampung           -5.4460713         105.2643742
45.420134
8332   Kupang, Timor          -10.1432432         123.6585378
452.825279
8333         Jakarta            -6.175247         106.8270488
```

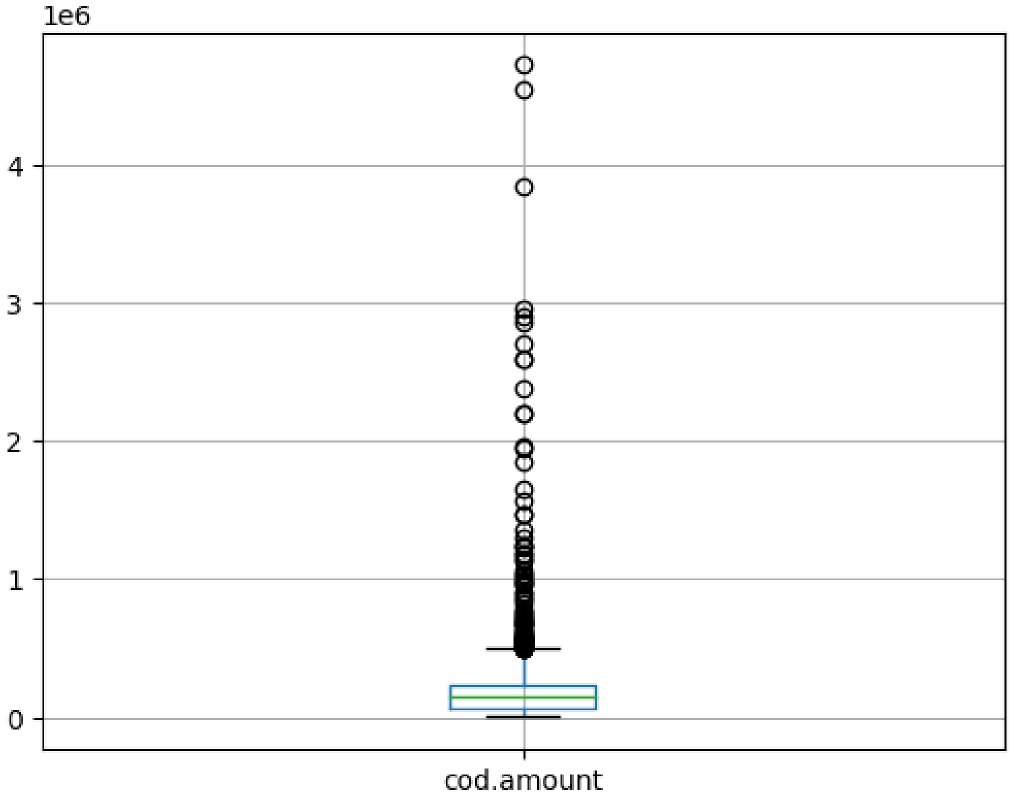
```
NaN

     time_diff(s) average_speed(kmph)
0          1744.0          151.252991
1         13961.0           11.544165
2         18050.0            8.793010
3         34632.0            3.316010
4          7842.0           18.996069
...           ...                 ...
8329       1852.0                 NaN
8330        970.0           35.544927
8331         71.0         2302.992698
8332       7933.0          205.492374
8333       6697.0                 NaN

[8334 rows x 25 columns]
```

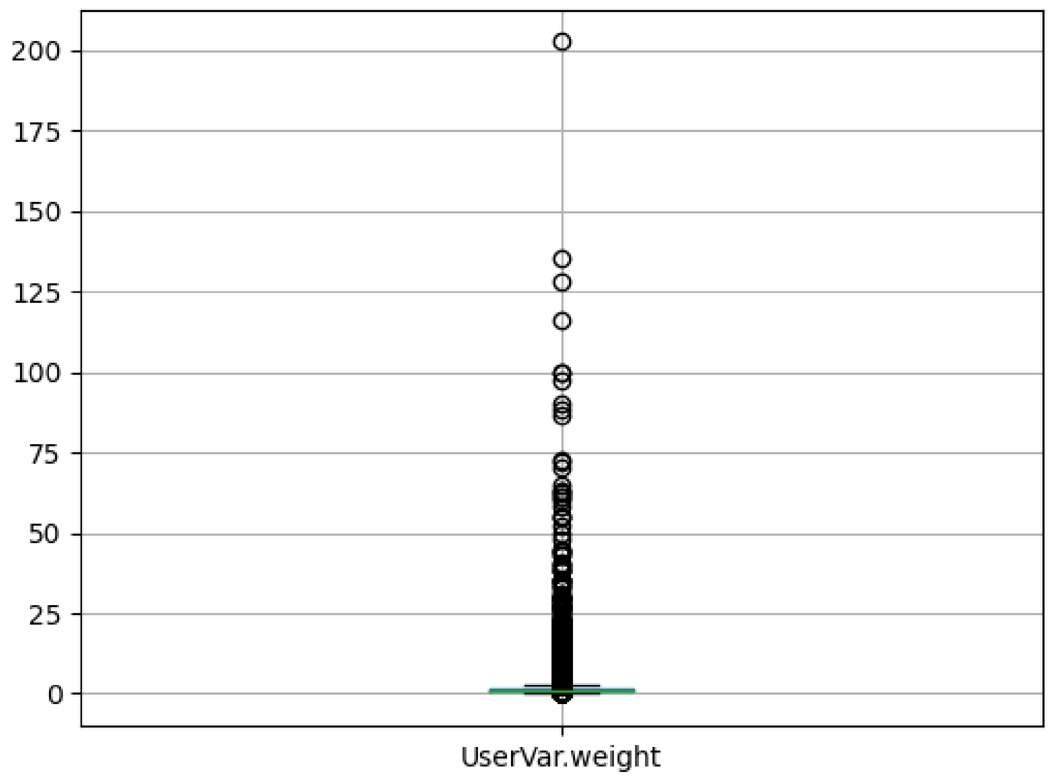
```
df_all.describe()
```

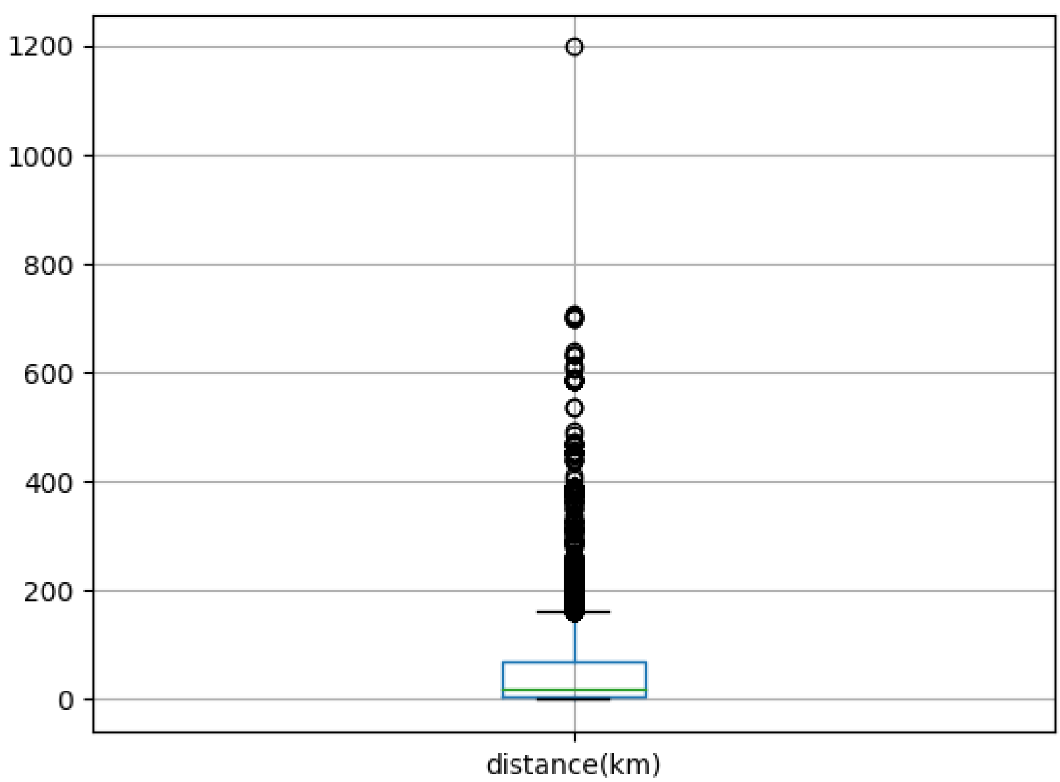
```
       taskLocationDone.lon  taskLocationDone.lat    cod.amount  \
count           7566.000000           7566.000000  2.358000e+03
mean              75.355852             -3.610514  1.911411e+05
std               52.492016              3.647171  2.723770e+05
min                0.000000            -10.493658  8.370000e+02
25%                0.000000             -7.061575  6.100000e+04
50%              106.843097             -3.329263  1.533750e+05
75%              112.182877              0.000000  2.350000e+05
max              140.806424              5.564040  4.730000e+06

       UserVar.weight  distance(km)  time_diff(s)  average_speed(kmph)

count     8334.000000   5115.000000   7566.000000          5115.000000

mean         2.448298     57.786945   4370.355802           682.579084

std          6.188171     92.433385   6052.779119          3387.264499

min          0.000000      0.021052     15.000000             0.021074

25%          1.000000      6.233397    599.750000             6.713558

50%          1.000000     20.864941   2235.500000            26.372692

75%          1.600000     68.665146   4927.250000           175.988185

max        202.500000   1196.874767  47760.000000         69106.635749
```

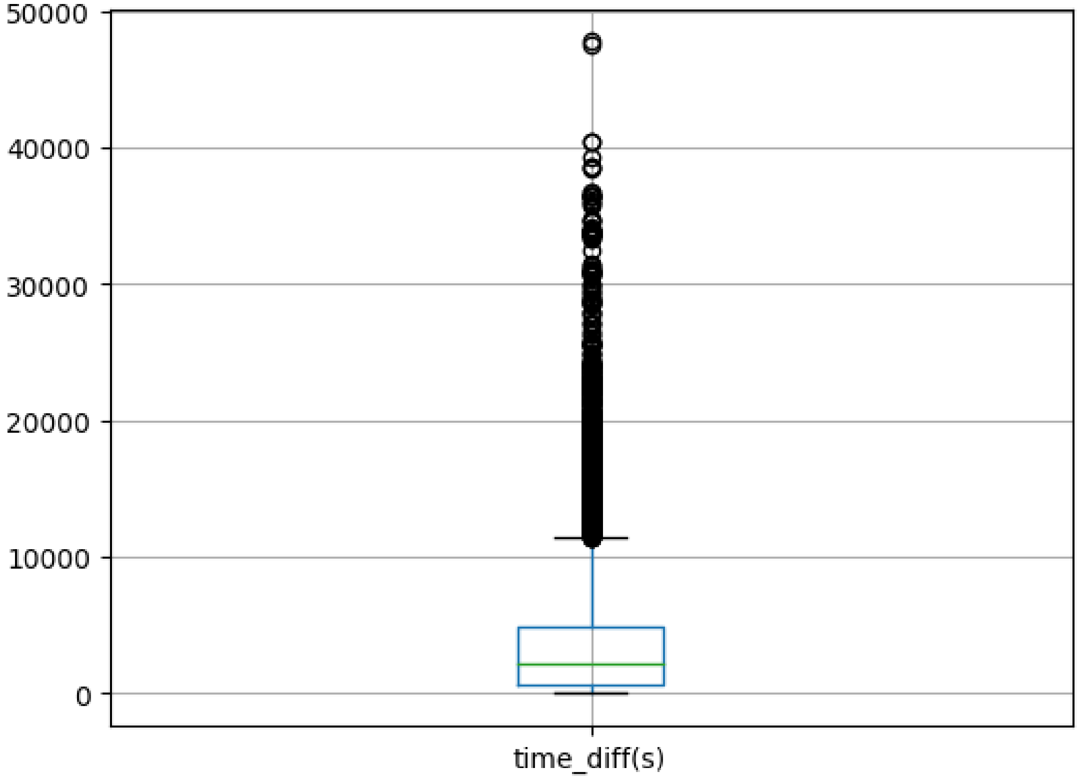
```
#Check the outliers distribution
num_columns = ["cod.amount","UserVar.weight", "distance(km)",
"time_diff(s)", "average_speed(kmph)"]
```

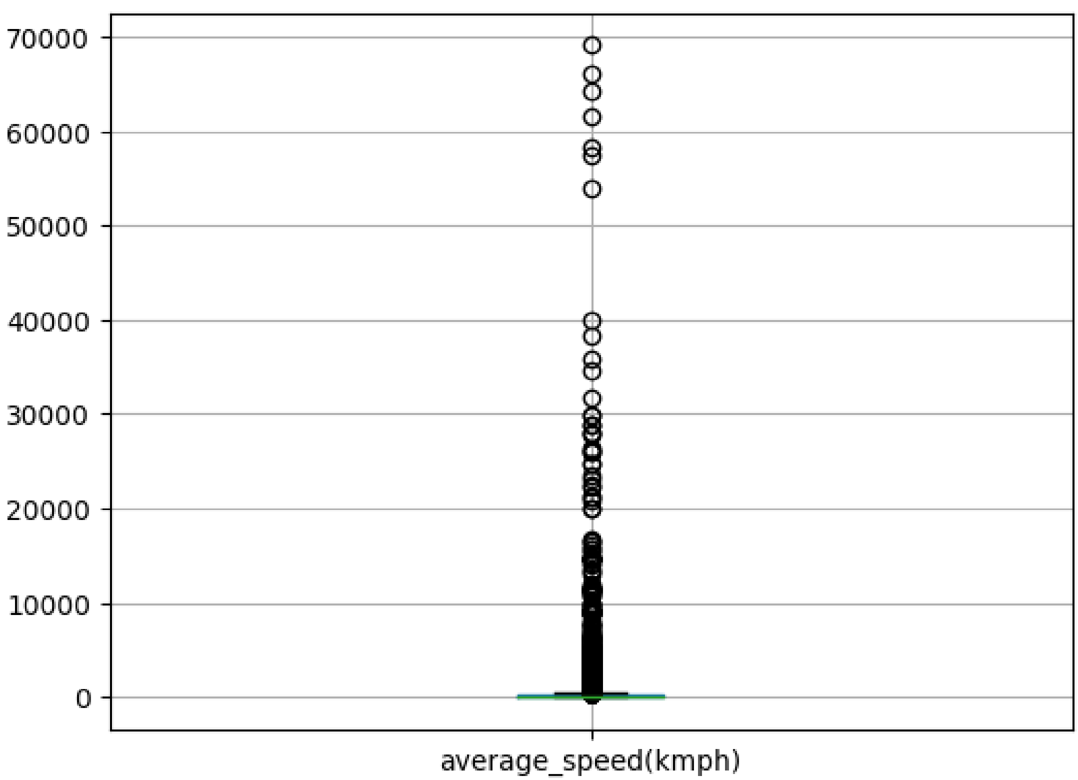
```python
for i in num_columns:
    ax=df_all.boxplot(column=i)
    plt.show()
```




200
175
150
125
100
75
50
25
0
UserVar.weight

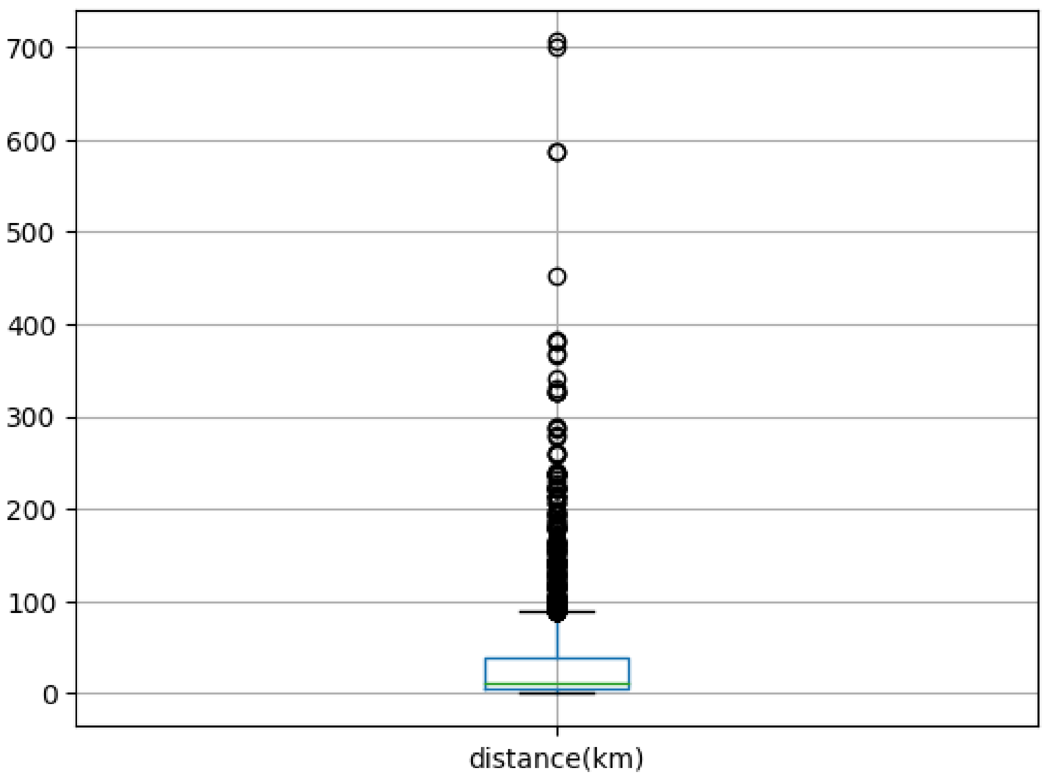

1200
1000
800
600
400
200
0
distance(km)

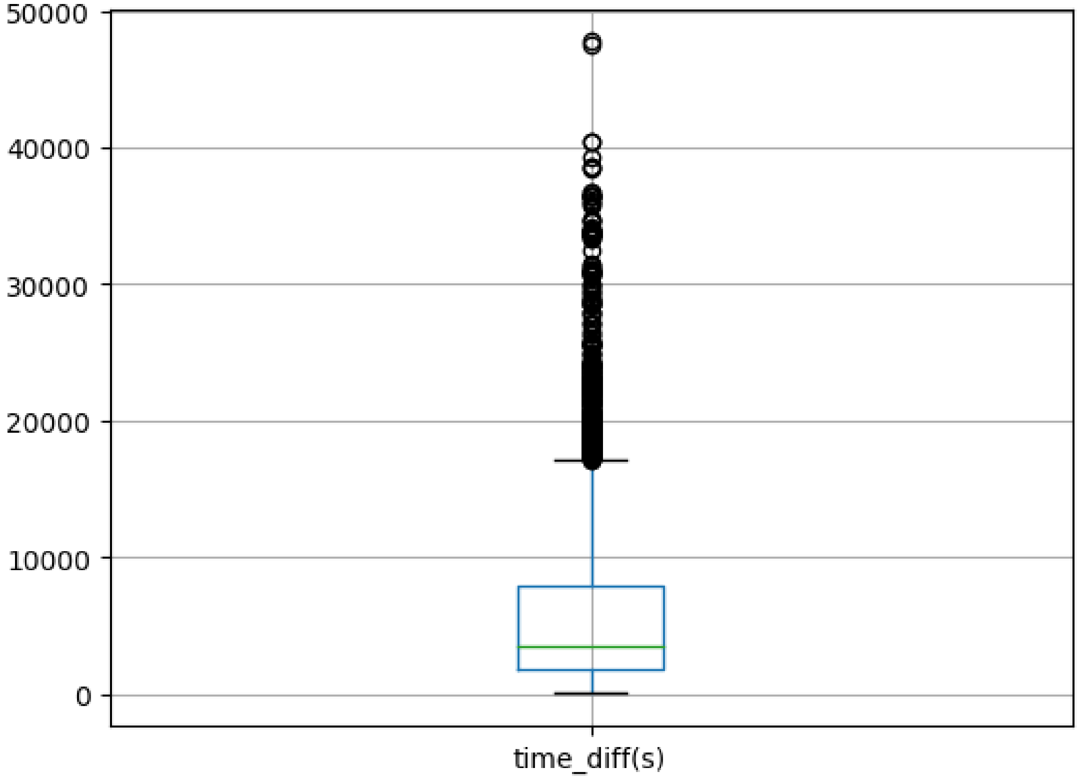


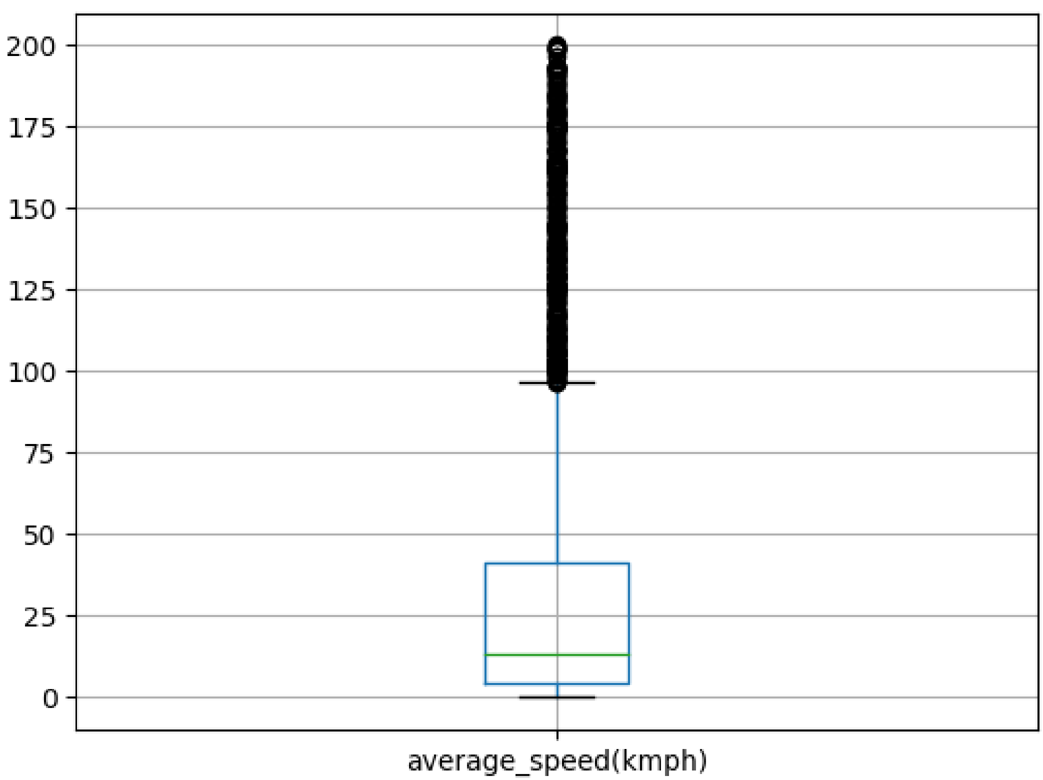



```
#Remove outlier
for x in ["distance(km)", "time_diff(s)", "average_speed(kmph)"]:
```

```
    q75,q25 = np.percentile(df_all.loc[:,x],[75,25])
    intr_qr = q75-q25

    max = q75+(1.5*intr_qr)
    min = q25-(1.5*intr_qr)

    df_all.loc[df_all[x] < min,x] = np.nan
    df_all.loc[df_all[x] > max,x] = np.nan

#Because there are still outliers in the average speed column, I'll
just manually set the maximum average speed limit (even though 200 is
still impossible, but many things are not considered here)
df_all.loc[df_all["average_speed(kmph)"] > 200,"average_speed(kmph)"]
= np.nan
df_all.dropna(subset=['distance(km)', 'time_diff(s)',
'average_speed(kmph)'], inplace=True)

#Check again the distribution after removing outliers
num_columns = ["distance(km)", "time_diff(s)", "average_speed(kmph)"]

for i in num_columns:
    ax=df_all.boxplot(column=i)
    plt.show()
```




50000
40000
30000
20000
10000
0
time_diff(s)


200
175
150
125
100
75
50
25
0
average_speed(kmph)

```
df_all.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3899 entries, 0 to 8330
Data columns (total 25 columns):
 #   Column                         Non-Null Count  Dtype

---  ------                         --------------  -----

 0   taskCreatedTime                3899 non-null   datetime64[ns,
UTC]
 1   taskAssignedTo                 3899 non-null   object

 2   taskCompletedTime              3899 non-null   datetime64[ns,
UTC]
 3   taskStatus                     3899 non-null   object

 4   flow                           3899 non-null   object

 5   taskId                         3899 non-null   object

 6   taskLocationDone.lon           3899 non-null   float64

 7   taskLocationDone.lat           3899 non-null   float64

 8   cod.amount                     1027 non-null   float64

 9   cod.received                   3899 non-null   object

 10  UserVar.branch_dest            3899 non-null   object

 11  UserVar.taskStatusLabel        3899 non-null   object

 12  UserVar.receiver_city          3880 non-null   object

 13  UserVar.taskDetailStatusLabel  3899 non-null   object

 14  UserVar.taskDetailStatus       3899 non-null   object

 15  UserVar.weight                 3899 non-null   float64

 16  UserVar.branch_origin          3899 non-null   object

 17  UserVar.taskStatus             3899 non-null   object

 18  receiver_city_clean            3880 non-null   object

 19  dest_city_name                 3899 non-null   object

 20  dest_lat_city                  3899 non-null   object
```

```
 21  dest_lon_city                  3899 non-null   object

 22  distance(km)                   3899 non-null   float64

 23  time_diff(s)                   3899 non-null   float64

 24  average_speed(kmph)            3899 non-null   float64

dtypes: datetime64[ns, UTC](2), float64(7), object(16)
memory usage: 792.0+ KB
```

```
#Remove the unknown receiver city
df_all=df_all.dropna(subset=['receiver_city_clean'])

#Fill the nan value of these 3 columns
df_all.loc[:, ('cod.amount')] = df_all['cod.amount'].fillna(0)
#Assuming online payment used
df_all.loc[:, ('cod.received')] =
df_all['cod.received'].replace('nan', 'no COD') #Assuming no COD used
df_all.loc[:, ('cod.received')] = df_all['cod.received'].fillna('No
COD') #Assuming no COD used
df_all.loc[:, ('UserVar.branch_origin')] =
df_all['UserVar.branch_origin'].fillna('CGK') #Assuming CGK as the
origin because of the most used branch origin
```

```
<ipython-input-59-2d4865c040f8>:2: 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-copy
  df_all.loc[:, ('cod.amount')] = df_all['cod.amount'].fillna(0)
#Assuming online payment used
<ipython-input-59-2d4865c040f8>: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-copy
  df_all.loc[:, ('cod.received')] =
df_all['cod.received'].replace('nan', 'no COD') #Assuming no COD used
<ipython-input-59-2d4865c040f8>: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-copy
  df_all.loc[:, ('cod.received')] = df_all['cod.received'].fillna('No
COD') #Assuming no COD used
<ipython-input-59-2d4865c040f8>: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-copy
  df_all.loc[:, ('UserVar.branch_origin')] =
df_all['UserVar.branch_origin'].fillna('CGK') #Assuming CGK as the
origin because of the most used branch origin
```

```
df_all.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3880 entries, 0 to 8330
Data columns (total 25 columns):
 #   Column                        Non-Null Count  Dtype

---  ------                        --------------  -----

 0   taskCreatedTime               3880 non-null   datetime64[ns,
UTC]
 1   taskAssignedTo                3880 non-null   object

 2   taskCompletedTime             3880 non-null   datetime64[ns,
UTC]
 3   taskStatus                    3880 non-null   object

 4   flow                          3880 non-null   object

 5   taskId                        3880 non-null   object

 6   taskLocationDone.lon          3880 non-null   float64

 7   taskLocationDone.lat          3880 non-null   float64

 8   cod.amount                    3880 non-null   float64

 9   cod.received                  3880 non-null   object

 10  UserVar.branch_dest           3880 non-null   object

 11  UserVar.taskStatusLabel       3880 non-null   object

 12  UserVar.receiver_city         3880 non-null   object
```

```
 13  UserVar.taskDetailStatusLabel  3880 non-null   object

 14  UserVar.taskDetailStatus       3880 non-null   object

 15  UserVar.weight                 3880 non-null   float64

 16  UserVar.branch_origin          3880 non-null   object

 17  UserVar.taskStatus             3880 non-null   object

 18  receiver_city_clean            3880 non-null   object

 19  dest_city_name                 3880 non-null   object

 20  dest_lat_city                  3880 non-null   object

 21  dest_lon_city                  3880 non-null   object

 22  distance(km)                   3880 non-null   float64

 23  time_diff(s)                   3880 non-null   float64

 24  average_speed(kmph)            3880 non-null   float64

dtypes: datetime64[ns, UTC](2), float64(7), object(16)
memory usage: 788.1+ KB
```

```
#Save the data to fetch to bigquery later
df_all.to_csv('all_data.csv')
```

```
df_all_filtered = df_all.copy()
```

##Modelling Machine Learning

```
df_all_filtered
```

```
               taskCreatedTime    taskAssignedTo
taskCompletedTime  \
0    2022-11-01 13:17:26+00:00     pacifiedLion0 2022-11-01
13:46:30+00:00
1    2022-11-01 01:41:07+00:00    peacefulTacos6 2022-11-01
05:33:48+00:00
2    2022-11-01 01:41:07+00:00    peacefulTacos6 2022-11-01
06:41:57+00:00
3    2022-11-01 01:41:07+00:00    peacefulTacos6 2022-11-01
11:18:19+00:00
4    2022-11-01 01:41:07+00:00    peacefulTacos6 2022-11-01
03:51:49+00:00
...                        ...               ...
...
```

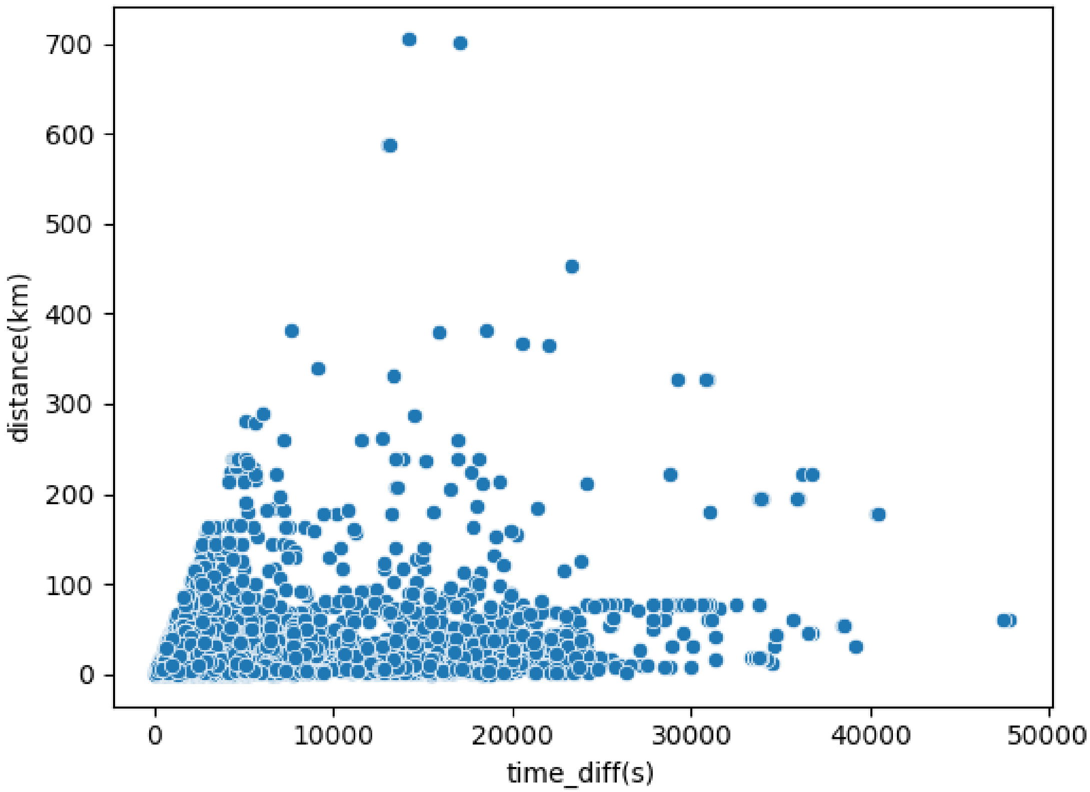
```
8319 2022-11-10 00:50:16+00:00  grudgingBittern7 2022-11-10
02:37:41+00:00
8321 2022-11-10 01:13:30+00:00   humorousPiglet8 2022-11-10
02:38:02+00:00
8323 2022-11-10 01:56:48+00:00       giddyShads0 2022-11-10
02:37:58+00:00
8327 2022-11-10 00:27:51+00:00      dearWhiting2 2022-11-10
02:38:02+00:00
8330 2022-11-10 02:21:42+00:00     murkyThrushe3 2022-11-10
02:37:52+00:00

     taskStatus      flow                taskId  taskLocationDone.lon
\
0          done  Delivery  4fe3b237c832ca4841a2            109.762910

1          done  Delivery  08a4da25256affae8446            110.033986

2          done  Delivery  2ff0dc469826158b7684            109.999733

3          done  Delivery  331c172c2b383f774328            110.003708

4          done  Delivery  a9d53fa96c80baee8b23            110.013887

...         ...       ...                   ...                   ...

8319       done  Delivery  2bf6ce01d5b6a8ac8f34            107.899584

8321       done  Delivery  85f340c19c6cffd3135e            107.694447

8323       done  Delivery  abb2cc73275d23947762             98.736924

8327       done  Delivery  4df98016923e193d39ec            101.438664

8330       done  Delivery  5cc952d9e9f8066dbf24            110.352054


      taskLocationDone.lat  cod.amount cod.received  ...
UserVar.weight  \
0                -6.926608    685000.0         True  ...
13.000
1                -7.876154     53500.0         True  ...
1.300
2                -7.849777    179500.0         True  ...
3.000
3                -7.710998     31815.0         True  ...
0.625
4                -7.829742    144562.0         True  ...
3.000
...                    ...         ...          ...  ...              ..
```

```
.
8319             -7.089875        0.0       no COD  ...
1.000
8321             -6.924457        0.0       no COD  ...
54.800
8323              3.536418        0.0       no COD  ...
1.000
8327              0.479580        0.0       no COD  ...
1.000
8330             -7.892571        0.0       no COD  ...
1.000

     UserVar.branch_origin UserVar.taskStatus   receiver_city_clean  \
0                      CGK             COLF01       BATANG   BATANG
1                      CGK             COLF01   PURWODADI PURWOREJO
2                      CGK             COLF01   PURWODADI PURWOREJO
3                      CGK             COLF01   PURWODADI PURWOREJO
4                      CGK             COLF01     BAGELEN PURWOREJO
...                    ...                ...                   ...
8319                   CGK             COLF01                 GARUT
8321                   CGK             COLF01  UJUNGBERUNG  BANDUNG
8323                   MES             COLF02          MEDAN  MEDAN
8327                   CGK             COLF01  MARPOYAN DAMAI  PEKA
8330                   TGR             COLF02                BANTUL

          dest_city_name        dest_lat_city       dest_lon_city  \
0               Semarang           -6.9903988         110.4229104
1               Magelang          -7.51361445   110.2145132553504
2               Magelang          -7.51361445   110.2145132553504
3               Magelang          -7.51361445   110.2145132553504
4               Magelang          -7.51361445   110.2145132553504
...                  ...                  ...                 ...
8319       Bandung, Java           -6.9215529         107.6110212
8321       Bandung, Java           -6.9215529         107.6110212
8323      Medan, Sumatra            3.5896654          98.6738261
8327  Pekanbaru, Sumatra   0.6111032000000001  101.54284256313278
8330          Yogyakarta  -7.9778383999999996  110.36722565020224

     distance(km) time_diff(s) average_speed(kmph)
0       73.273671       1744.0          151.252991
1       44.768913      13961.0           11.544165
2       44.087177      18050.0            8.793010
3       31.900013      34632.0            3.316010
4       41.379771       7842.0           18.996069
...           ...          ...                 ...
8319    36.920554       6445.0           20.622808
8321     9.225213       5072.0            6.547864
8323     9.155145       2470.0           13.343532
8327    18.600634       7811.0            8.572818
8330     9.577383        970.0           35.544927
```

[3880 rows x 25 columns]

```
sns.scatterplot(data=df_all_filtered, y='distance(km)',
x='time_diff(s)')
```

<Axes: xlabel='time_diff(s)', ylabel='distance(km)'>




```
df_all_filtered.isnull().sum()
```

```
taskCreatedTime                  0
taskAssignedTo                   0
taskCompletedTime                0
taskStatus                       0
flow                             0
taskId                           0
taskLocationDone.lon             0
taskLocationDone.lat             0
cod.amount                       0
cod.received                     0
UserVar.branch_dest              0
UserVar.taskStatusLabel          0
UserVar.receiver_city            0
UserVar.taskDetailStatusLabel    0
UserVar.taskDetailStatus         0
UserVar.weight                   0
```

```
UserVar.branch_origin           0
UserVar.taskStatus              0
receiver_city_clean             0
dest_city_name                  0
dest_lat_city                   0
dest_lon_city                   0
distance(km)                    0
time_diff(s)                    0
average_speed(kmph)             0
dtype: int64
```

```
df_all_filtered.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3880 entries, 0 to 8330
Data columns (total 25 columns):
 #   Column                        Non-Null Count  Dtype

---  ------                        --------------  -----

 0   taskCreatedTime               3880 non-null   datetime64[ns,
UTC]
 1   taskAssignedTo                3880 non-null   object

 2   taskCompletedTime             3880 non-null   datetime64[ns,
UTC]
 3   taskStatus                    3880 non-null   object

 4   flow                          3880 non-null   object

 5   taskId                        3880 non-null   object

 6   taskLocationDone.lon          3880 non-null   float64

 7   taskLocationDone.lat          3880 non-null   float64

 8   cod.amount                    3880 non-null   float64

 9   cod.received                  3880 non-null   object

 10  UserVar.branch_dest           3880 non-null   object

 11  UserVar.taskStatusLabel       3880 non-null   object

 12  UserVar.receiver_city         3880 non-null   object

 13  UserVar.taskDetailStatusLabel  3880 non-null   object

 14  UserVar.taskDetailStatus      3880 non-null   object
```

```
 15  UserVar.weight                  3880 non-null   float64

 16  UserVar.branch_origin           3880 non-null   object

 17  UserVar.taskStatus              3880 non-null   object

 18  receiver_city_clean             3880 non-null   object

 19  dest_city_name                  3880 non-null   object

 20  dest_lat_city                   3880 non-null   object

 21  dest_lon_city                   3880 non-null   object

 22  distance(km)                    3880 non-null   float64

 23  time_diff(s)                    3880 non-null   float64

 24  average_speed(kmph)             3880 non-null   float64

dtypes: datetime64[ns, UTC](2), float64(7), object(16)
memory usage: 788.1+ KB
```

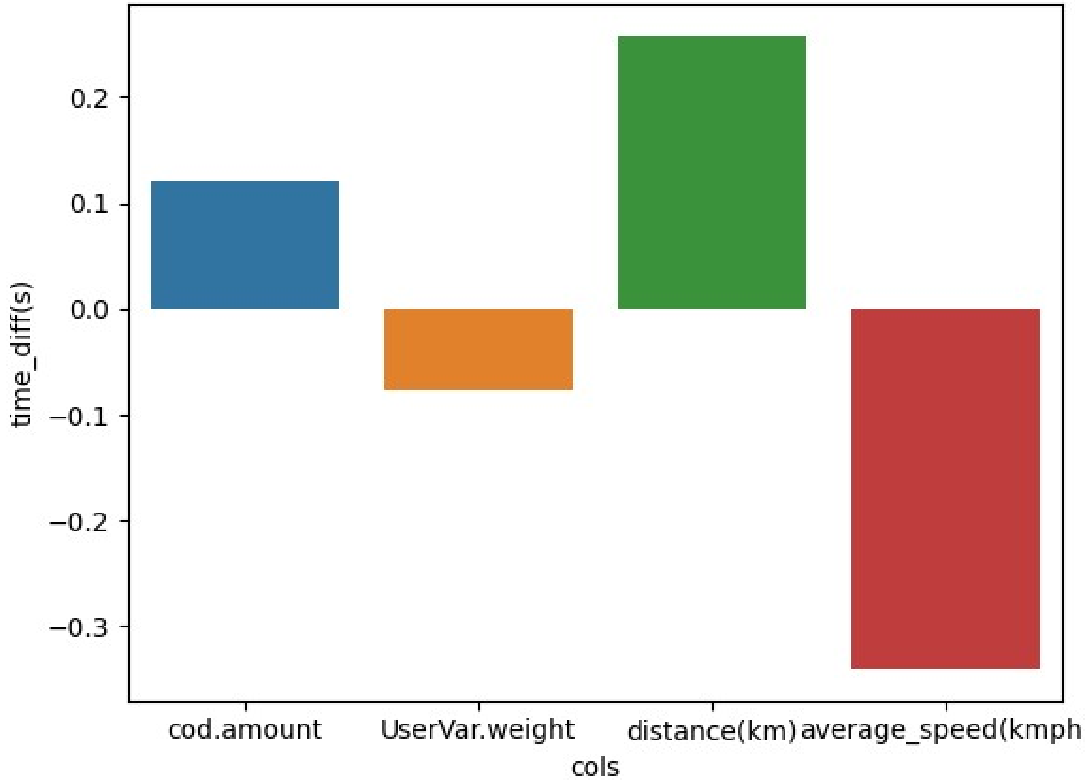
```
#Only use the important columns
df_prepared = df_all_filtered[["taskAssignedTo", "cod.amount",
"cod.received", "UserVar.branch_origin", "UserVar.branch_dest",
"UserVar.taskDetailStatus",
                 "UserVar.weight", "UserVar.taskStatus",
"receiver_city_clean", "distance(km)", "time_diff(s)",
"average_speed(kmph)"]]

df_prepared.head()
```

```
   taskAssignedTo  cod.amount cod.received UserVar.branch_origin  \
0   pacifiedLion0    685000.0         True                   CGK
1  peacefulTacos6     53500.0         True                   CGK
2  peacefulTacos6    179500.0         True                   CGK
3  peacefulTacos6     31815.0         True                   CGK
4  peacefulTacos6    144562.0         True                   CGK

  UserVar.branch_dest UserVar.taskDetailStatus  UserVar.weight  \
0                 SRG                      D01          13.000
1                 MGL                      D01           1.300
2                 MGL                      D01           3.000
3                 MGL                      D01           0.625
4                 MGL                      D01           3.000

  UserVar.taskStatus  receiver_city_clean  distance(km)  time_diff(s)
\
```

```
0              COLF01      BATANG   BATANG     73.273671        1744.0

1              COLF01  PURWODADI PURWOREJO     44.768913       13961.0

2              COLF01  PURWODADI PURWOREJO     44.087177       18050.0

3              COLF01  PURWODADI PURWOREJO     31.900013       34632.0

4              COLF01    BAGELEN PURWOREJO     41.379771        7842.0


   average_speed(kmph)
0           151.252991
1            11.544165
2             8.793010
3             3.316010
4            18.996069
```

```
corrM=df_prepared.corr()
sns.barplot(data=corrM.drop('time_diff(s)').reset_index(names='cols'),
x='cols', y='time_diff(s)')
```

```
<ipython-input-54-bc74f6db1b8d>:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it
will default to False. Select only valid columns or specify the value
of numeric_only to silence this warning.
  corrM=df_prepared.corr()

<Axes: xlabel='cols', ylabel='time_diff(s)'>
```



Modelling using One-Hot Encoding

```
#Perform OHE
df_dummy = pd.get_dummies(df_prepared, drop_first=True)
df_dummy.head()
```

```
   cod.amount  UserVar.weight  distance(km)  time_diff(s)  \
0    685000.0          13.000     73.273671        1744.0
1     53500.0           1.300     44.768913       13961.0
2    179500.0           3.000     44.087177       18050.0
3     31815.0           0.625     31.900013       34632.0
4    144562.0           3.000     41.379771        7842.0

   average_speed(kmph)  taskAssignedTo_abjectCaribou1  \
0           151.252991                              0
1            11.544165                              0
2             8.793010                              0
3             3.316010                              0
4            18.996069                              0

   taskAssignedTo_abjectCur0  taskAssignedTo_abjectFerret4  \
0                          0                             0
1                          0                             0
2                          0                             0
3                          0                             0
4                          0                             0
```

```
   taskAssignedTo_abjectPepper4  taskAssignedTo_abjectSausage7  ...  \
0                             0                              0  ...   
1                             0                              0  ...   
2                             0                              0  ...   
3                             0                              0  ...   
4                             0                              0  ...   

   receiver_city_clean_WOLIO BAU-BAU  \
0                                  0   
1                                  0   
2                                  0   
3                                  0   
4                                  0   

   receiver_city_clean_WONGSOREJO BANYUWANG  \
0                                         0   
1                                         0   
2                                         0   
3                                         0   
4                                         0   

   receiver_city_clean_WONOAYU SIDOARJO  \
0                                     0   
1                                     0   
2                                     0   
3                                     0   
4                                     0   

   receiver_city_clean_WONOCOLO  SURABAYA  \
0                                       0   
1                                       0   
2                                       0   
3                                       0   
4                                       0   

   receiver_city_clean_WONOGIRI  WONOGIR  \
0                                      0   
1                                      0   
2                                      0   
3                                      0   
4                                      0   

   receiver_city_clean_WONOSARI  GN KIDU  \
0                                      0   
1                                      0   
2                                      0   
3                                      0   
4                                      0   
```

```
   receiver_city_clean_WONOSEGORO BOYOLALI
receiver_city_clean_WONOSOBO  \
0                                         0
0
1                                         0
0
2                                         0
0
3                                         0
0
4                                         0
0

   receiver_city_clean_WRINGINANOM GRESIK  receiver_city_clean_WUA-WUA
KENDARI
0                                        0
0
1                                        0
0
2                                        0
0
3                                        0
0
4                                        0
0

[5 rows x 2780 columns]
```

```
#Standardize the value of numeric columns
scaler = MinMaxScaler()
df_dummy[['cod.amount',     'UserVar.weight',     'distance(km)',
'average_speed(kmph)']] = scaler.fit_transform(df_dummy[['cod.amount',
      'UserVar.weight',     'distance(km)', 'average_speed(kmph)']])

df_dummy.head()
```

```
   cod.amount  UserVar.weight  distance(km)  time_diff(s)  \
0    0.144820         0.13000      0.103678        1744.0
1    0.011311         0.01300      0.063334       13961.0
2    0.037949         0.03000      0.062369       18050.0
3    0.006726         0.00625      0.045120       34632.0
4    0.030563         0.03000      0.058537        7842.0

   average_speed(kmph)  taskAssignedTo_abjectCaribou1  \
0             0.756322                              0
1             0.057628                              0
2             0.043869                              0
3             0.016478                              0
4             0.094895                              0
```

```
   taskAssignedTo_abjectCur0  taskAssignedTo_abjectFerret4  \
0                          0                             0   
1                          0                             0   
2                          0                             0   
3                          0                             0   
4                          0                             0   

   taskAssignedTo_abjectPepper4  taskAssignedTo_abjectSausage7  ...  \
0                             0                              0  ...   
1                             0                              0  ...   
2                             0                              0  ...   
3                             0                              0  ...   
4                             0                              0  ...   

   receiver_city_clean_WOLIO BAU-BAU  \
0                                  0   
1                                  0   
2                                  0   
3                                  0   
4                                  0   

   receiver_city_clean_WONGSOREJO BANYUWANG  \
0                                         0   
1                                         0   
2                                         0   
3                                         0   
4                                         0   

   receiver_city_clean_WONOAYU SIDOARJO  \
0                                     0   
1                                     0   
2                                     0   
3                                     0   
4                                     0   

   receiver_city_clean_WONOCOLO  SURABAYA  \
0                                       0   
1                                       0   
2                                       0   
3                                       0   
4                                       0   

   receiver_city_clean_WONOGIRI  WONOGIR  \
0                                      0   
1                                      0   
2                                      0   
3                                      0   
4                                      0   
```

```
   receiver_city_clean_WONOSARI  GN KIDU  \
0                                      0
1                                      0
2                                      0
3                                      0
4                                      0

   receiver_city_clean_WONOSEGORO BOYOLALI
receiver_city_clean_WONOSOBO  \
0                                        0
0
1                                        0
0
2                                        0
0
3                                        0
0
4                                        0
0

   receiver_city_clean_WRINGINANOM GRESIK  receiver_city_clean_WUA-WUA
KENDARI
0                                       0
0
1                                       0
0
2                                       0
0
3                                       0
0
4                                       0
0

[5 rows x 2780 columns]
```

```
X = df_dummy.drop('time_diff(s)', axis=1)
y = df_dummy['time_diff(s)']

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=0)

#Try 3 different models
lgbm = LGBMRegressor()
rfc = RandomForestRegressor()
xgb = XGBRegressor()

models = [lgbm, rfc, xgb]
models_name = ["LGBM","Random Forest", "XGBoost"]
```

```
#The result and performance of 3 different models
model_fitted = []
scoring_results=[]
scorings = ["neg_mean_absolute_error", "neg_root_mean_squared_error",
"r2"]

for model in models:
  score=[]
  for scoring in scorings:
    results = model_selection.cross_val_score(model, X_train, y_train,
cv=3, scoring=scoring)
    score.append(results.mean())
  scoring_results.append(score)

#Show the model performance
model_results = pd.DataFrame(scoring_results, columns=scorings,
index=models_name)
model_results
```

```
               neg_mean_absolute_error  neg_root_mean_squared_error
r2
LGBM                       -555.382040                 -1207.627670
0.971424
Random Forest              -509.617642                 -1336.240150
0.964822
XGBoost                    -597.140313                 -1231.116828
0.970437
```

```
print(model_results["neg_root_mean_squared_error"].idxmax())
```

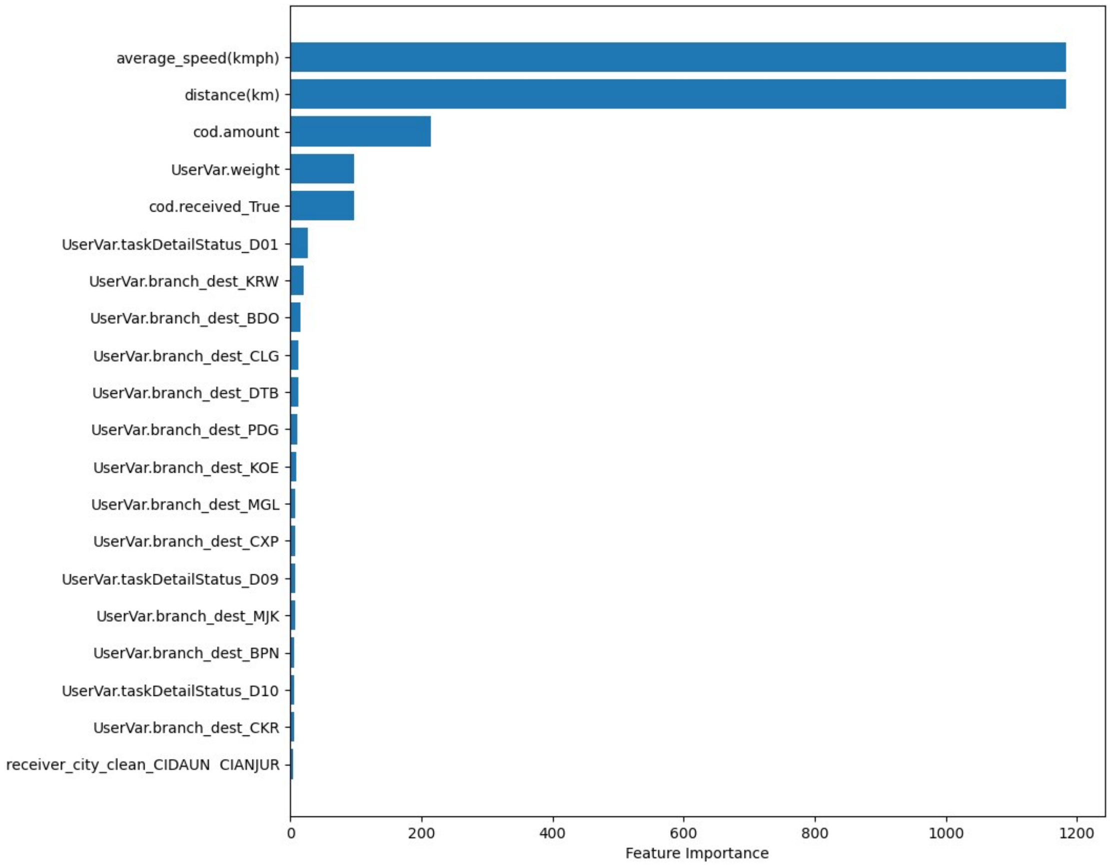
```
LGBM
```

```
#Pick only one the best model to train
model =
models[models_name.index(model_results["neg_root_mean_squared_error"].
idxmax())].fit(X_train, y_train)
y_pred = model.predict(X_test)

print("The r2 score is: ", r2_score(y_test, y_pred))
print("The mean absolute error is: ", mean_absolute_error(y_test,
y_pred))
print("The root mean squared error is: ", (mean_squared_error(y_test,
y_pred)**0.5))
```

```
The r2 score is:  0.9751861841207782
The mean absolute error is:  526.1110804204747
The root mean squared error is:  1220.2104476523316
```

```
#Get the top 20 of most important features
sorted_idx = model.feature_importances_.argsort()
fig, ax = plt.subplots(figsize=(10, 10))
plt.barh(X.columns[sorted_idx][-20:],
```

```
model.feature_importances_[sorted_idx][-20:])
plt.xlabel("Feature Importance")
plt.show()
```




```
test_data=y_test.copy()
test_data=test_data.reset_index()
test_data['type']="test"
test_data.drop("index", axis=1, inplace=True)
test_data.columns=["time_value", "type"]
test_data.head()
```

```
   time_value  type
0      3833.0  test
1     18745.0  test
2       937.0  test
3      1789.0  test
4       927.0  test
```

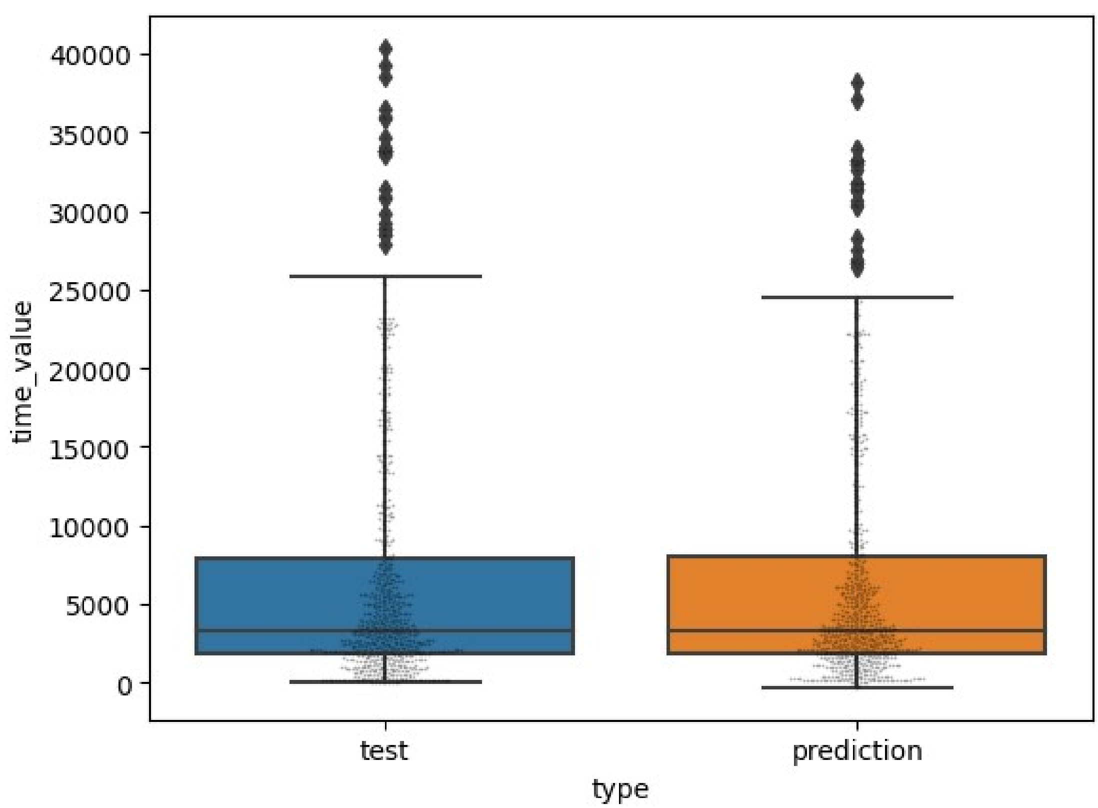
```
pred_data=pd.DataFrame(y_pred, columns=["time_value"])
pred_data["type"]="prediction"
pred_data
```

```
       time_value        type
0     3678.216253  prediction
1    18363.943402  prediction
```

```
2      999.304618  prediction
3     1629.824572  prediction
4     1107.870522  prediction
..            ...         ...
771    381.476026  prediction
772   2919.361332  prediction
773    240.486696  prediction
774   6120.796213  prediction
775   4165.906637  prediction

[776 rows x 2 columns]
```

```
#Shows the difference of distributions between test and prediction
value
pred_result=pd.concat([test_data, pred_data], axis=0,
ignore_index=True)
sns.boxplot(x="type", y="time_value", data=pred_result, whis=3.0)
sns.swarmplot(x="type", y="time_value", data=pred_result, size=1.2,
color="k", alpha=0.3)
```

```
<Axes: xlabel='type', ylabel='time_value'>
```



Modelling using Label Encoding

```
#Perform Label Encoding
from sklearn import preprocessing

df_label_encoded=df_prepared.copy()
label_encoder = preprocessing.LabelEncoder()
df_label_encoded[['taskAssignedTo', 'cod.received',
'UserVar.branch_origin', 'UserVar.branch_dest',
'UserVar.taskDetailStatus', 'UserVar.taskStatus',
'receiver_city_clean']]= df_label_encoded[['taskAssignedTo',
'cod.received', 'UserVar.branch_origin', 'UserVar.branch_dest',
'UserVar.taskDetailStatus', 'UserVar.taskStatus',
'receiver_city_clean']].apply(label_encoder.fit_transform)

#Standardizing the numeric value

scaler = MinMaxScaler()
df_label_encoded[['cod.amount',  'UserVar.weight',     'distance(km)',
'average_speed(kmph)']] =
scaler.fit_transform(df_label_encoded[['cod.amount',  'UserVar.weight'
,    'distance(km)', 'average_speed(kmph)']])

X = df_label_encoded.drop('time_diff(s)', axis=1)
y = df_label_encoded['time_diff(s)']

X
```

```
      taskAssignedTo  cod.amount  cod.received  UserVar.branch_origin
\
0               1040    0.144820             1                     10

1               1073    0.011311             1                     10

2               1073    0.037949             1                     10

3               1073    0.006726             1                     10

4               1073    0.030563             1                     10

...              ...         ...           ...                    ...

8319             670    0.000000             2                     10

8321             735    0.000000             2                     10

8323             619    0.000000             2                     27

8327             384    0.000000             2                     10

8330             964    0.000000             2                     48
```

```
      UserVar.branch_dest  UserVar.taskDetailStatus  UserVar.weight  \
0                      50                         3         0.13000   
1                      31                         3         0.01300   
2                      31                         3         0.03000   
3                      31                         3         0.00625   
4                      31                         3         0.03000   
...                   ...                       ...             ...   
8319                    3                        10         0.01000   
8321                    3                         3         0.54800   
8323                   30                        14         0.01000   
8327                   38                         3         0.01000   
8330                   23                        14         0.01000   

      UserVar.taskStatus  receiver_city_clean  distance(km)  \
0                      0                   85      0.103678   
1                      0                  765      0.063334   
2                      0                  765      0.062369   
3                      0                  765      0.045120   
4                      0                   31      0.058537   
...                  ...                  ...           ...   
8319                   0                  294      0.052226   
8321                   0                 1046      0.013027   
8323                   1                  569      0.012928   
8327                   0                  564      0.026297   
8330                   1                   75      0.013526   

      average_speed(kmph)  
0                0.756322  
1                0.057628  
2                0.043869  
3                0.016478  
4                0.094895  
...                   ...  
8319             0.103031  
8321             0.032641  
8323             0.066627  
8327             0.042768  
8330             0.177657  

[3880 rows x 11 columns]
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, 
test_size=0.2, random_state=0)

#Try 3 different models
lgbm = LGBMRegressor()
rfc = RandomForestRegressor()
xgb = XGBRegressor()
```

```
models = [lgbm, rfc, xgb]
models_name = ["LGBM","Random Forest", "XGBoost"]

#The result and performance of 3 different models
model_fitted = []
scoring_results=[]
scorings = ["neg_mean_absolute_error", "neg_root_mean_squared_error",
"r2"]

for model in models:
  score=[]
  for scoring in scorings:
    results = model_selection.cross_val_score(model, X_train, y_train,
cv=3, scoring=scoring)
    score.append(results.mean())
  scoring_results.append(score)

#Shows the score results
model_results = pd.DataFrame(scoring_results, columns=scorings,
index=models_name)
model_results
```

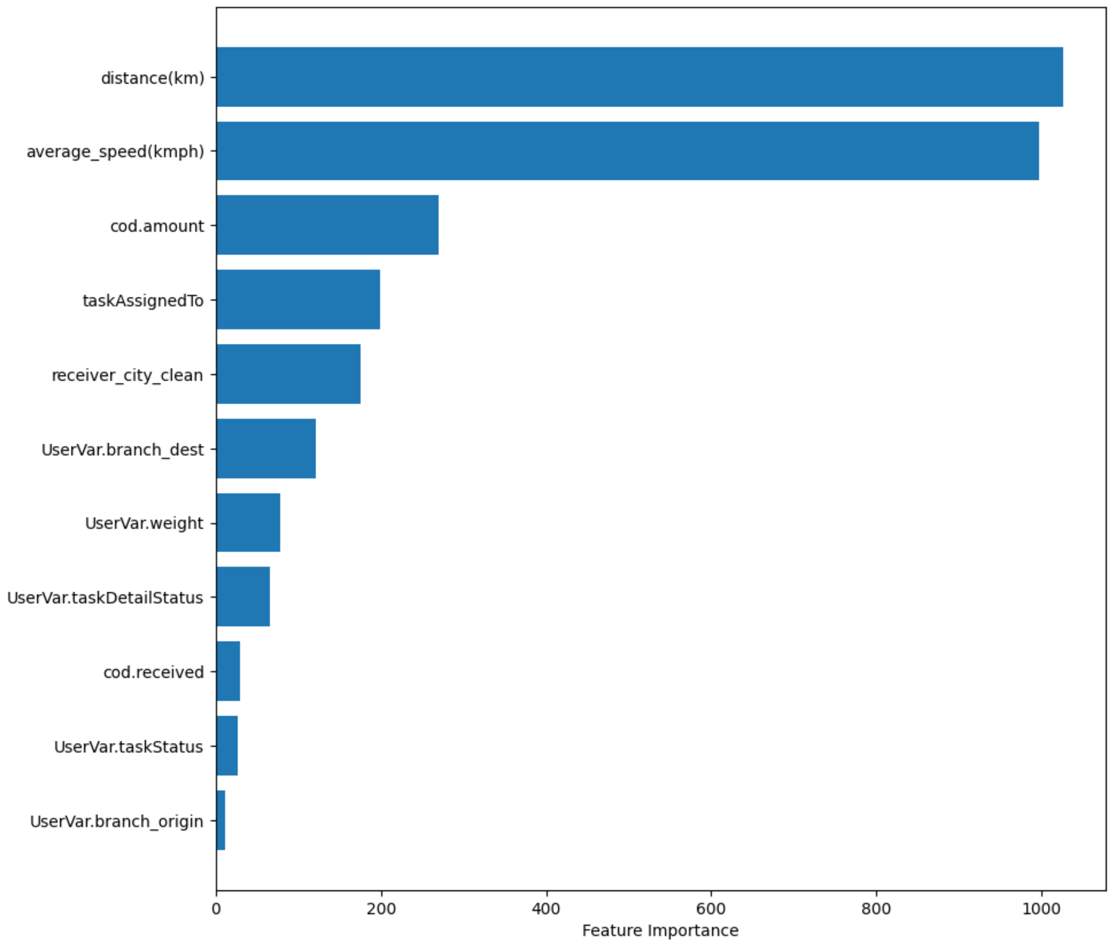
```
               neg_mean_absolute_error  neg_root_mean_squared_error
r2
LGBM                       -554.061547                 -1184.335853
0.972474
Random Forest              -531.376747                 -1289.838295
0.966317
XGBoost                    -560.811494                 -1187.510675
0.972266
```

```
#Train the best model
model =
models[models_name.index(model_results["neg_root_mean_squared_error"].
idxmax())].fit(X_train, y_train)
y_pred = model.predict(X_test)

print("The r2 score is: ", r2_score(y_test, y_pred))
print("The mean absolute error is: ", mean_absolute_error(y_test,
y_pred))
print("The root mean squared error is: ", (mean_squared_error(y_test,
y_pred)**0.5))
```

```
The r2 score is:  0.9750621764350887
The mean absolute error is:  541.6471271117142
The root mean squared error is:  1223.2556643690739
```

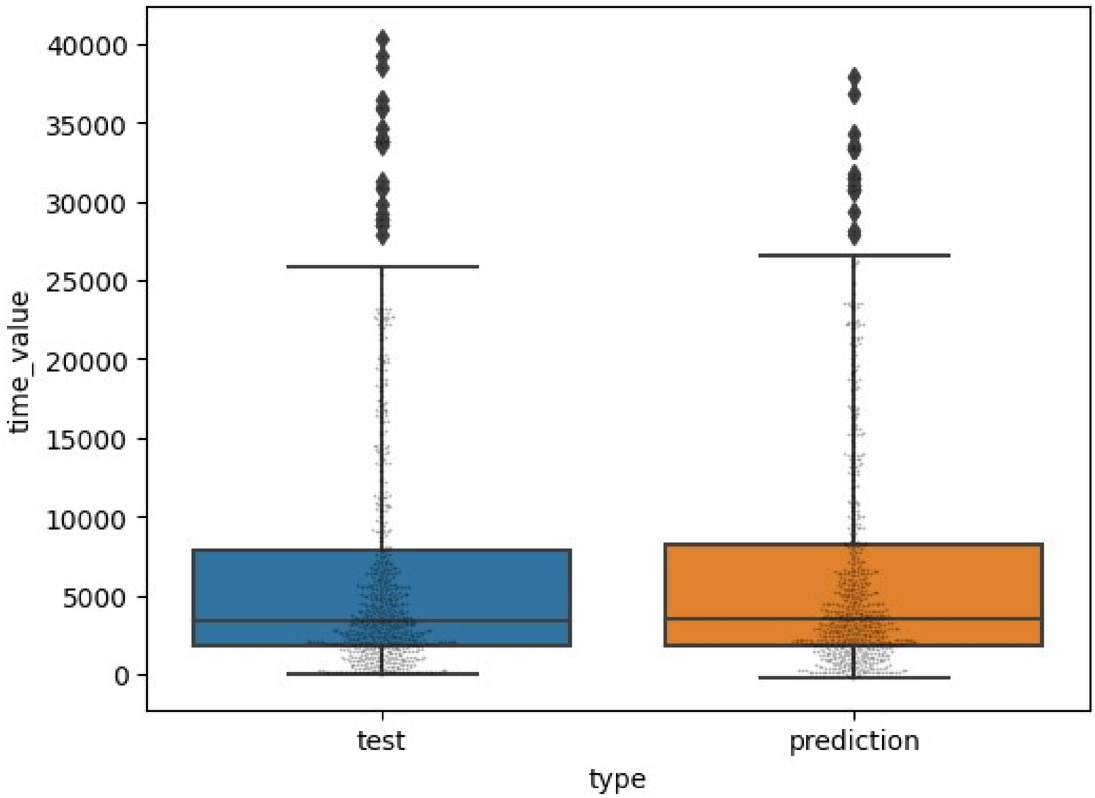
```
#Get the top 20 of most important features
sorted_idx = model.feature_importances_.argsort()
fig, ax = plt.subplots(figsize=(10, 10))
```

```
plt.barh(X.columns[sorted_idx][-20:],
model.feature_importances_[sorted_idx][-20:])
plt.xlabel("Feature Importance")
plt.show()
```




```
#Shows the difference of distributions between test and prediction
value
test_data=y_test.copy()
test_data=test_data.reset_index()
test_data['type']="test"
test_data.drop("index", axis=1, inplace=True)
test_data.columns=["time_value", "type"]

pred_data=pd.DataFrame(y_pred, columns=["time_value"])
pred_data["type"]="prediction"

pred_result=pd.concat([test_data, pred_data], axis=0,
ignore_index=True)
sns.boxplot(x="type", y="time_value", data=pred_result, whis=3.0)
sns.swarmplot(x="type", y="time_value", data=pred_result, size=1.2,
color="k", alpha=0.3)
```

```
<Axes: xlabel='type', ylabel='time_value'>
```



Fetching Data to BigQuery

```
df_all.columns = df_all.columns.str.replace('.', '_')
df_all.columns = df_all.columns.str.replace('(', '_')
df_all.columns = df_all.columns.str.replace(')', '')
```

```
<ipython-input-64-dc63bb670e90>:1: FutureWarning: The default value of
regex will change from True to False in a future version. In addition,
single character regular expressions will *not* be treated as literal
strings when regex=True.
  df_all.columns = df_all.columns.str.replace('.', '_')
<ipython-input-64-dc63bb670e90>:2: FutureWarning: The default value of
regex will change from True to False in a future version. In addition,
single character regular expressions will *not* be treated as literal
strings when regex=True.
  df_all.columns = df_all.columns.str.replace('(', '_')
<ipython-input-64-dc63bb670e90>:3: FutureWarning: The default value of
regex will change from True to False in a future version. In addition,
single character regular expressions will *not* be treated as literal
strings when regex=True.
  df_all.columns = df_all.columns.str.replace(')', '')
```

```
df_all
```

```
                 taskCreatedTime    taskAssignedTo
taskCompletedTime  \
0    2022-11-01 13:17:26+00:00     pacifiedLion0 2022-11-01
13:46:30+00:00
1    2022-11-01 01:41:07+00:00    peacefulTacos6 2022-11-01
05:33:48+00:00
2    2022-11-01 01:41:07+00:00    peacefulTacos6 2022-11-01
06:41:57+00:00
3    2022-11-01 01:41:07+00:00    peacefulTacos6 2022-11-01
11:18:19+00:00
4    2022-11-01 01:41:07+00:00    peacefulTacos6 2022-11-01
03:51:49+00:00
...                        ...               ...
...
8319 2022-11-10 00:50:16+00:00  grudgingBittern7 2022-11-10
02:37:41+00:00
8321 2022-11-10 01:13:30+00:00   humorousPiglet8 2022-11-10
02:38:02+00:00
8323 2022-11-10 01:56:48+00:00       giddyShads0 2022-11-10
02:37:58+00:00
8327 2022-11-10 00:27:51+00:00      dearWhiting2 2022-11-10
02:38:02+00:00
8330 2022-11-10 02:21:42+00:00     murkyThrushe3 2022-11-10
02:37:52+00:00

     taskStatus      flow                taskId  taskLocationDone_lon
\
0          done  Delivery  4fe3b237c832ca4841a2            109.762910

1          done  Delivery  08a4da25256affae8446            110.033986

2          done  Delivery  2ff0dc469826158b7684            109.999733

3          done  Delivery  331c172c2b383f774328            110.003708

4          done  Delivery  a9d53fa96c80baee8b23            110.013887

...         ...       ...                   ...                   ...

8319       done  Delivery  2bf6ce01d5b6a8ac8f34            107.899584

8321       done  Delivery  85f340c19c6cffd3135e            107.694447

8323       done  Delivery  abb2cc73275d23947762             98.736924

8327       done  Delivery  4df98016923e193d39ec            101.438664

8330       done  Delivery  5cc952d9e9f8066dbf24            110.352054
```

```
      taskLocationDone_lat  cod_amount cod_received  ...
UserVar_weight  \
0                -6.926608    685000.0         True  ...
13.000
1                -7.876154     53500.0         True  ...
1.300
2                -7.849777    179500.0         True  ...
3.000
3                -7.710998     31815.0         True  ...
0.625
4                -7.829742    144562.0         True  ...
3.000
...                    ...         ...          ...  ...             ..
.
8319             -7.089875         0.0       no COD  ...
1.000
8321             -6.924457         0.0       no COD  ...
54.800
8323              3.536418         0.0       no COD  ...
1.000
8327              0.479580         0.0       no COD  ...
1.000
8330             -7.892571         0.0       no COD  ...
1.000

     UserVar_branch_origin UserVar_taskStatus   receiver_city_clean  \
0                      CGK             COLF01       BATANG   BATANG
1                      CGK             COLF01   PURWODADI PURWOREJO
2                      CGK             COLF01   PURWODADI PURWOREJO
3                      CGK             COLF01   PURWODADI PURWOREJO
4                      CGK             COLF01     BAGELEN PURWOREJO
...                    ...                ...                   ...
8319                   CGK             COLF01                 GARUT
8321                   CGK             COLF01  UJUNGBERUNG  BANDUNG
8323                   MES             COLF02          MEDAN  MEDAN
8327                   CGK             COLF01  MARPOYAN DAMAI  PEKA
8330                   TGR             COLF02                BANTUL

          dest_city_name        dest_lat_city       dest_lon_city
distance_km  \
0              Semarang           -6.9903988         110.4229104
73.273671
1              Magelang          -7.51361445   110.2145132553504
44.768913
2              Magelang          -7.51361445   110.2145132553504
44.087177
3              Magelang          -7.51361445   110.2145132553504
31.900013
4              Magelang          -7.51361445   110.2145132553504
```

```
41.379771
...                   ...                    ...                  ...
...
8319       Bandung, Java           -6.9215529        107.6110212
36.920554
8321       Bandung, Java           -6.9215529        107.6110212
9.225213
8323      Medan, Sumatra            3.5896654         98.6738261
9.155145
8327  Pekanbaru, Sumatra   0.6111032000000001  101.54284256313278
18.600634
8330          Yogyakarta  -7.9778383999999996  110.36722565020224
9.577383

     time_diff_s average_speed_kmph
0         1744.0         151.252991
1        13961.0          11.544165
2        18050.0           8.793010
3        34632.0           3.316010
4         7842.0          18.996069
...          ...                ...
8319      6445.0          20.622808
8321      5072.0           6.547864
8323      2470.0          13.343532
8327      7811.0           8.572818
8330       970.0          35.544927

[3880 rows x 25 columns]
```

```
credentials =
service_account.Credentials.from_service_account_file('/content/latiha
n-345909-89e4eb39e2b1.json')

project_id = 'latihan-345909'
table_id = 'latihan-345909.tabel_apapun.mileapp_table'
client = bigquery.Client(credentials= credentials,project=project_id)
job_config = bigquery.LoadJobConfig(
    # Optionally, set the write disposition. BigQuery appends loaded
rows
    # to an existing table by default, but with WRITE_TRUNCATE write
    # disposition it replaces the table with the loaded data.
    write_disposition="WRITE_TRUNCATE",
)

job = client.load_table_from_dataframe(df_all, table_id,
job_config=job_config)
job.result()
# pandas_gbq.to_gbq(df_all, table_id, project_id=project_id,
if_exists='append')
```

```
LoadJob<project=latihan-345909, location=US, id=89af31c0-1569-440d-9d34-8b5ea2916ebf>
```

From BigQuery, I made the visualization using Google Data Studio. I chose Google Data Studio because it's free and reliable for real-time data.

You can check the dashboard here

https://lookerstudio.google.com/reporting/95f21f2b-ddfc-473a-8e93-80a8b3ce799c

##Conclusion

Based on the analysis above, we can see that the CGK branch is the centre for sending goods, and the PLM, CGK, SRG, BDO, and KOE branches are the branches that receive the most goods. From here, we can start optimizing the operations of important branches. Then, the routes of these branches can be optimized so as to speed up time and reduce the cost of sending goods to smaller branches.

Regarding the delivery of goods, we have conducted an analysis of the variables that most influence the delivery time. The result is the speed of delivery, distance, number of cod, and weight of goods. These four variables need further analysis for optimization so that travel time can be accelerated. Regarding the delivery of goods, we have conducted an analysis of the variables that most influence the delivery time. The result is the speed of delivery, distance, number of cod, and weight of goods. These four variables need further analysis for optimization so that travel time can be accelerated.