```
import pandas as pd
import os
import numpy as np
import glob
import lightgbm as lgb
import pathlib
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
```

Reading and Processing CSV File

```
df = pd.read_csv ('Catalogue.csv', encoding='latin-1')
dfProcessed = pd.DataFrame(df, columns = ['Zaman (UTC)', 'Enlem', 'Boylam', 'Derinlik', 'Büyüklük'])
dfProcessed['Ay'] = pd.DatetimeIndex(df['Zaman (UTC)']).month
dfProcessed = dfProcessed.drop(columns="Zaman·(UTC)")
dfProcessed·=·dfProcessed[['Enlem',·'Boylam',·'Ay',·'Derinlik',·'Büyüklük']]
print(dfProcessed)
```

```
           Enlem   Boylam  Ay  Derinlik  Büyüklük
0        35.7380  24.8870   6     37.50       4.0
1        36.4261  27.0918   6     18.97       4.1
2        36.4461  27.1643   6     11.09       4.1
3        37.5625  36.1321   6     14.37       4.2
4        35.3220  32.8156   5     25.63       4.3
...          ...      ...  ..       ...       ...
12196    38.1200  31.2900   2     10.00       5.7
12197    37.8300  27.7100   1     10.00       4.4
12198    37.6300  37.3700   1     10.00       4.8
12199    39.5400  26.1400   1     10.00       5.2
12200    37.7900  28.2100   1     10.00       4.4

[12201 rows x 5 columns]
```

Dividing Data into 3 pieces as Train, Validation and Test data

```
r, c = dfProcessed.shape
print(r)
trainSize = int(0.7 * r)
validSize = int(0.1 * r)
trainLastIndex = trainSize
validLastIndex = trainLastIndex + validSize
dfTrain = dfProcessed.iloc[:trainLastIndex]
dfValid = dfProcessed.iloc[trainLastIndex+1:validLastIndex]
dfTest = dfProcessed.iloc[validLastIndex+1:]
```

```
     12201
```

```
dfTrainFeature = dfTrain.drop(columns="Büyüklük")
dfTrainTarget = dfTrain[['Büyüklük']]

dfValidFeature = dfValid.drop(columns="Büyüklük")
dfValidTarget = dfValid[['Büyüklük']]

dfTestFeature = dfTest.drop(columns="Büyüklük")
dfTestTarget = dfTest[['Büyüklük']]
```

Setting Parameters of Model

```
train_data = lgb.Dataset(dfTrainFeature, label=dfTrainTarget)
valid_data = lgb.Dataset(dfValidFeature, label=dfValidTarget)

parameters = {'objective': 'regression',
              'metric': 'root_mean_squared_error',
              'boosting': 'gbdt',
              'num_leaves': 63,
              'bagging_freq': 20,
             'learning_rate': 0.01,
             'verbose': -1
            }
```

Creating Model ve Training

```
model_lgbm = lgb.train(parameters,
                       train_data,
                       valid_sets=valid_data,
                       num_boost_round=1000,
                       early_stopping_rounds=50)
```

```
[231]   valid_0's rmse: 0.385039
[232]   valid_0's rmse: 0.385043
[233]   valid_0's rmse: 0.384999
[234]   valid_0's rmse: 0.384994
[235]   valid_0's rmse: 0.385007
[236]   valid_0's rmse: 0.384975
[237]   valid_0's rmse: 0.384989
[238]   valid_0's rmse: 0.385004
[239]   valid_0's rmse: 0.385001
[240]   valid_0's rmse: 0.385012
[241]   valid_0's rmse: 0.385029
[242]   valid_0's rmse: 0.385042
[243]   valid_0's rmse: 0.385037
[244]   valid_0's rmse: 0.385051
[245]   valid_0's rmse: 0.385063
[246]   valid_0's rmse: 0.385065
[247]   valid_0's rmse: 0.385073
[248]   valid_0's rmse: 0.385107
[249]   valid_0's rmse: 0.385121
[250]   valid_0's rmse: 0.385118
[251]   valid_0's rmse: 0.385126
[252]   valid_0's rmse: 0.385129
[253]   valid_0's rmse: 0.38508
[254]   valid_0's rmse: 0.385078
[255]   valid_0's rmse: 0.385076
[256]   valid_0's rmse: 0.385087
[257]   valid_0's rmse: 0.385086
```

```
[258]	valid_0's rmse: 0.385089
[259]	valid_0's rmse: 0.38509
[260]	valid_0's rmse: 0.385094
[261]	valid_0's rmse: 0.385099
[262]	valid_0's rmse: 0.385121
[263]	valid_0's rmse: 0.385145
[264]	valid_0's rmse: 0.385158
[265]	valid_0's rmse: 0.385164
[266]	valid_0's rmse: 0.385178
[267]	valid_0's rmse: 0.385195
[268]	valid_0's rmse: 0.385217
[269]	valid_0's rmse: 0.385211
[270]	valid_0's rmse: 0.38521
[271]	valid_0's rmse: 0.385226
[272]	valid_0's rmse: 0.385241
[273]	valid_0's rmse: 0.385221
[274]	valid_0's rmse: 0.385224
[275]	valid_0's rmse: 0.385201
[276]	valid_0's rmse: 0.385197
[277]	valid_0's rmse: 0.385211
[278]	valid_0's rmse: 0.385237
[279]	valid_0's rmse: 0.38525
[280]	valid_0's rmse: 0.385275
[281]	valid_0's rmse: 0.385304
[282]	valid_0's rmse: 0.385305
[283]	valid_0's rmse: 0.385307
[284]	valid_0's rmse: 0.385328
[285]	valid_0's rmse: 0.385351
[286]	valid_0's rmse: 0.385367
Early stopping, best iteration is:
[236]	valid_0's rmse: 0.384975
```

Saving and Loading Model

```
model_lgbm.save_model('lgbmModel.txt', num_iteration=model_lgbm.best_iteration)
```

```
loadedModel = lgb.Booster(model_file='lgbmModel.txt')
```

```
pred = loadedModel.predict(dfTestFeature,  num_iteration = loadedModel.best_iteration)
```

```
npTestVal = dfTestTarget.to_numpy().flatten()
```

Calculating RMS Error

```
from sklearn.metrics import mean_squared_error

rms = mean_squared_error(npTestVal, pred, squared=False)
print(rms)
```

```
    0.793554760418947
```

✓ 0 sn tamamlanma zamanı: 23:40

0 sn. tamamlanma zamanı: 23:16