Import

In [ ]:
```python
import gc
import os
import datetime

import pandas as pd
import numpy as np
import plotly
import plotly.express as px
import plotly.graph_objects as go

from sklearn.preprocessing import LabelEncoder
import lightgbm as lgb

import requests
import time
from datetime import datetime, timedelta
from bs4 import BeautifulSoup
```

csv to parquet

메모리에 효율적인 데이터 유형을 사용하여 용량을 크게 줄이고 빠른 작업이 가능합니다.

In [ ]:
```
pip install fastparquet
```

In [2]:
```python
def csv_to_parquet(csv_path, save_name):
    df = pd.read_csv(csv_path)
    df.to_parquet(f"./{save_name}.parquet")
    del df
    gc.collect()
    print(save_name, "Done.")
```

In [3]:
```python
os.getcwd()
```

Out[3]:
```
'C:\\deep1\\User\\Untitled Folder'
```

In [4]:
```python
csv_to_parquet('./train.csv', 'train')
csv_to_parquet('./test.csv', 'test')
```

```
train Done.
test Done.
```

In [5]:
```python
train = pd.read_parquet('./train.parquet')
test = pd.read_parquet('./test.parquet')
```

EDA

In [6]:
```python
train.head()
```

Out[6]:

	id	base_date	day_of_week	base_hour	lane_count	road_rating	road_name	multi_linked	connect_code	maximum_speed_lim
0	TRAIN_0000000	20220623	목	17	1	106	지방도1112호선	0	0	60
1	TRAIN_0000001	20220728	목	21	2	103	일반국도11호선	0	0	60
2	TRAIN_0000002	20211010	일	7	2	103	일반국도16호선	0	0	80
3	TRAIN_0000003	20220311	금	13	2	107	태평로	0	0	50
4	TRAIN_0000004	20211005	화	8	2	103	일반국도12호선	0	0	80

5 rows × 23 columns

In [7]:
```python
pd.read_csv('./data_info.csv')
```

Out[7]:

	변수명	변수 설명
0	id	아이디
1	base_date	날짜
2	day_of_week	요일
3	base_hour	시간대
4	road_in_use	도로사용여부
5	lane_count	차로수
6	road_rating	도로등급
7	multi_linked	중용구간 여부
8	connect_code	연결로 코드
9	maximum_speed_limit	최고속도제한
10	weight_restricted	통과제한하중
11	hight_restricted	통과제한높이
12	road_type	도로유형
13	start_latitude	시작지점의 위도
14	start_longitude	시작지점의 경도
15	start_turn_restricted	시작 지점의 회전제한 유무
16	end_latitude	도착지점의 위도
17	end_longitude	도착지점의 경도
18	end_turn_restricted	도작지점의 회전제한 유무
19	road_name	도로명
20	start_node_name	시작지점명
21	end_node_name	도착지점명
22	vehicle_restricted	통과제한차량
23	target	평균속도(km)

In [8]:
```python
train.isnull().sum()   #null 값 존재 유무
```

Out[8]:
```
id                       0
base_date                0
day_of_week              0
base_hour                0
lane_count               0
road_rating              0
road_name                0
multi_linked             0
connect_code             0
maximum_speed_limit      0
vehicle_restricted       0
weight_restricted        0
height_restricted        0
road_type                0
start_node_name          0
start_latitude           0
start_longitude          0
start_turn_restricted    0
end_node_name            0
end_latitude             0
end_longitude            0
end_turn_restricted      0
target                   0
dtype: int64
```

In [9]:
```python
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4701217 entries, 0 to 4701216
Data columns (total 23 columns):
 #   Column                 Dtype
---  ------                 -----
 0   id                     object
 1   base_date              int64
 2   day_of_week            object
 3   base_hour              int64
 4   lane_count             int64
 5   road_rating            int64
 6   road_name              object
 7   multi_linked           int64
 8   connect_code           int64
 9   maximum_speed_limit    float64
 10  vehicle_restricted     float64
 11  weight_restricted      float64
 12  height_restricted      float64
 13  road_type              int64
 14  start_node_name        object
 15  start_latitude         float64
 16  start_longitude        float64
 17  start_turn_restricted  object
 18  end_node_name          object
 19  end_latitude           float64
 20  end_longitude          float64
 21  end_turn_restricted    object
 22  target                 float64
dtypes: float64(9), int64(7), object(7)
memory usage: 825.0+ MB
```

데이터전처리

In [10]:
```python
train["base_date"] = train["base_date"].apply(str)
test["base_date"] = test["base_date"].apply(str)
```

In [14]:
```python
test
```

Out[14]:

	id	base_date	day_of_week	base_hour	lane_count	road_rating	road_name	multi_linked	connect_code	maximum_speed_
0	TEST_000000	20220825	목	17	3	107	연삼로	0	0	
1	TEST_000001	20220809	화	12	2	103	일반국도12호선	0	0	
2	TEST_000002	20220805	금	2	1	103	일반국도16호선	0	0	
3	TEST_000003	20220818	목	23	3	103	일반국도11호선	0	0	
4	TEST_000004	20220810	수	17	3	106	번영로	0	0	
...	...	...	...	...	...	...	...	...	...	
291236	TEST_291236	20220827	토	5	1	103	일반국도16호선	0	0	
291237	TEST_291237	20220819	금	20	2	103	일반국도11호선	0	0	
291238	TEST_291238	20220805	금	11	1	107	신대로	0	0	
291239	TEST_291239	20220812	금	7	2	107	경찰로	0	0	
291240	TEST_291240	20220812	금	10	3	106	지방도1132호선	0	0	

291241 rows × 22 columns

In [18]:
```python
import datetime

train["base_date"]= [
 datetime.datetime.strptime(timestamp, "%Y%m%d") for timestamp in train["base_date"]
]

test["base_date"]= [
 datetime.datetime.strptime(timestamp2, "%Y%m%d") for timestamp2 in test["base_date"]
]
```

In [19]:
```python
test
```

Out[19]:

	id	base_date	day_of_week	base_hour	lane_count	road_rating	road_name	multi_linked	connect_code	maximum_speed_
0	TEST_000000	2022-08-25	목	17	3	107	연삼로	0	0	
1	TEST_000001	2022-08-09	화	12	2	103	일반국도12호선	0	0	
2	TEST_000002	2022-08-05	금	2	1	103	일반국도16호선	0	0	
3	TEST_000003	2022-08-18	목	23	3	103	일반국도11호선	0	0	
4	TEST_000004	2022-08-10	수	17	3	106	번영로	0	0	
...	...	...	...	...	...	...	...	...	...	
291236	TEST_291236	2022-08-27	토	5	1	103	일반국도16호선	0	0	
291237	TEST_291237	2022-08-19	금	20	2	103	일반국도11호선	0	0	
291238	TEST_291238	2022-08-05	금	11	1	107	신대로	0	0	
291239	TEST_291239	2022-08-12	금	7	2	107	경찰로	0	0	
291240	TEST_291240	2022-08-12	금	10	3	106	지방도1132호선	0	0	

291241 rows × 22 columns

In [20]:
```python
train["Year"]= train['base_date'].dt.year
train["Month"]= train['base_date'].dt.month
train["day"]= train['base_date'].dt.day

test["Year"]= test['base_date'].dt.year
test["Month"]= test['base_date'].dt.month
test["day"]= test['base_date'].dt.day
```

In [ ]:

In [156…
```
pip install holidays
```

```
Requirement already satisfied: holidays in c:\users\jjun6\anaconda3\lib\site-packages (0.16)Note: you may need t
o restart the kernel to use updated packages.
Requirement already satisfied: python-dateutil in c:\users\jjun6\anaconda3\lib\site-packages (from holidays) (2.
8.2)
Requirement already satisfied: hijri-converter in c:\users\jjun6\anaconda3\lib\site-packages (from holidays) (2.
2.4)
Requirement already satisfied: korean-lunar-calendar in c:\users\jjun6\anaconda3\lib\site-packages (from holiday
s) (0.3.1)
Requirement already satisfied: convertdate>=2.3.0 in c:\users\jjun6\anaconda3\lib\site-packages (from holidays)
(2.4.0)
Requirement already satisfied: pymeeus<=1,>=0.3.13 in c:\users\jjun6\anaconda3\lib\site-packages (from convertda
te>=2.3.0->holidays) (0.5.11)
Requirement already satisfied: six>=1.5 in c:\users\jjun6\anaconda3\lib\site-packages (from python-dateutil->hol
idays) (1.16.0)
```

In [21]:
```python
import holidays

# 한국 휴일 객체 생성
kr_holidays = holidays.KR()

# generate holiday table

train['base_date'] = sorted(train['base_date'])
test['base_date'] = sorted(test['base_date'])
```

In [158…
```python
kr_holidays
```

Out[158]:
```
holidays.country_holidays('KR')
```

In [22]:
```python
train['holiday'] = train.base_date.apply(lambda x: 'holiday' if x in kr_holidays else 'non-holiday')
test['holiday'] = test.base_date.apply(lambda x: 'holiday' if x in kr_holidays else 'non-holiday')
```

In [23]:
```python
str_col = ['day_of_week','start_turn_restricted','end_turn_restricted', 'holiday']
for i in str_col:
    le = LabelEncoder()
    le=le.fit(train[i])
    train[i]=le.transform(train[i])

    for label in np.unique(test[i]):
        if label not in le.classes_:
            le.classes_ = np.append(le.classes_, label)
    test[i]=le.transform(test[i])

train['day_of_week'].info()
```

```
<class 'pandas.core.series.Series'>
RangeIndex: 4701217 entries, 0 to 4701216
Series name: day_of_week
Non-Null Count    Dtype
--------------    -----
4701217 non-null  int32
dtypes: int32(1)
memory usage: 17.9 MB
```

In [39]:
```python
# 표준화
def data_standardization(x):
    x_np = np.asarray(x)
    return (x_np - x_np.mean()) / x_np.std()

# 너무 작거나 너무 큰 값이 학습을 방해하는 것을 방지하고자 정규화한다
# x가 양수라는 가정하에 최소값과 최대값을 이용하여 0~1사이의 값으로 변환
# Min-Max scaling
def min_max_scaling(x):
    x_np = np.asarray(x)
    return (x_np - x_np.min()) / (x_np.max() - x_np.min() + 1e-7) # 1e-7은 0으로 나누는 오류 예방차원

# 정규화된 값을 원래의 값으로 되돌린다
# 정규화하기 이전의 org_x값과 되돌리고 싶은 x를 입력하면 역정규화된 값을 리턴한다
def reverse_min_max_scaling(org_x, x):
    org_x_np = np.asarray(org_x)
    x_np = np.asarray(x)
    return (x_np * (org_x_np.max() - org_x_np.min() + 1e-7)) + org_x_np.min()
```

In [24]:
```python
y_train = train['target']

X_train = train.drop(['id','base_date', 'target','road_name', 'start_node_name', 'end_node_name','vehicle_restri

test = test.drop(['id','base_date', 'road_name', 'start_node_name', 'end_node_name','vehicle_restricted'], axis=

print(X_train.shape)
print(y_train.shape)
print(test.shape)
```

```
(4701217, 20)
(4701217,)
(291241, 20)
```

In [42]:
```python
norm_X_train = min_max_scaling(X_train) # 가격형태 데이터 정규화 처리
norm_test = min_max_scaling(test)
```

모델 선언 및 학습

In [25]:
```python
LR = lgb.LGBMRegressor(random_state=42).fit(X_train, y_train)
```

예측

In [26]:
```python
pred = LR.predict(test)
```

In [27]:
```python
sample_submission = pd.read_csv('./sample_submission.csv')
```

In [28]:
```python
sample_submission['target'] = pred
sample_submission.to_csv("./submit.csv", index = False)
```

In [29]:
```python
sample_submission
```

Out[29]:

	id	target
0	TEST_000000	27.183604
1	TEST_000001	43.829473
2	TEST_000002	59.772398
3	TEST_000003	36.681372
4	TEST_000004	37.398105
...	...	...
291236	TEST_291236	45.664110
291237	TEST_291237	51.887405
291238	TEST_291238	20.688816
291239	TEST_291239	25.357124
291240	TEST_291240	41.904254

291241 rows × 2 columns