

```
In [164]: from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"

import warnings
import logging
from re import sub
from functools import reduce
import boto3
import io

import numpy as np
import pandas as pd
import cudf
import cupy as cp

from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.metrics import mean_squared_error
from sklearn.model_selection import KFold
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import FunctionTransformer, OrdinalEncoder
from category_encoders import CatBoostEncoder
import preprocessing as pp

import xgboost as xgb
import catboost as cb
import lightgbm as lgb

import optuna
import joblib

import matplotlib.pyplot as plt
import plotly
import plotly.express as px
```

Global Settings

```
In [2]: seed = 12
rs = np.random.RandomState(seed)

plt.rcParams['figure.figsize'] = (12, 10)

warnings.filterwarnings("ignore")

plotly.offline.init_notebook_mode()

logging.getLogger('matplotlib.font_manager').setLevel(logging.ERROR)

# Model path
model_path = '../output/models/'
eval_path = '../output/evals/'
prep_path = '../output/preprocessors/'
```

S3

```
In [162]: s3 = boto3.client('s3')

AWS_S3_BUCKET = 'yang-ml-sagemaker'
```

Data

```
In [86]: train, test = pd.read_csv("../data/train_sanitized.csv"), pd.read_csv("../data/test_sanitized.csv")
train.shape, test.shape
```

```
Out[86]: ((338988, 32), (80000, 32))
```

```
In [87]: X_train, y_train = train.drop(['interest_rate'], axis=1), train.interest_rate.to_numpy()
X_test = test.drop(['interest_rate'], axis=1)
X_train.shape, X_test.shape, y_train.shape
```

```
Out[87]: ((338988, 31), (80000, 31), (338988,))
```

Pipeline Ingredients

We will use three boosted tree frameworks that support GPU training--- XGBoost, CatBoost, and LightGBM. Catboost supports categorical features out of the box while XGBoost and LightGBM have support for pandas or integer-encoded categorical features, respectively. Therefore, our preprocessing pipelines will be slightly different between these three frameworks. Nevertheless, the preprocessing workloads share similar ingredients such as imputation. We define them below so that they can be reused.

```
In [5]: # Numerical features
num_cols = ['loan_amt_requested', 'loan_amt_investor_funded_portion', 'borrower_annual_income', 'monthly_debt_to_income_ratio', 'num_of_past_dues', 'num_of_creditor_inquiries', 'num_of_months_since_delinquency', 'num_of_open_credit_lines', 'num_of_derog_public_rec', 'total_credit_rev_balance', 'rev_line_util_rate', 'total_credit_line']

# Categorical features
cat_cols = ['num_of_payment_months', 'loan_subgrade', 'num_of_years_employed', 'home_ownership_status', 'verified_income', 'loan_issued_date', 'borrower_provided_loan_category', 'zip_first_three', 'borrower_state', 'borrower_earliest_credit_open_date', 'init_loan_status']

# Categorical features to be encoded
encode_cols = ['num_of_payment_months', 'loan_subgrade', 'num_of_years_employed', 'home_ownership_status', 'verified_income', 'loan_issued_year', 'loan_issued_month', 'borrower_provided_loan_category', 'zip_first_three', 'borrower_state', 'borrower_earliest_credit_open_year', 'borrower_earliest_credit_open_month', 'init_loan_status']

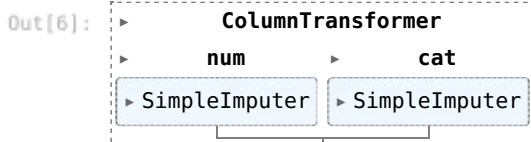
# Extracted date features
date_cols = ['loan_issued_year', 'loan_issued_month', 'borrower_earliest_credit_open_year', 'borrower_earliest_credit_open_month']
```

For imputation, the KNN imputation implementation from Sklearn is not really scalable to 338,988 rows. KNN using the kd-tree method generally has complexity $O(d N \log N)$; according to [this](#) issue, the sklearn implementation also involves $O(n^2)$ computations, which can further slow down the training time. Unfortunately, the `cuml` implementation of KNN imputer has not been released yet. Due to all of this, we will use the simple imputer for all of the missing features. Also, we are using ensemble models, and, according to the sklearn documentation:

In a prediction context, simple imputation usually performs poorly when associated with a weak learner. However, with a powerful learner, it can lead to as good or better performance than complex imputation such as IterativeImputer or KNNImputer.

With more computing resources, I may opt to try the KNN imputation model and see if the results are better. But for now, we will proceed as best as we could.

```
In [6]: imputers = ColumnTransformer([
    ('num', SimpleImputer(strategy='median').set_output(transform='pandas'), num_cols),
    ('cat', SimpleImputer(strategy='constant', fill_value='missing').set_output(transform='pandas'), cat_cols)
], remainder='drop').set_output(transform='pandas')
imputers
```



The feature engineering pieces are encapsulated in the `preprocessor.py` module. The following steps are carried out:

1. Extract year and month from the the two date features, creating four categorical features
2. For each category in each of the categorical features, create primitive aggregate features--- max, sum, mean, std--- of the numerical features. This creates $11 \times 11 \times 4 = 484$ numerical features in total.
3. The categorical feature will be handled differently:
 - For XGBoost, the categorical features will be encoded using `CatBoostEncoder`, which is an implementation of target encoding
 - For CatBoost, the categorical features will be handled natively
 - For LightGBM, the categorical features will be encoded using `OrdinalEncoder`, which will then be handled by the LightGBM internals

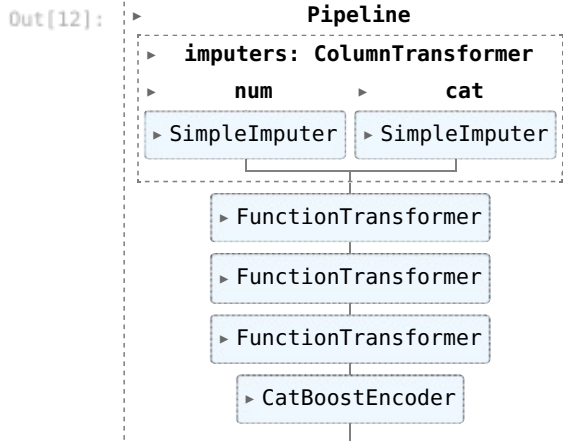
XGBoost

Pipeline

```
In [12]: xgboost_preprocessor = Pipeline([
    ('imputers', imputers),
    ('restore_cols', FunctionTransformer(pp.restore_columns)),
```

```
(('date_transformer', FunctionTransformer(pp.extract_date_features)),
 ('num_feat_eng', FunctionTransformer(pp.num_feat_eng)),
 ('cat_encoder', CatBoostEncoder(cols=encode_cols, handle_missing='value', handle_unknown='value'))) # Catboost
])
joblib.dump(xgboost_preprocessor, prep_path + 'xgboost_preprocessor.joblib')
xgboost_preprocessor
```

Out[12]: ['../output/preprocessors/xgboost_preprocessor.joblib']



Hyperparameter Search

The hyperparameter search will be carried out using Bayesian optimization, specifically, the Tree Parzen Estimator algorithm. Because we have limited compute budget where grid search can be hard to scale given the data size, we will use bayesian optimization, which generally requires fewer iterations to achieve acceptable results. In addition, we will use the implementation from **Optuna** rather than from **Hyperopt**. Optuna has more utilities and support for pruning. For all three frameworks, we will limit our budget to 20 trials.

```
In [43]: def objective_xgboost(trial):

    # Fold and seed
    train = pd.read_csv("../data/train_sanitized.csv")
    X_train, y_train = train.drop(['interest_rate'], axis=1), train.interest_rate.to_numpy()
    folds = 5
    seed = 1227

    # Parameters
    search_space = {
        # Booster parameters
        'booster_params': {
            'booster': 'gbtree',
            'objective': 'reg:squarederror',
            'eval_metric': 'rmse', # Use RMSE for evaluation metric on train and validation sets
            'learning_rate': trial.suggest_float(name='learning_rate', low=0.001, high=0.5), # Range: [0, 1
            'gamma': trial.suggest_int('gamma', 0, 20), # Range: [0, inf], the larger the more conservative
            'max_delta_step': trial.suggest_int('max_delta_step', 1, 10), # Range: [0, inf], values from 1-
            'lambda': trial.suggest_categorical('lambda', [10, 100, 500]), # Range: [0, inf], L2 regulariza
            'alpha': trial.suggest_categorical('alpha', [10, 100, 500]), # Range: [0, inf], L1 regularizati
            'colsample_bylevel': trial.suggest_categorical('colsample_bylevel', np.linspace(0.3, 1, 6).toli
            'colsample_bynode': trial.suggest_categorical('colsample_bynode', np.linspace(0.3, 1, 6).tolist
            'colsample_bytree': trial.suggest_categorical('colsample_bytree', np.linspace(0.3, 1, 6).tolist
            'subsample': trial.suggest_categorical('subsample', np.linspace(0.3, 1, 6).tolist()), # Range:
            'sampling_method': 'gradient_based', # Only supported for 'gpu_hist'
            'max_depth': trial.suggest_categorical('max_depth', np.arange(3, 12, dtype=np.int16).tolist()),
            'tree_method': 'gpu_hist',
            'predictor': 'gpu_predictor'
        },
        # Non-booster parameters
        'num_boost_round': trial.suggest_int('num_boost_round', low=500, high=2000, step=100), # Range: [0, inf
    }

    # K-fold cross validation
    kf = KFold(n_splits=folds, shuffle=True, random_state=rs)
    rmse_scores = np.empty(folds)

    for fold, (train_indx, val_indx) in enumerate(kf.split(X_train, y_train)):

        # Train and validation sets
        fold_X_train, fold_y_train = X_train.iloc[train_indx], y_train[train_indx]
        fold_X_val, fold_y_val = X_train.iloc[val_indx], y_train[val_indx]

        # Preprocessing using a fresh copy of the pipeline for every fold to prevent leakage
        preprocessor = joblib.load('../output/preprocessors/xgboost_preprocessor.joblib')
        print(f'Start processing fold {fold + 1}...')
        fold_X_train = preprocessor.fit_transform(fold_X_train, fold_y_train)
```

```

fold_X_val = preprocessor.transform(fold_X_val)

# Data for modeling
feature_names = fold_X_train.columns.tolist()
dtrain = xgb.DMatrix(data=fold_X_train, label=fold_y_train, feature_names=feature_names)
dvalid = xgb.DMatrix(data=fold_X_val, label=fold_y_val, feature_names=feature_names)

# Model
model = xgb.train(
    params=search_space['booster_params'],
    dtrain=dtrain,
    num_boost_round=search_space['num_boost_round'],
    early_stopping_rounds=200,
    evals=[(dtrain, 'train'), (dvalid, 'validate')],
    verbose_eval=200 # Print eval every 200 boosting rounds
)

# Out-of-fold prediction
print(f'Predicting for fold {fold + 1}...')
oof_pred = model.predict(data=dvalid)
rmse_scores[fold] = mean_squared_error(fold_y_val, oof_pred, squared=False) # Use RMSE

# Average across 5 folds
mean_rmse = np.mean(rmse_scores)

return mean_rmse

```

```

In [44]: study_xgboost = optuna.create_study(sampler=optuna.samplers.TPESampler(), study_name='min_rmse_xgboost', direction='minimize')
study_xgboost.optimize(objective_xgboost, n_trials=20)

```

[I 2023-02-12 08:57:23,065] A new study created in memory with name: min_rmse_xgboost

```

Start Processing fold 1...
[0]    train-rmse:13.61135    validate-rmse:13.61680
[200]  train-rmse:1.05672    validate-rmse:1.57748
[400]  train-rmse:1.05477    validate-rmse:1.57762
[412]  train-rmse:1.05477    validate-rmse:1.57762
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:13.61195    validate-rmse:13.61439
[200]  train-rmse:1.05722    validate-rmse:1.61548
[344]  train-rmse:1.05612    validate-rmse:1.61511
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:13.61381    validate-rmse:13.60694
[200]  train-rmse:1.05113    validate-rmse:1.61564
[400]  train-rmse:1.05002    validate-rmse:1.61536
[600]  train-rmse:1.04932    validate-rmse:1.61517
[800]  train-rmse:1.04828    validate-rmse:1.61422
[1000] train-rmse:1.04817    validate-rmse:1.61421
[1136] train-rmse:1.04817    validate-rmse:1.61421
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:13.60789    validate-rmse:13.63062
[200]  train-rmse:1.05917    validate-rmse:1.57633
[400]  train-rmse:1.05749    validate-rmse:1.57641
[476]  train-rmse:1.05641    validate-rmse:1.57640
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:13.61719    validate-rmse:13.59342
[200]  train-rmse:1.06073    validate-rmse:1.70401
[400]  train-rmse:1.05867    validate-rmse:1.70225
[600]  train-rmse:1.05851    validate-rmse:1.70222
[631]  train-rmse:1.05851    validate-rmse:1.70222

```

[I 2023-02-12 08:59:48,848] Trial 0 finished with value: 1.6171119920003556 and parameters: {'learning_rate': 0.1856901734449837, 'gamma': 6, 'max_delta_step': 3, 'lambda': 100, 'alpha': 500, 'colsample_bylevel': 0.8599999999999999, 'colsample_bynode': 0.8599999999999999, 'colsample_bytree': 0.8599999999999999, 'subsample': 0.3, 'max_depth': 11, 'num_boost_round': 1800}. Best is trial 0 with value: 1.6171119920003556.

```
Predicting for fold 5...
Start Processing fold 1...
[0]   train-rmse:13.78072   validate-rmse:13.77241
[200] train-rmse:1.09313   validate-rmse:1.72533
[350] train-rmse:1.09291   validate-rmse:1.72502
Predicting for fold 1...
Start Processing fold 2...
[0]   train-rmse:13.77274   validate-rmse:13.80433
[200] train-rmse:1.09583   validate-rmse:1.89320
[257] train-rmse:1.09534   validate-rmse:1.89275
Predicting for fold 2...
Start Processing fold 3...
[0]   train-rmse:13.77995   validate-rmse:13.77550
[200] train-rmse:1.09807   validate-rmse:1.89783
[257] train-rmse:1.09787   validate-rmse:1.89750
Predicting for fold 3...
Start Processing fold 4...
[0]   train-rmse:13.77471   validate-rmse:13.79647
[200] train-rmse:1.10069   validate-rmse:2.04729
[255] train-rmse:1.10063   validate-rmse:2.04720
Predicting for fold 4...
Start Processing fold 5...
[0]   train-rmse:13.78718   validate-rmse:13.74652
[200] train-rmse:1.10100   validate-rmse:1.78084
[255] train-rmse:1.10010   validate-rmse:1.77956
[I 2023-02-12 09:01:32,849] Trial 1 finished with value: 1.8684071040199797 and parameters: {'learning_rate': 0.38122271808122965, 'gamma': 8, 'max_delta_step': 1, 'lambda': 10, 'alpha': 500, 'colsample_bylevel': 0.58, 'colsample_bynode': 0.72, 'colsample_bytree': 1.0, 'subsample': 0.43999999999999995, 'max_depth': 5, 'num_boost_round': 1100}. Best is trial 0 with value: 1.6171119920003556.
Predicting for fold 5...
Start Processing fold 1...
[0]   train-rmse:13.75529   validate-rmse:13.76219
[200] train-rmse:1.00390   validate-rmse:2.05411
[251] train-rmse:1.00328   validate-rmse:2.05398
Predicting for fold 1...
Start Processing fold 2...
[0]   train-rmse:13.75912   validate-rmse:13.74684
[200] train-rmse:1.00619   validate-rmse:2.19115
[248] train-rmse:1.00614   validate-rmse:2.19121
Predicting for fold 2...
Start Processing fold 3...
[0]   train-rmse:13.76288   validate-rmse:13.73180
[200] train-rmse:0.99972   validate-rmse:2.22276
[249] train-rmse:0.99972   validate-rmse:2.22278
Predicting for fold 3...
Start Processing fold 4...
[0]   train-rmse:13.74981   validate-rmse:13.78405
[200] train-rmse:1.00156   validate-rmse:2.03566
[249] train-rmse:1.00156   validate-rmse:2.03553
Predicting for fold 4...
Start Processing fold 5...
[0]   train-rmse:13.75623   validate-rmse:13.75840
[200] train-rmse:1.00738   validate-rmse:2.20355
[247] train-rmse:1.00738   validate-rmse:2.20352
[I 2023-02-12 09:03:16,472] Trial 2 finished with value: 2.141384130449744 and parameters: {'learning_rate': 0.20242149604554308, 'gamma': 15, 'max_delta_step': 2, 'lambda': 10, 'alpha': 10, 'colsample_bylevel': 0.58, 'colsample_bynode': 0.43999999999999995, 'colsample_bytree': 1.0, 'subsample': 0.3, 'max_depth': 7, 'num_boost_round': 1000}. Best is trial 0 with value: 1.6171119920003556.
```

```

Predicting for fold 5...
Start Processing fold 1...
[0]    train-rmse:13.24581    validate-rmse:13.26646
[200]  train-rmse:1.09246    validate-rmse:2.84035
[218]  train-rmse:1.09246    validate-rmse:2.84036
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:13.25413    validate-rmse:13.23318
[200]  train-rmse:1.09991    validate-rmse:2.40889
[220]  train-rmse:1.09936    validate-rmse:2.40826
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:13.24187    validate-rmse:13.28218
[200]  train-rmse:1.09343    validate-rmse:2.86448
[221]  train-rmse:1.09166    validate-rmse:2.86735
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:13.25979    validate-rmse:13.21049
[200]  train-rmse:1.09940    validate-rmse:2.18569
[222]  train-rmse:1.09864    validate-rmse:2.18678
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:13.24811    validate-rmse:13.25729
[200]  train-rmse:1.08898    validate-rmse:2.37687

[221]  train-rmse:1.08887    validate-rmse:2.37672
[I 2023-02-12 09:04:56,413] Trial 3 finished with value: 2.5358941210218386 and parameters: {'learning_rate': 0.3135042098887452, 'gamma': 12, 'max_delta_step': 3, 'lambda': 100, 'alpha': 500, 'colsample_bylevel': 0.58, 'colsample_bynode': 0.3, 'colsample_bytree': 0.8599999999999999, 'subsample': 0.3, 'max_depth': 7, 'num_boost_round': 600}. Best is trial 0 with value: 1.6171119920003556.
Predicting for fold 5...
Start Processing fold 1...
[0]    train-rmse:13.71661    validate-rmse:13.72693
[200]  train-rmse:0.72238    validate-rmse:2.68698
[244]  train-rmse:0.72238    validate-rmse:2.68702
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:13.72157    validate-rmse:13.70705
[200]  train-rmse:0.74387    validate-rmse:2.48270
[242]  train-rmse:0.72794    validate-rmse:2.48599
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:13.71790    validate-rmse:13.72174
[200]  train-rmse:0.73526    validate-rmse:2.31127
[245]  train-rmse:0.73374    validate-rmse:2.30800
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:13.71782    validate-rmse:13.72206
[200]  train-rmse:0.73111    validate-rmse:2.07091
[247]  train-rmse:0.72284    validate-rmse:2.07096
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:13.71945    validate-rmse:13.71557
[200]  train-rmse:0.73229    validate-rmse:2.54331
[243]  train-rmse:0.73229    validate-rmse:2.54302
[I 2023-02-12 09:06:41,071] Trial 4 finished with value: 2.418971190006795 and parameters: {'learning_rate': 0.44492831493475143, 'gamma': 3, 'max_delta_step': 1, 'lambda': 10, 'alpha': 10, 'colsample_bylevel': 0.3, 'colsample_bynode': 0.72, 'colsample_bytree': 0.58, 'subsample': 0.43999999999999995, 'max_depth': 11, 'num_boost_round': 1700}. Best is trial 0 with value: 1.6171119920003556.

```

```
Predicting for fold 5...
Start Processing fold 1...
[0]   train-rmse:10.67927   validate-rmse:10.67258
[200] train-rmse:1.12128   validate-rmse:3.63684
[202] train-rmse:1.12128   validate-rmse:3.63684
Predicting for fold 1...
Start Processing fold 2...
[0]   train-rmse:10.67290   validate-rmse:10.71015
[200] train-rmse:1.11220   validate-rmse:3.23638
[203] train-rmse:1.11220   validate-rmse:3.23638
Predicting for fold 2...
Start Processing fold 3...
[0]   train-rmse:10.67315   validate-rmse:10.71594
[200] train-rmse:1.11511   validate-rmse:2.16465
[207] train-rmse:1.11511   validate-rmse:2.16465
Predicting for fold 3...
Start Processing fold 4...
[0]   train-rmse:10.68600   validate-rmse:10.65731
[200] train-rmse:1.12155   validate-rmse:2.48460
[205] train-rmse:1.12155   validate-rmse:2.48460
Predicting for fold 4...
Start Processing fold 5...
[0]   train-rmse:10.68350   validate-rmse:10.67180
[200] train-rmse:1.12486   validate-rmse:3.91663
[204] train-rmse:1.12486   validate-rmse:3.91663
```

[I 2023-02-12 09:08:18,153] Trial 5 finished with value: 3.0878222246594556 and parameters: {'learning_rate': 0.46512956426074153, 'gamma': 19, 'max_delta_step': 8, 'lambda': 10, 'alpha': 500, 'colsample_bylevel': 0.43999999999999995, 'colsample_bynode': 0.43999999999999995, 'colsample_bytree': 0.43999999999999995, 'subsample': 0.43999999999999995, 'max_depth': 9, 'num_boost_round': 1100}. Best is trial 0 with value: 1.6171119920003556.

```
Predicting for fold 5...
Start Processing fold 1...
[0]   train-rmse:11.66352   validate-rmse:11.67413
[200] train-rmse:1.02067   validate-rmse:1.88004
[399] train-rmse:1.02067   validate-rmse:1.88004
Predicting for fold 1...
Start Processing fold 2...
[0]   train-rmse:11.66751   validate-rmse:11.65813
[200] train-rmse:1.02251   validate-rmse:1.91466
[219] train-rmse:1.02251   validate-rmse:1.91466
Predicting for fold 2...
Start Processing fold 3...
[0]   train-rmse:11.66345   validate-rmse:11.67440
[200] train-rmse:1.02256   validate-rmse:1.83904
[241] train-rmse:1.02256   validate-rmse:1.83904
Predicting for fold 3...
Start Processing fold 4...
[0]   train-rmse:11.66044   validate-rmse:11.68641
[200] train-rmse:1.02750   validate-rmse:2.05087
[210] train-rmse:1.02750   validate-rmse:2.05085
Predicting for fold 4...
Start Processing fold 5...
[0]   train-rmse:11.67327   validate-rmse:11.63506
[200] train-rmse:1.02991   validate-rmse:1.73431
[241] train-rmse:1.02991   validate-rmse:1.73431
```

[I 2023-02-12 09:09:59,549] Trial 6 finished with value: 1.8837807095844283 and parameters: {'learning_rate': 0.3291632736936288, 'gamma': 16, 'max_delta_step': 8, 'lambda': 500, 'alpha': 10, 'colsample_bylevel': 0.43999999999999995, 'colsample_bynode': 0.72, 'colsample_bytree': 0.8599999999999999, 'subsample': 0.8599999999999999, 'max_depth': 11, 'num_boost_round': 1300}. Best is trial 0 with value: 1.6171119920003556.

```
Predicting for fold 5...
Start Processing fold 1...
[0]   train-rmse:11.00012   validate-rmse:11.09714
[200] train-rmse:1.10970   validate-rmse:2.44340
[207] train-rmse:1.10970   validate-rmse:2.44340
Predicting for fold 1...
Start Processing fold 2...
[0]   train-rmse:10.99569   validate-rmse:11.04706
[200] train-rmse:1.11477   validate-rmse:2.90475
[204] train-rmse:1.11477   validate-rmse:2.90475
Predicting for fold 2...
Start Processing fold 3...
[0]   train-rmse:10.99840   validate-rmse:11.03770
[200] train-rmse:1.12079   validate-rmse:2.90805
[205] train-rmse:1.12079   validate-rmse:2.90805
Predicting for fold 3...
Start Processing fold 4...
[0]   train-rmse:11.00727   validate-rmse:10.98143
[200] train-rmse:1.11034   validate-rmse:2.05557
[391] train-rmse:1.11034   validate-rmse:2.05557
Predicting for fold 4...
Start Processing fold 5...
[0]   train-rmse:11.00323   validate-rmse:11.07957
[200] train-rmse:1.11217   validate-rmse:4.59298

[203] train-rmse:1.11217   validate-rmse:4.59298
```

```
[I 2023-02-12 09:11:39,748] Trial 7 finished with value: 2.980947780569891 and parameters: {'learning_rate': 0.4211242619425227, 'gamma': 17, 'max_delta_step': 8, 'lambda': 100, 'alpha': 500, 'colsample_bylevel': 0.8599999999999999, 'colsample_bynode': 0.58, 'colsample_bytree': 0.3, 'subsample': 0.8599999999999999, 'max_depth': 6, 'num_boost_round': 1300}. Best is trial 0 with value: 1.6171119920003556.
```

```
Predicting for fold 5...
Start Processing fold 1...
[0]   train-rmse:13.46352   validate-rmse:13.42886
[200] train-rmse:1.10464   validate-rmse:1.73801
[275] train-rmse:1.10464   validate-rmse:1.73801
Predicting for fold 1...
Start Processing fold 2...
[0]   train-rmse:13.45282   validate-rmse:13.47169
[200] train-rmse:1.10295   validate-rmse:1.81770
[259] train-rmse:1.10295   validate-rmse:1.81769
Predicting for fold 2...
Start Processing fold 3...
[0]   train-rmse:13.45011   validate-rmse:13.48253
[200] train-rmse:1.09284   validate-rmse:1.62334
[264] train-rmse:1.09284   validate-rmse:1.62334
Predicting for fold 3...
Start Processing fold 4...
[0]   train-rmse:13.45606   validate-rmse:13.45875
[200] train-rmse:1.10297   validate-rmse:1.82276
[400] train-rmse:1.10297   validate-rmse:1.82276
[600] train-rmse:1.10297   validate-rmse:1.82275
[799] train-rmse:1.10297   validate-rmse:1.82275
Predicting for fold 4...
Start Processing fold 5...
[0]   train-rmse:13.46047   validate-rmse:13.44109
[200] train-rmse:1.09747   validate-rmse:1.72968
[248] train-rmse:1.09747   validate-rmse:1.72968
```

```
[I 2023-02-12 09:13:31,748] Trial 8 finished with value: 1.7462946607825853 and parameters: {'learning_rate': 0.360883123040774, 'gamma': 15, 'max_delta_step': 2, 'lambda': 10, 'alpha': 500, 'colsample_bylevel': 0.8599999999999999, 'colsample_bynode': 0.8599999999999999, 'colsample_bytree': 1.0, 'subsample': 0.8599999999999999, 'max_depth': 6, 'num_boost_round': 800}. Best is trial 0 with value: 1.6171119920003556.
```



```
Predicting for fold 5...
Start Processing fold 1...
[0]    train-rmse:13.32452    validate-rmse:13.32109
[200]  train-rmse:1.14399    validate-rmse:3.12260
[222]  train-rmse:1.13826    validate-rmse:3.11612
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:13.31993    validate-rmse:13.33944
[200]  train-rmse:1.14634    validate-rmse:3.70437
[220]  train-rmse:1.14292    validate-rmse:3.70234
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:13.32988    validate-rmse:13.29960
[200]  train-rmse:1.13775    validate-rmse:2.95273
[223]  train-rmse:1.13338    validate-rmse:2.95096
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:13.32106    validate-rmse:13.33492
[200]  train-rmse:1.13890    validate-rmse:2.23706
[230]  train-rmse:1.13468    validate-rmse:2.23927
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:13.32377    validate-rmse:13.32407
[200]  train-rmse:1.14207    validate-rmse:3.33409
[221]  train-rmse:1.13799    validate-rmse:3.33413
[I 2023-02-12 09:15:12,272] Trial 9 finished with value: 3.068565869817596 and parameters: {'learning_rate': 0.14370882609821814, 'gamma': 16, 'max_delta_step': 6, 'lambda': 10, 'alpha': 500, 'colsample_bylevel': 0.58, 'colsample_bynode': 0.3, 'colsample_bytree': 0.3, 'subsample': 1.0, 'max_depth': 6, 'num_boost_round': 1600}. Best is trial 0 with value: 1.6171119920003556.
```

```
Predicting for fold 5...
Start Processing fold 1...
[0]    train-rmse:14.09967    validate-rmse:14.09900
[200]  train-rmse:6.43725    validate-rmse:6.48721
[400]  train-rmse:2.15911    validate-rmse:2.44526
[600]  train-rmse:1.26720    validate-rmse:1.63678
[800]  train-rmse:1.14816    validate-rmse:1.56684
[1000] train-rmse:1.09222    validate-rmse:1.55436
[1200] train-rmse:1.05869    validate-rmse:1.55089
[1400] train-rmse:1.03783    validate-rmse:1.54949
[1600] train-rmse:1.02369    validate-rmse:1.54873
[1800] train-rmse:1.01210    validate-rmse:1.54798
[1999] train-rmse:1.00099    validate-rmse:1.54770
```

```
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:14.09903    validate-rmse:14.10158
[200]  train-rmse:6.43870    validate-rmse:6.49728
[400]  train-rmse:2.16483    validate-rmse:2.35226
[600]  train-rmse:1.27027    validate-rmse:1.56996
[800]  train-rmse:1.15194    validate-rmse:1.53106
[1000] train-rmse:1.09627    validate-rmse:1.53061
[1047] train-rmse:1.08703    validate-rmse:1.53070
```

```
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:14.10236    validate-rmse:14.08825
[200]  train-rmse:6.43878    validate-rmse:6.44853
[400]  train-rmse:2.16200    validate-rmse:2.46763
[600]  train-rmse:1.26955    validate-rmse:1.72867
[800]  train-rmse:1.14949    validate-rmse:1.67829
[1000] train-rmse:1.09276    validate-rmse:1.66919
[1200] train-rmse:1.05814    validate-rmse:1.66591
[1400] train-rmse:1.03746    validate-rmse:1.66451
[1600] train-rmse:1.02264    validate-rmse:1.66346
[1800] train-rmse:1.01074    validate-rmse:1.66215
[1999] train-rmse:1.00082    validate-rmse:1.66121
```

```
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:14.10089    validate-rmse:14.09414
[200]  train-rmse:6.43637    validate-rmse:6.48468
[400]  train-rmse:2.16122    validate-rmse:2.47474
[600]  train-rmse:1.27097    validate-rmse:1.69821
[800]  train-rmse:1.15170    validate-rmse:1.63152
[1000] train-rmse:1.09473    validate-rmse:1.61735
[1200] train-rmse:1.06066    validate-rmse:1.61297
[1400] train-rmse:1.04041    validate-rmse:1.61173
[1600] train-rmse:1.02587    validate-rmse:1.61092
[1800] train-rmse:1.01358    validate-rmse:1.61102
[1808] train-rmse:1.01313    validate-rmse:1.61095
```

```
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:14.09575    validate-rmse:14.11471
[200]  train-rmse:6.43442    validate-rmse:6.48507
[400]  train-rmse:2.16076    validate-rmse:2.42545
[600]  train-rmse:1.27132    validate-rmse:1.62423
[800]  train-rmse:1.15049    validate-rmse:1.55291
[1000] train-rmse:1.09377    validate-rmse:1.54547
[1200] train-rmse:1.05963    validate-rmse:1.54395
[1400] train-rmse:1.03920    validate-rmse:1.54343
[1600] train-rmse:1.02403    validate-rmse:1.54274
[1800] train-rmse:1.01173    validate-rmse:1.54221
[1999] train-rmse:1.00259    validate-rmse:1.54203
```

```
Predicting for fold 5...
[I 2023-02-12 09:22:18,304] Trial 10 finished with value: 1.5785182273533465 and parameters: {'learning_rate': 0.008732104299950916, 'gamma': 0, 'max_delta_step': 5, 'lambda': 100, 'alpha': 100, 'colsample_bylevel': 1.0, 'colsample_bynode': 0.8599999999999999, 'colsample_bytree': 0.72, 'subsample': 0.72, 'max_depth': 8, 'num_boost_round': 2000}. Best is trial 10 with value: 1.5785182273533465.
```

```
Start Processing fold 1...
[0]    train-rmse:14.13397    validate-rmse:14.10772
[200]  train-rmse:11.69814    validate-rmse:11.67440
[400]  train-rmse:9.35713     validate-rmse:9.34420
[600]  train-rmse:7.17455     validate-rmse:7.18802
[800]  train-rmse:5.25738     validate-rmse:5.30804
[1000] train-rmse:3.72695     validate-rmse:3.85967
[1200] train-rmse:2.60994     validate-rmse:2.87784
[1400] train-rmse:1.88710     validate-rmse:2.25312
[1600] train-rmse:1.48107     validate-rmse:1.85150
[1800] train-rmse:1.30284     validate-rmse:1.70586
[1999] train-rmse:1.22457     validate-rmse:1.65428
```

Predicting for fold 1...

```
Start Processing fold 2...
[0]    train-rmse:14.13140    validate-rmse:14.11805
[200]  train-rmse:11.69523    validate-rmse:11.68458
[400]  train-rmse:9.35365     validate-rmse:9.34726

[600]  train-rmse:7.17059     validate-rmse:7.17952
[800]  train-rmse:5.25251     validate-rmse:5.28027
[1000] train-rmse:3.72192     validate-rmse:3.80385
[1200] train-rmse:2.60603     validate-rmse:2.78167
[1400] train-rmse:1.88523     validate-rmse:2.14837
[1600] train-rmse:1.48138     validate-rmse:1.74928
[1800] train-rmse:1.30478     validate-rmse:1.61979
[1999] train-rmse:1.22768     validate-rmse:1.57558
```

Predicting for fold 2...

```
Start Processing fold 3...
[0]    train-rmse:14.12618    validate-rmse:14.13890
[200]  train-rmse:11.69021    validate-rmse:11.70422
[400]  train-rmse:9.34899     validate-rmse:9.37275
[600]  train-rmse:7.16640     validate-rmse:7.20733
[800]  train-rmse:5.24993     validate-rmse:5.33130
[1000] train-rmse:3.72042     validate-rmse:3.84747
[1200] train-rmse:2.60415     validate-rmse:2.80831
[1400] train-rmse:1.88346     validate-rmse:2.15211
[1600] train-rmse:1.47936     validate-rmse:1.75646
[1800] train-rmse:1.30224     validate-rmse:1.60716
[1999] train-rmse:1.22481     validate-rmse:1.55662
```

Predicting for fold 3...

```
Start Processing fold 4...
[0]    train-rmse:14.12377    validate-rmse:14.14855
[200]  train-rmse:11.68804    validate-rmse:11.71210
[400]  train-rmse:9.34734     validate-rmse:9.37441
[600]  train-rmse:7.16524     validate-rmse:7.21863
[800]  train-rmse:5.24835     validate-rmse:5.31440
[1000] train-rmse:3.71912     validate-rmse:3.83682
[1200] train-rmse:2.60242     validate-rmse:2.82909
[1400] train-rmse:1.88085     validate-rmse:2.19421
[1600] train-rmse:1.47571     validate-rmse:1.83438
[1800] train-rmse:1.29838     validate-rmse:1.69494
[1999] train-rmse:1.22099     validate-rmse:1.65127
```

Predicting for fold 4...

```
Start Processing fold 5...
[0]    train-rmse:14.12831    validate-rmse:14.13038
[200]  train-rmse:11.69260    validate-rmse:11.69542
[400]  train-rmse:9.35184     validate-rmse:9.36404
[600]  train-rmse:7.16980     validate-rmse:7.20008
[800]  train-rmse:5.25275     validate-rmse:5.30778
[1000] train-rmse:3.72252     validate-rmse:3.89208
[1200] train-rmse:2.60589     validate-rmse:2.93501
[1400] train-rmse:1.88338     validate-rmse:2.35941
[1600] train-rmse:1.47702     validate-rmse:2.02498
[1800] train-rmse:1.29859     validate-rmse:1.88268
[1999] train-rmse:1.22128     validate-rmse:1.81581
```

Predicting for fold 5...

[I 2023-02-12 09:29:51,801] Trial 11 finished with value: 1.65071199035213 and parameters: {'learning_rate': 0.003239647394147671, 'gamma': 0, 'max_delta_step': 4, 'lambda': 100, 'alpha': 100, 'colsample_bylevel': 1.0, 'colsample_bynode': 0.8599999999999999, 'colsample_bytree': 0.72, 'subsample': 0.72, 'max_depth': 8, 'num_boost_round': 2000}. Best is trial 10 with value: 1.5785182273533465.

```

Start Processing fold 1...
[0]    train-rmse:13.82508    validate-rmse:13.82588
[200]  train-rmse:1.03000    validate-rmse:1.75568
[289]  train-rmse:0.99708    validate-rmse:1.75600
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:13.82463    validate-rmse:13.82768
[200]  train-rmse:1.03744    validate-rmse:1.66573
[400]  train-rmse:0.98481    validate-rmse:1.66251
[600]  train-rmse:0.98336    validate-rmse:1.66232
[800]  train-rmse:0.98306    validate-rmse:1.66244
[851]  train-rmse:0.98281    validate-rmse:1.66242
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:13.82044    validate-rmse:13.84441
[200]  train-rmse:1.03329    validate-rmse:1.62814
[400]  train-rmse:0.97996    validate-rmse:1.62641
[555]  train-rmse:0.97872    validate-rmse:1.62649
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:13.82724    validate-rmse:13.81724
[200]  train-rmse:1.03335    validate-rmse:1.62623
[320]  train-rmse:0.98871    validate-rmse:1.62681
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:13.82881    validate-rmse:13.81096
[200]  train-rmse:1.03105    validate-rmse:1.62970
[400]  train-rmse:0.98302    validate-rmse:1.62161
[600]  train-rmse:0.98035    validate-rmse:1.62150
[800]  train-rmse:0.98012    validate-rmse:1.62151
[812]  train-rmse:0.98012    validate-rmse:1.62151

```

[I 2023-02-12 09:32:30,360] Trial 12 finished with value: 1.6586449952709361 and parameters: {'learning_rate': 0.06650579647187355, 'gamma': 5, 'max_delta_step': 5, 'lambda': 100, 'alpha': 100, 'colsample_bylevel': 1.0, 'colsample_bynode': 0.8599999999999999, 'colsample_bytree': 0.72, 'subsample': 0.58, 'max_depth': 8, 'num_boost_round': 1900}. Best is trial 10 with value: 1.5785182273533465.

```

Predicting for fold 5...
Start Processing fold 1...
[0]    train-rmse:13.52768    validate-rmse:13.52922
[200]  train-rmse:0.91626    validate-rmse:1.71135
[305]  train-rmse:0.84717    validate-rmse:1.71252
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:13.53375    validate-rmse:13.50493
[200]  train-rmse:0.91147    validate-rmse:1.59494
[400]  train-rmse:0.79016    validate-rmse:1.59515
[419]  train-rmse:0.78031    validate-rmse:1.59474
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:13.52694    validate-rmse:13.53219
[200]  train-rmse:0.91223    validate-rmse:1.55065
[325]  train-rmse:0.83144    validate-rmse:1.55146
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:13.52414    validate-rmse:13.54336
[200]  train-rmse:0.91433    validate-rmse:1.64927
[400]  train-rmse:0.79184    validate-rmse:1.64534
[600]  train-rmse:0.70449    validate-rmse:1.64487
[800]  train-rmse:0.64221    validate-rmse:1.64315
[1000] train-rmse:0.59846    validate-rmse:1.64214
[1200] train-rmse:0.56849    validate-rmse:1.64182
[1400] train-rmse:0.54747    validate-rmse:1.64119
[1599] train-rmse:0.53419    validate-rmse:1.64115
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:13.52743    validate-rmse:13.53020
[200]  train-rmse:0.91675    validate-rmse:1.57986
[400]  train-rmse:0.79683    validate-rmse:1.57523
[594]  train-rmse:0.71242    validate-rmse:1.57608

```

[I 2023-02-12 09:35:56,783] Trial 13 finished with value: 1.615193231760476 and parameters: {'learning_rate': 0.1292587786925063, 'gamma': 0, 'max_delta_step': 5, 'lambda': 100, 'alpha': 100, 'colsample_bylevel': 0.72, 'colsample_bynode': 1.0, 'colsample_bytree': 0.8599999999999999, 'subsample': 0.72, 'max_depth': 10, 'num_boost_round': 1600}. Best is trial 10 with value: 1.5785182273533465.

```

Predicting for fold 5...
Start Processing fold 1...
[0]    train-rmse:13.64363    validate-rmse:13.61703
[200]  train-rmse:1.01083    validate-rmse:1.75106
[400]  train-rmse:0.93613    validate-rmse:1.75049
[466]  train-rmse:0.91606    validate-rmse:1.74992
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:13.64031    validate-rmse:13.63034
[200]  train-rmse:1.01135    validate-rmse:1.58934
[400]  train-rmse:0.93676    validate-rmse:1.58142
[600]  train-rmse:0.87957    validate-rmse:1.57671
[800]  train-rmse:0.82624    validate-rmse:1.57548
[1000] train-rmse:0.78076    validate-rmse:1.57487
[1200] train-rmse:0.74160    validate-rmse:1.57518
[1217] train-rmse:0.73868    validate-rmse:1.57548
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:13.63727    validate-rmse:13.64248
[200]  train-rmse:1.00443    validate-rmse:1.68187
[400]  train-rmse:0.93131    validate-rmse:1.68137
[600]  train-rmse:0.87223    validate-rmse:1.68158
[636]  train-rmse:0.86274    validate-rmse:1.68197
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:13.63305    validate-rmse:13.65935
[200]  train-rmse:1.00994    validate-rmse:1.61762
[400]  train-rmse:0.93366    validate-rmse:1.61307
[600]  train-rmse:0.87659    validate-rmse:1.61144
[800]  train-rmse:0.82339    validate-rmse:1.61008
[1000] train-rmse:0.77861    validate-rmse:1.60944
[1146] train-rmse:0.74950    validate-rmse:1.60924
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:13.63731    validate-rmse:13.64235
[200]  train-rmse:1.00286    validate-rmse:1.67368
[400]  train-rmse:0.93024    validate-rmse:1.66789
[600]  train-rmse:0.87006    validate-rmse:1.66707
[800]  train-rmse:0.81866    validate-rmse:1.66716
[927]  train-rmse:0.79016    validate-rmse:1.66752
Predicting for fold 5...

```

```

[I 2023-02-12 09:40:12,960] Trial 14 finished with value: 1.6568194289521483 and parameters: {'learning_rate':
0.08829098394038953, 'gamma': 0, 'max_delta_step': 6, 'lambda': 500, 'alpha': 100, 'colsample_bylevel': 0.72, '
colsample_bynode': 1.0, 'colsample_bytree': 0.72, 'subsample': 0.72, 'max_depth': 10, 'num_boost_round': 1500}.
Best is trial 10 with value: 1.5785182273533465.

```

```
Start Processing fold 1...
[0]    train-rmse:13.99836    validate-rmse:13.97079
[200]  train-rmse:1.44716    validate-rmse:1.90520
[400]  train-rmse:1.09100    validate-rmse:1.73992
[600]  train-rmse:1.02769    validate-rmse:1.73405
[800]  train-rmse:0.99650    validate-rmse:1.73253
[1000] train-rmse:0.97375    validate-rmse:1.73128
[1200] train-rmse:0.95438    validate-rmse:1.72984
[1400] train-rmse:0.94050    validate-rmse:1.72942
[1586] train-rmse:0.93814    validate-rmse:1.72939
```

Predicting for fold 1...

```
Start Processing fold 2...
[0]    train-rmse:13.99727    validate-rmse:13.97427
[200]  train-rmse:1.44993    validate-rmse:1.74928
[400]  train-rmse:1.09366    validate-rmse:1.62826
[527]  train-rmse:1.04661    validate-rmse:1.62873
```

Predicting for fold 2...

```
Start Processing fold 3...
[0]    train-rmse:13.99479    validate-rmse:13.98535
[200]  train-rmse:1.44939    validate-rmse:1.83241
[400]  train-rmse:1.09481    validate-rmse:1.66864
[596]  train-rmse:1.03186    validate-rmse:1.66955
```

Predicting for fold 3...

```
Start Processing fold 4...
[0]    train-rmse:13.98809    validate-rmse:14.01071
[200]  train-rmse:1.44171    validate-rmse:1.94646
[400]  train-rmse:1.08872    validate-rmse:1.75135
[600]  train-rmse:1.02578    validate-rmse:1.74576
[800]  train-rmse:0.99501    validate-rmse:1.74246
[1000] train-rmse:0.97300    validate-rmse:1.73922
[1200] train-rmse:0.95423    validate-rmse:1.73802
[1400] train-rmse:0.94286    validate-rmse:1.73724
[1600] train-rmse:0.94077    validate-rmse:1.73711
[1800] train-rmse:0.94033    validate-rmse:1.73709
[1999] train-rmse:0.93989    validate-rmse:1.73708
```

Predicting for fold 4...

```
Start Processing fold 5...
[0]    train-rmse:13.98456    validate-rmse:14.02583
```

```
[200]  train-rmse:1.44892    validate-rmse:1.83310
[400]  train-rmse:1.09566    validate-rmse:1.64464
[600]  train-rmse:1.03255    validate-rmse:1.63514
[800]  train-rmse:1.00090    validate-rmse:1.63092
[1000] train-rmse:0.97833    validate-rmse:1.62819
[1200] train-rmse:0.95988    validate-rmse:1.62661
[1400] train-rmse:0.94657    validate-rmse:1.62583
[1600] train-rmse:0.94455    validate-rmse:1.62554
[1800] train-rmse:0.94432    validate-rmse:1.62552
[1999] train-rmse:0.94412    validate-rmse:1.62548
```

Predicting for fold 5...

[I 2023-02-12 09:45:15,308] Trial 15 finished with value: 1.6780456653941687 and parameters: {'learning_rate': 0.01594004130908005, 'gamma': 3, 'max_delta_step': 10, 'lambda': 100, 'alpha': 100, 'colsample_bylevel': 0.72, 'colsample_bynode': 1.0, 'colsample_bytree': 0.58, 'subsample': 0.72, 'max_depth': 10, 'num_boost_round': 2000}. Best is trial 10 with value: 1.5785182273533465.

```

Start Processing fold 1...
[0]   train-rmse:13.64987   validate-rmse:13.61743
[200] train-rmse:1.21326   validate-rmse:1.74837
[400] train-rmse:1.15099   validate-rmse:1.74096
[600] train-rmse:1.12435   validate-rmse:1.73884
[693] train-rmse:1.12395   validate-rmse:1.73888
Predicting for fold 1...
Start Processing fold 2...
[0]   train-rmse:13.63988   validate-rmse:13.65743
[200] train-rmse:1.21538   validate-rmse:1.76469
[400] train-rmse:1.14720   validate-rmse:1.75432
[600] train-rmse:1.12328   validate-rmse:1.74678
[732] train-rmse:1.12241   validate-rmse:1.74644
Predicting for fold 2...
Start Processing fold 3...
[0]   train-rmse:13.64567   validate-rmse:13.63428
[200] train-rmse:1.21526   validate-rmse:1.78725
[400] train-rmse:1.15200   validate-rmse:1.78055
[600] train-rmse:1.12697   validate-rmse:1.77846
[639] train-rmse:1.12670   validate-rmse:1.77827
Predicting for fold 3...
Start Processing fold 4...
[0]   train-rmse:13.63665   validate-rmse:13.67032
[200] train-rmse:1.21406   validate-rmse:1.74962
[400] train-rmse:1.14915   validate-rmse:1.74052
[600] train-rmse:1.12631   validate-rmse:1.73807
[734] train-rmse:1.12607   validate-rmse:1.73809
Predicting for fold 4...
Start Processing fold 5...
[0]   train-rmse:13.64488   validate-rmse:13.63744
[200] train-rmse:1.21593   validate-rmse:1.75158
[400] train-rmse:1.15045   validate-rmse:1.73365
[600] train-rmse:1.12742   validate-rmse:1.72628
[800] train-rmse:1.12649   validate-rmse:1.72606
[880] train-rmse:1.12649   validate-rmse:1.72607

```

[I 2023-02-12 09:47:51,934] Trial 16 finished with value: 1.7455499846270015 and parameters: {'learning_rate': 0.10487715415287856, 'gamma': 11, 'max_delta_step': 5, 'lambda': 100, 'alpha': 100, 'colsample_bylevel': 1.0, 'colsample_bynode': 1.0, 'colsample_bytree': 0.43999999999999995, 'subsample': 0.72, 'max_depth': 3, 'num_boost_round': 1500}. Best is trial 10 with value: 1.5785182273533465.

```

Predicting for fold 5...
Start Processing fold 1...
[0]   train-rmse:12.55862   validate-rmse:12.58956
[200] train-rmse:1.08693   validate-rmse:1.72481
[400] train-rmse:1.05506   validate-rmse:1.71992
[493] train-rmse:1.05506   validate-rmse:1.71992
Predicting for fold 1...
Start Processing fold 2...
[0]   train-rmse:12.56958   validate-rmse:12.54575
[200] train-rmse:1.09197   validate-rmse:1.81194
[400] train-rmse:1.06172   validate-rmse:1.81610
[435] train-rmse:1.06023   validate-rmse:1.81894
Predicting for fold 2...
Start Processing fold 3...
[0]   train-rmse:12.57165   validate-rmse:12.53745
[200] train-rmse:1.09326   validate-rmse:1.83965
[221] train-rmse:1.08471   validate-rmse:1.83965
Predicting for fold 3...
Start Processing fold 4...
[0]   train-rmse:12.55830   validate-rmse:12.59084
[200] train-rmse:1.08223   validate-rmse:1.78730
[293] train-rmse:1.05984   validate-rmse:1.78973
Predicting for fold 4...
Start Processing fold 5...
[0]   train-rmse:12.56593   validate-rmse:12.56038
[200] train-rmse:1.08902   validate-rmse:1.80942
[400] train-rmse:1.05828   validate-rmse:1.80495
[481] train-rmse:1.05687   validate-rmse:1.80525

```

[I 2023-02-12 09:49:52,479] Trial 17 finished with value: 1.7946984871997351 and parameters: {'learning_rate': 0.2384070959457482, 'gamma': 3, 'max_delta_step': 7, 'lambda': 500, 'alpha': 100, 'colsample_bylevel': 0.72, 'colsample_bynode': 0.58, 'colsample_bytree': 0.8599999999999999, 'subsample': 0.72, 'max_depth': 4, 'num_boost_round': 1800}. Best is trial 10 with value: 1.5785182273533465.

```

Predicting for fold 5...
Start Processing fold 1...
[0]    train-rmse:13.90105    validate-rmse:13.91708
[200]  train-rmse:1.06363    validate-rmse:2.27384
[279]  train-rmse:1.02663    validate-rmse:2.26951
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:13.90115    validate-rmse:13.91668
[200]  train-rmse:1.06231    validate-rmse:2.22709
[284]  train-rmse:1.01903    validate-rmse:2.21955
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:13.91049    validate-rmse:13.87930
[200]  train-rmse:1.06803    validate-rmse:2.29854
[280]  train-rmse:1.02817    validate-rmse:2.28679
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:13.90223    validate-rmse:13.91235
[200]  train-rmse:1.05879    validate-rmse:1.97315
[292]  train-rmse:1.01672    validate-rmse:1.96600
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:13.90636    validate-rmse:13.89582
[200]  train-rmse:1.05849    validate-rmse:1.77650
[307]  train-rmse:1.01203    validate-rmse:1.77131
[I 2023-02-12 09:51:56,795] Trial 18 finished with value: 2.102634994926789 and parameters: {'learning_rate': 0.06231353236268271, 'gamma': 0, 'max_delta_step': 4, 'lambda': 100, 'alpha': 100, 'colsample_bylevel': 0.3, 'colsample_bynode': 1.0, 'colsample_bytree': 0.72, 'subsample': 1.0, 'max_depth': 8, 'num_boost_round': 1500}. Best is trial 10 with value: 1.5785182273533465.
Predicting for fold 5...
Start Processing fold 1...
[0]    train-rmse:12.73862    validate-rmse:12.78280
[200]  train-rmse:0.98532    validate-rmse:1.63571
[400]  train-rmse:0.98198    validate-rmse:1.63494
[448]  train-rmse:0.98198    validate-rmse:1.63494
Predicting for fold 1...
Start Processing fold 2...
[0]    train-rmse:12.74389    validate-rmse:12.75112
[200]  train-rmse:0.98627    validate-rmse:1.75819
[279]  train-rmse:0.98495    validate-rmse:1.75818
Predicting for fold 2...
Start Processing fold 3...
[0]    train-rmse:12.74078    validate-rmse:12.76740
[200]  train-rmse:0.98417    validate-rmse:1.68106
[400]  train-rmse:0.98410    validate-rmse:1.68106
[444]  train-rmse:0.98410    validate-rmse:1.68106
Predicting for fold 3...
Start Processing fold 4...
[0]    train-rmse:12.74427    validate-rmse:12.75129
[200]  train-rmse:0.98263    validate-rmse:1.75061
[400]  train-rmse:0.98259    validate-rmse:1.75049
[600]  train-rmse:0.98254    validate-rmse:1.75050
[605]  train-rmse:0.98254    validate-rmse:1.75050
Predicting for fold 4...
Start Processing fold 5...
[0]    train-rmse:12.74733    validate-rmse:12.72373
[200]  train-rmse:0.98971    validate-rmse:1.57522
[400]  train-rmse:0.98894    validate-rmse:1.57522
[442]  train-rmse:0.98894    validate-rmse:1.57523
[I 2023-02-12 09:54:03,190] Trial 19 finished with value: 1.6799802107072614 and parameters: {'learning_rate': 0.15069330553741483, 'gamma': 7, 'max_delta_step': 10, 'lambda': 100, 'alpha': 100, 'colsample_bylevel': 0.72, 'colsample_bynode': 1.0, 'colsample_bytree': 0.72, 'subsample': 0.58, 'max_depth': 10, 'num_boost_round': 1700}. Best is trial 10 with value: 1.5785182273533465.
Predicting for fold 5...

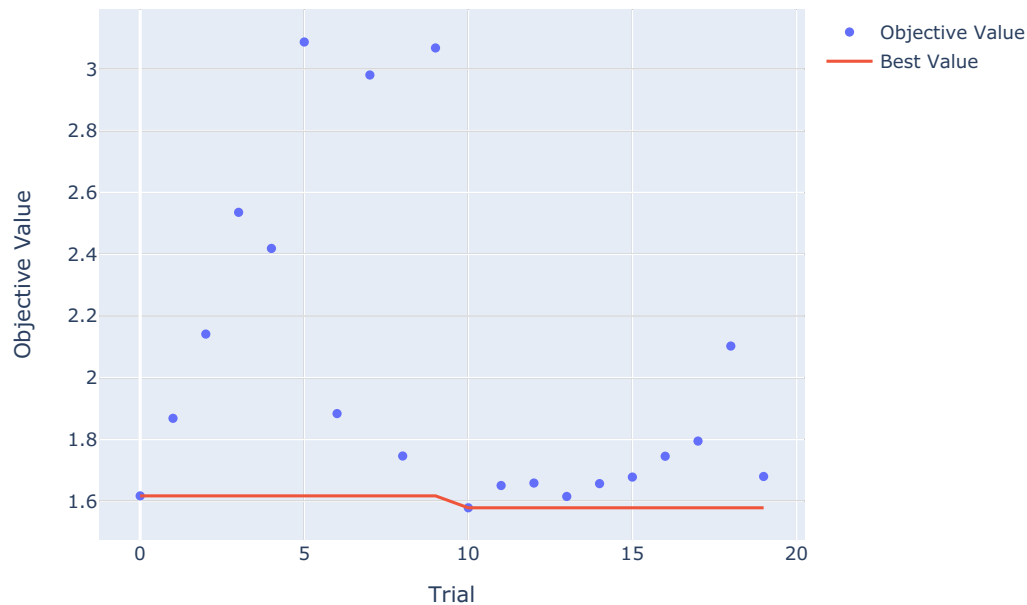
```

```

In [53]: fig_xgboost = optuna.visualization.plot_optimization_history(study_xgboost)
fig_xgboost.show();

```


Optimization History Plot



Model Training

The set of parameters that resulted in the lowest RMSE is as follows:

```
In [54]: study_xgboost.best_params
```

```
Out[54]: {'learning_rate': 0.008732104299950916,
'gamma': 0,
'max_delta_step': 5,
'lambda': 100,
'alpha': 100,
'colsample_bylevel': 1.0,
'colsample_bynode': 0.8599999999999999,
'colsample_bytree': 0.72,
'subsample': 0.72,
'max_depth': 8,
'num_boost_round': 2000}
```

We now train the model:

```
In [118]: # Out-of-fold prediction dictionary
oof_xgboost = {}
# Feature importance container
feat_imp_xgboost = []
# K-fold cross validation
kf_xgboost = KFold(n_splits=5, shuffle=True, random_state=rs)

for fold, (train_indx, val_indx) in enumerate(kf_xgboost.split(X_train, y_train)):
    # Train and validation sets
    fold_X_train, fold_y_train = X_train.iloc[train_indx], y_train[train_indx]
    fold_X_val, fold_y_val = X_train.iloc[val_indx], y_train[val_indx]

    # Preprocessing using fresh copy of the pipeline for every fold
    preprocessor = joblib.load('../output/preprocessors/xgboost_preprocessor.joblib')
    print(f'Start processing fold {fold + 1}...')
    fold_X_train = preprocessor.fit_transform(fold_X_train, fold_y_train)
    fold_X_val = preprocessor.transform(fold_X_val)
    # Write fitted preprocessor to disk
    joblib.dump(preprocessor, model_path + f'xgboost/preprocessor_fold_{fold + 1}.joblib')

    # Data for modeling
    feature_names = fold_X_train.columns.tolist()
    dtrain = xgb.DMatrix(data=fold_X_train, label=fold_y_train, feature_names=feature_names)
    dvalid = xgb.DMatrix(data=fold_X_val, label=fold_y_val, feature_names=feature_names)

    # Model
    evals_result = {}
    model = xgb.train(
        params={'learning_rate': 0.009,
                'gamma': 0,
```

```

        'max_delta_step': 5,
        'lambda': 100,
        'alpha': 100,
        'colsample_bylevel': 1,
        'colsample_bynode': 0.8599999999999999,
        'colsample_bytree': 0.72,
        'subsample': 0.72,
        'max_depth': 8,
        'sampling_method': 'gradient_based',
        'tree_method': 'gpu_hist',
        'predictor': 'gpu_predictor'},
    dtrain=dtrain,
    num_boost_round=study_xgboost.best_params['num_boost_round'],
    early_stopping_rounds=200,
    evals=[(dtrain, 'train'), (dvalid, 'validate')],
    evals_result=evals_result,
    verbose_eval=200 # Print eval every 200 boosting rounds
)
model.save_model(model_path + f'xgboost/model_fold_{fold + 1}.xgb')
joblib.dump(evals_result, model_path + f'xgboost/eval_fold_{fold + 1}.joblib')

# Feature importance for top 20 features for the current fold
# The booster object has a get_score method that returns a dictionary of feature names and their importance
feat_imp = model.get_score(importance_type='weight')
df = pd.DataFrame({'feature': feat_imp.keys(), f'importance_{fold + 1}': feat_imp.values()})
feat_imp_xgboost.append(df)

# Predictions
print(f'predicting for fold {fold + 1}...')
oof_pred = model.predict(data=dvalid)
oof_xgboost[f'fold_{fold + 1}'] = {'target': fold_y_val, 'predictions': oof_pred}

del dtrain, dvalid, preprocessor, model, evals_result, feat_imp, df, oof_pred

```

Start processing fold 1...

Out[118]: ['../output/models/xgboost/preprocessor_fold_1.joblib']

```

[0]    train-rmse:14.10195    validate-rmse:14.08351
[200]  train-rmse:6.24335    validate-rmse:6.29366
[400]  train-rmse:2.03898    validate-rmse:2.42746
[600]  train-rmse:1.25203    validate-rmse:1.72568
[800]  train-rmse:1.14223    validate-rmse:1.64578
[1000] train-rmse:1.08700    validate-rmse:1.63519
[1200] train-rmse:1.05480    validate-rmse:1.63334
[1400] train-rmse:1.03457    validate-rmse:1.63284
[1600] train-rmse:1.02108    validate-rmse:1.63264
[1800] train-rmse:1.00929    validate-rmse:1.63254
[1999] train-rmse:0.99884    validate-rmse:1.63293

```

Out[118]: ['../output/models/xgboost/eval_fold_1.joblib']

predicting for fold 1...

Start processing fold 2...

Out[118]: ['../output/models/xgboost/preprocessor_fold_2.joblib']

```

[0]    train-rmse:14.09135    validate-rmse:14.12589
[200]  train-rmse:6.23148    validate-rmse:6.31004
[400]  train-rmse:2.03349    validate-rmse:2.31435
[600]  train-rmse:1.25053    validate-rmse:1.64104
[800]  train-rmse:1.14159    validate-rmse:1.61230
[1000] train-rmse:1.08716    validate-rmse:1.60911
[1098] train-rmse:1.06923    validate-rmse:1.60915

```

Out[118]: ['../output/models/xgboost/eval_fold_2.joblib']

predicting for fold 2...

Start processing fold 3...

Out[118]: ['../output/models/xgboost/preprocessor_fold_3.joblib']

```

[0]    train-rmse:14.10141    validate-rmse:14.08567
[200]  train-rmse:6.24173    validate-rmse:6.25762
[400]  train-rmse:2.03730    validate-rmse:2.29337
[600]  train-rmse:1.25080    validate-rmse:1.66738
[800]  train-rmse:1.14343    validate-rmse:1.62506
[1000] train-rmse:1.08942    validate-rmse:1.62046
[1200] train-rmse:1.05792    validate-rmse:1.62104
[1212] train-rmse:1.05646    validate-rmse:1.62110

```

Out[118]: ['../output/models/xgboost/eval_fold_3.joblib']

predicting for fold 3...

Start processing fold 4...

Out[118]: ['../output/models/xgboost/preprocessor_fold_4.joblib']

```
[0]    train-rmse:14.09775    validate-rmse:14.10034
[200]  train-rmse:6.23707    validate-rmse:6.27878
[400]  train-rmse:2.03466    validate-rmse:2.28574
[600]  train-rmse:1.24949    validate-rmse:1.60826
[800]  train-rmse:1.14053    validate-rmse:1.55816
[1000] train-rmse:1.08597    validate-rmse:1.54573
[1200] train-rmse:1.05402    validate-rmse:1.54003
[1400] train-rmse:1.03457    validate-rmse:1.53832
[1600] train-rmse:1.02071    validate-rmse:1.53708
[1800] train-rmse:1.00923    validate-rmse:1.53580
[1999] train-rmse:0.99950    validate-rmse:1.53528
```

```
Out[118]: ['../output/models/xgboost/eval_fold_4.joblib']
```

predicting for fold 4...
Start processing fold 5...

```
Out[118]: ['../output/models/xgboost/preprocessor_fold_5.joblib']
```

```
[0]    train-rmse:14.09886    validate-rmse:14.09588
[200]  train-rmse:6.23780    validate-rmse:6.32863
[400]  train-rmse:2.03405    validate-rmse:2.35200
[600]  train-rmse:1.24865    validate-rmse:1.57282
[800]  train-rmse:1.13978    validate-rmse:1.51728
[1000] train-rmse:1.08546    validate-rmse:1.51096
[1200] train-rmse:1.05330    validate-rmse:1.50851
[1400] train-rmse:1.03444    validate-rmse:1.50786
[1600] train-rmse:1.02027    validate-rmse:1.50699
[1800] train-rmse:1.00859    validate-rmse:1.50684
[1999] train-rmse:0.99859    validate-rmse:1.50644
```

```
Out[118]: ['../output/models/xgboost/eval_fold_5.joblib']
```

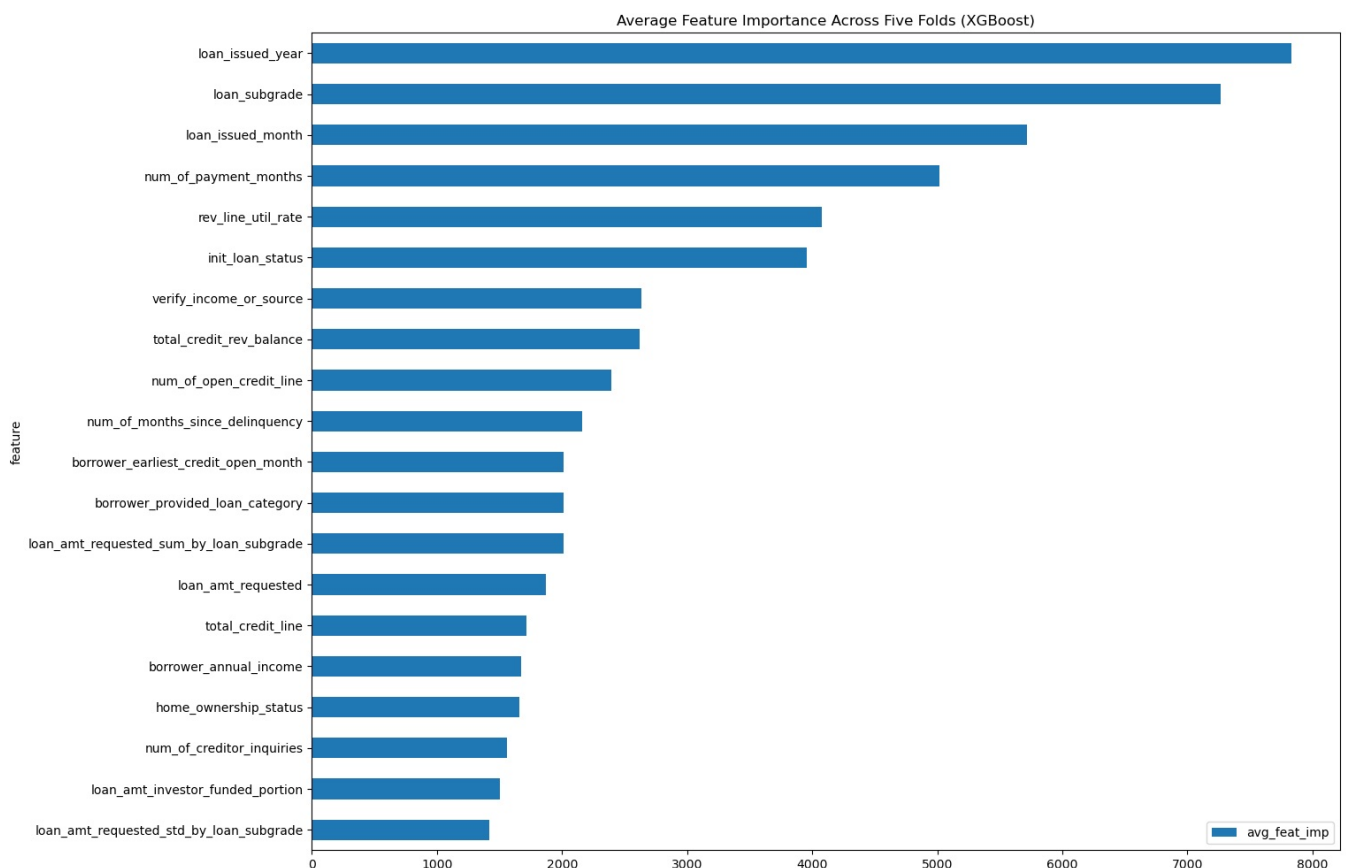
predicting for fold 5...

Features Importance

Feature importance can be visualized as follows:

```
In [119]: # Join feature importance
feat_imp_xgboost = reduce(lambda x, y: pd.merge(x, y, on='feature', how='left'), feat_imp_xgboost)
feat_imp_xgboost['avg_feat_imp'] = feat_imp_xgboost.iloc[:, 1:].apply(lambda row: row.mean(), axis=1)

# Plot top feature importance
feat_imp_xgboost.sort_values(by='avg_feat_imp', ascending=True).iloc[-20:].plot(
    kind='barh', x='feature', y='avg_feat_imp',
    figsize=(15, 12),
    title='Average Feature Importance Across Five Folds (XGBoost)'
)
plt.show();
```



A few of these features are generated; it appears that subgrade is one of the most important categorical features. Interestingly, the year

and month in which the loans were issued have strong predictive power.

Learning Curves

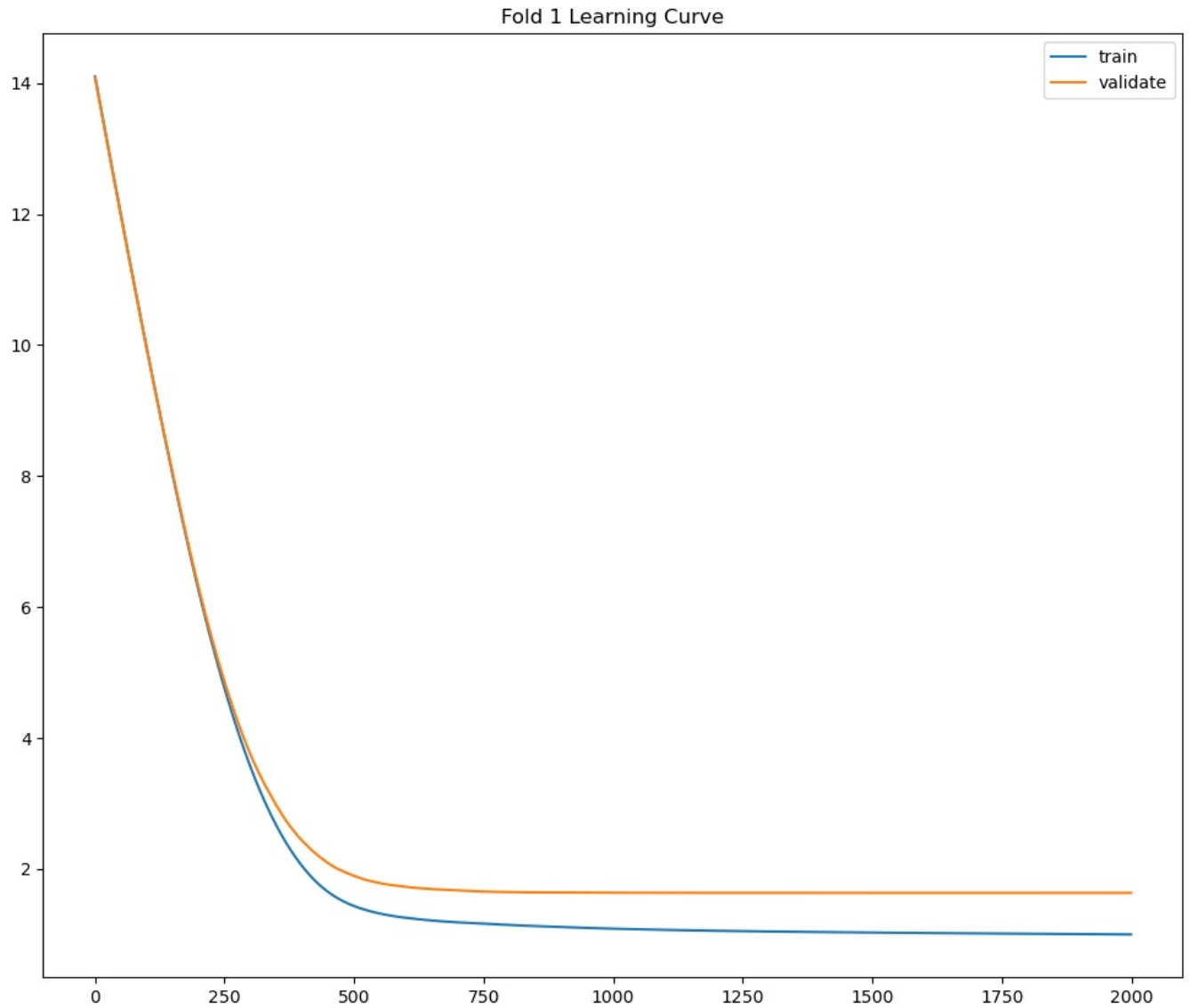
```
In [120]: for fold in range(5):  
eval_result = joblib.load(model_path + f'xgboost/eval_fold_{fold + 1}.joblib')  
plt.plot(eval_result['train']['rmse'], label='train');  
plt.plot(eval_result['validate']['rmse'], label='validate');  
plt.legend();  
plt.title(f'Fold {fold + 1} Learning Curve');  
plt.show();
```

Out[120]: [<matplotlib.lines.Line2D at 0x7f23633d27c0>]

Out[120]: [<matplotlib.lines.Line2D at 0x7f23633b03a0>]

Out[120]: <matplotlib.legend.Legend at 0x7f23633d2340>

Out[120]: Text(0.5, 1.0, 'Fold 1 Learning Curve')

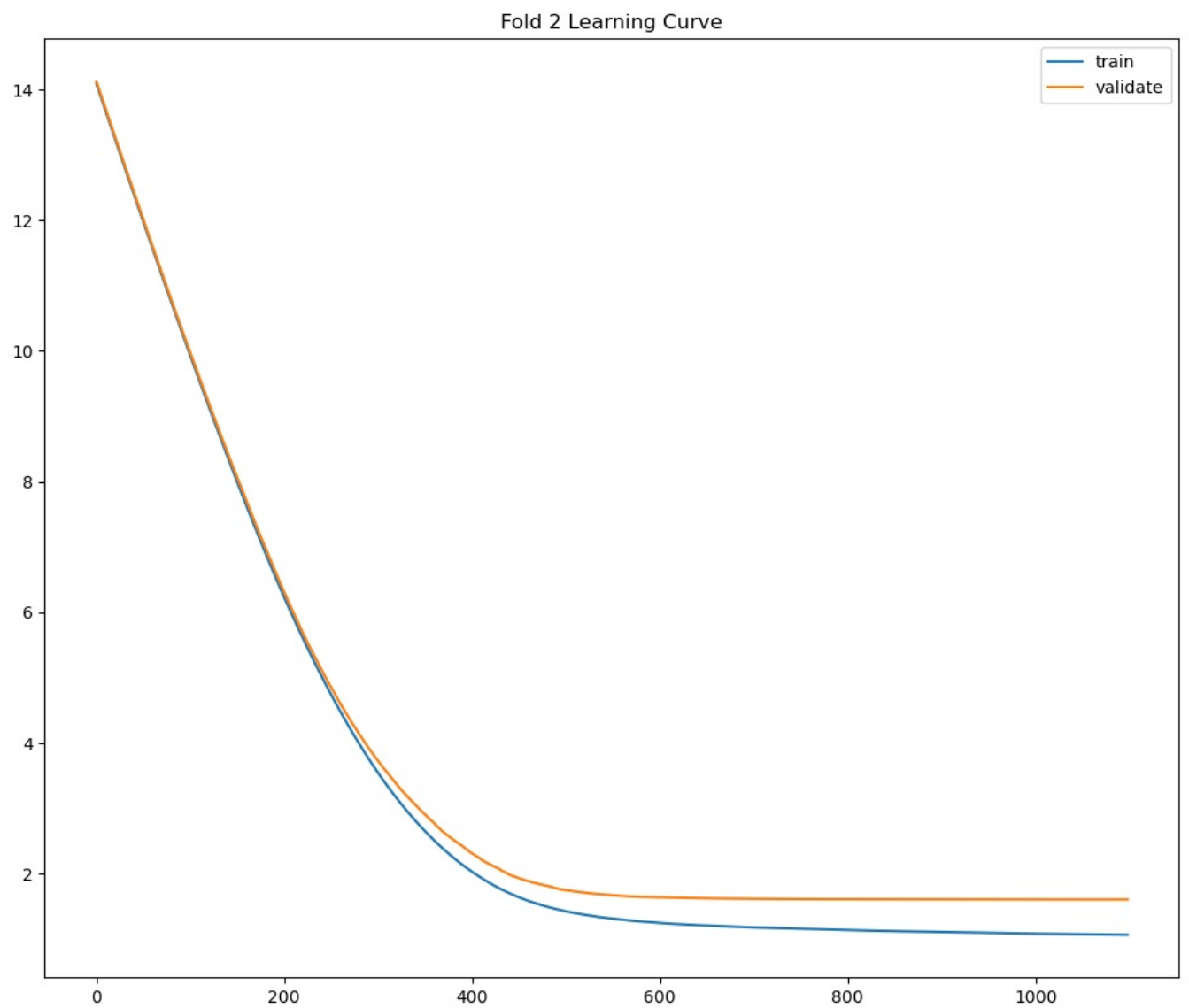


Out[120]: [<matplotlib.lines.Line2D at 0x7f230dd18ac0>]

Out[120]: [<matplotlib.lines.Line2D at 0x7f230d930b20>]

Out[120]: <matplotlib.legend.Legend at 0x7f231a0b87c0>

Out[120]: Text(0.5, 1.0, 'Fold 2 Learning Curve')

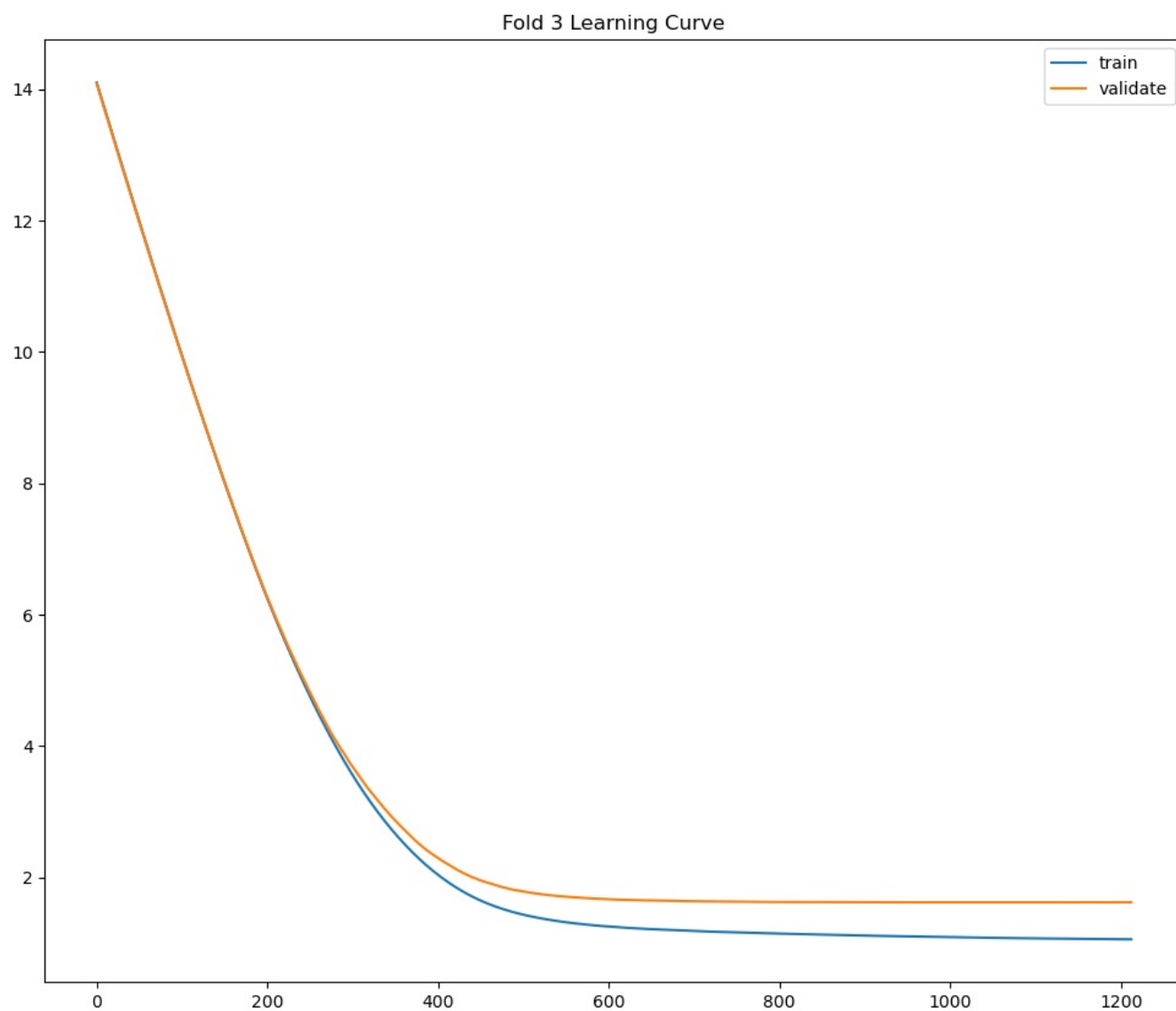


Out[120]: [<matplotlib.lines.Line2D at 0x7f2362981190>]

Out[120]: [<matplotlib.lines.Line2D at 0x7f23629813d0>]

Out[120]: <matplotlib.legend.Legend at 0x7f2362981850>

Out[120]: Text(0.5, 1.0, 'Fold 3 Learning Curve')

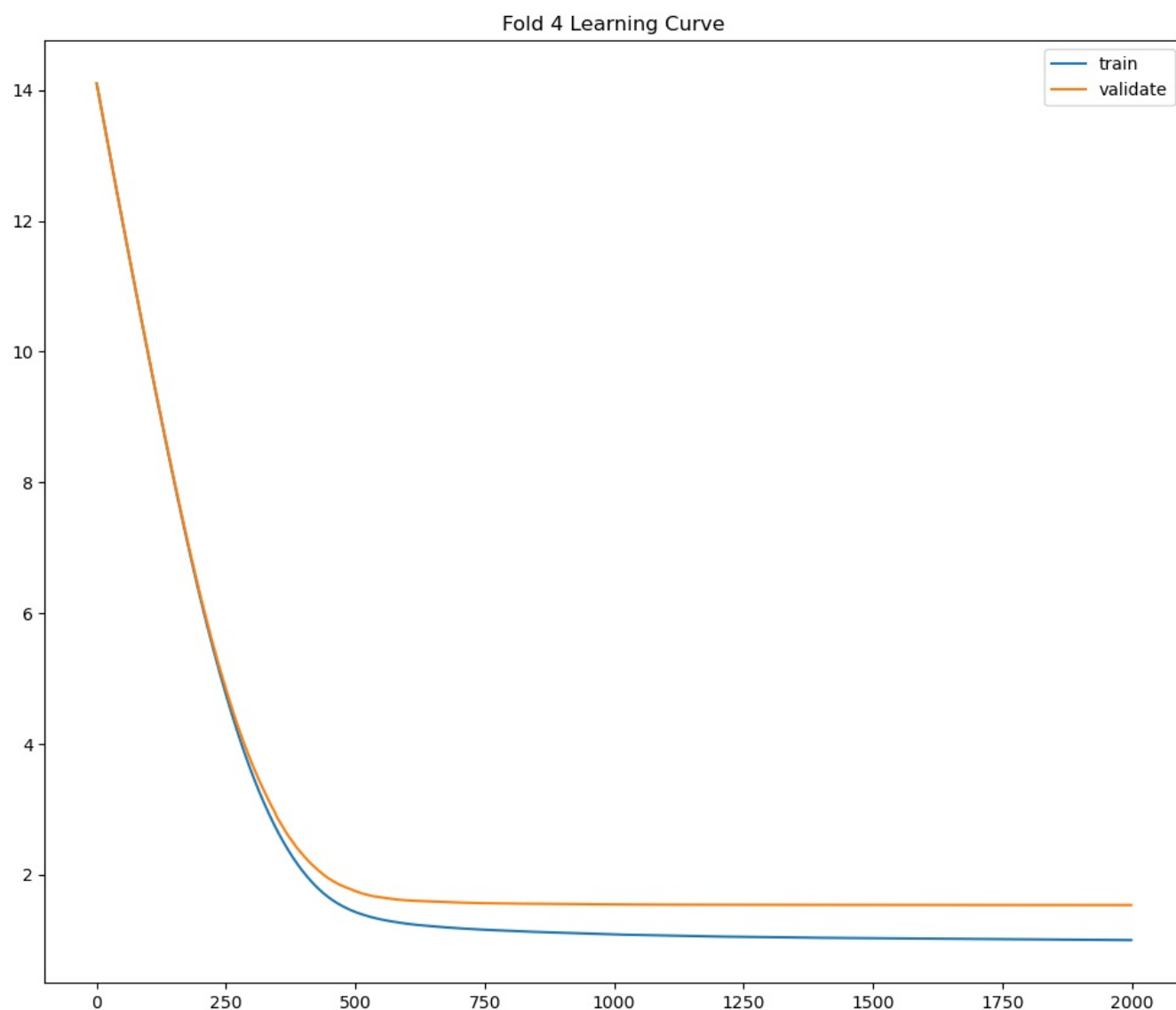


Out[120]: [<matplotlib.lines.Line2D at 0x7f23643a13d0>]

Out[120]: [<matplotlib.lines.Line2D at 0x7f23643a1070>]

Out[120]: <matplotlib.legend.Legend at 0x7f2362a65ac0>

Out[120]: Text(0.5, 1.0, 'Fold 4 Learning Curve')

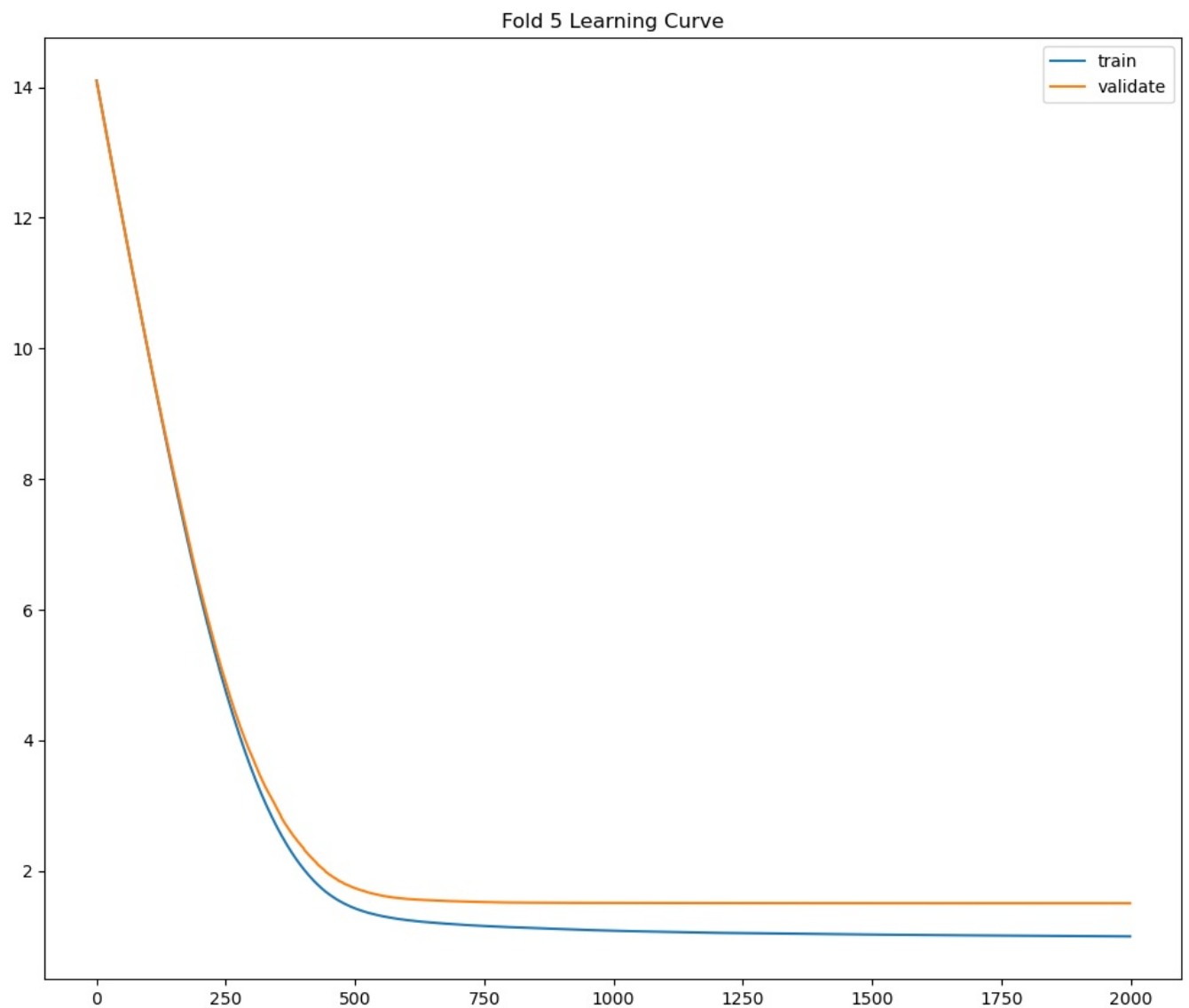


Out[120]: [<matplotlib.lines.Line2D at 0x7f2363571e50>]

Out[120]: [<matplotlib.lines.Line2D at 0x7f2363e4f310>]

Out[120]: <matplotlib.legend.Legend at 0x7f236468ceb0>

Out[120]: Text(0.5, 1.0, 'Fold 5 Learning Curve')



Both the training and validation sets begin to converge at around 500 boosting rounds.

Performance on Validation Sets

```
In [121]: oof_xgboost_rmse = []
target_frame = cudf.DataFrame(index=['count', 'mean', 'std', 'min', '25%', '50%', '75%', 'max'])

for key in oof_xgboost:
    oof_xgboost_rmse.append(
        mean_squared_error(oof_xgboost[key]['target'], oof_xgboost[key]['predictions'], squared=False)
    )
    print(f'Finished computing rmse for {key}')

    target_frame[f'{key}_target_descriptive_stats'] = cudf.Series(oof_xgboost[key]['target']).describe()
    print(f'Finished computing descriptive stats for {key} target')
```



```

Finished computing rmse for fold_1
Finished computing descriptive stats for fold_1 target
Finished computing rmse for fold_2
Finished computing descriptive stats for fold_2 target
Finished computing rmse for fold_3
Finished computing descriptive stats for fold_3 target
Finished computing rmse for fold_4
Finished computing descriptive stats for fold_4 target
Finished computing rmse for fold_5
Finished computing descriptive stats for fold_5 target

```

```
In [122]: cdf.Series(oof_xgboost_rmse).describe()
```

```

Out[122]: count      5.000000
          mean       1.580978
          std        0.056452
          min        1.506437
          25%        1.535277
          50%        1.609146
          75%        1.621096
          max        1.632933
          dtype: float64

```

On average, the predictions are off by \$1.580978\$ percentage points with a standard deviation of about \$0.056452\$ percentage points. This can be compared to the distributions of the true target interest rates.

```
In [118]: target_frame
```

```

Out[118]:

```

	fold_1_target_descriptive_stats	fold_2_target_descriptive_stats	fold_3_target_descriptive_stats	fold_4_target_descriptive_stats	fold_5_target_descriptive_stats
count	67798.000000	67798.000000	67798.000000	67797.000000	67797.000000
mean	13.943553	13.956328	13.940899	13.963181	13.963181
std	4.399556	4.354424	4.376767	4.384787	4.384787
min	5.420000	5.420000	5.420000	5.420000	5.420000
25%	10.990000	10.990000	10.990000	10.990000	10.990000
50%	13.680000	13.920000	13.680000	13.980000	13.980000
75%	16.780000	16.780000	16.780000	16.780000	16.780000
max	26.060000	26.060000	26.060000	26.060000	26.060000

The middle \$50\%\$ of interest rates in the validation sets range between \$10.99\%\$ and \$16.78\%\$; and so the RMSE of \$1.580978\$ percentage points is acceptable. Although with more time, we would like to explore ways to perhaps reduce RMSE down to \$1\$ percentage points or even lower.

CatBoost

Pipeline

For catboost, as mentioned above, we do not include the catboost encode step and allow catboost to handle the text features as categorical variables natively.

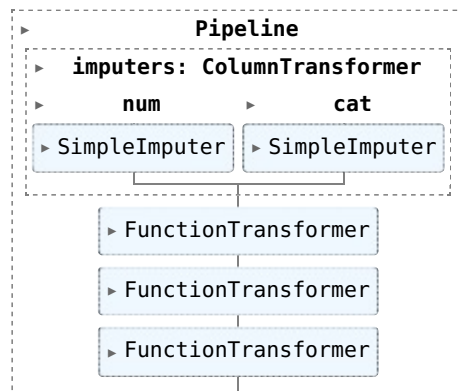
```

In [8]: catboost_preprocessor = Pipeline([
        ('imputers', imputers),
        ('restore_cols', FunctionTransformer(pp.restore_columns)),
        ('date_transformer', FunctionTransformer(pp.extract_date_features)),
        ('num_feat_eng', FunctionTransformer(pp.num_feat_eng))
    ])
joblib.dump(catboost_preprocessor, prep_path + 'catboost_preprocessor.joblib')
catboost_preprocessor

```

```
Out[8]: ['../output/preprocessors/catboost_preprocessor.joblib']
```

Out[8]:



Hyperparameter Search

```
In [260] def objective_catboost(trial):

    # Fold and seed
    train = pd.read_csv("../data/train_sanitized.csv")
    X_train, y_train = train.drop(['interest_rate'], axis=1), train.interest_rate.to_numpy()
    folds = 5
    seed = 1227

    # Parameters
    search_space = {
        'objective': 'RMSE',
        'eval_metric': 'RMSE',
        'task_type': 'GPU', # GPU training
        'boosting_type': 'Plain', # Boosting scheme
        'border_count': 254, # Number of splits for numerical features (recommended 254 for best possible quality)
        'use_best_model': True, # Use the validation dataset to identify the iteration with the optimal value of the objective function
        'iterations': trial.suggest_int('iterations', low=500, high=2000, step=100), # Range: [0, inf], number of iterations
        'learning_rate': trial.suggest_float(name='learning_rate', low=0.001, high=0.1), # Decrease the learning rate
        'depth': trial.suggest_int('depth', 6, 10), # Depth of trees, where values in the range from 6 to 10 are recommended
        'l2_leaf_reg': trial.suggest_categorical('l2_leaf_reg', [10, 100, 500]), # Range: [0, inf], L2 regularization
        'random_strength': trial.suggest_float('random_strength', 100, 500), # Range: Positive floating point number
        'colsample_bylevel': None, # Range (0;1], also 'rsm', the percentage of features to use at each split
        'bootstrap_type': trial.suggest_categorical(
            'bootstrap_type', ['Bayesian', 'Bernoulli']
        ), # The weight of each training example is varied over steps of choosing different splits (not over samples)
        'score_function': trial.suggest_categorical(
            'score_function', ['L2', 'Cosine']
        ) # The score function measures the quality of the gradient approximation, which is used to select the best iteration
    }

    # These parameters are depended on the 'bootstrap_type' chosen
    if search_space['bootstrap_type'] == 'Bayesian':
        search_space['bagging_temperature'] = trial.suggest_float('bagging_temperature', 0, 50) # Range: [0;inf]
    elif search_space['bootstrap_type'] == 'Bernoulli':
        search_space['subsample'] = trial.suggest_float("subsample", 0.1, 1, log=True) # Sample rate for bagging

    # K-fold cross validation
    kf = KFold(n_splits=folds, shuffle=True, random_state=rs)
    rmse_scores = np.empty(folds)

    for fold, (train_idx, val_idx) in enumerate(kf.split(X_train, y_train)):

        # Train and validation sets
        fold_X_train, fold_y_train = X_train.iloc[train_idx], y_train[train_idx]
        fold_X_val, fold_y_val = X_train.iloc[val_idx], y_train[val_idx]

        # Preprocessing using a fresh copy of the pipeline for every fold to prevent leakage
        preprocessor = joblib.load('../output/preprocessors/catboost_preprocessor.joblib')
        print(f'Start processing fold {fold + 1}...')
```

```

fold_X_train = preprocessor.fit_transform(fold_X_train, fold_y_train)
fold_X_val = preprocessor.transform(fold_X_val)

# Data for modeling
feature_names = fold_X_train.columns.tolist()
dtrain = cb.Pool(data=fold_X_train, label=fold_y_train, feature_names=feature_names, cat_features=encode_co
dvalid = cb.Pool(data=fold_X_val, label=fold_y_val, feature_names=feature_names, cat_features=encode_co

# Model
model = cb.train(
    params=search_space,
    dtrain=dtrain,
    early_stopping_rounds=200,
    eval_set=dvalid,
    verbose=200 # Report every 200 rounds
)

# Out-of-fold prediction
print(f'Predicting for fold {fold + 1}...')
oof_pred = model.predict(data=dvalid)
rmse_scores[fold] = mean_squared_error(fold_y_val, oof_pred, squared=False) # Use RMSE

# Average across 5 folds
mean_rmse = np.mean(rmse_scores)

return mean_rmse

```

```

In [ ]: study_catboost = optuna.create_study(sampler=optuna.samplers.TPESampler(), study_name='min_rmse_catboost', dire
study_catboost.optimize(objective_catboost, n_trials=20)

```

```

[I 2023-02-12 17:27:50,209] A new study created in memory with name: min_rmse_catboost

```

```

Start processing fold 1...
0:      learn: 4.0848596      test: 4.0723751 best: 4.0723751 (0)      total: 37ms      remaining: 1m 2s
200:    learn: 1.5473853      test: 2.7024220 best: 2.6968588 (45)      total: 6.71s     remaining: 50.1s
bestTest = 2.696858788
bestIteration = 45
Shrink model to first 46 iterations.
Predicting for fold 1...
Start processing fold 2...
0:      learn: 4.1203519      test: 4.0742373 best: 4.0742373 (0)      total: 34.2ms    remaining: 58.1s
200:    learn: 1.5436217      test: 1.9945311 best: 1.9940960 (196)      total: 6.7s      remaining: 50s
400:    learn: 1.4338617      test: 1.9515568 best: 1.9514760 (389)      total: 13.4s     remaining: 43.3s
600:    learn: 1.3628402      test: 1.9294044 best: 1.9290701 (589)      total: 19.9s     remaining: 36.3s
800:    learn: 1.3122895      test: 1.9076298 best: 1.9076298 (800)      total: 26.4s     remaining: 29.6s
1000:   learn: 1.2774315      test: 1.9012140 best: 1.9000863 (991)      total: 32.9s     remaining: 23s
1200:   learn: 1.2526392      test: 1.8931661 best: 1.8927445 (1188)      total: 39.5s     remaining: 16.4s
1400:   learn: 1.2305494      test: 1.8850506 best: 1.8847245 (1389)      total: 46.1s     remaining: 9.83s
1600:   learn: 1.2120301      test: 1.8843298 best: 1.8842594 (1449)      total: 52.8s     remaining: 3.26s
1699:   learn: 1.2039046      test: 1.8846747 best: 1.8832456 (1685)      total: 56s       remaining: 0us
bestTest = 1.883245572
bestIteration = 1685
Shrink model to first 1686 iterations.
Predicting for fold 2...
Start processing fold 3...
0:      learn: 4.1231846      test: 4.1023352 best: 4.1023352 (0)      total: 35.8ms    remaining: 1m
200:    learn: 1.5511281      test: 2.2707612 best: 2.2680511 (134)      total: 6.57s     remaining: 49s
400:    learn: 1.4269661      test: 2.2208782 best: 2.2208782 (400)      total: 13.1s     remaining: 42.5s
600:    learn: 1.3536703      test: 2.1869745 best: 2.1869745 (600)      total: 19.8s     remaining: 36.1s
800:    learn: 1.3056872      test: 2.1709075 best: 2.1707334 (796)      total: 26.3s     remaining: 29.5s
1000:   learn: 1.2752853      test: 2.1577518 best: 2.1572338 (976)      total: 32.9s     remaining: 23s
1200:   learn: 1.2449648      test: 2.1533405 best: 2.1531355 (1195)      total: 39.4s     remaining: 16.4s
1400:   learn: 1.2224910      test: 2.1435578 best: 2.1434680 (1396)      total: 46s       remaining: 9.82s
1600:   learn: 1.2033667      test: 2.1414529 best: 2.1413359 (1597)      total: 52.5s     remaining: 3.25s
1699:   learn: 1.1947402      test: 2.1395602 best: 2.1390459 (1668)      total: 55.7s     remaining: 0us
bestTest = 2.139045939
bestIteration = 1668
Shrink model to first 1669 iterations.
Predicting for fold 3...
Start processing fold 4...
0:      learn: 4.2131120      test: 4.2899211 best: 4.2899211 (0)      total: 35.3ms    remaining: 59.9s
200:    learn: 1.5600864      test: 2.3840657 best: 2.3840657 (200)      total: 6.67s     remaining: 49.7s
400:    learn: 1.4283281      test: 2.3622502 best: 2.3609446 (392)      total: 13.2s     remaining: 42.9s
600:    learn: 1.3520794      test: 2.3468624 best: 2.3463829 (580)      total: 19.8s     remaining: 36.2s
800:    learn: 1.3029012      test: 2.3419366 best: 2.3419366 (800)      total: 26.3s     remaining: 29.6s
1000:   learn: 1.2741654      test: 2.3393090 best: 2.3391486 (997)      total: 32.9s     remaining: 22.9s
1200:   learn: 1.2519078      test: 2.3308172 best: 2.3290042 (1144)      total: 39.3s     remaining: 16.3s
1400:   learn: 1.2251457      test: 2.3272872 best: 2.3257725 (1308)      total: 45.8s     remaining: 9.77s
bestTest = 2.325772452
bestIteration = 1308
Shrink model to first 1309 iterations.
Predicting for fold 4...
Start processing fold 5...
0:      learn: 4.2117981      test: 4.2022596 best: 4.2022596 (0)      total: 38.7ms    remaining: 1m 5s
200:    learn: 1.5642409      test: 2.0729604 best: 2.0729604 (200)      total: 6.71s     remaining: 50s
400:    learn: 1.4269476      test: 2.0189208 best: 2.0189208 (400)      total: 13.4s     remaining: 43.3s
600:    learn: 1.3668995      test: 2.0056792 best: 2.0054244 (584)      total: 20s       remaining: 36.6s
800:    learn: 1.3098762      test: 1.9842738 best: 1.9842455 (799)      total: 26.6s     remaining: 29.9s
1000:   learn: 1.2730691      test: 1.9749355 best: 1.9723649 (953)      total: 33.3s     remaining: 23.2s
1200:   learn: 1.2453016      test: 1.9607700 best: 1.9607700 (1200)      total: 40s       remaining: 16.6s
1400:   learn: 1.2267419      test: 1.9564191 best: 1.9556433 (1361)      total: 46.6s     remaining: 9.94s
1600:   learn: 1.2072765      test: 1.9446174 best: 1.9443208 (1582)      total: 53.1s     remaining: 3.28s
1699:   learn: 1.1997228      test: 1.9433751 best: 1.9432321 (1688)      total: 56.3s     remaining: 0us
bestTest = 1.943232137
bestIteration = 1688
Shrink model to first 1689 iterations.
Predicting for fold 5...

```

```

[I 2023-02-12 17:32:22,079] Trial 0 finished with value: 2.19763139748944 and parameters: {'iterations': 1700,
'learning_rate': 0.09243529468898837, 'depth': 7, 'l2_leaf_reg': 100, 'random_strength': 443.98205697337323, 'b
ootstrap_type': 'Bayesian', 'score_function': 'Cosine', 'bagging_temperature': 21.533709015144137}. Best is tri
al 0 with value: 2.19763139748944.

```

```

Start processing fold 1...
0:      learn: 3.4968570      test: 3.6328176 best: 3.6328176 (0)      total: 51ms      remaining: 25.4s
200:    learn: 1.2163275      test: 2.7258795 best: 2.7239090 (196)    total: 9.55s     remaining: 14.2s
400:    learn: 1.0750178      test: 2.7250388 best: 2.7161948 (219)    total: 19s       remaining: 4.7s
bestTest = 2.716194754
bestIteration = 219
Shrink model to first 220 iterations.
Predicting for fold 1...
Start processing fold 2...
0:      learn: 3.4511862      test: 3.3999753 best: 3.3999753 (0)      total: 45.6ms    remaining: 22.8s
200:    learn: 1.1982550      test: 1.9702158 best: 1.9690059 (196)    total: 9.31s     remaining: 13.9s
400:    learn: 1.0596488      test: 1.9755263 best: 1.9605864 (245)    total: 18.6s     remaining: 4.58s
bestTest = 1.96058642

bestIteration = 245
Shrink model to first 246 iterations.
Predicting for fold 2...
Start processing fold 3...
0:      learn: 3.4970091      test: 3.7061443 best: 3.7061443 (0)      total: 50.3ms    remaining: 25.1s
200:    learn: 1.2160255      test: 2.3135732 best: 2.2862840 (66)      total: 9.54s     remaining: 14.2s
bestTest = 2.286284025
bestIteration = 66
Shrink model to first 67 iterations.
Predicting for fold 3...
Start processing fold 4...
0:      learn: 3.4334628      test: 3.4710775 best: 3.4710775 (0)      total: 47.1ms    remaining: 23.5s
200:    learn: 1.2024274      test: 2.5969827 best: 2.5755178 (145)    total: 9.47s     remaining: 14.1s
bestTest = 2.575517765
bestIteration = 145
Shrink model to first 146 iterations.
Predicting for fold 4...
Start processing fold 5...
0:      learn: 3.4583370      test: 3.7710421 best: 3.7710421 (0)      total: 51.7ms    remaining: 25.8s
200:    learn: 1.2007035      test: 2.3290866 best: 2.3149984 (178)    total: 9.36s     remaining: 13.9s
400:    learn: 1.0505498      test: 2.2997804 best: 2.2938160 (383)    total: 18.7s     remaining: 4.62s
499:    learn: 0.9991515      test: 2.2972953 best: 2.2938160 (383)    total: 23.3s     remaining: 0us
bestTest = 2.293816045
bestIteration = 383
Shrink model to first 384 iterations.
Predicting for fold 5...
[I 2023-02-12 17:34:40,479] Trial 1 finished with value: 2.3664799006597526 and parameters: {'iterations': 500,
'learning_rate': 0.2876547994556458, 'depth': 9, 'l2_leaf_reg': 10, 'random_strength': 352.66894226927604, 'boo
tstrap_type': 'Bayesian', 'score_function': 'Cosine', 'bagging_temperature': 20.270770136272372}. Best is trial
0 with value: 2.19763139748944.

```

```

Start processing fold 1...
0:      learn: 4.0777904      test: 4.2151198 best: 4.2151198 (0)      total: 43.7ms      remaining: 39.2s
200:    learn: 1.1580801      test: 2.1777521 best: 2.1771445 (198)    total: 8s          remaining: 27.8s
400:    learn: 1.0787310      test: 2.1724619 best: 2.1684399 (361)    total: 16s         remaining: 19.9s
bestTest = 2.168439923
bestIteration = 361
Shrink model to first 362 iterations.
Predicting for fold 1...
Start processing fold 2...
0:      learn: 4.0499645      test: 4.0376765 best: 4.0376765 (0)      total: 41.5ms      remaining: 37.3s
200:    learn: 1.1595877      test: 2.4730881 best: 2.4723799 (198)    total: 8.06s       remaining: 28s
400:    learn: 1.0628059      test: 2.4682939 best: 2.4665543 (393)    total: 16.1s       remaining: 20s
600:    learn: 1.0231657      test: 2.4657692 best: 2.4643577 (475)    total: 24.1s       remaining: 12s
bestTest = 2.464357713
bestIteration = 475
Shrink model to first 476 iterations.
Predicting for fold 2...
Start processing fold 3...
0:      learn: 4.0400257      test: 3.9585732 best: 3.9585732 (0)      total: 46.4ms      remaining: 41.7s
200:    learn: 1.1522551      test: 2.4385676 best: 2.4385676 (200)    total: 8.01s       remaining: 27.9s
400:    learn: 1.0664529      test: 2.4093584 best: 2.4093584 (400)    total: 15.9s       remaining: 19.8s
600:    learn: 1.0258395      test: 2.4074760 best: 2.4058980 (534)    total: 23.8s       remaining: 11.9s
800:    learn: 0.9930525      test: 2.4058924 best: 2.4056433 (785)    total: 31.9s       remaining: 3.94s
899:    learn: 0.9775517      test: 2.4092458 best: 2.4056433 (785)    total: 35.9s       remaining: 0us
bestTest = 2.405643252
bestIteration = 785
Shrink model to first 786 iterations.
Predicting for fold 3...
Start processing fold 4...
0:      learn: 4.0485282      test: 4.0965720 best: 4.0965720 (0)      total: 46.4ms      remaining: 41.7s
200:    learn: 1.1545757      test: 2.0374261 best: 2.0371794 (199)    total: 7.93s       remaining: 27.6s
400:    learn: 1.0734457      test: 2.0130988 best: 2.0128848 (399)    total: 15.7s       remaining: 19.5s
600:    learn: 1.0342822      test: 2.0112936 best: 2.0111861 (520)    total: 23.6s       remaining: 11.7s
800:    learn: 1.0006668      test: 2.0029292 best: 2.0027239 (775)    total: 31.6s       remaining: 3.91s
899:    learn: 0.9857980      test: 2.0030489 best: 2.0023056 (827)    total: 35.6s       remaining: 0us
bestTest = 2.00230565
bestIteration = 827
Shrink model to first 828 iterations.
Predicting for fold 4...

Start processing fold 5...
0:      learn: 4.0984718      test: 4.1526316 best: 4.1526316 (0)      total: 47.7ms      remaining: 42.9s
200:    learn: 1.1606541      test: 2.2896682 best: 2.2893970 (199)    total: 8.18s       remaining: 28.5s
400:    learn: 1.0736631      test: 2.2779428 best: 2.2777106 (385)    total: 16.2s       remaining: 20.1s
600:    learn: 1.0332423      test: 2.2753544 best: 2.2750193 (544)    total: 24.1s       remaining: 12s
800:    learn: 1.0008421      test: 2.2803834 best: 2.2747051 (655)    total: 32.1s       remaining: 3.96s
bestTest = 2.27470507
bestIteration = 655
Shrink model to first 656 iterations.
Predicting for fold 5...

[I 2023-02-12 17:38:05,358] Trial 2 finished with value: 2.263091107876284 and parameters: {'iterations': 900,
'learning_rate': 0.10545772184117341, 'depth': 8, 'l2_leaf_reg': 10, 'random_strength': 132.02105296824692, 'bo
otstrap_type': 'Bernoulli', 'score_function': 'Cosine', 'subsample': 0.13212373310354877}. Best is trial 0 with
value: 2.19763139748944.

```

```

Start processing fold 1...
0:      learn: 3.7618030      test: 3.7852330 best: 3.7852330 (0)      total: 51.8ms      remaining: 1m 22s
200:    learn: 1.2801650      test: 2.0991671 best: 2.0948465 (145)    total: 9.75s      remaining: 1m 7s
400:    learn: 1.2132436      test: 2.0993209 best: 2.0943914 (344)    total: 19.4s      remaining: 58s
bestTest = 2.094391383
bestIteration = 344
Shrink model to first 345 iterations.
Predicting for fold 1...
Start processing fold 2...
0:      learn: 3.7750651      test: 3.8001021 best: 3.8001021 (0)      total: 47.6ms      remaining: 1m 16s
200:    learn: 1.2822394      test: 1.9222712 best: 1.9221592 (196)    total: 9.66s      remaining: 1m 7s
400:    learn: 1.2202816      test: 1.9020375 best: 1.9019627 (398)    total: 19.3s      remaining: 57.8s
600:    learn: 1.1776971      test: 1.8966319 best: 1.8961188 (570)    total: 28.8s      remaining: 48s
800:    learn: 1.1440243      test: 1.8931051 best: 1.8921635 (781)    total: 38.3s      remaining: 38.2s
1000:   learn: 1.1147822      test: 1.8927794 best: 1.8916243 (857)    total: 47.8s      remaining: 28.6s
1200:   learn: 1.0897026      test: 1.8917715 best: 1.8905452 (1086)   total: 57.3s      remaining: 19s
bestTest = 1.890545225
bestIteration = 1086
Shrink model to first 1087 iterations.
Predicting for fold 2...
Start processing fold 3...
0:      learn: 3.7605555      test: 4.0347971 best: 4.0347971 (0)      total: 52.5ms      remaining: 1m 24s
200:    learn: 1.2668996      test: 2.0028822 best: 2.0028011 (199)    total: 9.45s      remaining: 1m 5s
400:    learn: 1.2096007      test: 1.9844220 best: 1.9844016 (399)    total: 18.8s      remaining: 56.3s
600:    learn: 1.1650471      test: 1.9791947 best: 1.9791947 (600)    total: 28.2s      remaining: 47s
800:    learn: 1.1336122      test: 1.9728185 best: 1.9728004 (799)    total: 37.6s      remaining: 37.5s
1000:   learn: 1.1060521      test: 1.9684674 best: 1.9683143 (997)    total: 47.1s      remaining: 28.2s
1200:   learn: 1.0814493      test: 1.9692201 best: 1.9682965 (1008)   total: 56.6s      remaining: 18.8s
bestTest = 1.96829649
bestIteration = 1008
Shrink model to first 1009 iterations.
Predicting for fold 3...
Start processing fold 4...
0:      learn: 3.7808159      test: 3.9535966 best: 3.9535966 (0)      total: 51.7ms      remaining: 1m 22s
200:    learn: 1.2877353      test: 2.0804752 best: 2.0793087 (195)    total: 9.51s      remaining: 1m 6s
400:    learn: 1.2165688      test: 2.0564227 best: 2.0563241 (394)    total: 19.1s      remaining: 57s
600:    learn: 1.1743506      test: 2.0568480 best: 2.0557177 (416)    total: 28.5s      remaining: 47.4s
bestTest = 2.055717674
bestIteration = 416
Shrink model to first 417 iterations.
Predicting for fold 4...
Start processing fold 5...
0:      learn: 3.7848651      test: 3.8846603 best: 3.8846603 (0)      total: 48ms        remaining: 1m 16s
200:    learn: 1.2682368      test: 1.9256688 best: 1.9252221 (198)    total: 9.47s      remaining: 1m 5s
400:    learn: 1.2120032      test: 1.9001580 best: 1.9001580 (400)    total: 18.9s      remaining: 56.4s
600:    learn: 1.1686133      test: 1.8792366 best: 1.8792148 (597)    total: 28.2s      remaining: 46.9s
800:    learn: 1.1336706      test: 1.8695701 best: 1.8687617 (764)    total: 37.8s      remaining: 37.7s
1000:   learn: 1.1068532      test: 1.8698741 best: 1.8675164 (964)    total: 47.3s      remaining: 28.3s
1200:   learn: 1.0836473      test: 1.8631398 best: 1.8629739 (1197)   total: 56.7s      remaining: 18.8s
1400:   learn: 1.0609423      test: 1.8604759 best: 1.8598902 (1388)   total: 1m 6s      remaining: 9.39s
1599:   learn: 1.0407010      test: 1.8540146 best: 1.8539517 (1596)   total: 1m 15s     remaining: 0us
bestTest = 1.853951729
bestIteration = 1596
Shrink model to first 1597 iterations.
Predicting for fold 5...

```

```

[I 2023-02-12 17:43:02,821] Trial 3 finished with value: 1.9725806096138503 and parameters: {'iterations': 1600, 'learning_rate': 0.21065008859344805, 'depth': 9, 'l2_leaf_reg': 500, 'random_strength': 122.57161445434504, 'bootstrap_type': 'Bayesian', 'score_function': 'L2', 'bagging_temperature': 7.960423129118915}. Best is trial 3 with value: 1.9725806096138503.

```

```
Start processing fold 1...
0:      learn: 3.6509066      test: 3.8018097 best: 3.8018097 (0)      total: 52.7ms      remaining: 36.8s
200:    learn: 1.0792543      test: 3.5708527 best: 3.5708527 (200)    total: 9.5s      remaining: 23.6s
400:    learn: 1.0292823      test: 3.5773411 best: 3.5660097 (249)    total: 19.1s      remaining: 14.2s
```

bestTest = 3.566009686

bestIteration = 249

Shrink model to first 250 iterations.

Predicting for fold 1...

Start processing fold 2...

```
0:      learn: 3.6678501      test: 3.8660457 best: 3.8660457 (0)      total: 48.7ms      remaining: 34s
200:    learn: 1.0840214      test: 3.0704321 best: 3.0628918 (171)    total: 9.65s      remaining: 24s
```

bestTest = 3.062891839

bestIteration = 171

Shrink model to first 172 iterations.

Predicting for fold 2...

Start processing fold 3...

```
0:      learn: 3.5893861      test: 3.9051212 best: 3.9051212 (0)      total: 57.7ms      remaining: 40.3s
200:    learn: 1.0727138      test: 3.2999636 best: 3.2996514 (199)    total: 9.48s      remaining: 23.5s
400:    learn: 1.0308172      test: 3.2883986 best: 3.2823433 (347)    total: 18.8s      remaining: 14s
```

bestTest = 3.282343284

bestIteration = 347

Shrink model to first 348 iterations.

Predicting for fold 3...

Start processing fold 4...

```
0:      learn: 3.5586934      test: 3.6621947 best: 3.6621947 (0)      total: 53.3ms      remaining: 37.3s
200:    learn: 1.0727155      test: 2.7152242 best: 2.7150923 (199)    total: 9.62s      remaining: 23.9s
400:    learn: 1.0263183      test: 2.7109000 best: 2.7074339 (364)    total: 19.1s      remaining: 14.2s
```

bestTest = 2.707433936

bestIteration = 364

Shrink model to first 365 iterations.

Predicting for fold 4...

Start processing fold 5...

```
0:      learn: 3.7240634      test: 3.8098981 best: 3.8098981 (0)      total: 60.1ms      remaining: 42s
200:    learn: 1.0791372      test: 3.1748232 best: 3.1684538 (178)    total: 9.65s      remaining: 24s
```

bestTest = 3.168453765

bestIteration = 178

Shrink model to first 179 iterations.

Predicting for fold 5...

[I 2023-02-12 17:45:41,321] Trial 4 finished with value: 3.1574260620229095 and parameters: {'iterations': 700, 'learning_rate': 0.27296590435850876, 'depth': 9, 'l2_leaf_reg': 500, 'random_strength': 482.8209104403226, 'bootstrap_type': 'Bernoulli', 'score_function': 'Cosine', 'subsample': 0.562988418591445}. Best is trial 3 with value: 1.9725806096138503.


```

Start processing fold 1...
0:      learn: 3.7032788      test: 3.7047728 best: 3.7047728 (0)      total: 39.2ms      remaining: 1m 18s
200:    learn: 1.4036859      test: 2.4250500 best: 2.4105033 (178)    total: 7.8s        remaining: 1m 9s
400:    learn: 1.2959469      test: 2.4218086 best: 2.3980371 (281)    total: 15.5s       remaining: 1m 1s
bestTest = 2.398037146
bestIteration = 281
Shrink model to first 282 iterations.
Predicting for fold 1...
Start processing fold 2...
0:      learn: 3.6641951      test: 3.7780716 best: 3.7780716 (0)      total: 42.7ms      remaining: 1m 25s
200:    learn: 1.4133520      test: 2.1027656 best: 2.1027656 (200)    total: 7.74s       remaining: 1m 9s
400:    learn: 1.2861666      test: 2.0066951 best: 2.0061327 (399)    total: 15.4s       remaining: 1m 1s
600:    learn: 1.2162303      test: 1.9912529 best: 1.9865342 (441)    total: 23s         remaining: 53.6s
800:    learn: 1.1656672      test: 1.9810809 best: 1.9743689 (762)    total: 30.7s       remaining: 46s
1000:   learn: 1.1258115      test: 1.9750002 best: 1.9721495 (930)    total: 38.5s       remaining: 38.5s
bestTest = 1.972149482
bestIteration = 930
Shrink model to first 931 iterations.
Predicting for fold 2...
Start processing fold 3...
0:      learn: 3.6875692      test: 3.8285542 best: 3.8285542 (0)      total: 43.4ms      remaining: 1m 26s
200:    learn: 1.4422921      test: 2.3547477 best: 2.3547477 (200)    total: 7.73s       remaining: 1m 9s
400:    learn: 1.3016802      test: 2.2937991 best: 2.2931333 (372)    total: 15.6s       remaining: 1m 2s
600:    learn: 1.2371372      test: 2.2895046 best: 2.2874688 (554)    total: 23.3s       remaining: 54.3s
800:    learn: 1.1717906      test: 2.3076659 best: 2.2858239 (641)    total: 31.2s       remaining: 46.7s
bestTest = 2.285823914
bestIteration = 641
Shrink model to first 642 iterations.
Predicting for fold 3...
Start processing fold 4...
0:      learn: 3.7724871      test: 3.8512094 best: 3.8512094 (0)      total: 40.6ms      remaining: 1m 21s
200:    learn: 1.4440473      test: 2.3258132 best: 2.3249970 (175)    total: 7.83s       remaining: 1m 10s
400:    learn: 1.3030968      test: 2.3285040 best: 2.3235688 (370)    total: 15.7s       remaining: 1m 2s
600:    learn: 1.2331889      test: 2.3263343 best: 2.3197331 (576)    total: 23.5s       remaining: 54.6s
800:    learn: 1.1819443      test: 2.3178971 best: 2.3126289 (695)    total: 31.3s       remaining: 46.8s
bestTest = 2.312628896
bestIteration = 695
Shrink model to first 696 iterations.
Predicting for fold 4...
Start processing fold 5...
0:      learn: 3.6938607      test: 3.4972496 best: 3.4972496 (0)      total: 42.5ms      remaining: 1m 25s
200:    learn: 1.4498925      test: 2.1687930 best: 2.1683824 (154)    total: 7.81s       remaining: 1m 9s
400:    learn: 1.3078301      test: 2.1482541 best: 2.1421694 (273)    total: 15.5s       remaining: 1m 1s
bestTest = 2.142169438
bestIteration = 273
Shrink model to first 274 iterations.
Predicting for fold 5...
[I 2023-02-12 17:48:55,771] Trial 5 finished with value: 2.222161643786955 and parameters: {'iterations': 2000,
'learning_rate': 0.2457514208222952, 'depth': 8, 'l2_leaf_reg': 100, 'random_strength': 225.5153077107271, 'boo
tstrap_type': 'Bayesian', 'score_function': 'L2', 'bagging_temperature': 28.983593683335464}. Best is trial 3 w
ith value: 1.9725806096138503.

```

```

Start processing fold 1...
0:      learn: 4.3003305      test: 4.3342562 best: 4.3342562 (0)      total: 28.5ms      remaining: 57s
200:    learn: 1.3643183      test: 2.4698198 best: 2.4545976 (87)      total: 5.59s      remaining: 50s
400:    learn: 1.2021255      test: 2.4368202 best: 2.4365513 (399)      total: 11.1s      remaining: 44.4s
600:    learn: 1.1405471      test: 2.4328034 best: 2.4325062 (587)      total: 16.7s      remaining: 38.8s
800:    learn: 1.1079051      test: 2.4329232 best: 2.4319892 (738)      total: 22.1s      remaining: 33.1s
bestTest = 2.431989226
bestIteration = 738
Shrink model to first 739 iterations.
Predicting for fold 1...
Start processing fold 2...
0:      learn: 4.3041845      test: 4.3015893 best: 4.3015893 (0)      total: 29.8ms      remaining: 59.5s
200:    learn: 1.3743687      test: 2.4198597 best: 2.3034945 (78)      total: 5.56s      remaining: 49.7s
bestTest = 2.303494543
bestIteration = 78
Shrink model to first 79 iterations.
Predicting for fold 2...
Start processing fold 3...
0:      learn: 4.3021296      test: 4.3149652 best: 4.3149652 (0)      total: 27.8ms      remaining: 55.7s
200:    learn: 1.3757776      test: 2.3153795 best: 2.2162449 (78)      total: 5.6s      remaining: 50.1s
bestTest = 2.216244909
bestIteration = 78
Shrink model to first 79 iterations.
Predicting for fold 3...
Start processing fold 4...
0:      learn: 4.3025354      test: 4.3061905 best: 4.3061905 (0)      total: 27.8ms      remaining: 55.5s
200:    learn: 1.3737359      test: 2.6186714 best: 2.3887200 (55)      total: 5.46s      remaining: 48.9s
bestTest = 2.388719956
bestIteration = 55
Shrink model to first 56 iterations.
Predicting for fold 4...

Start processing fold 5...
0:      learn: 4.3057406      test: 4.3039510 best: 4.3039510 (0)      total: 30.5ms      remaining: 1m
200:    learn: 1.3745383      test: 2.4326439 best: 2.3428188 (83)      total: 5.55s      remaining: 49.7s
[I 2023-02-12 17:50:38,022] Trial 6 finished with value: 2.336653728565746 and parameters: {'iterations': 2000,
'learning_rate': 0.02056948406702553, 'depth': 6, 'l2_leaf_reg': 10, 'random_strength': 247.61142570207707, 'bo
otstrap_type': 'Bernoulli', 'score_function': 'L2', 'subsample': 0.3699579076859651}. Best is trial 3 with valu
e: 1.9725806096138503.
bestTest = 2.34281876
bestIteration = 83
Shrink model to first 84 iterations.
Predicting for fold 5...
Start processing fold 1...
0:      learn: 3.9260586      test: 3.9941262 best: 3.9941262 (0)      total: 47.2ms      remaining: 33s
200:    learn: 1.4521354      test: 2.3555567 best: 2.3551621 (198)      total: 9.28s      remaining: 23s
bestTest = 2.266953027
bestIteration = 975
Shrink model to first 976 iterations.
Predicting for fold 4...
Start processing fold 5...
0:      learn: 4.3696102      test: 4.3773716 best: 4.3773716 (0)      total: 34.9ms      remaining: 1m 2s
200:    learn: 1.0365771      test: 1.9675331 best: 1.9675331 (200)      total: 11.8s      remaining: 1m 33s
400:    learn: 0.9856710      test: 1.9644075 best: 1.9638971 (368)      total: 23.8s      remaining: 1m 23s
600:    learn: 0.9439555      test: 1.9616074 best: 1.9615820 (599)      total: 35.9s      remaining: 1m 11s
800:    learn: 0.9075639      test: 1.9614794 best: 1.9614794 (800)      total: 48.1s      remaining: 60s
1000:   learn: 0.8745723      test: 1.9606410 best: 1.9606063 (996)      total: 1m      remaining: 48.2s
1200:   learn: 0.8423074      test: 1.9592372 best: 1.9592372 (1200)      total: 1m 12s      remaining: 36.1s
1400:   learn: 0.8115288      test: 1.9613687 best: 1.9592022 (1202)      total: 1m 24s      remaining: 24.1s
bestTest = 1.959202248
bestIteration = 1202
Shrink model to first 1203 iterations.
Predicting for fold 4...
Start processing fold 5...
0:      learn: 3.7388224      test: 3.7827831 best: 3.7827831 (0)      total: 59.5ms      remaining: 1m 46s
200:    learn: 1.0363323      test: 2.4924087 best: 2.3888544 (4)      total: 11.7s      remaining: 1m 33s
[I 2023-02-12 18:29:17,464] Trial 18 finished with value: 2.203492765210925 and parameters: {'iterations': 1800
, 'learning_rate': 0.18553819894546034, 'depth': 10, 'l2_leaf_reg': 500, 'random_strength': 277.893968218993, '
bootstrap_type': 'Bernoulli', 'score_function': 'L2', 'subsample': 0.8296535954749351}. Best is trial 3 with va
lue: 1.9725806096138503.

```

```

bestTest = 2.38854351
bestIteration = 4
Shrink model to first 5 iterations.
Predicting for fold 5...
Start processing fold 1...
0:      learn: 4.1543841      test: 4.1270020 best: 4.1270020 (0)      total: 40.3ms      remaining: 44.3s
200:    learn: 1.4679340      test: 2.2525211 best: 2.2501099 (185)    total: 7.88s      remaining: 35.3s
400:    learn: 1.3294188      test: 2.2363308 best: 2.2363308 (400)    total: 15.8s      remaining: 27.5s
600:    learn: 1.2690285      test: 2.2393402 best: 2.2294294 (517)    total: 23.6s      remaining: 19.6s
bestTest = 2.229429392
bestIteration = 517
Shrink model to first 518 iterations.
Predicting for fold 1...

Start processing fold 2...
0:      learn: 4.1795961      test: 4.2100459 best: 4.2100459 (0)      total: 43ms       remaining: 47.3s
200:    learn: 1.4269875      test: 2.2032511 best: 2.2026144 (196)    total: 8.03s      remaining: 35.9s
400:    learn: 1.3229699      test: 2.1739985 best: 2.1714599 (394)    total: 16s        remaining: 27.8s
bestTest = 2.171459921
bestIteration = 394
Shrink model to first 395 iterations.
Predicting for fold 2...
Start processing fold 3...
0:      learn: 4.1724948      test: 4.3693814 best: 4.3693814 (0)      total: 39.4ms     remaining: 43.3s
200:    learn: 1.4480570      test: 2.0793979 best: 2.0751591 (152)    total: 8s         remaining: 35.8s
400:    learn: 1.3281120      test: 2.0443287 best: 2.0439836 (387)    total: 16s        remaining: 27.8s
600:    learn: 1.2616592      test: 2.0245049 best: 2.0245049 (600)    total: 23.9s      remaining: 19.9s
800:    learn: 1.2189871      test: 2.0210007 best: 2.0183766 (793)    total: 31.9s      remaining: 11.9s
1000:   learn: 1.1870541      test: 2.0164697 best: 2.0163148 (989)    total: 39.9s      remaining: 3.94s
1099:   learn: 1.1730093      test: 2.0163640 best: 2.0157656 (1079)   total: 43.8s      remaining: 0us
bestTest = 2.015765555
bestIteration = 1079
Shrink model to first 1080 iterations.
Predicting for fold 3...
Start processing fold 4...
0:      learn: 4.1499020      test: 4.2005455 best: 4.2005455 (0)      total: 43.4ms     remaining: 47.7s
200:    learn: 1.4370082      test: 2.2490725 best: 2.2444321 (82)     total: 8.07s      remaining: 36.1s
400:    learn: 1.3224691      test: 2.2102485 best: 2.2097509 (394)    total: 16s        remaining: 28s
600:    learn: 1.2619467      test: 2.1992855 best: 2.1990257 (586)    total: 24s        remaining: 20s
800:    learn: 1.2156625      test: 2.1931757 best: 2.1931757 (800)    total: 32s        remaining: 12s
1000:   learn: 1.1813572      test: 2.1836485 best: 2.1836204 (992)    total: 40.1s      remaining: 3.96s
1099:   learn: 1.1663943      test: 2.1828387 best: 2.1828387 (1099)   total: 44s        remaining: 0us
bestTest = 2.182838736
bestIteration = 1099
Predicting for fold 4...
Start processing fold 5...
0:      learn: 4.1562173      test: 4.1976456 best: 4.1976456 (0)      total: 43.6ms     remaining: 48s
200:    learn: 1.4419683      test: 2.1979926 best: 2.1857741 (163)    total: 8.09s      remaining: 36.2s
400:    learn: 1.3176110      test: 2.1812219 best: 2.1774058 (392)    total: 16s        remaining: 28s
600:    learn: 1.2513098      test: 2.1681635 best: 2.1669276 (572)    total: 24s        remaining: 19.9s
800:    learn: 1.2075787      test: 2.1578364 best: 2.1578364 (800)    total: 32s        remaining: 11.9s
1000:   learn: 1.1761045      test: 2.1487069 best: 2.1487069 (1000)   total: 39.9s      remaining: 3.94s
1099:   learn: 1.1620604      test: 2.1487532 best: 2.1481219 (1096)   total: 43.8s      remaining: 0us
bestTest = 2.148121863
bestIteration = 1096
Shrink model to first 1097 iterations.
Predicting for fold 5...
[I 2023-02-12 18:33:07,031] Trial 19 finished with value: 2.149522811224378 and parameters: {'iterations': 1100, 'learning_rate': 0.06594318493279969, 'depth': 8, 'l2_leaf_reg': 10, 'random_strength': 196.4654452403077, 'bootstrap_type': 'Bayesian', 'score_function': 'Cosine', 'bagging_temperature': 15.182113058121327}. Best is trial 3 with value: 1.9725806096138503.

```

```

In [264]: fig_catboost = optuna.visualization.plot_optimization_history(study_catboost)
fig_catboost.show();

```

The objective values do appear to be trending downwards. Perhaps with more trials allocated, we would be able to achieve finer-tuned models.

Model Training

The best parameters returned can be further fine-tuned manually. Below, we will tweak one of the hyperparameters--- lowering the learning rate.

```
In [266]: study_catboost.best_params
```

```
Out[266]: {'iterations': 1600,
           'learning_rate': 0.21065008859344805,
           'depth': 9,
           'l2_leaf_reg': 500,
           'random_strength': 122.57161445434504,
           'bootstrap_type': 'Bayesian',
           'score_function': 'L2',
           'bagging_temperature': 7.960423129118915}
```

```
In [105]: # Out-of-fold prediction dictionary
oof_catboost = {}
# Feature importance container
feat_imp_catboost = []
# K-fold cross validation
kf_catboost = KFold(n_splits=5, shuffle=True, random_state=rs)

for fold, (train_idx, val_idx) in enumerate(kf_catboost.split(X_train, y_train)):

    # Train and validation sets
    fold_X_train, fold_y_train = X_train.iloc[train_idx], y_train[train_idx]
    fold_X_val, fold_y_val = X_train.iloc[val_idx], y_train[val_idx]

    # Preprocessing using a fresh copy of the pipeline for every fold to prevent leakage
    preprocessor = joblib.load('../output/preprocessors/catboost_preprocessor.joblib')
    print(f'Start processing fold {fold + 1}...')
    fold_X_train = preprocessor.fit_transform(fold_X_train, fold_y_train)
    fold_X_val = preprocessor.transform(fold_X_val)
    # Write fitted preprocessor to disk
    joblib.dump(preprocessor, model_path + f'catboost/preprocessor_fold_{fold + 1}.joblib')

    # Data for modeling
    feature_names = fold_X_train.columns.tolist()
    dtrain = cb.Pool(data=fold_X_train, label=fold_y_train, feature_names=feature_names, cat_features=encode_cols)
    dvalid = cb.Pool(data=fold_X_val, label=fold_y_val, feature_names=feature_names, cat_features=encode_cols)

    # Model
    model = cb.train(
        params={'iterations': 1600,
               'learning_rate': 0.03,
```

```

        'depth': 9,
        'l2_leaf_reg': 500,
        'random_strength': 122.57161445434504,
        'bootstrap_type': 'Bayesian',
        'score_function': 'L2',
        'bagging_temperature': 7.960423129118915,
        'objective': 'RMSE',
        'eval_metric': 'RMSE',
        'task_type': 'GPU', # GPU training
        'border_count': 254,
        'use_best_model': True,
        'boosting_type': 'Plain'},
    dtrain=dtrain,
    early_stopping_rounds=200,
    eval_set=dvalid,
    verbose=200 # Report every 200 rounds
)
model.save_model(model_path + f'catboost/model_fold_{fold + 1}.cbm')
joblib.dump(model.get_evals_result(), model_path + f'catboost/eval_fold_{fold + 1}.joblib')

# Return feature importance as a list of (feature_id, feature importance)
feat_imp_catboost.append(model.get_feature_importance(type='FeatureImportance', prettified=True))

# Predictions
print(f'Predicting for fold {fold + 1}...')
oof_pred = model.predict(data=dvalid)
oof_catboost[f'fold_{fold + 1}'] = {'target': fold_y_val, 'predictions': oof_pred}

del dtrain, dvalid, preprocessor, model, oof_pred

```

Start processing fold 1...

Out[105]: ['../output/models/catboost/preprocessor_fold_1.joblib']

0:	learn: 4.2942860	test: 4.3481472	best: 4.3481472 (0)	total: 52.3ms	remaining: 1m 23s
200:	learn: 1.4846321	test: 2.0264708	best: 2.0264708 (200)	total: 9.6s	remaining: 1m 6s
400:	learn: 1.3742982	test: 1.9483631	best: 1.9483551 (397)	total: 19.2s	remaining: 57.4s
600:	learn: 1.3332005	test: 1.9358437	best: 1.9357838 (599)	total: 28.9s	remaining: 48.1s
800:	learn: 1.3054368	test: 1.9269489	best: 1.9263232 (799)	total: 38.4s	remaining: 38.3s
1000:	learn: 1.2833531	test: 1.9170145	best: 1.9169861 (997)	total: 48s	remaining: 28.7s
1200:	learn: 1.2680810	test: 1.9136113	best: 1.9131233 (1123)	total: 57.6s	remaining: 19.1s
1400:	learn: 1.2534695	test: 1.9083046	best: 1.9080637 (1399)	total: 1m 7s	remaining: 9.53s
1599:	learn: 1.2414106	test: 1.9044369	best: 1.9041122 (1553)	total: 1m 16s	remaining: 0us

bestTest = 1.904112249
bestIteration = 1553
Shrink model to first 1554 iterations.

Out[105]: ['../output/models/catboost/eval_fold_1.joblib']

Predicting for fold 1...
Start processing fold 2...

Out[105]: ['../output/models/catboost/preprocessor_fold_2.joblib']

0:	learn: 4.2972292	test: 4.3050925	best: 4.3050925 (0)	total: 53ms	remaining: 1m 24s
200:	learn: 1.4682031	test: 2.0364287	best: 2.0364287 (200)	total: 9.63s	remaining: 1m 6s
400:	learn: 1.3678735	test: 1.9756398	best: 1.9750068 (395)	total: 19.2s	remaining: 57.5s
600:	learn: 1.3260084	test: 1.9540601	best: 1.9532478 (596)	total: 29s	remaining: 48.1s
800:	learn: 1.2979784	test: 1.9415586	best: 1.9415586 (800)	total: 38.7s	remaining: 38.6s
1000:	learn: 1.2782562	test: 1.9310207	best: 1.9309814 (995)	total: 48.4s	remaining: 29s
1200:	learn: 1.2642259	test: 1.9257172	best: 1.9255771 (1194)	total: 58.2s	remaining: 19.3s
1400:	learn: 1.2513337	test: 1.9208633	best: 1.9208505 (1399)	total: 1m 7s	remaining: 9.65s
1599:	learn: 1.2388092	test: 1.9146963	best: 1.9146254 (1597)	total: 1m 17s	remaining: 0us

bestTest = 1.914625425
bestIteration = 1597
Shrink model to first 1598 iterations.

Out[105]: ['../output/models/catboost/eval_fold_2.joblib']

Predicting for fold 2...
Start processing fold 3...

Out[105]: ['../output/models/catboost/preprocessor_fold_3.joblib']

0:	learn: 4.2973890	test: 4.2776678	best: 4.2776678 (0)	total: 53.5ms	remaining: 1m 25s
200:	learn: 1.4602346	test: 2.0156389	best: 2.0156389 (200)	total: 9.72s	remaining: 1m 7s
400:	learn: 1.3690786	test: 1.9731652	best: 1.9728916 (393)	total: 19.4s	remaining: 58.1s
600:	learn: 1.3264653	test: 1.9573255	best: 1.9573244 (599)	total: 29.1s	remaining: 48.4s
800:	learn: 1.2982251	test: 1.9470217	best: 1.9470099 (794)	total: 38.6s	remaining: 38.5s
1000:	learn: 1.2770995	test: 1.9388541	best: 1.9387614 (992)	total: 48.4s	remaining: 28.9s
1200:	learn: 1.2603996	test: 1.9342572	best: 1.9342572 (1200)	total: 58s	remaining: 19.3s
1400:	learn: 1.2468703	test: 1.9308011	best: 1.9308011 (1400)	total: 1m 7s	remaining: 9.62s
1599:	learn: 1.2345420	test: 1.9267397	best: 1.9267269 (1598)	total: 1m 17s	remaining: 0us

bestTest = 1.926726947
bestIteration = 1598
Shrink model to first 1599 iterations.

Out[105]: ['../output/models/catboost/eval_fold_3.joblib']

Predicting for fold 3...
Start processing fold 4...

```
Out[105]: ['../output/models/catboost/preprocessor_fold_4.joblib']

0:      learn: 4.2942688      test: 4.3187485 best: 4.3187485 (0)      total: 51.9ms      remaining: 1m 22s
200:    learn: 1.4656432      test: 1.9762537 best: 1.9762537 (200)    total: 9.56s      remaining: 1m 6s
400:    learn: 1.3677487      test: 1.9062980 best: 1.9062980 (400)    total: 19.1s      remaining: 57.1s
600:    learn: 1.3255955      test: 1.8836513 best: 1.8836513 (600)    total: 28.7s      remaining: 47.7s
800:    learn: 1.2981092      test: 1.8679248 best: 1.8679206 (799)    total: 38.2s      remaining: 38.1s
1000:   learn: 1.2764594      test: 1.8567059 best: 1.8567059 (1000)  total: 47.9s      remaining: 28.6s
1200:   learn: 1.2590459      test: 1.8482693 best: 1.8482693 (1200)  total: 57.5s      remaining: 19.1s
1400:   learn: 1.2470191      test: 1.8418291 best: 1.8418291 (1400)  total: 1m 7s      remaining: 9.53s
1599:   learn: 1.2350771      test: 1.8354936 best: 1.8353143 (1578)  total: 1m 16s     remaining: 0us
bestTest = 1.835314325
bestIteration = 1578
Shrink model to first 1579 iterations.
```

```
Out[105]: ['../output/models/catboost/eval_fold_4.joblib']

Predicting for fold 4...
Start processing fold 5...
```

```
Out[105]: ['../output/models/catboost/preprocessor_fold_5.joblib']

0:      learn: 4.3050323      test: 4.3343444 best: 4.3343444 (0)      total: 50.8ms      remaining: 1m 21s
200:    learn: 1.4696808      test: 2.1308574 best: 2.1308574 (200)    total: 9.72s      remaining: 1m 7s
400:    learn: 1.3702882      test: 2.0761662 best: 2.0761662 (400)    total: 19.3s      remaining: 57.7s
600:    learn: 1.3224202      test: 2.0474226 best: 2.0473729 (599)    total: 28.9s      remaining: 48.1s
800:    learn: 1.2930671      test: 2.0327357 best: 2.0327357 (800)    total: 38.5s      remaining: 38.4s
1000:   learn: 1.2723030      test: 2.0214952 best: 2.0214952 (1000)  total: 48s        remaining: 28.8s
1200:   learn: 1.2560228      test: 2.0134869 best: 2.0134794 (1199)  total: 57.8s      remaining: 19.2s
1400:   learn: 1.2421490      test: 2.0087456 best: 2.0086940 (1396)  total: 1m 7s      remaining: 9.58s
1599:   learn: 1.2323502      test: 2.0049508 best: 2.0049508 (1599)  total: 1m 16s     remaining: 0us
bestTest = 2.004950766
bestIteration = 1599
```

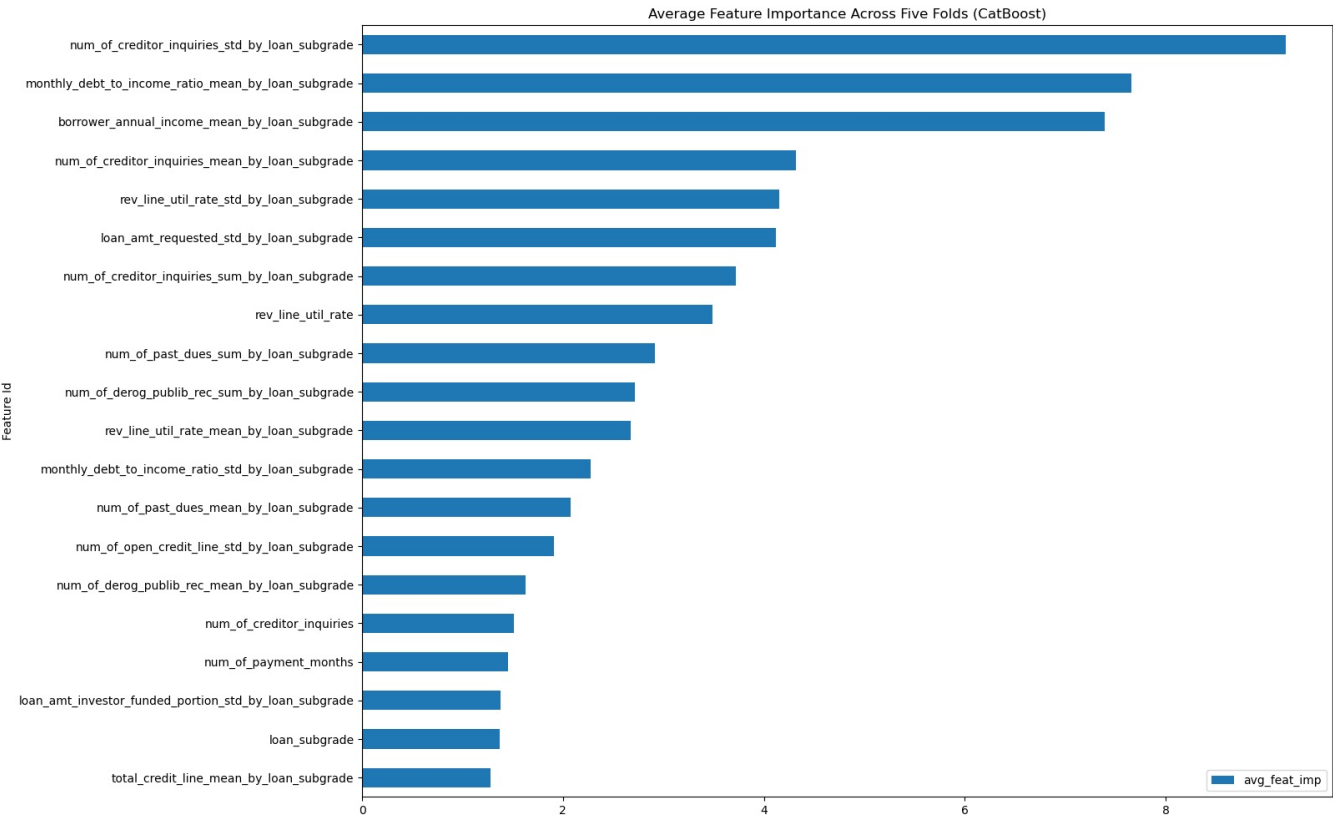
```
Out[105]: ['../output/models/catboost/eval_fold_5.joblib']

Predicting for fold 5...
```

Feature Importance

```
In [106.. # Join feature importance
feat_imp_catboost = reduce(lambda x, y: pd.merge(x, y, on='Feature Id', how='left'), feat_imp_catboost)
feat_imp_catboost['avg_feat_imp'] = feat_imp_catboost.iloc[:, 1:].apply(lambda row: row.mean(), axis=1)

# Plot top feature importance
feat_imp_catboost.sort_values(by='avg_feat_imp', ascending=True).iloc[-20:].plot(
    kind='barh', x='Feature Id', y='avg_feat_imp',
    figsize=(15, 12),
    title='Average Feature Importance Across Five Folds (CatBoost)'
)
plt.show();
```



Again, similar to the output of XGBoost, the loan subgrade feature is of crucial importance. Many of the generated features based on this

grade feature are also ranked highly in terms of importance.

Learning Curves

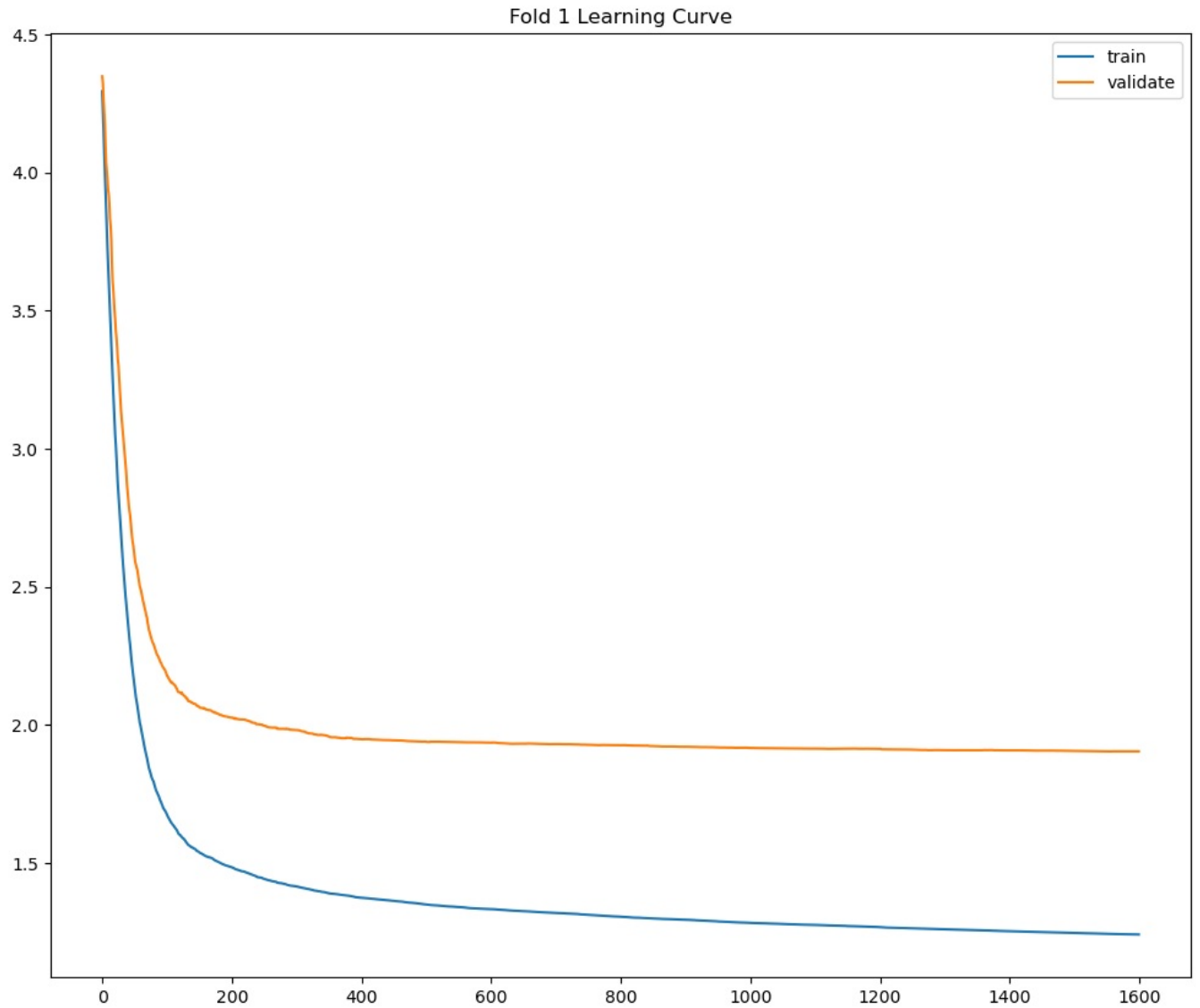
```
In [107]: for fold in range(5):  
            eval_result = joblib.load(model_path + f'catboost/eval_fold_{fold + 1}.joblib')  
            plt.plot(eval_result['learn']['RMSE'], label='train');  
            plt.plot(eval_result['validation']['RMSE'], label='validate');  
            plt.legend();  
            plt.title(f'Fold {fold + 1} Learning Curve');  
            plt.show();
```

Out[107]: [<matplotlib.lines.Line2D at 0x7f2364608a30>]

Out[107]: [<matplotlib.lines.Line2D at 0x7f2364608df0>]

Out[107]: <matplotlib.legend.Legend at 0x7f2364608cd0>

Out[107]: Text(0.5, 1.0, 'Fold 1 Learning Curve')

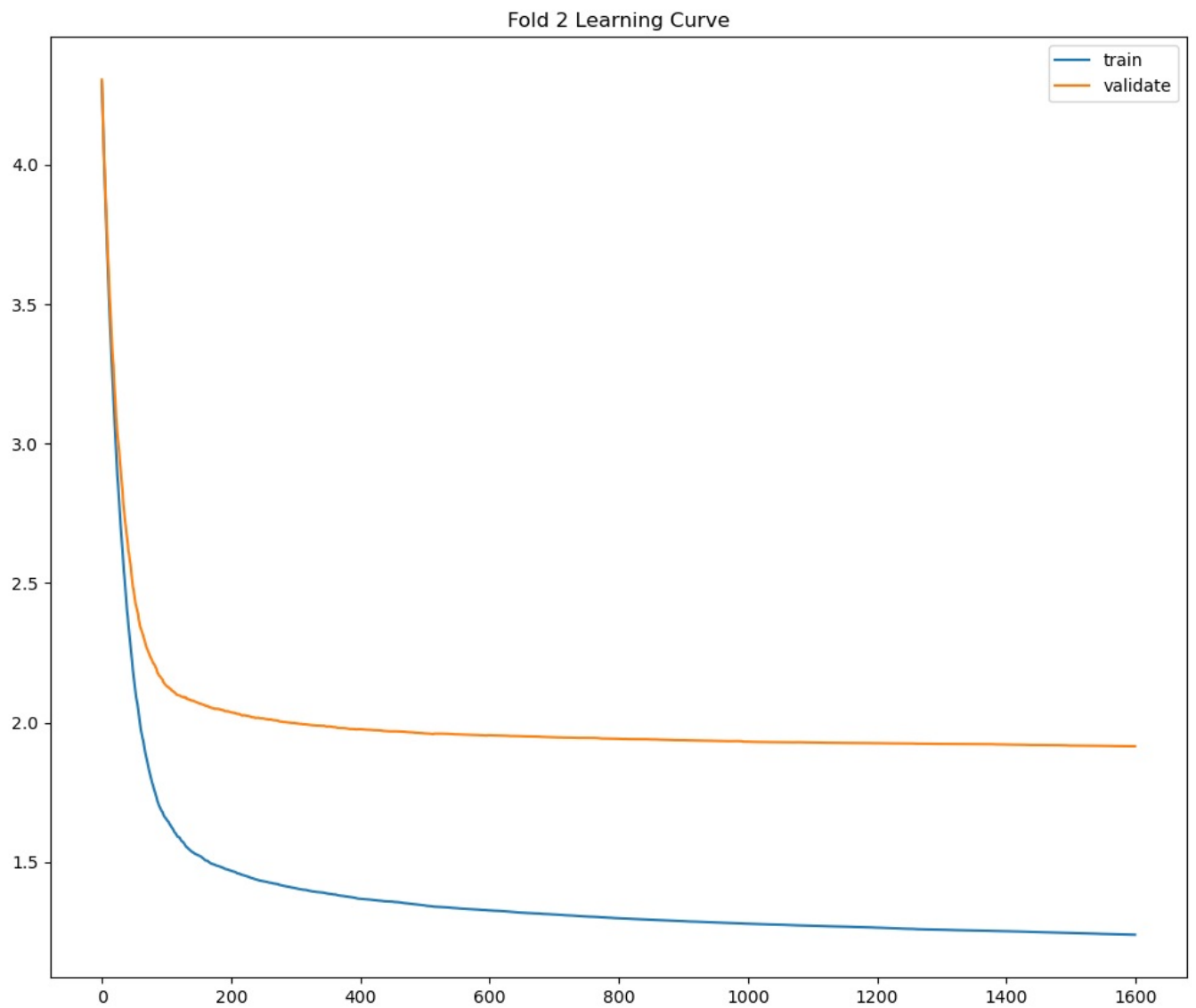


Out[107]: [<matplotlib.lines.Line2D at 0x7f2362a125e0>]

Out[107]: [<matplotlib.lines.Line2D at 0x7f2362a12970>]

Out[107]: <matplotlib.legend.Legend at 0x7f2362a12850>

Out[107]: Text(0.5, 1.0, 'Fold 2 Learning Curve')

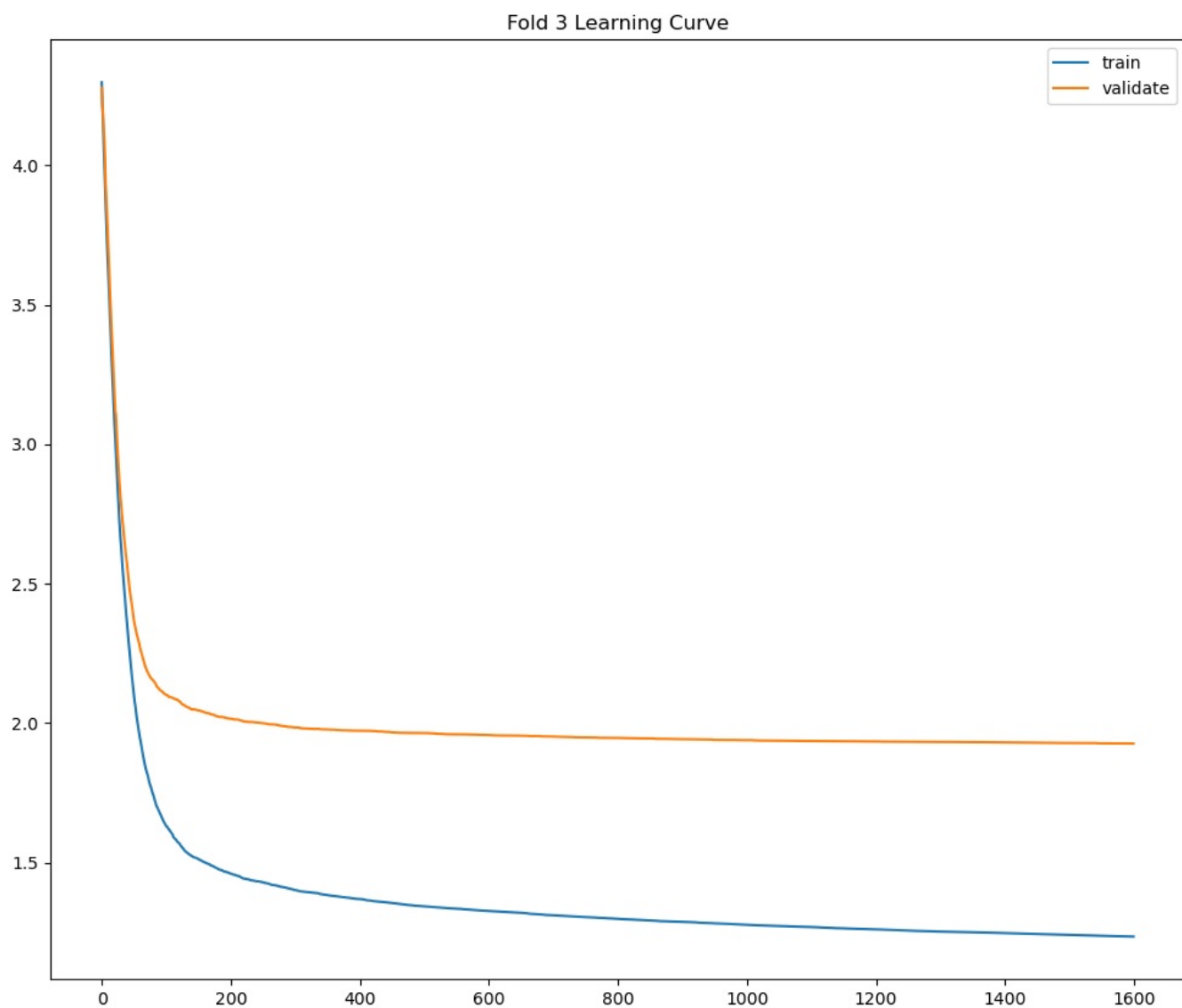


Out[107]: [<matplotlib.lines.Line2D at 0x7f23633d3f10>]

Out[107]: [<matplotlib.lines.Line2D at 0x7f23633ec1f0>]

Out[107]: <matplotlib.legend.Legend at 0x7f236420b7c0>

Out[107]: Text(0.5, 1.0, 'Fold 3 Learning Curve')

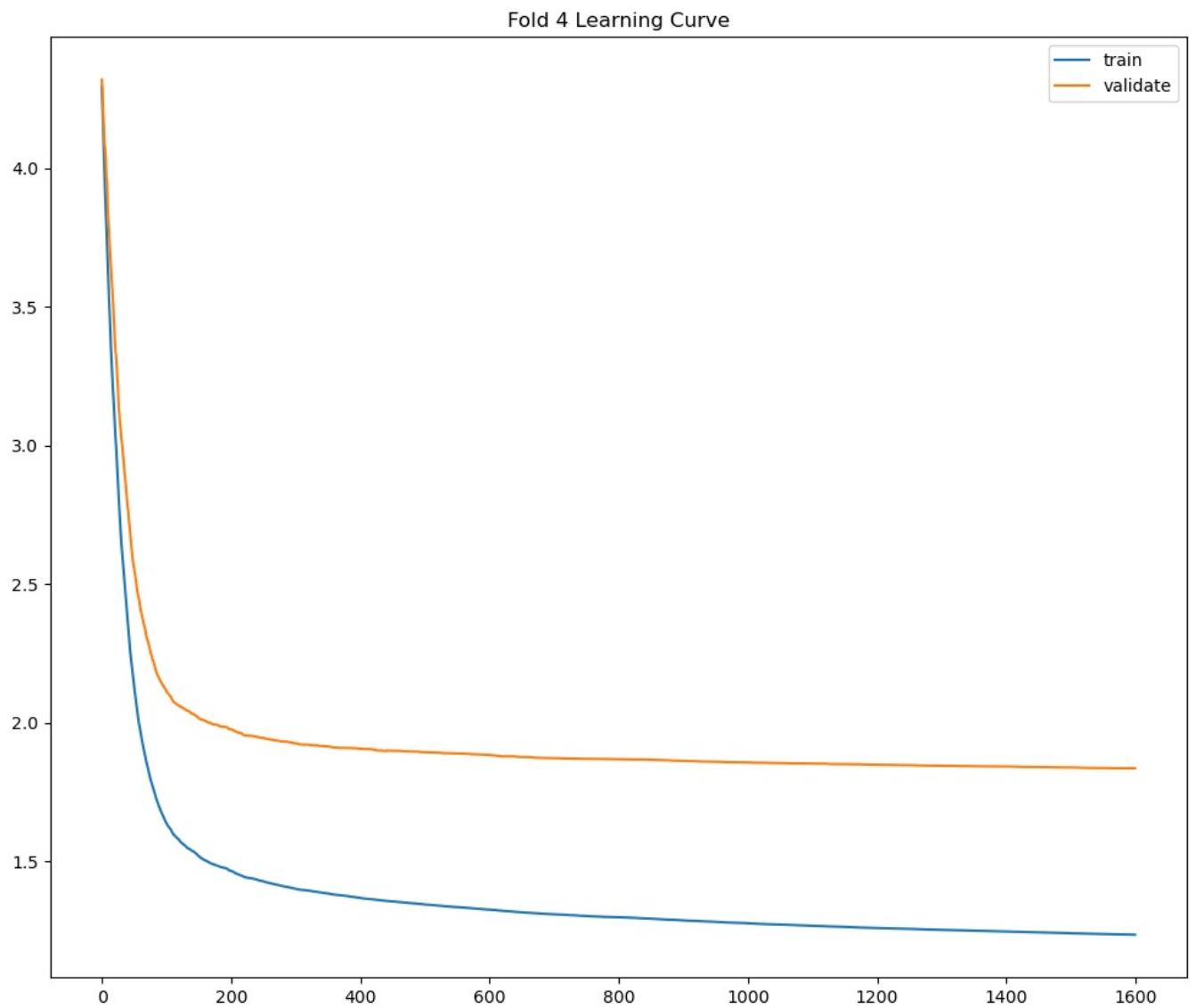


Out[107]: [<matplotlib.lines.Line2D at 0x7f2363725190>]

Out[107]: [<matplotlib.lines.Line2D at 0x7f2363725400>]

Out[107]: <matplotlib.legend.Legend at 0x7f2364879580>

Out[107]: Text(0.5, 1.0, 'Fold 4 Learning Curve')

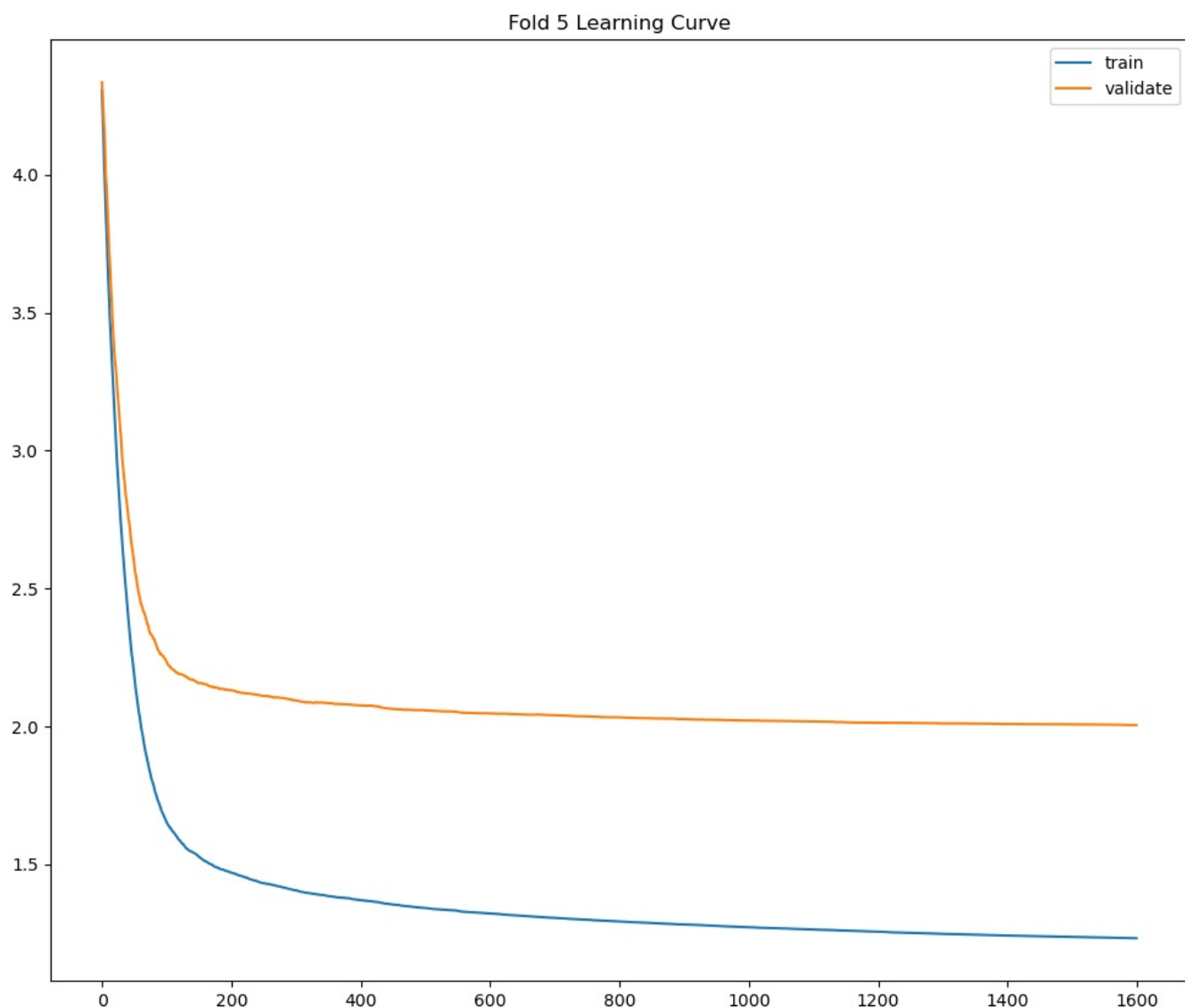


Out[107]: [<matplotlib.lines.Line2D at 0x7f2364c5ff10>]

Out[107]: [<matplotlib.lines.Line2D at 0x7f2364c4e1c0>]

Out[107]: <matplotlib.legend.Legend at 0x7f2364c5ff70>

Out[107]: Text(0.5, 1.0, 'Fold 5 Learning Curve')



Contrary to XGBoost, the learning curves for CatBoost show that the models begin to overfit as soon as we reach about 200 rounds. We also set the parameter `use_best_model` to true in the train method to identify the iteration with the optimal value of the metric.

Performance on Validation Sets

```
In [108]: oof_catboost_rmse = []
target_frame = cudf.DataFrame(index=['count', 'mean', 'std', 'min', '25%', '50%', '75%', 'max'])

for key in oof_catboost:
    oof_catboost_rmse.append(
        mean_squared_error(oof_catboost[key]['target'], oof_catboost[key]['predictions'], squared=False)
    )
    print(f'Finished computing rmse for {key}')

    target_frame[f'{key}_target_descriptive_stats'] = cudf.Series(oof_catboost[key]['target']).describe()
    print(f'Finished computing descriptive stats for {key} target')
```

```

Finished computing rmse for fold_1
Finished computing descriptive stats for fold_1 target
Finished computing rmse for fold_2
Finished computing descriptive stats for fold_2 target
Finished computing rmse for fold_3
Finished computing descriptive stats for fold_3 target
Finished computing rmse for fold_4
Finished computing descriptive stats for fold_4 target
Finished computing rmse for fold_5
Finished computing descriptive stats for fold_5 target

```

```
In [109]: cdf.Series(oof_catboost_rmse).describe()
```

```

Out[109]: count    5.000000
          mean     1.917146
          std      0.060569
          min     1.835314
          25%     1.904114
          50%     1.914625
          75%     1.926727
          max     2.004952
          dtype: float64

```

```
In [301]: target_frame
```

```

Out[301]:

```

	fold_1_target_descriptive_stats	fold_2_target_descriptive_stats	fold_3_target_descriptive_stats	fold_4_target_descriptive_stats	fold_5_target_descriptive_stats
count	67798.000000	67798.000000	67798.000000	67797.000000	67797.000000
mean	13.946219	13.940915	13.924338	13.952788	13.952788
std	4.354851	4.380408	4.363100	4.404952	4.404952
min	5.420000	5.420000	5.420000	5.420000	5.420000
25%	10.990000	10.990000	10.990000	10.990000	10.990000
50%	13.980000	13.680000	13.670000	13.680000	13.680000
75%	16.780000	16.780000	16.770000	16.780000	16.780000
max	26.060000	26.060000	26.060000	26.060000	26.060000

On average, we are off by \$1.913222\$ percentage points. This value is higher than that of XGBoost. However, cross-validation scores are usually better than the real test scores anyways, since it is likely that our system is fine-tuned to perform well on the validation data but will likely not perform as well on unknown datasets. Therefore, these models may perform better on certain training examples than the XGBoost models even when their performances on the validation sets are relatively worse.

LightGBM

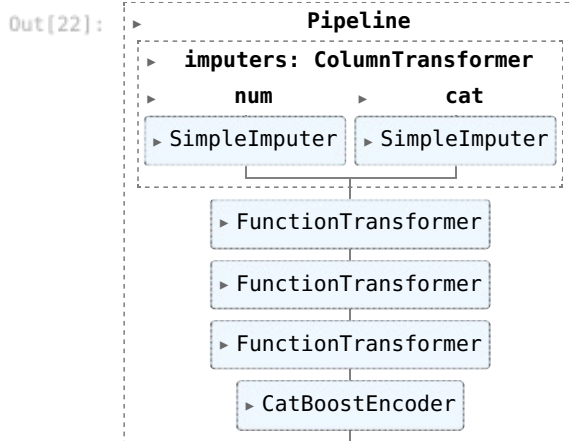
Pipeline

```

In [22]: lightgbm_preprocessor = Pipeline([
          ('imputers', imputers),
          ('restore_cols', FunctionTransformer(pp.restore_columns)),
          ('date_transformer', FunctionTransformer(pp.extract_date_features)),
          ('num_feat_eng', FunctionTransformer(pp.num_feat_eng)),
          ('cat_encoder', CatBoostEncoder(cols=encode_cols, handle_missing='value', handle_unknown='value'))
        ])
joblib.dump(lightgbm_preprocessor, prep_path + 'lightgbm_preprocessor.joblib')
lightgbm_preprocessor

```

```
Out[22]: ['../output/preprocessors/lightgbm_preprocessor.joblib']
```



Hyperparameter Search

```
In [44]: def objective_lightgbm(trial):

    # Fold and seed
    train = pd.read_csv("../data/train_sanitized.csv")
    X_train, y_train = train.drop(['interest_rate'], axis=1), train.interest_rate.to_numpy()
    folds = 5
    seed = 1227

    # Parameters
    search_space = {
        'objective': 'rmse',
        'metric': 'rmse',
        'device_type': 'gpu',
        'verbosity': -1,
        'early_stopping_round': 200,
        'boosting': 'gbdt',
        # For better accuracy
        'num_iterations': trial.suggest_int('num_iterations', low=500, high=2000, step=100), # Range: [0, inf],
        'learning_rate': trial.suggest_float(name='learning_rate', low=0.001, high=0.1), # Shrinkage rate
        'num_leaves': trial.suggest_int('num_leaves', 31, 100), # Constrained: 1 < num_leaves <= 131072, max num
        # Regularizers
        'max_depth': trial.suggest_int('max_depth', low=4, high=12), # Regularizer that controls max depth for
        'max_bin': trial.suggest_int('max_bin', low=150, high=255), # Constrained: max_bin > 1, small values may
        'bagging_fraction': trial.suggest_float('bagging_fraction', 0.1, 0.6), # Constrained: 0.0 < bagging_fra
        'bagging_freq': trial.suggest_int('bagging_freq', 20, 100), # Every k-th iteration, LightGBM will random
        'feature_fraction': trial.suggest_float('feature_fraction', 0.1, 0.6), # Constrained: 0.0 < feature_fra
        'feature_fraction_bynode': trial.suggest_float('feature_fraction_bynode', 0.1, 0.6), # Constrained: 0.0
        'lambda_l1': trial.suggest_int('lambda_l1', low=100, high=1000), # Constrained: lambda_l1 >= 0.0 (regul
        'lambda_l2': trial.suggest_int('lambda_l2', low=100, high=1000), # Constrained: lambda_l2 >= 0.0 (regul
        'extra_trees': trial.suggest_categorical('extra_trees', [True, False]), # If set to true, when evaluati
        'path_smooth': trial.suggest_int('path_smooth', low=100, high=1000) # Controls smoothing applied to tree
    }

    # K-fold cross validation
    kf = KFold(n_splits=folds, shuffle=True, random_state=rs)
    rmse_scores = np.empty(folds)

    for fold, (train_idx, val_idx) in enumerate(kf.split(X_train, y_train)):

        # Train and validation sets
        fold_X_train, fold_y_train = X_train.iloc[train_idx], y_train[train_idx]
        fold_X_val, fold_y_val = X_train.iloc[val_idx], y_train[val_idx]

        # Preprocessing using a fresh copy of the pipeline for every fold to prevent leakage
        preprocessor = joblib.load('../output/preprocessors/lightgbm_preprocessor.joblib')
        print(f'Start processing fold {fold + 1}...')
        fold_X_train = preprocessor.fit_transform(fold_X_train, fold_y_train)
        fold_X_val = preprocessor.transform(fold_X_val)

        # Data for modeling
        feature_names = fold_X_train.columns.tolist()
        dtrain = lgb.Dataset(data=fold_X_train, label=fold_y_train, feature_name=feature_names)
        dvalid = lgb.Dataset(data=fold_X_val, label=fold_y_val, feature_name=feature_names, reference=dtrain)

        # Model
        model = lgb.train(
            params=search_space,
            train_set=dtrain,
            valid_sets=[dtrain, dvalid],
            valid_names=['train', 'valid'],
            callbacks=[lgb.log_evaluation(period=200), lgb.early_stopping(stopping_rounds=200)] # Log evaluation
        )

        # Out-of-fold prediction
        print(f'Predicting for fold {fold + 1}...')
        oof_pred = model.predict(data=fold_X_val)
        rmse_scores[fold] = mean_squared_error(fold_y_val, oof_pred, squared=False) # Use RMSE

    # Average across 5 folds
    mean_rmse = np.mean(rmse_scores)

    return mean_rmse
```

```
In [ ]: study_lightgbm = optuna.create_study(sampler=optuna.samplers.TPESampler(), study_name='min_rmse_lightgbm', dire
study_lightgbm.optimize(objective_lightgbm, n_trials=20)
```

[I 2023-02-13 05:49:58,496] A new study created in memory with name: min_rmse_lightgbm

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.64927 valid's rmse: 2.41888
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.67731 valid's rmse: 2.05804
[400] train's rmse: 1.61628 valid's rmse: 2.03081
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.67464 valid's rmse: 2.09431
Early stopping, best iteration is:
[120] train's rmse: 1.73483 valid's rmse: 2.07136
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.69319 valid's rmse: 2.41033
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.68245 valid's rmse: 2.7383
[400] train's rmse: 1.62956 valid's rmse: 2.70514
[600] train's rmse: 1.59952 valid's rmse: 2.69378
[800] train's rmse: 1.57291 valid's rmse: 2.67713
[1000] train's rmse: 1.56285 valid's rmse: 2.67525
Predicting for fold 5...
```

```
[I 2023-02-13 05:52:13,914] Trial 0 finished with value: 2.310950566601096 and parameters: {'num_iterations': 1900, 'learning_rate': 0.054172139977479834, 'num_leaves': 66, 'max_depth': 10, 'max_bin': 204, 'bagging_fraction': 0.48543255102322513, 'bagging_freq': 68, 'feature_fraction': 0.3114237062389975, 'feature_fraction_bynode': 0.1277763428763173, 'lambda_l1': 780, 'lambda_l2': 730, 'extra_trees': True, 'path_smooth': 455}. Best is trial 0 with value: 2.310950566601096.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.56156 valid's rmse: 2.16705
Early stopping, best iteration is:
[60] train's rmse: 1.75489 valid's rmse: 2.13457
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.55126 valid's rmse: 2.23048
[400] train's rmse: 1.46439 valid's rmse: 2.21595
[600] train's rmse: 1.42421 valid's rmse: 2.19913
[800] train's rmse: 1.40755 valid's rmse: 2.19024
[1000] train's rmse: 1.39621 valid's rmse: 2.18313
[1200] train's rmse: 1.39142 valid's rmse: 2.17547
[1400] train's rmse: 1.3874 valid's rmse: 2.17116
[1600] train's rmse: 1.38455 valid's rmse: 2.1707
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.55056 valid's rmse: 2.2756
Early stopping, best iteration is:
[46] train's rmse: 1.84841 valid's rmse: 2.14609
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.55253 valid's rmse: 2.19018
[400] train's rmse: 1.45906 valid's rmse: 2.14005
[600] train's rmse: 1.42694 valid's rmse: 2.12182
[800] train's rmse: 1.40526 valid's rmse: 2.10873
[1000] train's rmse: 1.39055 valid's rmse: 2.10049
[1200] train's rmse: 1.38353 valid's rmse: 2.09632
[1400] train's rmse: 1.37932 valid's rmse: 2.09187
[1600] train's rmse: 1.37403 valid's rmse: 2.08828
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.55719 valid's rmse: 2.46119
Predicting for fold 5...
```

```
[I 2023-02-13 05:54:43,899] Trial 1 finished with value: 2.168692672835806 and parameters: {'num_iterations': 1700, 'learning_rate': 0.050868051185383324, 'num_leaves': 32, 'max_depth': 9, 'max_bin': 227, 'bagging_fraction': 0.5913533561366116, 'bagging_freq': 87, 'feature_fraction': 0.28333655555498927, 'feature_fraction_bynode': 0.4234284426180617, 'lambda_l1': 770, 'lambda_l2': 530, 'extra_trees': True, 'path_smooth': 782}. Best is trial 1 with value: 2.168692672835806.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.46234 valid's rmse: 2.35082
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.46204 valid's rmse: 2.4012
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.46884 valid's rmse: 2.63302
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.46269 valid's rmse: 2.71386
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.46037 valid's rmse: 2.49855
Early stopping, best iteration is:
[50] train's rmse: 1.71775 valid's rmse: 2.45922
Predicting for fold 5...
```

```
[I 2023-02-13 05:56:50,761] Trial 2 finished with value: 2.4217747108098626 and parameters: {'num_iterations': 700, 'learning_rate': 0.054821458866835995, 'num_leaves': 37, 'max_depth': 8, 'max_bin': 188, 'bagging_fraction': 0.491918566176633, 'bagging_freq': 57, 'feature_fraction': 0.3806587982326842, 'feature_fraction_bynode': 0.4716864331566988, 'lambda_l1': 679, 'lambda_l2': 110, 'extra_trees': True, 'path_smooth': 573}. Best is trial 1 with value: 2.168692672835806.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.3906 valid's rmse: 2.30627
Early stopping, best iteration is:
[59] train's rmse: 1.72789 valid's rmse: 2.25125
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.38302 valid's rmse: 2.14953
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.38137 valid's rmse: 2.27134
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.38463 valid's rmse: 2.35745
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.38301 valid's rmse: 2.23029
Predicting for fold 5...
```

```
[I 2023-02-13 05:58:57,664] Trial 3 finished with value: 2.166137540986789 and parameters: {'num_iterations': 1200, 'learning_rate': 0.03907785871161757, 'num_leaves': 98, 'max_depth': 6, 'max_bin': 205, 'bagging_fraction': 0.2127273020483946, 'bagging_freq': 30, 'feature_fraction': 0.5041493984213664, 'feature_fraction_bynode': 0.4668826599178957, 'lambda_l1': 989, 'lambda_l2': 693, 'extra_trees': False, 'path_smooth': 159}. Best is trial 3 with value: 2.166137540986789.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.21362 valid's rmse: 2.23698
[400] train's rmse: 1.15776 valid's rmse: 2.22505
[600] train's rmse: 1.14455 valid's rmse: 2.2162
[800] train's rmse: 1.14037 valid's rmse: 2.21041
[1000] train's rmse: 1.13969 valid's rmse: 2.20694
[1200] train's rmse: 1.13928 valid's rmse: 2.20715
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.21974 valid's rmse: 2.50154
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.21372 valid's rmse: 2.21689
[400] train's rmse: 1.16279 valid's rmse: 2.20191
[600] train's rmse: 1.14748 valid's rmse: 2.19061
[800] train's rmse: 1.14375 valid's rmse: 2.18551
[1000] train's rmse: 1.14373 valid's rmse: 2.18475
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.21606 valid's rmse: 2.52833
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.21766 valid's rmse: 2.26421
Predicting for fold 5...
```

[I 2023-02-13 06:01:21,314] Trial 4 finished with value: 2.3095182634840503 and parameters: {'num_iterations': 1900, 'learning_rate': 0.08366112060614134, 'num_leaves': 49, 'max_depth': 6, 'max_bin': 156, 'bagging_fraction': 0.25077131876786296, 'bagging_freq': 78, 'feature_fraction': 0.2768133548592929, 'feature_fraction_bynode': 0.5015097305437328, 'lambda_l1': 303, 'lambda_l2': 641, 'extra_trees': False, 'path_smooth': 454}. Best is trial 3 with value: 2.166137540986789.

Start processing fold 1...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.70803	valid's rmse: 2.21066
[400]	train's rmse: 1.60585	valid's rmse: 2.19251
[600]	train's rmse: 1.56459	valid's rmse: 2.17857
[800]	train's rmse: 1.53477	valid's rmse: 2.16581
[1000]	train's rmse: 1.51526	valid's rmse: 2.15955
[1200]	train's rmse: 1.50354	valid's rmse: 2.15164
[1400]	train's rmse: 1.49296	valid's rmse: 2.14384
[1600]	train's rmse: 1.47851	valid's rmse: 2.13366
[1800]	train's rmse: 1.47303	valid's rmse: 2.13046

Predicting for fold 1...

Start processing fold 2...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.68307	valid's rmse: 2.26088
-------	-----------------------	-----------------------

Predicting for fold 2...

Start processing fold 3...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.69058	valid's rmse: 2.37223
-------	-----------------------	-----------------------

Predicting for fold 3...

Start processing fold 4...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.68421	valid's rmse: 2.29047
[400]	train's rmse: 1.59012	valid's rmse: 2.28373
[600]	train's rmse: 1.55046	valid's rmse: 2.26423
[800]	train's rmse: 1.52515	valid's rmse: 2.24627
[1000]	train's rmse: 1.50481	valid's rmse: 2.2348
[1200]	train's rmse: 1.49213	valid's rmse: 2.22473
[1400]	train's rmse: 1.47865	valid's rmse: 2.22082
[1600]	train's rmse: 1.4727	valid's rmse: 2.21543
[1800]	train's rmse: 1.46705	valid's rmse: 2.21246

Predicting for fold 4...

Start processing fold 5...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.67352	valid's rmse: 2.30415
-------	-----------------------	-----------------------

Predicting for fold 5...

[I 2023-02-13 06:03:56,866] Trial 5 finished with value: 2.2396725821774566 and parameters: {'num_iterations': 1800, 'learning_rate': 0.021046353497940293, 'num_leaves': 32, 'max_depth': 7, 'max_bin': 224, 'bagging_fraction': 0.4231621172767821, 'bagging_freq': 70, 'feature_fraction': 0.35761730860375707, 'feature_fraction_bynode': 0.3400963668981716, 'lambda_l1': 965, 'lambda_l2': 238, 'extra_trees': True, 'path_smooth': 716}. Best is trial 3 with value: 2.166137540986789.

Start processing fold 1...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.49569	valid's rmse: 2.35132
-------	-----------------------	-----------------------

Early stopping, best iteration is:

[62]	train's rmse: 1.72761	valid's rmse: 2.33279
------	-----------------------	-----------------------

Predicting for fold 1...

Start processing fold 2...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.49979	valid's rmse: 2.45678
-------	-----------------------	-----------------------

Early stopping, best iteration is:

[60]	train's rmse: 1.7214	valid's rmse: 2.38647
------	----------------------	-----------------------

Predicting for fold 2...

Start processing fold 3...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.55534	valid's rmse: 2.77
[400]	train's rmse: 1.46394	valid's rmse: 2.75881
[600]	train's rmse: 1.42407	valid's rmse: 2.73325
[800]	train's rmse: 1.38915	valid's rmse: 2.72983
[1000]	train's rmse: 1.36196	valid's rmse: 2.72108
[1200]	train's rmse: 1.3438	valid's rmse: 2.71692
[1400]	train's rmse: 1.32769	valid's rmse: 2.70267

Did not meet early stopping. Best iteration is:

[1400]	train's rmse: 1.32769	valid's rmse: 2.70267
--------	-----------------------	-----------------------

Predicting for fold 3...

Start processing fold 4...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.48978	valid's rmse: 2.59573
-------	-----------------------	-----------------------

Predicting for fold 4...

Start processing fold 5...

Training until validation scores don't improve for 200 rounds

[200]	train's rmse: 1.55247	valid's rmse: 2.52909
-------	-----------------------	-----------------------

Early stopping, best iteration is:

[48]	train's rmse: 1.84192	valid's rmse: 2.40486
------	-----------------------	-----------------------

Predicting for fold 5...

[I 2023-02-13 06:06:19,524] Trial 6 finished with value: 2.478435536361656 and parameters: {'num_iterations': 1400, 'learning_rate': 0.0544956777620593, 'num_leaves': 78, 'max_depth': 8, 'max_bin': 177, 'bagging_fraction': 0.5806205714680905, 'bagging_freq': 74, 'feature_fraction': 0.15273229633909682, 'feature_fraction_bynode': 0.5282478549298597, 'lambda_l1': 158, 'lambda_l2': 347, 'extra_trees': True, 'path_smooth': 379}. Best is trial 3 with value: 2.166137540986789.

Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.75339 valid's rmse: 3.81066
[400] train's rmse: 3.2699 valid's rmse: 3.38635
[600] train's rmse: 2.90753 valid's rmse: 3.0685
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.77072 valid's rmse: 3.83586
[400] train's rmse: 3.25832 valid's rmse: 3.37997
[600] train's rmse: 2.88111 valid's rmse: 3.06251
Did not meet early stopping. Best iteration is:
[700] train's rmse: 2.73521 valid's rmse: 2.94747
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.75016 valid's rmse: 3.87192
[400] train's rmse: 3.25717 valid's rmse: 3.44222
[600] train's rmse: 2.88062 valid's rmse: 3.12311
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.69123 valid's rmse: 3.78724
[400] train's rmse: 3.20183 valid's rmse: 3.37638
[600] train's rmse: 2.82453 valid's rmse: 3.0814
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.74662 valid's rmse: 3.80306
[400] train's rmse: 3.22761 valid's rmse: 3.35421
[600] train's rmse: 2.85073 valid's rmse: 3.02627
Did not meet early stopping. Best iteration is:
[700] train's rmse: 2.70332 valid's rmse: 2.90901
Predicting for fold 5...

[I 2023-02-13 06:08:51,114] Trial 7 finished with value: 2.9559594725688063 and parameters: {'num_iterations': 700, 'learning_rate': 0.002361148499413034, 'num_leaves': 78, 'max_depth': 11, 'max_bin': 186, 'bagging_fraction': 0.36098437826641994, 'bagging_freq': 55, 'feature_fraction': 0.16251137303935112, 'feature_fraction_bynode': 0.1735965200394863, 'lambda_l1': 991, 'lambda_l2': 100, 'extra_trees': True, 'path_smooth': 619}. Best is trial 3 with value: 2.166137540986789.

Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.56168 valid's rmse: 2.66043
Early stopping, best iteration is:
[25] train's rmse: 2.16066 valid's rmse: 2.60515
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.57295 valid's rmse: 2.27199
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.53645 valid's rmse: 2.37533
Early stopping, best iteration is:
[55] train's rmse: 1.73999 valid's rmse: 2.34478
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.55742 valid's rmse: 2.25673
Early stopping, best iteration is:
[64] train's rmse: 1.75157 valid's rmse: 2.19077
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.5606 valid's rmse: 2.01204
[400] train's rmse: 1.48873 valid's rmse: 1.97625
[600] train's rmse: 1.46658 valid's rmse: 1.96355
[800] train's rmse: 1.45187 valid's rmse: 1.95196
[1000] train's rmse: 1.44311 valid's rmse: 1.94773
[1200] train's rmse: 1.43872 valid's rmse: 1.94485
[1400] train's rmse: 1.43285 valid's rmse: 1.93815
[1600] train's rmse: 1.42943 valid's rmse: 1.9334
[1800] train's rmse: 1.42654 valid's rmse: 1.93178
Did not meet early stopping. Best iteration is:
[1896] train's rmse: 1.42544 valid's rmse: 1.93039
Predicting for fold 5...

[I 2023-02-13 06:11:06,467] Trial 8 finished with value: 2.263700901979423 and parameters: {'num_iterations': 1900, 'learning_rate': 0.07431865160244198, 'num_leaves': 74, 'max_depth': 12, 'max_bin': 157, 'bagging_fraction': 0.38119792838663136, 'bagging_freq': 37, 'feature_fraction': 0.29175219304369626, 'feature_fraction_bynode': 0.3158936152544254, 'lambda_l1': 630, 'lambda_l2': 919, 'extra_trees': True, 'path_smooth': 862}. Best is trial 3 with value: 2.166137540986789.

Start processing fold 1...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.68182 valid's rmse: 2.25289

Predicting for fold 1...

Start processing fold 2...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.6886 valid's rmse: 2.20654

Early stopping, best iteration is:

[189] train's rmse: 1.71437 valid's rmse: 2.20096

Predicting for fold 2...

Start processing fold 3...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.68687 valid's rmse: 2.08613

Early stopping, best iteration is:

[158] train's rmse: 1.80694 valid's rmse: 2.06761

Predicting for fold 3...

Start processing fold 4...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.67712 valid's rmse: 2.20878

Early stopping, best iteration is:

[195] train's rmse: 1.68706 valid's rmse: 2.20542

Predicting for fold 4...

Start processing fold 5...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.67569 valid's rmse: 2.19507

Early stopping, best iteration is:

[155] train's rmse: 1.80677 valid's rmse: 2.18331

Predicting for fold 5...

[I 2023-02-13 06:13:27,869] Trial 9 finished with value: 2.1758976464437363 and parameters: {'num_iterations': 2000, 'learning_rate': 0.012857540941278965, 'num_leaves': 40, 'max_depth': 10, 'max_bin': 236, 'bagging_fraction': 0.49238374442513777, 'bagging_freq': 42, 'feature_fraction': 0.4597985102238875, 'feature_fraction_bynode': 0.31411235562741135, 'lambda_l1': 138, 'lambda_l2': 147, 'extra_trees': True, 'path_smooth': 466}. Best is trial 3 with value: 2.166137540986789.

Start processing fold 1...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.48917 valid's rmse: 2.0923

[400] train's rmse: 1.3678 valid's rmse: 2.05314

[600] train's rmse: 1.31886 valid's rmse: 2.03906

[800] train's rmse: 1.29218 valid's rmse: 2.03514

[1000] train's rmse: 1.28195 valid's rmse: 2.03282

Predicting for fold 1...

Start processing fold 2...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.49284 valid's rmse: 2.077

Predicting for fold 2...

Start processing fold 3...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.49065 valid's rmse: 2.09892

Predicting for fold 3...

Start processing fold 4...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.48507 valid's rmse: 2.15302

Predicting for fold 4...

Start processing fold 5...

Training until validation scores don't improve for 200 rounds

[200] train's rmse: 1.47887 valid's rmse: 2.08394

Predicting for fold 5...

[I 2023-02-13 06:15:40,006] Trial 10 finished with value: 2.0651381565806988 and parameters: {'num_iterations': 1100, 'learning_rate': 0.0328345041349871, 'num_leaves': 100, 'max_depth': 4, 'max_bin': 251, 'bagging_fraction': 0.11036750976974558, 'bagging_freq': 22, 'feature_fraction': 0.5861806533992988, 'feature_fraction_bynode': 0.5908282078616656, 'lambda_l1': 435, 'lambda_l2': 933, 'extra_trees': False, 'path_smooth': 112}. Best is trial 10 with value: 2.0651381565806988.

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.52353 valid's rmse: 2.16383
Early stopping, best iteration is:
[105] train's rmse: 1.72707 valid's rmse: 2.12064
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.51493 valid's rmse: 2.23422
Early stopping, best iteration is:
[83] train's rmse: 1.84844 valid's rmse: 2.16677
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.52177 valid's rmse: 2.23285
Early stopping, best iteration is:
[85] train's rmse: 1.83727 valid's rmse: 2.17546
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.51481 valid's rmse: 2.10721
Early stopping, best iteration is:
[97] train's rmse: 1.75413 valid's rmse: 2.0752
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.5257 valid's rmse: 2.23387
Early stopping, best iteration is:
[90] train's rmse: 1.80001 valid's rmse: 2.17471
Predicting for fold 5...
```

```
[I 2023-02-13 06:17:46,560] Trial 11 finished with value: 2.1425559803946372 and parameters: {'num_iterations': 1100, 'learning_rate': 0.029564302707576727, 'num_leaves': 100, 'max_depth': 4, 'max_bin': 252, 'bagging_fraction': 0.10165081754719657, 'bagging_freq': 23, 'feature_fraction': 0.5837261924570463, 'feature_fraction_bynode': 0.5956078593308411, 'lambda_l1': 409, 'lambda_l2': 966, 'extra_trees': False, 'path_smooth': 100}. Best is trial 10 with value: 2.0651381565806988.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.51058 valid's rmse: 2.05683
[400] train's rmse: 1.3789 valid's rmse: 2.05736
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.51105 valid's rmse: 2.19926
Early stopping, best iteration is:
[94] train's rmse: 1.77039 valid's rmse: 2.13619
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.51764 valid's rmse: 2.19799
Early stopping, best iteration is:
[85] train's rmse: 1.82571 valid's rmse: 2.15206
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.50671 valid's rmse: 2.15723
Early stopping, best iteration is:
[92] train's rmse: 1.77465 valid's rmse: 2.08526
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.51487 valid's rmse: 2.09586
[400] train's rmse: 1.37565 valid's rmse: 2.06612
[600] train's rmse: 1.32244 valid's rmse: 2.05901
[800] train's rmse: 1.29367 valid's rmse: 2.05343
[1000] train's rmse: 1.27919 valid's rmse: 2.04792
Did not meet early stopping. Best iteration is:
[1100] train's rmse: 1.27418 valid's rmse: 2.04545
Predicting for fold 5...
```

```
[I 2023-02-13 06:20:00,587] Trial 12 finished with value: 2.09395157299241 and parameters: {'num_iterations': 1100, 'learning_rate': 0.029248303227431938, 'num_leaves': 98, 'max_depth': 4, 'max_bin': 253, 'bagging_fraction': 0.11158755880972135, 'bagging_freq': 20, 'feature_fraction': 0.5943456334510211, 'feature_fraction_bynode': 0.598246113288877, 'lambda_l1': 425, 'lambda_l2': 978, 'extra_trees': False, 'path_smooth': 123}. Best is trial 10 with value: 2.0651381565806988.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49113 valid's rmse: 2.17561
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49769 valid's rmse: 2.29857
Early stopping, best iteration is:
[73] train's rmse: 1.8353 valid's rmse: 2.1974
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49609 valid's rmse: 2.16458
Early stopping, best iteration is:
[83] train's rmse: 1.7743 valid's rmse: 2.11773
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49249 valid's rmse: 2.07369
Early stopping, best iteration is:
[106] train's rmse: 1.6627 valid's rmse: 2.06085
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49211 valid's rmse: 2.27149
Early stopping, best iteration is:
[71] train's rmse: 1.84851 valid's rmse: 2.21603
Predicting for fold 5...
```

```
[I 2023-02-13 06:22:06,846] Trial 13 finished with value: 2.144853683502202 and parameters: {'num_iterations': 1000, 'learning_rate': 0.03366642375020089, 'num_leaves': 89, 'max_depth': 4, 'max_bin': 251, 'bagging_fraction': 0.1078212238588392, 'bagging_freq': 21, 'feature_fraction': 0.5808134728863427, 'feature_fraction_bynode': 0.5987502779211402, 'lambda_l1': 456, 'lambda_l2': 844, 'extra_trees': False, 'path_smooth': 253}. Best is trial 10 with value: 2.0651381565806988.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49544 valid's rmse: 2.0405
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49852 valid's rmse: 1.98958
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49022 valid's rmse: 2.1172
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.49736 valid's rmse: 2.19857
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.4985 valid's rmse: 1.98466
[400] train's rmse: 1.34317 valid's rmse: 1.96816
[600] train's rmse: 1.27574 valid's rmse: 1.95512
[800] train's rmse: 1.2373 valid's rmse: 1.94511
[1000] train's rmse: 1.21338 valid's rmse: 1.93815
[1200] train's rmse: 1.1974 valid's rmse: 1.93306
[1400] train's rmse: 1.18756 valid's rmse: 1.92878
Predicting for fold 5...
```

```
[I 2023-02-13 06:24:31,743] Trial 14 finished with value: 2.02999775969229 and parameters: {'num_iterations': 1500, 'learning_rate': 0.019908450044620562, 'num_leaves': 90, 'max_depth': 5, 'max_bin': 239, 'bagging_fraction': 0.18078231391179356, 'bagging_freq': 43, 'feature_fraction': 0.5980226845999467, 'feature_fraction_bynode': 0.5458756726159959, 'lambda_l1': 310, 'lambda_l2': 827, 'extra_trees': False, 'path_smooth': 239}. Best is trial 14 with value: 2.02999775969229.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.61021 valid's rmse: 2.14739
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.62201 valid's rmse: 2.09132
Early stopping, best iteration is:
[170] train's rmse: 1.70029 valid's rmse: 2.07812
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.62022 valid's rmse: 2.08822
Early stopping, best iteration is:
[173] train's rmse: 1.69108 valid's rmse: 2.08209
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.60295 valid's rmse: 2.16058
Early stopping, best iteration is:
[165] train's rmse: 1.70653 valid's rmse: 2.14566
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.61333 valid's rmse: 2.14137
Early stopping, best iteration is:
[173] train's rmse: 1.68632 valid's rmse: 2.13445
Predicting for fold 5...
```

```
[I 2023-02-13 06:26:45,015] Trial 15 finished with value: 2.116061404845992 and parameters: {'num_iterations': 1500, 'learning_rate': 0.013865556971201391, 'num_leaves': 88, 'max_depth': 5, 'max_bin': 238, 'bagging_fraction': 0.19985466469554847, 'bagging_freq': 44, 'feature_fraction': 0.5140938499929191, 'feature_fraction_bynode': 0.535719438164022, 'lambda_l1': 248, 'lambda_l2': 816, 'extra_trees': False, 'path_smooth': 282}. Best is trial 14 with value: 2.02999775969229.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.29747 valid's rmse: 2.15004
Early stopping, best iteration is:
[53] train's rmse: 1.64814 valid's rmse: 2.0915
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.30183 valid's rmse: 2.31749
Early stopping, best iteration is:
[47] train's rmse: 1.72273 valid's rmse: 2.21408
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.29591 valid's rmse: 2.36067
Early stopping, best iteration is:
[45] train's rmse: 1.76163 valid's rmse: 2.22147
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.30382 valid's rmse: 2.38471
Early stopping, best iteration is:
[45] train's rmse: 1.76655 valid's rmse: 2.20511
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.30149 valid's rmse: 2.26148
Early stopping, best iteration is:
[45] train's rmse: 1.75533 valid's rmse: 2.23433
Predicting for fold 5...
```

```
[I 2023-02-13 06:28:55,055] Trial 16 finished with value: 2.193299515964761 and parameters: {'num_iterations': 900, 'learning_rate': 0.041537492969298456, 'num_leaves': 88, 'max_depth': 6, 'max_bin': 220, 'bagging_fraction': 0.2720917357944324, 'bagging_freq': 50, 'feature_fraction': 0.4467626729732193, 'feature_fraction_bynode': 0.4169601333058024, 'lambda_l1': 532, 'lambda_l2': 476, 'extra_trees': False, 'path_smooth': 274}. Best is trial 14 with value: 2.02999775969229.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.54584 valid's rmse: 2.03758
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.54877 valid's rmse: 2.28252
Early stopping, best iteration is:
[137] train's rmse: 1.68797 valid's rmse: 2.23004
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.54846 valid's rmse: 1.95586
Early stopping, best iteration is:
[182] train's rmse: 1.57668 valid's rmse: 1.94108
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.54459 valid's rmse: 2.02977
Early stopping, best iteration is:
[184] train's rmse: 1.57162 valid's rmse: 2.01584
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.54545 valid's rmse: 2.1808
Early stopping, best iteration is:
[129] train's rmse: 1.71619 valid's rmse: 2.14362
Predicting for fold 5...
```

```
[I 2023-02-13 06:31:07,324] Trial 17 finished with value: 2.072769909508447 and parameters: {'num_iterations': 1500, 'learning_rate': 0.020493069375226356, 'num_leaves': 49, 'max_depth': 5, 'max_bin': 240, 'bagging_fraction': 0.1586821506388803, 'bagging_freq': 33, 'feature_fraction': 0.5364333045281309, 'feature_fraction_bynode': 0.5490690795299651, 'lambda_l1': 295, 'lambda_l2': 822, 'extra_trees': False, 'path_smooth': 999}. Best is trial 14 with value: 2.02999775969229.
```

```
Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.17959 valid's rmse: 2.20299
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.17915 valid's rmse: 2.27052
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.17673 valid's rmse: 2.2384
Early stopping, best iteration is:
[19] train's rmse: 1.7677 valid's rmse: 2.1228
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.18382 valid's rmse: 2.24007
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 1.17896 valid's rmse: 2.08443
Predicting for fold 5...
```

```
[I 2023-02-13 06:33:13,817] Trial 18 finished with value: 2.1197424083102896 and parameters: {'num_iterations': 1300, 'learning_rate': 0.09581185341057605, 'num_leaves': 91, 'max_depth': 5, 'max_bin': 215, 'bagging_fraction': 0.2968247202931393, 'bagging_freq': 31, 'feature_fraction': 0.5436453150060586, 'feature_fraction_bynode': 0.5446131732326209, 'lambda_l1': 350, 'lambda_l2': 634, 'extra_trees': False, 'path_smooth': 186}. Best is trial 14 with value: 2.02999775969229.
```

```

Start processing fold 1...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.00748 valid's rmse: 3.0146
[400] train's rmse: 2.28672 valid's rmse: 2.41993
Predicting for fold 1...
Start processing fold 2...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.00229 valid's rmse: 3.05832
[400] train's rmse: 2.28295 valid's rmse: 2.41542
Predicting for fold 2...
Start processing fold 3...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.00726 valid's rmse: 3.04171
[400] train's rmse: 2.28413 valid's rmse: 2.41983
Did not meet early stopping. Best iteration is:
[500] train's rmse: 2.06901 valid's rmse: 2.2868
Predicting for fold 3...
Start processing fold 4...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.00542 valid's rmse: 3.04034
[400] train's rmse: 2.28593 valid's rmse: 2.42163
Predicting for fold 4...
Start processing fold 5...
Training until validation scores don't improve for 200 rounds
[200] train's rmse: 3.00545 valid's rmse: 3.00927
[400] train's rmse: 2.28872 valid's rmse: 2.40772
Did not meet early stopping. Best iteration is:
[500] train's rmse: 2.07431 valid's rmse: 2.27024
Predicting for fold 5...
[I 2023-02-13 06:35:36,154] Trial 19 finished with value: 2.2739637834415745 and parameters: {'num_iterations': 500, 'learning_rate': 0.003105282285168102, 'num_leaves': 60, 'max_depth': 7, 'max_bin': 243, 'bagging_fraction': 0.16632944108350842, 'bagging_freq': 47, 'feature_fraction': 0.5989188643494747, 'feature_fraction_bynode': 0.3932645009593174, 'lambda_l1': 558, 'lambda_l2': 880, 'extra_trees': False, 'path_smooth': 328}. Best is trial 14 with value: 2.02999775969229.

```

```

In [46]: fig_lightgbm = optuna.visualization.plot_optimization_history(study_lightgbm)
fig_lightgbm.show();

```

There appears to be a downward trend with only a few number of trials. What is important is the fact that we can use Bayesian optimization to give us a good starting point to carry out some manual tuning.

Model Training

```

In [ ]: study_lightgbm.best_params

```

```
Out[ ]: {'num_iterations': 1500,
        'learning_rate': 0.019908450044620562,
        'num_leaves': 90,
        'max_depth': 5,
        'max_bin': 239,
        'bagging_fraction': 0.18078231391179356,
        'bagging_freq': 43,
        'feature_fraction': 0.5980226845999467,
        'feature_fraction_bynode': 0.5458756726159959,
        'lambda_l1': 310,
        'lambda_l2': 827,
        'extra_trees': False,
        'path_smooth': 239}
```

```
In [66]: # Out-of-fold prediction dictionary
oof_lightgbm = {}
# Feature importance container
feat_imp_lightgbm = []
# K-fold cross validation
kf_lightgbm = KFold(n_splits=5, shuffle=True, random_state=rs)

for fold, (train_indx, val_indx) in enumerate(kf_lightgbm.split(X_train, y_train)):

    # Train and validation sets
    fold_X_train, fold_y_train = X_train.iloc[train_indx], y_train[train_indx]
    fold_X_val, fold_y_val = X_train.iloc[val_indx], y_train[val_indx]

    # Preprocessing using a fresh copy of the pipeline for every fold to prevent leakage
    preprocessor = joblib.load('../output/preprocessors/lightgbm_preprocessor.joblib')
    print(f'Start processing fold {fold + 1}...')
    fold_X_train = preprocessor.fit_transform(fold_X_train, fold_y_train)
    fold_X_val = preprocessor.transform(fold_X_val)
    # Write fitted preprocessor to disk
    joblib.dump(preprocessor, model_path + f'lightgbm/preprocessor_fold_{fold + 1}.joblib')

    # Data for modeling
    feature_names = fold_X_train.columns.tolist()
    dtrain = lgb.Dataset(data=fold_X_train, label=fold_y_train, feature_name=feature_names)
    dvalid = lgb.Dataset(data=fold_X_val, label=fold_y_val, feature_name=feature_names, reference=dtrain)

    # Model
    eval_results = {}
    model = lgb.train(
        params={'objective': 'rmse',
                'metric': 'rmse',
                'device_type': 'gpu',
                'verbosity': -1,
                'early_stopping_round': 200,
                'boosting': 'gbdt',
                'num_iterations': 1500,
                'learning_rate': 0.01,
                'num_leaves': 100,
                'max_depth': 5,
                'max_bin': 239,
                'bagging_fraction': 0.2,
                'bagging_freq': 43,
                'feature_fraction': 0.6,
                'feature_fraction_bynode': 0.6,
                'lambda_l1': 310,
                'lambda_l2': 827,
                'extra_trees': False,
                'path_smooth': 239},
        train_set=dtrain,
        valid_sets=[dtrain, dvalid],
        valid_names=['train', 'valid'],
        callbacks=[lgb.log_evaluation(period=50), lgb.early_stopping(stopping_rounds=200), lgb.record_evaluation
        ])
    model.save_model(model_path + f'lightgbm/model_fold_{fold + 1}.txt', importance_type='gain') # Save gain-based
    joblib.dump(eval_results, model_path + f'lightgbm/eval_fold_{fold + 1}.joblib')

    # Feature importance
    df = pd.DataFrame({'features': fold_X_val.columns.tolist(), 'feat_imp': model.feature_importance(importance
    feat_imp_lightgbm.append(df)

    # Predictions
    print(f'Predicting for fold {fold + 1}...')
    oof_pred = model.predict(data=fold_X_val)
    oof_lightgbm[f'fold_{fold + 1}'] = {'target': fold_y_val, 'predictions': oof_pred}

    del dtrain, dvalid, preprocessor, model, eval_results, df, oof_pred
```

Start processing fold 1...

```
Out[66]: ['../output/models/lightgbm/preprocessor_fold_1.joblib']
```



```

Training until validation scores don't improve for 200 rounds
[50] train's rmse: 3.15877 valid's rmse: 3.18044
[100] train's rmse: 2.442 valid's rmse: 2.50662
[150] train's rmse: 2.0306 valid's rmse: 2.20206
[200] train's rmse: 1.79768 valid's rmse: 2.08868
[250] train's rmse: 1.66129 valid's rmse: 2.05533
[300] train's rmse: 1.57837 valid's rmse: 2.05621
[350] train's rmse: 1.52252 valid's rmse: 2.06422
[400] train's rmse: 1.48046 valid's rmse: 2.07768
[450] train's rmse: 1.45079 valid's rmse: 2.08082

Out[66]: <lightgbm.basic.Booster at 0x7f22ec491040>

Out[66]: ['../output/models/lightgbm/eval_fold_1.joblib']
Predicting for fold 1...
Start processing fold 2...

Out[66]: ['../output/models/lightgbm/preprocessor_fold_2.joblib']
Training until validation scores don't improve for 200 rounds
[50] train's rmse: 3.15987 valid's rmse: 3.14101
[100] train's rmse: 2.4468 valid's rmse: 2.45851
[150] train's rmse: 2.03853 valid's rmse: 2.14168
[200] train's rmse: 1.80417 valid's rmse: 2.02628
[250] train's rmse: 1.66945 valid's rmse: 2.00035
[300] train's rmse: 1.58584 valid's rmse: 2.00402
[350] train's rmse: 1.5317 valid's rmse: 2.01464
[400] train's rmse: 1.48987 valid's rmse: 2.02046
[450] train's rmse: 1.45823 valid's rmse: 2.0195

Out[66]: <lightgbm.basic.Booster at 0x7f22fe77b070>

Out[66]: ['../output/models/lightgbm/eval_fold_2.joblib']
Predicting for fold 2...
Start processing fold 3...

Out[66]: ['../output/models/lightgbm/preprocessor_fold_3.joblib']
Training until validation scores don't improve for 200 rounds
[50] train's rmse: 3.16028 valid's rmse: 3.12902
[100] train's rmse: 2.44853 valid's rmse: 2.44402
[150] train's rmse: 2.03809 valid's rmse: 2.12483
[200] train's rmse: 1.80589 valid's rmse: 1.99557
[250] train's rmse: 1.66739 valid's rmse: 1.95735
[300] train's rmse: 1.58758 valid's rmse: 1.96054
[350] train's rmse: 1.53206 valid's rmse: 1.96208
[400] train's rmse: 1.48973 valid's rmse: 1.97646
[450] train's rmse: 1.45691 valid's rmse: 1.9839
Early stopping, best iteration is:
[256] train's rmse: 1.65545 valid's rmse: 1.95659

Out[66]: <lightgbm.basic.Booster at 0x7f23072c5730>

Out[66]: ['../output/models/lightgbm/eval_fold_3.joblib']
Predicting for fold 3...
Start processing fold 4...

Out[66]: ['../output/models/lightgbm/preprocessor_fold_4.joblib']
Training until validation scores don't improve for 200 rounds
[50] train's rmse: 3.15984 valid's rmse: 3.14697
[100] train's rmse: 2.44539 valid's rmse: 2.47591
[150] train's rmse: 2.03673 valid's rmse: 2.17032
[200] train's rmse: 1.80254 valid's rmse: 2.0478
[250] train's rmse: 1.66627 valid's rmse: 2.02398
[300] train's rmse: 1.58329 valid's rmse: 2.03934
[350] train's rmse: 1.52493 valid's rmse: 2.05101
[400] train's rmse: 1.48307 valid's rmse: 2.05466

Out[66]: <lightgbm.basic.Booster at 0x7f2307c34490>

Out[66]: ['../output/models/lightgbm/eval_fold_4.joblib']
Predicting for fold 4...
Start processing fold 5...

Out[66]: ['../output/models/lightgbm/preprocessor_fold_5.joblib']
Training until validation scores don't improve for 200 rounds
[50] train's rmse: 3.15796 valid's rmse: 3.11707
[100] train's rmse: 2.43985 valid's rmse: 2.45579
[150] train's rmse: 2.02856 valid's rmse: 2.18495
[200] train's rmse: 1.79659 valid's rmse: 2.11167
[250] train's rmse: 1.66124 valid's rmse: 2.10853
[300] train's rmse: 1.57794 valid's rmse: 2.13395
[350] train's rmse: 1.524 valid's rmse: 2.16732
[400] train's rmse: 1.48257 valid's rmse: 2.17625

Out[66]: <lightgbm.basic.Booster at 0x7f22ec4a8c10>

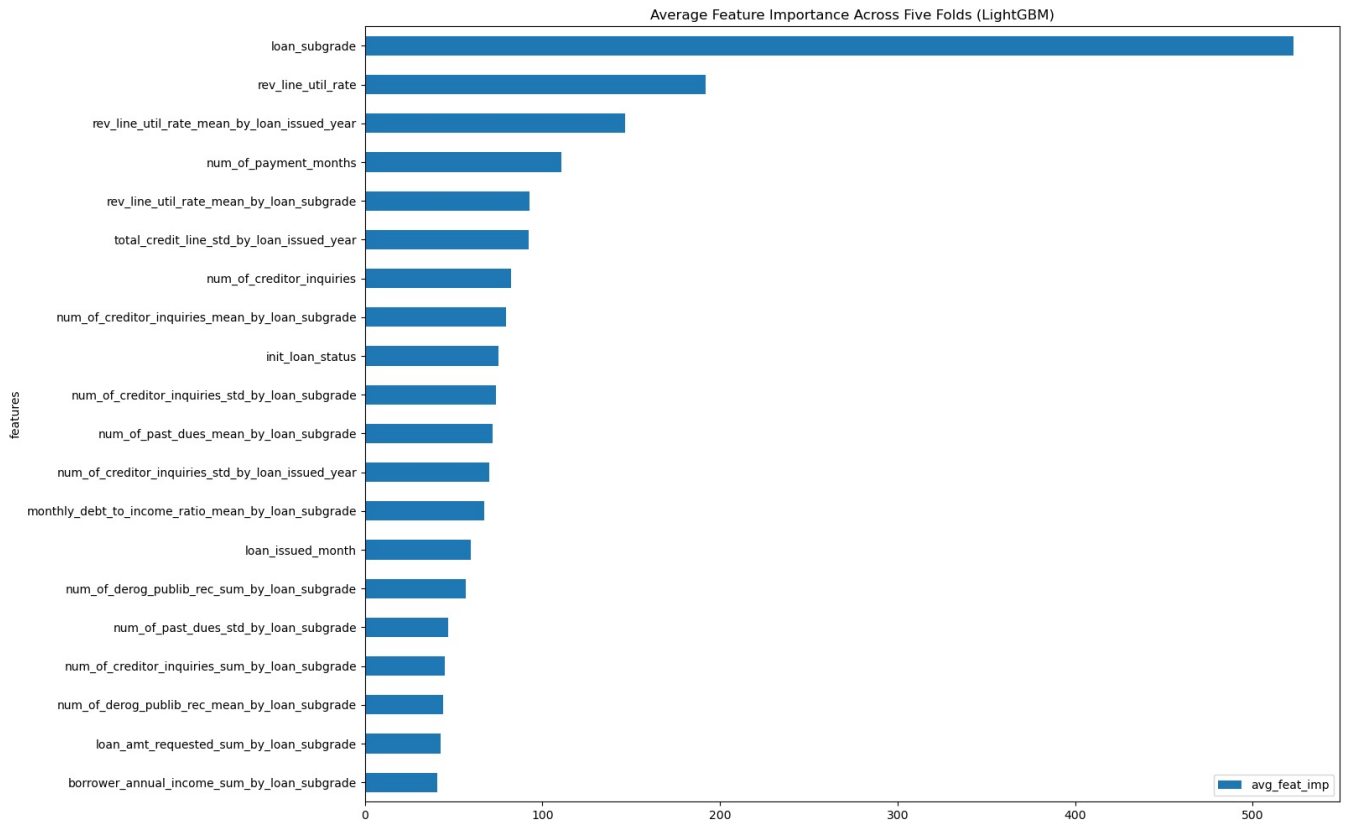
Out[66]: ['../output/models/lightgbm/eval_fold_5.joblib']
Predicting for fold 5...

```

Feature Importance

```
In [67]: # Join feature importance
feat_imp_lightgbm = reduce(lambda x, y: pd.merge(x, y, on='features', how='left'), feat_imp_lightgbm)
feat_imp_lightgbm['avg_feat_imp'] = feat_imp_lightgbm.iloc[:, 1:].apply(lambda row: row.mean(), axis=1)

# Plot top feature importance
feat_imp_lightgbm.sort_values(by='avg_feat_imp', ascending=True).iloc[-20:].plot(
    kind='barh', x='features', y='avg_feat_imp',
    figsize=(15, 12),
    title='Average Feature Importance Across Five Folds (LightGBM)'
)
plt.show();
```



Learning Curves

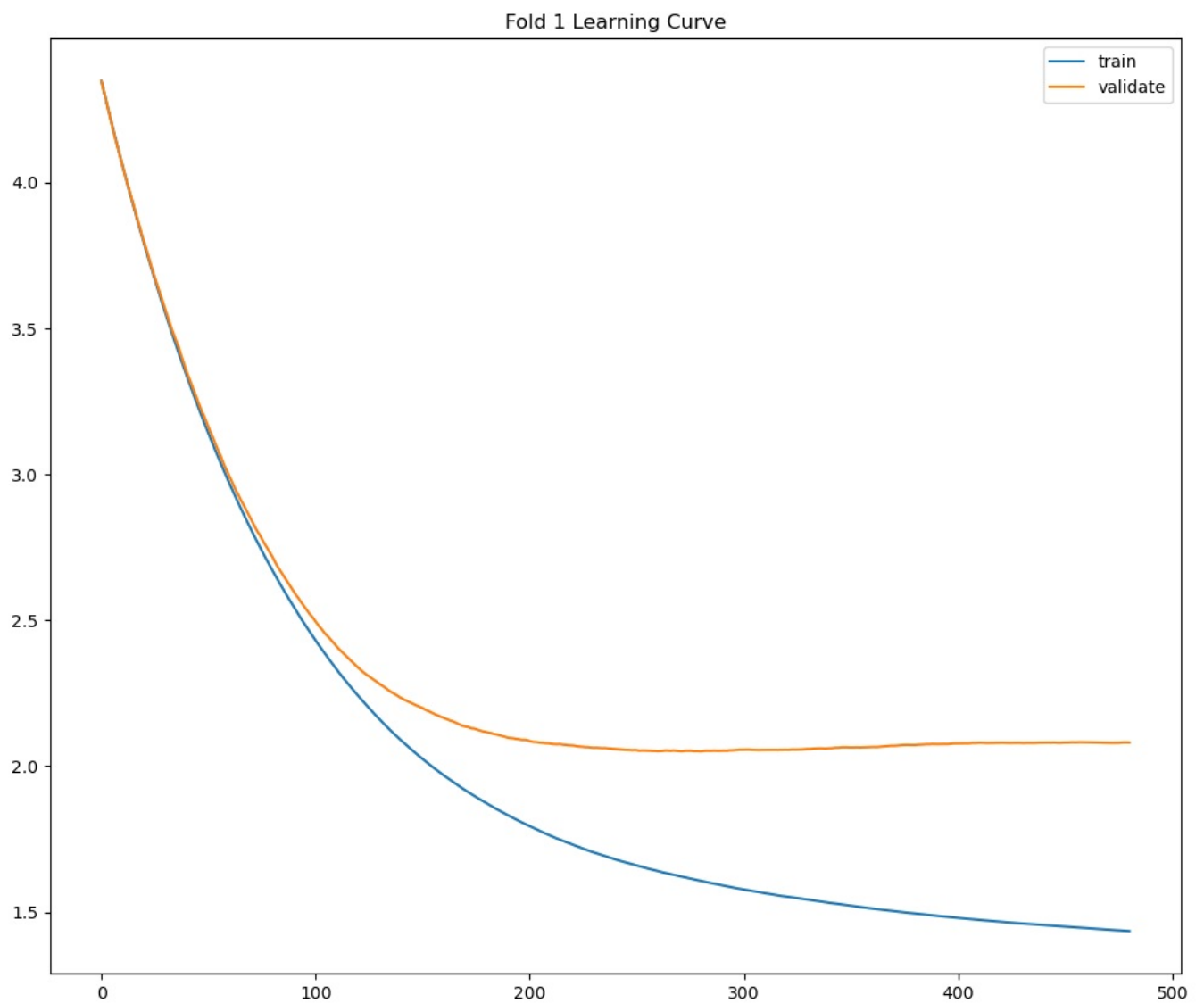
```
In [68]: for fold in range(5):
eval_result = joblib.load(model_path + f'lightgbm/eval_fold_{fold + 1}.joblib')
plt.plot(eval_result['train']['rmse'], label='train');
plt.plot(eval_result['valid']['rmse'], label='validate');
plt.legend();
plt.title(f'Fold {fold + 1} Learning Curve');
plt.show();
```

Out[68]: [<matplotlib.lines.Line2D at 0x7f22fe2e8910>]

Out[68]: [<matplotlib.lines.Line2D at 0x7f22fe2e8f10>]

Out[68]: <matplotlib.legend.Legend at 0x7f22fe2e8ee0>

Out[68]: Text(0.5, 1.0, 'Fold 1 Learning Curve')

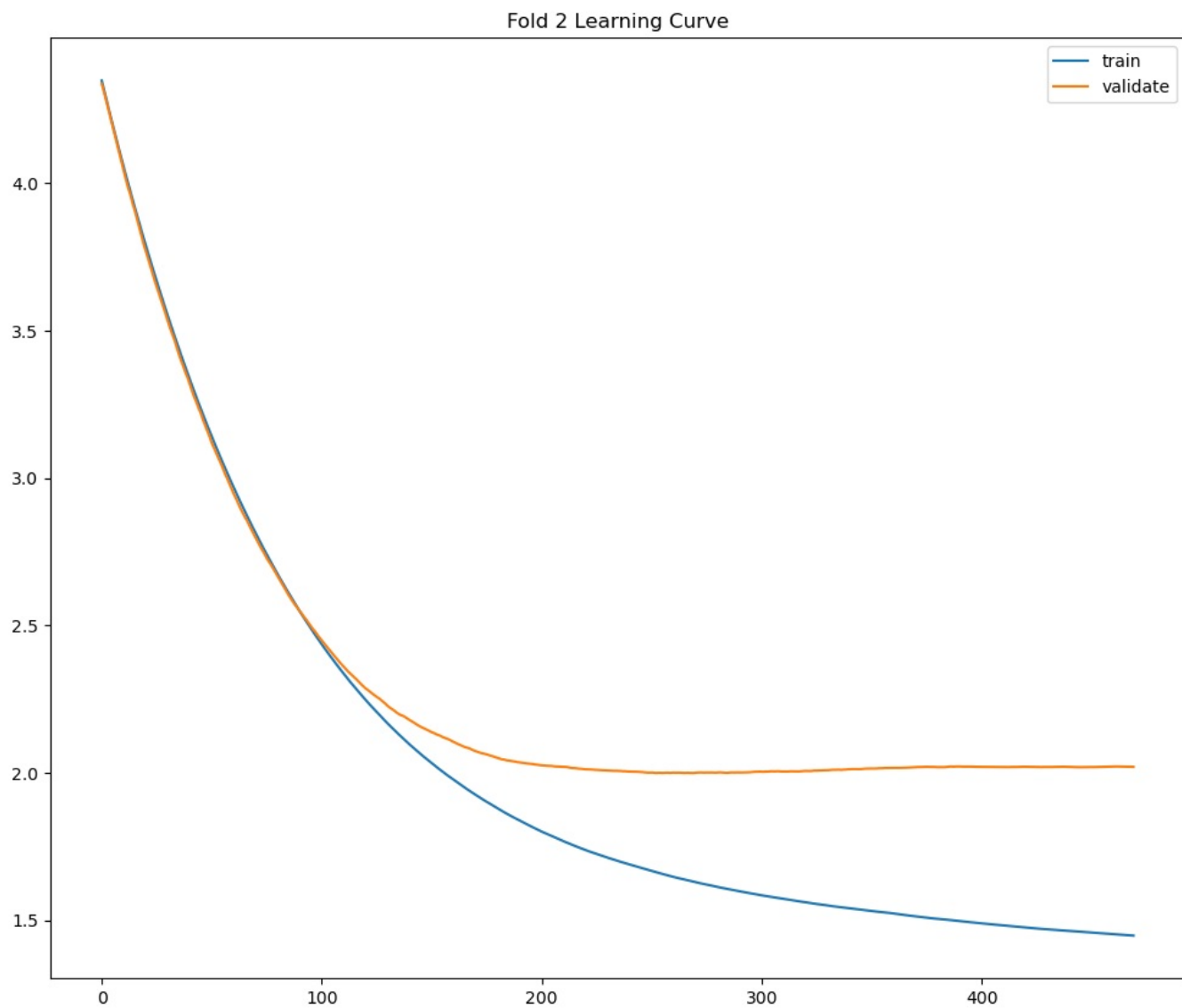


Out[68]: [<matplotlib.lines.Line2D at 0x7f22fe2f5a60>]

Out[68]: [<matplotlib.lines.Line2D at 0x7f231daaedc0>]

Out[68]: <matplotlib.legend.Legend at 0x7f22fe2f5340>

Out[68]: Text(0.5, 1.0, 'Fold 2 Learning Curve')

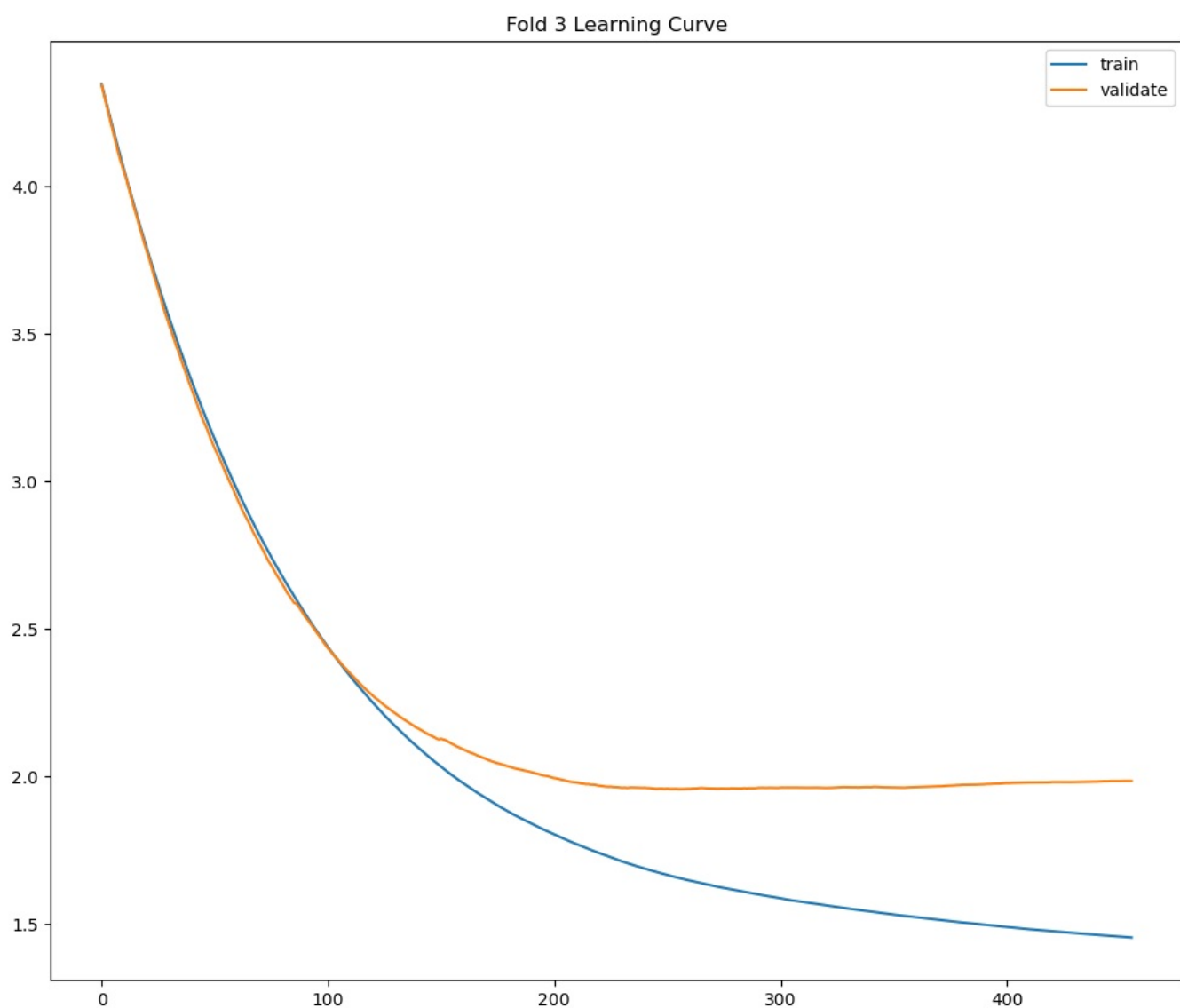


Out[68]: [<matplotlib.lines.Line2D at 0x7f22ee6e4640>]

Out[68]: [<matplotlib.lines.Line2D at 0x7f22fee46fa0>]

Out[68]: <matplotlib.legend.Legend at 0x7f22fee46880>

Out[68]: Text(0.5, 1.0, 'Fold 3 Learning Curve')

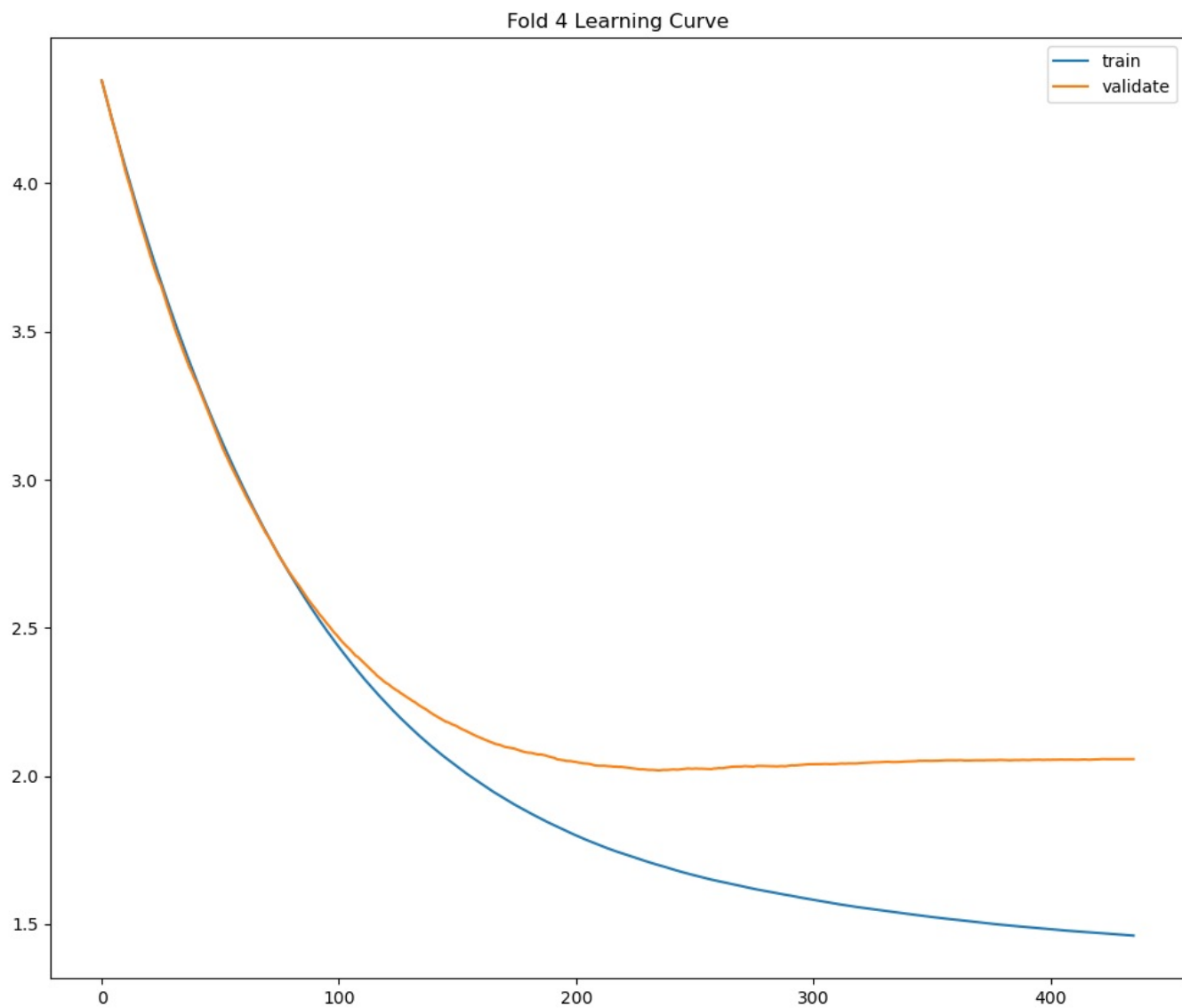


Out[68]: [<matplotlib.lines.Line2D at 0x7f2307856220>]

Out[68]: [<matplotlib.lines.Line2D at 0x7f2307855e80>]

Out[68]: <matplotlib.legend.Legend at 0x7f2307855040>

Out[68]: Text(0.5, 1.0, 'Fold 4 Learning Curve')

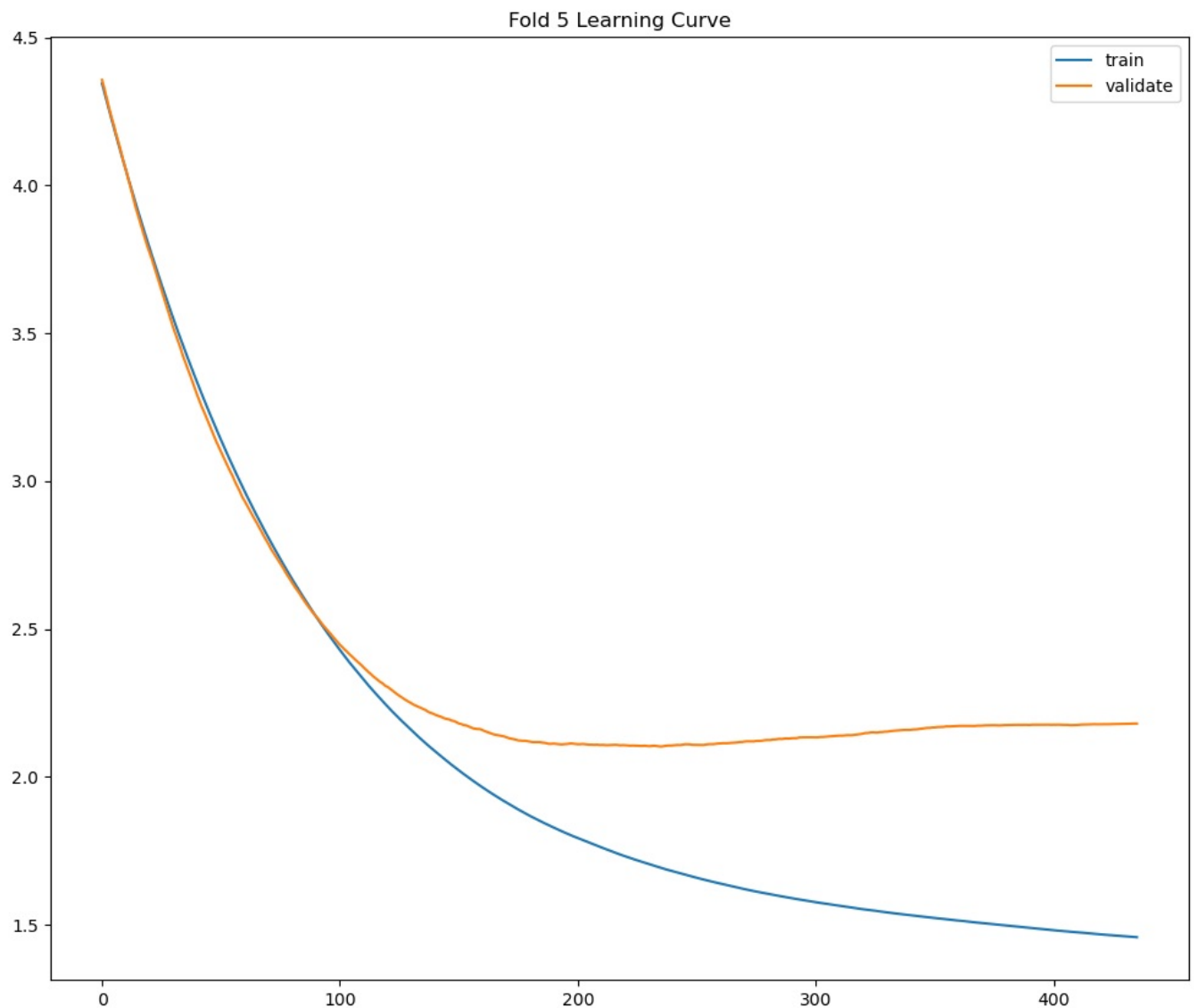


Out[68]: [<matplotlib.lines.Line2D at 0x7f230dd28ee0>]

Out[68]: [<matplotlib.lines.Line2D at 0x7f230dd28460>]

Out[68]: <matplotlib.legend.Legend at 0x7f230dd28f10>

Out[68]: Text(0.5, 1.0, 'Fold 5 Learning Curve')



Performance on Validation Sets

```
In [69]: oof_lightgbm_rmse = []
target_frame = cudf.DataFrame(index=['count', 'mean', 'std', 'min', '25%', '50%', '75%', 'max'])

for key in oof_lightgbm:
    oof_lightgbm_rmse.append(
        mean_squared_error(oof_lightgbm[key]['target'], oof_lightgbm[key]['predictions'], squared=False)
    )
    print(f'Finished computing rmse for {key}')

    target_frame[f'{key}_target_descriptive_stats'] = cudf.Series(oof_lightgbm[key]['target']).describe()
    print(f'Finished computing descriptive stats for {key} target')
```

```

Finished computing rmse for fold_1
Finished computing descriptive stats for fold_1 target
Finished computing rmse for fold_2
Finished computing descriptive stats for fold_2 target
Finished computing rmse for fold_3
Finished computing descriptive stats for fold_3 target
Finished computing rmse for fold_4
Finished computing descriptive stats for fold_4 target
Finished computing rmse for fold_5
Finished computing descriptive stats for fold_5 target

```

```
In [70]: cdf.Series(oof_lightgbm_rmse).describe()
```

```

Out[70]: count    5.000000
         mean     2.025601
         std      0.055010
         min      1.956591
         25%      1.999203
         50%      2.018575
         75%      2.050838
         max      2.102796
         dtype: float64

```

```
In [71]: target_frame
```

```

Out[71]:

```

	fold_1_target_descriptive_stats	fold_2_target_descriptive_stats	fold_3_target_descriptive_stats	fold_4_target_descriptive_stats	fold_5_target_descriptive_stats
count	67798.000000	67798.000000	67798.000000	67797.000000	
mean	13.94996	13.935819	13.954786	13.931679	
std	4.37422	4.369764	4.379598	4.378819	
min	5.42000	5.420000	5.420000	5.420000	
25%	10.99000	10.990000	10.990000	10.990000	
50%	13.79000	13.680000	13.680000	13.680000	
75%	16.78000	16.780000	16.780000	16.780000	
max	26.06000	26.060000	26.060000	26.060000	

Interestingly, for both LightGBM and CatBoost, the validation errors converges to around 2 percentage points, while XGBoost was able to reduce the validation error rates below 2 percentage points. Still, it may be that XGBoost was simply able to fine-tune the models based on the validation set; therefore, to improve generalization on unseen data, we will bag the three gradient boosting machines to create a meta-learner.

Ensemble Weighted Averaging

We will generate predictions for each of the three gradient boosted machines and average their predictions:

- XGBoost

```

In [89]: pred_xgboost = np.zeros(X_test.shape[0])

for fold in range(5):

    # Instantiate booster
    model_xgboost = xgb.Booster()
    # Load model
    model_xgboost.load_model(model_path + f'xgboost/model_fold_{fold + 1}.xgb')
    # Transform test data using fold preprocessor
    print(f'Preprocessing fold {fold + 1}...')
    fold_X_test = joblib.load(model_path + f'xgboost/preprocessor_fold_{fold + 1}.joblib').transform(X_test)
    # Make predictions on test set
    print(f'Predicting for fold {fold + 1}...')
    pred_xgboost += model_xgboost.predict(xgb.DMatrix(fold_X_test))
pred_xgboost /= 5
pred_xgboost

```

```

Preprocessing fold 1...
Predicting for fold 1...
Preprocessing fold 2...
Predicting for fold 2...
Preprocessing fold 3...
Predicting for fold 3...
Preprocessing fold 4...
Predicting for fold 4...
Preprocessing fold 5...
Predicting for fold 5...

```

```

Out[89]: array([15.15944023,  6.61361198, 14.25195141, ..., 16.88702888,
                14.72674465, 13.8499157 ])
```


- CatBoost:

```
In [97]: pred_catboost = np.zeros(X_test.shape[0])

for fold in range(5):

    # Instantiate booster
    model_catboost = cb.CatBoostRegressor()
    # Load model
    model_catboost.load_model(model_path + f'catboost/model_fold_{fold + 1}.cbm')
    # Transform test data using fold preprocessor
    print(f'Preprocessing fold {fold + 1}...')
    fold_X_test = joblib.load(model_path + f'catboost/preprocessor_fold_{fold + 1}.joblib').transform(X_test)
    # Make predictions on test set
    print(f'Predicting for fold {fold + 1}...')
    pred_catboost += model_catboost.predict(fold_X_test)
pred_catboost /= 5
pred_catboost
```

```
Out[97]: <catboost.core.CatBoostRegressor at 0x7f2362bfa820>
Preprocessing fold 1...
Predicting for fold 1...
```

```
Out[97]: <catboost.core.CatBoostRegressor at 0x7f231a30cf40>
Preprocessing fold 2...
Predicting for fold 2...
```

```
Out[97]: <catboost.core.CatBoostRegressor at 0x7f230dd202b0>
Preprocessing fold 3...
Predicting for fold 3...
```

```
Out[97]: <catboost.core.CatBoostRegressor at 0x7f235769ba90>
Preprocessing fold 4...
Predicting for fold 4...
```

```
Out[97]: <catboost.core.CatBoostRegressor at 0x7f2361fec4c0>
Preprocessing fold 5...
Predicting for fold 5...
```

```
Out[97]: array([17.2385936 ,  9.72445119, 16.22936023, ..., 16.0668276 ,
        16.35841502, 14.75556931])
```

- LightGBM

```
In [98]: pred_lightgbm = np.zeros(X_test.shape[0])

for fold in range(5):

    # Instantiate booster
    model_lightgbm = lgb.Booster(model_file = model_path + f'lightgbm/model_fold_{fold + 1}.txt')
    # Transform test data using fold preprocessor
    print(f'Preprocessing fold {fold + 1}...')
    fold_X_test = joblib.load(model_path + f'lightgbm/preprocessor_fold_{fold + 1}.joblib').transform(X_test)
    # Make predictions on test set
    print(f'Predicting for fold {fold + 1}...')
    pred_lightgbm += model_lightgbm.predict(fold_X_test)
pred_lightgbm /= 5
pred_lightgbm
```

```
Preprocessing fold 1...
Predicting for fold 1...
Preprocessing fold 2...
Predicting for fold 2...
Preprocessing fold 3...
Predicting for fold 3...
Preprocessing fold 4...
Predicting for fold 4...
Preprocessing fold 5...
Predicting for fold 5...
```

```
Out[98]: array([15.81290531,  8.98502879, 15.04477182, ..., 17.69637397,
        15.16434433, 14.48658767])
```

Computing Weights For Averaging

We will use weights that are inversely related to validation RMSE--- the lower the validation RMSE of the model, the higher the weights the predictions of that model receives.

```
In [129]: model_rmse = np.array([np.mean(oof_xgboost_rmse), np.mean(oof_catboost_rmse), np.mean(oof_lightgbm_rmse)])
model_rmse
```

```
Out[129]: array([1.58097789, 1.91714624, 2.02560063])
```

The model weights are the inverse of these errors:

```
In [133]: model_weights = 1 / (model_rsme / model_rsme.sum())
model_weights
```

```
Out[133]: array([3.49386592, 2.88122243, 2.72695648])
```

Generate matrix of predictions (\$80,000 \times 3\$) where each row vector is a training example and each column vector is a vector of predictions:

```
In [155]: predictions = np.column_stack((pred_xgboost, pred_catboost, pred_lightgbm))
predictions
```

```
Out[155]: array([[15.15944023, 17.2385936, 15.81290531],
 [ 6.61361198,  9.72445119,  8.98502879],
 [14.25195141, 16.22936023, 15.04477182],
 ...,
 [16.88702888, 16.0668276, 17.69637397],
 [14.72674465, 16.35841502, 15.16434433],
 [13.8499157, 14.75556931, 14.48658767]])
```

Take the weighted average:

```
In [158]: avg_predictions = np.average(predictions, axis=1, weights=model_weights)
avg_predictions
```

```
Out[158]: array([16.01336639,  8.3088102, 15.11542127, ..., 16.86987521,
 15.37434864, 14.32734317])
```

Finally, we attach the identification columns and write the output to disk:

```
In [160]: final_output = pd.DataFrame({
    'id_loan': X_test['id_loan'],
    'id_borrower': X_test['id_borrower'],
    'predicted_interest_rate': avg_predictions
})
final_output
```

```
Out[160]:
```

	id_loan	id_borrower	predicted_interest_rate
0	44409194.0	47416907.0	16.013366
1	44017917.0	47034722.0	8.308810
2	44259158.0	47306871.0	15.115421
3	44429213.0	47476932.0	16.124419
4	44299188.0	47346901.0	12.716305
...
79995	38272852.0	41056632.0	9.004351
79996	38232598.0	41016384.0	18.758735
79997	38282597.0	41066378.0	16.869875
79998	38232613.0	41016400.0	15.374349
79999	38262186.0	41045946.0	14.327343

80000 rows \times 3 columns

Write To S3

```
In [165]: with io.StringIO() as csv_buffer:

    final_output.to_csv(csv_buffer, index=False)

    response = s3.put_object(
        Bucket=AWS_S3_BUCKET, Key='Loan Prediction/Loan Results from Yang Wu 12373055.csv', Body=csv_buffer.getvalue()
    )

    status = response.get("ResponseMetadata", {}).get("HTTPStatusCode")

    if status == 200:
        print(f"Successful S3 put_object response. Status - {status}")
    else:
        print(f"Unsuccessful S3 put_object response. Status - {status}")
```

Successful S3 put_object response. Status - 200

