Part 2 Model Training and Prediction

December 23, 2021

1 Predict Future Sales Part 2: Model Training and Prediction

2 Table of contents

- 1. Model Training
  - 1.1 Validation scheme
  - 1.2 LightGBM Regressor
  - 1.3 XGBoost Regressor
    - 1.3.1 Training XGBoost Regressor on GPU
  - 1.4 CatBoost Regressor on GPU
  - 1.5 Linear models
    - 1.5.1 LinearRegression model
    - 1.5.2 Ridge model
    - 1.5.3 Lasso model
- 2. Ensembling Method
  - 2.1 Ensembling: LightGBM + Linear Regression
- 3. Predictions

[1]:
```
#load packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import copy

import os
import pickle
from tqdm import tqdm #progress bar
from itertools import product
import warnings
warnings.filterwarnings("ignore")
from IPython.display import clear_output

from sklearn.feature_extraction import text
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import KFold, StratifiedKFold
from sklearn.metrics import r2_score, mean_squared_error
```

```python
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA, TruncatedSVD, NMF
from sklearn.linear_model import LinearRegression, Ridge, Lasso, Lars,␣
↪ElasticNet
# from sklearn.externals import joblib
import joblib

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

[2]:
```python
#load processed data
df1 = pd.read_csv('./data_processed/data.csv', index_col=0)
```

[3]:
```python
data_folder = "./data/"
test = pd.read_csv(os.path.join(data_folder, "test.csv")).set_index('ID')
```

[4]:
```python
cat_features = ['shop_id_target_enc', 'item_id_target_enc',
                'city_code_target_enc', 'item_category_id_x_target_enc',
                'type_code_target_enc', 'subtype_code_target_enc']
```

3 1. Model Training

We will implement the following models:

- **Linear Regression**: preproccing is required and we will use `StandardScaler`;
- **XGBoost / LightBGM**: no preproccesing as decision tree are capable of handling features with different scales.

3.1 1.1 Validation scheme

We choose **holdout** as out validation scheme: out training set runs from Jan.2013 to Sep.2015, validation set runs on Oct.2015. The test set runs on Nov.2015.

[5]:
```python
all_data = copy.copy(df1)

X_train = all_data[all_data['date_block_num'] < 33].drop(['item_cnt_month'],␣
↪axis=1)
Y_train = all_data[all_data['date_block_num'] < 33]['item_cnt_month']

X_valid = all_data[all_data['date_block_num'] == 33].drop(['item_cnt_month'],␣
↪axis=1)
Y_valid = all_data[all_data['date_block_num'] == 33]['item_cnt_month']

X_test = all_data[all_data['date_block_num'] == 34].drop(['item_cnt_month'],␣
↪axis=1)
```

```python
print(X_train.shape)
print(Y_train.shape)
print(X_valid.shape)
print(Y_valid.shape)
print(X_test.shape)
```

```
(6186922, 38)
(6186922,)
(238172, 38)
(238172,)
(214200, 38)
```

[5]:

```python
X_test.head()
```

[5]:

```
         date_block_num  date_avg_item_cnt_lag_1  \
6425094              34                   0.2585
6425095              34                   0.0000
6425096              34                   0.2585
6425097              34                   0.2585
6425098              34                   0.0000

         date_item_avg_item_cnt_lag_1  date_item_avg_item_cnt_lag_2  \
6425094                        0.5684                         2.512
6425095                        0.0000                         0.000
6425096                        0.9546                         1.860
6425097                        0.6360                         1.116
6425098                        0.0000                         0.000

         date_item_avg_item_cnt_lag_3  date_item_avg_item_cnt_lag_6  \
6425094                         2.834                         1.978
6425095                         0.000                         0.000
6425096                         3.572                         1.613
6425097                         1.548                         0.000
6425098                         0.000                         0.000

         date_item_avg_item_cnt_lag_12  date_shop_avg_item_cnt_lag_1  \
6425094                            1.3                        0.1901
6425095                            0.0                        0.0000
6425096                            0.0                        0.1901
6425097                            0.0                        0.1901
6425098                            0.0                        0.0000

         date_shop_avg_item_cnt_lag_2  date_shop_avg_item_cnt_lag_3  …  \
6425094                        0.2059                        0.2451  …
6425095                        0.0000                        0.0000  …
6425096                        0.2059                        0.2451  …
6425097                        0.2059                        0.2451  …
6425098                        0.0000                        0.0000  …
```

```
         tfidf_interaction_1  tfidf_interaction_2  tfidf_interaction_3  \
6425094             0.000000             0.033116                  0.0
6425095             0.000395             0.034083                  0.0
6425096             0.000000             0.032825                  0.0
6425097             0.000000             0.033220                  0.0
6425098             0.000000             0.032375                  0.0

         tfidf_interaction_4  tfidf_interaction_5  tfidf_interaction_6  \
6425094                  0.0                  0.0             0.000134
6425095                  0.0                  0.0             0.000464
6425096                  0.0                  0.0             0.000014
6425097                  0.0                  0.0             0.000251
6425098                  0.0                  0.0             0.000000

         tfidf_interaction_7  tfidf_interaction_8  tfidf_interaction_9  \
6425094             0.001819             0.000000             0.000000
6425095             0.000000             0.001306             0.001373
6425096             0.015599             0.000000             0.000000
6425097             0.012365             0.000187             0.000000
6425098             0.019344             0.000000             0.000000

         tfidf_interaction_10
6425094              0.000046
6425095              0.000597
6425096              0.000000
6425097              0.000177
6425098              0.000000

[5 rows x 38 columns]
```

3.2 1.2 LightGBM Regressor

The correct metric to be used is **root mean squared error**

Our first model is regularized gradient boosting LightGBM.

[6]:
```
lgb_train = lgb.Dataset(X_train, Y_train)
lgb_valid = lgb.Dataset(X_valid, Y_valid)

params = {
    'boosting_type': 'dart',
    'metric': 'l2_root', # RMSE
    'verbose': 1,
    'seed': 0,
    'max_depth': 8,
    'learning_rate': 0.1,
    'reg_lambda': 2.0,
```

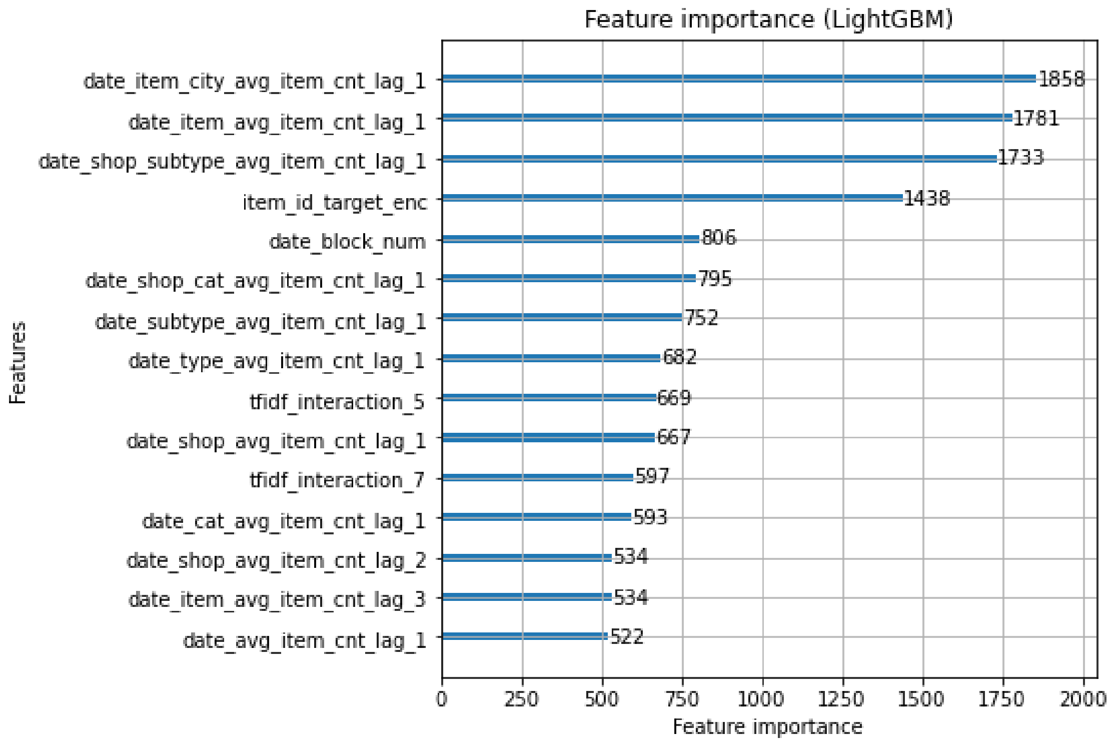
```
    'reg_alpha': 2.0,
    'subsample': 0.7,
    'num_leaves': 20,
    'feature_fraction': 0.8,
    'drop_rate': 0.2
}

#next time, try to use GPU rather than CPU
model_lgbm = lgb.train(params, lgb_train, num_boost_round=1000,␣
 ↪valid_sets=lgb_valid,
                       early_stopping_rounds=200,␣
 ↪categorical_feature=cat_features,
                        verbose_eval=50)
```

```
[LightGBM] [Warning] Auto-choosing row-wise multi-threading, the overhead of
testing was 0.141321 seconds.
You can set `force_row_wise=true` to remove the overhead.
And if memory is not enough, you can set `force_col_wise=true`.
[LightGBM] [Info] Total Bins 7654
[LightGBM] [Info] Number of data points in the train set: 6186922, number of
used features: 38
[LightGBM] [Info] Start training from score 0.288849
[50]    valid_0's rmse: 0.99683
[100]   valid_0's rmse: 0.972024
[150]   valid_0's rmse: 0.968635
[200]   valid_0's rmse: 0.964639
[250]   valid_0's rmse: 0.964203
[300]   valid_0's rmse: 0.959738
[350]   valid_0's rmse: 0.957481
[400]   valid_0's rmse: 0.956257
[450]   valid_0's rmse: 0.954682
[500]   valid_0's rmse: 0.954533
[550]   valid_0's rmse: 0.957562
[600]   valid_0's rmse: 0.955514
[650]   valid_0's rmse: 0.954121
[700]   valid_0's rmse: 0.957515
[750]   valid_0's rmse: 0.956778
[800]   valid_0's rmse: 0.956383
[850]   valid_0's rmse: 0.956392
[900]   valid_0's rmse: 0.957082
[950]   valid_0's rmse: 0.954556
[1000]  valid_0's rmse: 0.952658
```

[7]:
```
print(np.sqrt(mean_squared_error(Y_valid, model_lgbm.predict(X_valid))))
```

```
0.9526578913079428
```

```
[8]: lgb.plot_importance(model_lgbm, max_num_features=15, figsize=(6,6),␣
     ↪title='Feature importance (LightGBM)')
     plt.show()
```




```
[9]: joblib.dump(model_lgbm, 'models/model1_lgbm.pkl');
```

```
[10]: model_lgbm = joblib.load('models/model1_lgbm.pkl') #how to save and check the␣
      ↪parameters when load???
```

```
[ ]: # model_lgbm.save_model('models/model1_lgbm_txt.txt') #save
     # model_lgbm_txt = lgb.Booster(model_file='models/model1_lgbm_txt.txt') #load
```

3.3 1.3 XGBoost Regressor

Now we try XGBoost library.

```
[13]: model_xgb = xgb.XGBRegressor(max_depth=8, n_estimators=1000,␣
      ↪min_child_weight=300,
                                   colsample_bytree=0.8, subsample=0.8, eta=0.2,␣
      ↪seed=42,
                                   reg_alpha=0.2, reg_lambda=2.0)
```

```python
model_xgb.fit(X_train, Y_train, eval_metric='rmse', eval_set=[(X_valid,␣
↪Y_valid)],
            verbose=True, early_stopping_rounds=1000)  #start:10:29pm, end 1:
↪23am
```

```
[0]     validation_0-rmse:1.08661
[1]     validation_0-rmse:1.04106
[2]     validation_0-rmse:1.00700
[3]     validation_0-rmse:0.98522
[4]     validation_0-rmse:0.96973
[5]     validation_0-rmse:0.96625
[6]     validation_0-rmse:0.96619
[7]     validation_0-rmse:0.96430
[8]     validation_0-rmse:0.95899
[9]     validation_0-rmse:0.95977
[10]    validation_0-rmse:0.96491
[11]    validation_0-rmse:0.96315
[12]    validation_0-rmse:0.96090
[13]    validation_0-rmse:0.96517
[14]    validation_0-rmse:0.96373
[15]    validation_0-rmse:0.96412
[16]    validation_0-rmse:0.96361
[17]    validation_0-rmse:0.96730
[18]    validation_0-rmse:0.96633
[19]    validation_0-rmse:0.96688
[20]    validation_0-rmse:0.97346
[21]    validation_0-rmse:0.97422
[22]    validation_0-rmse:0.97439
[23]    validation_0-rmse:0.97414
[24]    validation_0-rmse:0.97231
[25]    validation_0-rmse:0.97294
[26]    validation_0-rmse:0.97258
[27]    validation_0-rmse:0.97255
[28]    validation_0-rmse:0.97114
[29]    validation_0-rmse:0.97069
[30]    validation_0-rmse:0.97063
[31]    validation_0-rmse:0.97169
[32]    validation_0-rmse:0.97169
[33]    validation_0-rmse:0.97148
[34]    validation_0-rmse:0.97216
[35]    validation_0-rmse:0.97327
[36]    validation_0-rmse:0.97345
[37]    validation_0-rmse:0.97447
[38]    validation_0-rmse:0.97422
[39]    validation_0-rmse:0.97395
[40]    validation_0-rmse:0.97393
[41]    validation_0-rmse:0.97397
[42]    validation_0-rmse:0.97361
```

```
[43]	validation_0-rmse:0.97345
[44]	validation_0-rmse:0.97357
[45]	validation_0-rmse:0.97376
[46]	validation_0-rmse:0.97535
[47]	validation_0-rmse:0.97555
[48]	validation_0-rmse:0.97621
[49]	validation_0-rmse:0.97599
[50]	validation_0-rmse:0.98225
[51]	validation_0-rmse:0.98238
[52]	validation_0-rmse:0.98395
[53]	validation_0-rmse:0.98359
[54]	validation_0-rmse:0.98367
[55]	validation_0-rmse:0.98346
[56]	validation_0-rmse:0.98375
[57]	validation_0-rmse:0.98343
[58]	validation_0-rmse:0.98360
[59]	validation_0-rmse:0.98523
[60]	validation_0-rmse:0.98533
[61]	validation_0-rmse:0.98519
[62]	validation_0-rmse:0.98510
[63]	validation_0-rmse:0.98497
[64]	validation_0-rmse:0.98555
[65]	validation_0-rmse:0.98522
[66]	validation_0-rmse:0.98510
[67]	validation_0-rmse:0.98616
[68]	validation_0-rmse:0.98638
[69]	validation_0-rmse:0.98642
[70]	validation_0-rmse:0.98701
[71]	validation_0-rmse:0.98749
[72]	validation_0-rmse:0.98776
[73]	validation_0-rmse:0.98722
[74]	validation_0-rmse:0.98714
[75]	validation_0-rmse:0.98693
[76]	validation_0-rmse:0.98715
[77]	validation_0-rmse:0.98668
[78]	validation_0-rmse:0.98692
[79]	validation_0-rmse:0.98719
[80]	validation_0-rmse:0.98715
[81]	validation_0-rmse:0.98755
[82]	validation_0-rmse:0.98755
[83]	validation_0-rmse:0.98823
[84]	validation_0-rmse:0.98839
[85]	validation_0-rmse:0.98864
[86]	validation_0-rmse:0.98783
[87]	validation_0-rmse:0.98749
[88]	validation_0-rmse:0.98794
[89]	validation_0-rmse:0.98786
[90]	validation_0-rmse:0.98863
```

```
[91]	validation_0-rmse:0.98856
[92]	validation_0-rmse:0.98823
[93]	validation_0-rmse:0.98793
[94]	validation_0-rmse:0.98797
[95]	validation_0-rmse:0.98796
[96]	validation_0-rmse:0.98889
[97]	validation_0-rmse:0.98876
[98]	validation_0-rmse:0.98863
[99]	validation_0-rmse:0.98903
[100]	validation_0-rmse:0.98871
[101]	validation_0-rmse:0.98934
[102]	validation_0-rmse:0.98943
[103]	validation_0-rmse:0.98847
[104]	validation_0-rmse:0.98863
[105]	validation_0-rmse:0.98815
[106]	validation_0-rmse:0.98810
[107]	validation_0-rmse:0.98805
[108]	validation_0-rmse:0.98787
[109]	validation_0-rmse:0.98798
[110]	validation_0-rmse:0.98812
[111]	validation_0-rmse:0.98788
[112]	validation_0-rmse:0.98844
[113]	validation_0-rmse:0.98913
[114]	validation_0-rmse:0.98934
[115]	validation_0-rmse:0.98919
[116]	validation_0-rmse:0.98949
[117]	validation_0-rmse:0.98878
[118]	validation_0-rmse:0.98895
[119]	validation_0-rmse:0.98889
[120]	validation_0-rmse:0.98889
[121]	validation_0-rmse:0.98909
[122]	validation_0-rmse:0.98935
[123]	validation_0-rmse:0.99247
[124]	validation_0-rmse:0.99225
[125]	validation_0-rmse:0.99225
[126]	validation_0-rmse:0.99213
[127]	validation_0-rmse:0.99204
[128]	validation_0-rmse:0.99283
[129]	validation_0-rmse:0.99271
[130]	validation_0-rmse:0.99271
[131]	validation_0-rmse:0.99546
[132]	validation_0-rmse:0.99572
[133]	validation_0-rmse:0.99565
[134]	validation_0-rmse:0.99699
[135]	validation_0-rmse:0.99703
[136]	validation_0-rmse:0.99657
[137]	validation_0-rmse:0.99657
[138]	validation_0-rmse:0.99657
```

```
[139]	validation_0-rmse:0.99712
[140]	validation_0-rmse:0.99759
[141]	validation_0-rmse:0.99761
[142]	validation_0-rmse:0.99705
[143]	validation_0-rmse:0.99714
[144]	validation_0-rmse:0.99710
[145]	validation_0-rmse:0.99679
[146]	validation_0-rmse:0.99675
[147]	validation_0-rmse:0.99692
[148]	validation_0-rmse:0.99700
[149]	validation_0-rmse:0.99709
[150]	validation_0-rmse:0.99771
[151]	validation_0-rmse:0.99782
[152]	validation_0-rmse:0.99776
[153]	validation_0-rmse:0.99767
[154]	validation_0-rmse:0.99799
[155]	validation_0-rmse:0.99811
[156]	validation_0-rmse:0.99892
[157]	validation_0-rmse:0.99746
[158]	validation_0-rmse:0.99738
[159]	validation_0-rmse:0.99715
[160]	validation_0-rmse:0.99711
[161]	validation_0-rmse:0.99772
[162]	validation_0-rmse:0.99764
[163]	validation_0-rmse:0.99699
[164]	validation_0-rmse:0.99743
[165]	validation_0-rmse:0.99694
[166]	validation_0-rmse:1.00046
[167]	validation_0-rmse:1.00040
[168]	validation_0-rmse:1.00071
[169]	validation_0-rmse:1.00067
[170]	validation_0-rmse:1.00022
[171]	validation_0-rmse:1.00068
[172]	validation_0-rmse:0.99892
[173]	validation_0-rmse:0.99892
[174]	validation_0-rmse:0.99885
[175]	validation_0-rmse:0.99854
[176]	validation_0-rmse:0.99868
[177]	validation_0-rmse:0.99859
[178]	validation_0-rmse:0.99963
[179]	validation_0-rmse:0.99982
[180]	validation_0-rmse:1.00005
[181]	validation_0-rmse:1.00009
[182]	validation_0-rmse:1.00039
[183]	validation_0-rmse:1.00074
[184]	validation_0-rmse:1.00047
[185]	validation_0-rmse:1.00058
[186]	validation_0-rmse:1.00032
```

```
[187]	validation_0-rmse:1.00075
[188]	validation_0-rmse:1.00110
[189]	validation_0-rmse:1.00098
[190]	validation_0-rmse:1.00059
[191]	validation_0-rmse:1.00088
[192]	validation_0-rmse:1.00122
[193]	validation_0-rmse:1.00117
[194]	validation_0-rmse:1.00120
[195]	validation_0-rmse:1.00145
[196]	validation_0-rmse:1.00165
[197]	validation_0-rmse:1.00347
[198]	validation_0-rmse:1.00356
[199]	validation_0-rmse:1.00369
[200]	validation_0-rmse:1.00350
[201]	validation_0-rmse:1.00354
[202]	validation_0-rmse:1.00351
[203]	validation_0-rmse:1.00330
[204]	validation_0-rmse:1.00334
[205]	validation_0-rmse:1.00590
[206]	validation_0-rmse:1.00637
[207]	validation_0-rmse:1.00663
[208]	validation_0-rmse:1.00618
[209]	validation_0-rmse:1.00613
[210]	validation_0-rmse:1.00658
[211]	validation_0-rmse:1.00689
[212]	validation_0-rmse:1.00691
[213]	validation_0-rmse:1.00693
[214]	validation_0-rmse:1.00793
[215]	validation_0-rmse:1.00800
[216]	validation_0-rmse:1.00863
[217]	validation_0-rmse:1.00862
[218]	validation_0-rmse:1.00854
[219]	validation_0-rmse:1.00832
[220]	validation_0-rmse:1.00833
[221]	validation_0-rmse:1.00852
[222]	validation_0-rmse:1.00913
[223]	validation_0-rmse:1.00917
[224]	validation_0-rmse:1.00975
[225]	validation_0-rmse:1.00975
[226]	validation_0-rmse:1.00977
[227]	validation_0-rmse:1.01002
[228]	validation_0-rmse:1.01008
[229]	validation_0-rmse:1.01014
[230]	validation_0-rmse:1.00988
[231]	validation_0-rmse:1.00985
[232]	validation_0-rmse:1.00994
[233]	validation_0-rmse:1.00986
[234]	validation_0-rmse:1.00966
```

```
[235]	validation_0-rmse:1.00944
[236]	validation_0-rmse:1.00964
[237]	validation_0-rmse:1.00956
[238]	validation_0-rmse:1.01180
[239]	validation_0-rmse:1.01156
[240]	validation_0-rmse:1.01205
[241]	validation_0-rmse:1.01381
[242]	validation_0-rmse:1.01382
[243]	validation_0-rmse:1.01377
[244]	validation_0-rmse:1.01419
[245]	validation_0-rmse:1.01491
[246]	validation_0-rmse:1.01487
[247]	validation_0-rmse:1.01486
[248]	validation_0-rmse:1.01490
[249]	validation_0-rmse:1.01497
[250]	validation_0-rmse:1.01487
[251]	validation_0-rmse:1.01482
[252]	validation_0-rmse:1.01484
[253]	validation_0-rmse:1.01494
[254]	validation_0-rmse:1.01493
[255]	validation_0-rmse:1.01543
[256]	validation_0-rmse:1.01544
[257]	validation_0-rmse:1.01542
[258]	validation_0-rmse:1.01543
[259]	validation_0-rmse:1.01590
[260]	validation_0-rmse:1.01604
[261]	validation_0-rmse:1.01612
[262]	validation_0-rmse:1.01578
[263]	validation_0-rmse:1.01564
[264]	validation_0-rmse:1.01552
[265]	validation_0-rmse:1.01547
[266]	validation_0-rmse:1.01566
[267]	validation_0-rmse:1.01568
[268]	validation_0-rmse:1.01562
[269]	validation_0-rmse:1.01585
[270]	validation_0-rmse:1.01613
[271]	validation_0-rmse:1.01625
[272]	validation_0-rmse:1.01608
[273]	validation_0-rmse:1.01605
[274]	validation_0-rmse:1.01603
[275]	validation_0-rmse:1.01949
[276]	validation_0-rmse:1.01947
[277]	validation_0-rmse:1.01965
[278]	validation_0-rmse:1.01946
[279]	validation_0-rmse:1.01951
[280]	validation_0-rmse:1.01949
[281]	validation_0-rmse:1.01947
[282]	validation_0-rmse:1.01930
```

```
[283]	validation_0-rmse:1.01931
[284]	validation_0-rmse:1.01911
[285]	validation_0-rmse:1.01915
[286]	validation_0-rmse:1.01905
[287]	validation_0-rmse:1.01906
[288]	validation_0-rmse:1.01923
[289]	validation_0-rmse:1.01926
[290]	validation_0-rmse:1.01926
[291]	validation_0-rmse:1.01946
[292]	validation_0-rmse:1.01940
[293]	validation_0-rmse:1.01953
[294]	validation_0-rmse:1.01948
[295]	validation_0-rmse:1.01946
[296]	validation_0-rmse:1.01957
[297]	validation_0-rmse:1.01979
[298]	validation_0-rmse:1.01984
[299]	validation_0-rmse:1.01991
[300]	validation_0-rmse:1.01965
[301]	validation_0-rmse:1.01955
[302]	validation_0-rmse:1.01954
[303]	validation_0-rmse:1.01953
[304]	validation_0-rmse:1.02052
[305]	validation_0-rmse:1.02039
[306]	validation_0-rmse:1.02031
[307]	validation_0-rmse:1.02029
[308]	validation_0-rmse:1.02120
[309]	validation_0-rmse:1.02121
[310]	validation_0-rmse:1.02104
[311]	validation_0-rmse:1.02135
[312]	validation_0-rmse:1.02149
[313]	validation_0-rmse:1.02126
[314]	validation_0-rmse:1.02109
[315]	validation_0-rmse:1.02116
[316]	validation_0-rmse:1.02300
[317]	validation_0-rmse:1.02276
[318]	validation_0-rmse:1.02253
[319]	validation_0-rmse:1.02270
[320]	validation_0-rmse:1.02290
[321]	validation_0-rmse:1.02313
[322]	validation_0-rmse:1.02295
[323]	validation_0-rmse:1.02292
[324]	validation_0-rmse:1.02329
[325]	validation_0-rmse:1.02347
[326]	validation_0-rmse:1.02357
[327]	validation_0-rmse:1.02339
[328]	validation_0-rmse:1.02518
[329]	validation_0-rmse:1.02510
[330]	validation_0-rmse:1.02517
```

```
[331]	validation_0-rmse:1.02525
[332]	validation_0-rmse:1.02525
[333]	validation_0-rmse:1.02536
[334]	validation_0-rmse:1.02575
[335]	validation_0-rmse:1.02565
[336]	validation_0-rmse:1.02579
[337]	validation_0-rmse:1.02530
[338]	validation_0-rmse:1.02547
[339]	validation_0-rmse:1.02517
[340]	validation_0-rmse:1.02549
[341]	validation_0-rmse:1.02559
[342]	validation_0-rmse:1.02549
[343]	validation_0-rmse:1.02537
[344]	validation_0-rmse:1.02547
[345]	validation_0-rmse:1.02638
[346]	validation_0-rmse:1.02600
[347]	validation_0-rmse:1.02580
[348]	validation_0-rmse:1.02572
[349]	validation_0-rmse:1.02612
[350]	validation_0-rmse:1.02586
[351]	validation_0-rmse:1.02591
[352]	validation_0-rmse:1.02594
[353]	validation_0-rmse:1.02615
[354]	validation_0-rmse:1.02697
[355]	validation_0-rmse:1.02684
[356]	validation_0-rmse:1.02735
[357]	validation_0-rmse:1.02729
[358]	validation_0-rmse:1.02764
[359]	validation_0-rmse:1.02833
[360]	validation_0-rmse:1.02879
[361]	validation_0-rmse:1.02879
[362]	validation_0-rmse:1.02889
[363]	validation_0-rmse:1.02917
[364]	validation_0-rmse:1.02894
[365]	validation_0-rmse:1.02894
[366]	validation_0-rmse:1.02914
[367]	validation_0-rmse:1.02869
[368]	validation_0-rmse:1.02847
[369]	validation_0-rmse:1.02857
[370]	validation_0-rmse:1.02856
[371]	validation_0-rmse:1.02875
[372]	validation_0-rmse:1.02891
[373]	validation_0-rmse:1.02884
[374]	validation_0-rmse:1.02903
[375]	validation_0-rmse:1.02919
[376]	validation_0-rmse:1.02922
[377]	validation_0-rmse:1.02919
[378]	validation_0-rmse:1.02919
```

```
[379]	validation_0-rmse:1.02919
[380]	validation_0-rmse:1.02893
[381]	validation_0-rmse:1.02974
[382]	validation_0-rmse:1.02972
[383]	validation_0-rmse:1.02993
[384]	validation_0-rmse:1.02989
[385]	validation_0-rmse:1.02980
[386]	validation_0-rmse:1.03036
[387]	validation_0-rmse:1.03033
[388]	validation_0-rmse:1.03037
[389]	validation_0-rmse:1.03095
[390]	validation_0-rmse:1.03111
[391]	validation_0-rmse:1.03068
[392]	validation_0-rmse:1.03098
[393]	validation_0-rmse:1.03081
[394]	validation_0-rmse:1.03086
[395]	validation_0-rmse:1.03095
[396]	validation_0-rmse:1.03108
[397]	validation_0-rmse:1.03105
[398]	validation_0-rmse:1.03205
[399]	validation_0-rmse:1.03208
[400]	validation_0-rmse:1.03212
[401]	validation_0-rmse:1.03217
[402]	validation_0-rmse:1.03217
[403]	validation_0-rmse:1.03218
[404]	validation_0-rmse:1.03234
[405]	validation_0-rmse:1.03247
[406]	validation_0-rmse:1.03261
[407]	validation_0-rmse:1.03270
[408]	validation_0-rmse:1.03288
[409]	validation_0-rmse:1.03305
[410]	validation_0-rmse:1.03293
[411]	validation_0-rmse:1.03304
[412]	validation_0-rmse:1.03267
[413]	validation_0-rmse:1.03265
[414]	validation_0-rmse:1.03270
[415]	validation_0-rmse:1.03156
[416]	validation_0-rmse:1.03149
[417]	validation_0-rmse:1.03126
[418]	validation_0-rmse:1.03117
[419]	validation_0-rmse:1.03094
[420]	validation_0-rmse:1.03090
[421]	validation_0-rmse:1.03119
[422]	validation_0-rmse:1.03122
[423]	validation_0-rmse:1.03135
[424]	validation_0-rmse:1.03138
[425]	validation_0-rmse:1.03145
[426]	validation_0-rmse:1.03163
```

```
[427]	validation_0-rmse:1.03163
[428]	validation_0-rmse:1.03160
[429]	validation_0-rmse:1.03158
[430]	validation_0-rmse:1.03174
[431]	validation_0-rmse:1.03150
[432]	validation_0-rmse:1.03309
[433]	validation_0-rmse:1.03354
[434]	validation_0-rmse:1.03329
[435]	validation_0-rmse:1.03381
[436]	validation_0-rmse:1.03388
[437]	validation_0-rmse:1.03386
[438]	validation_0-rmse:1.03358
[439]	validation_0-rmse:1.03357
[440]	validation_0-rmse:1.03409
[441]	validation_0-rmse:1.03478
[442]	validation_0-rmse:1.03486
[443]	validation_0-rmse:1.03486
[444]	validation_0-rmse:1.03513
[445]	validation_0-rmse:1.03508
[446]	validation_0-rmse:1.03507
[447]	validation_0-rmse:1.03517
[448]	validation_0-rmse:1.03517
[449]	validation_0-rmse:1.03510
[450]	validation_0-rmse:1.03510
[451]	validation_0-rmse:1.03537
[452]	validation_0-rmse:1.03625
[453]	validation_0-rmse:1.03617
[454]	validation_0-rmse:1.03623
[455]	validation_0-rmse:1.03621
[456]	validation_0-rmse:1.03567
[457]	validation_0-rmse:1.03578
[458]	validation_0-rmse:1.03575
[459]	validation_0-rmse:1.03588
[460]	validation_0-rmse:1.03596
[461]	validation_0-rmse:1.03632
[462]	validation_0-rmse:1.03629
[463]	validation_0-rmse:1.03745
[464]	validation_0-rmse:1.03743
[465]	validation_0-rmse:1.03746
[466]	validation_0-rmse:1.03755
[467]	validation_0-rmse:1.03719
[468]	validation_0-rmse:1.03728
[469]	validation_0-rmse:1.03736
[470]	validation_0-rmse:1.03740
[471]	validation_0-rmse:1.03730
[472]	validation_0-rmse:1.03787
[473]	validation_0-rmse:1.03771
[474]	validation_0-rmse:1.03791
```

```
[475]	validation_0-rmse:1.03807
[476]	validation_0-rmse:1.03798
[477]	validation_0-rmse:1.03812
[478]	validation_0-rmse:1.03822
[479]	validation_0-rmse:1.03806
[480]	validation_0-rmse:1.03837
[481]	validation_0-rmse:1.03842
[482]	validation_0-rmse:1.03834
[483]	validation_0-rmse:1.03867
[484]	validation_0-rmse:1.04182
[485]	validation_0-rmse:1.04184
[486]	validation_0-rmse:1.04180
[487]	validation_0-rmse:1.04178
[488]	validation_0-rmse:1.04179
[489]	validation_0-rmse:1.04179
[490]	validation_0-rmse:1.04149
[491]	validation_0-rmse:1.04149
[492]	validation_0-rmse:1.04180
[493]	validation_0-rmse:1.04151
[494]	validation_0-rmse:1.04119
[495]	validation_0-rmse:1.04226
[496]	validation_0-rmse:1.04221
[497]	validation_0-rmse:1.04229
[498]	validation_0-rmse:1.04239
[499]	validation_0-rmse:1.04269
[500]	validation_0-rmse:1.04270
[501]	validation_0-rmse:1.04263
[502]	validation_0-rmse:1.04257
[503]	validation_0-rmse:1.04281
[504]	validation_0-rmse:1.04302
[505]	validation_0-rmse:1.04311
[506]	validation_0-rmse:1.04266
[507]	validation_0-rmse:1.04290
[508]	validation_0-rmse:1.04293
[509]	validation_0-rmse:1.04289
[510]	validation_0-rmse:1.04301
[511]	validation_0-rmse:1.04291
[512]	validation_0-rmse:1.04267
[513]	validation_0-rmse:1.04267
[514]	validation_0-rmse:1.04249
[515]	validation_0-rmse:1.04236
[516]	validation_0-rmse:1.04229
[517]	validation_0-rmse:1.04218
[518]	validation_0-rmse:1.04346
[519]	validation_0-rmse:1.04350
[520]	validation_0-rmse:1.04359
[521]	validation_0-rmse:1.04319
[522]	validation_0-rmse:1.04323
```

```
[523]	validation_0-rmse:1.04344
[524]	validation_0-rmse:1.04335
[525]	validation_0-rmse:1.04338
[526]	validation_0-rmse:1.04354
[527]	validation_0-rmse:1.04392
[528]	validation_0-rmse:1.04398
[529]	validation_0-rmse:1.04396
[530]	validation_0-rmse:1.04404
[531]	validation_0-rmse:1.04415
[532]	validation_0-rmse:1.04412
[533]	validation_0-rmse:1.04419
[534]	validation_0-rmse:1.04416
[535]	validation_0-rmse:1.04416
[536]	validation_0-rmse:1.04436
[537]	validation_0-rmse:1.04438
[538]	validation_0-rmse:1.04447
[539]	validation_0-rmse:1.04472
[540]	validation_0-rmse:1.04669
[541]	validation_0-rmse:1.04633
[542]	validation_0-rmse:1.04633
[543]	validation_0-rmse:1.04632
[544]	validation_0-rmse:1.04641
[545]	validation_0-rmse:1.04641
[546]	validation_0-rmse:1.04652
[547]	validation_0-rmse:1.04647
[548]	validation_0-rmse:1.04614
[549]	validation_0-rmse:1.04608
[550]	validation_0-rmse:1.04603
[551]	validation_0-rmse:1.04700
[552]	validation_0-rmse:1.04710
[553]	validation_0-rmse:1.04702
[554]	validation_0-rmse:1.04708
[555]	validation_0-rmse:1.04720
[556]	validation_0-rmse:1.04736
[557]	validation_0-rmse:1.04718
[558]	validation_0-rmse:1.04588
[559]	validation_0-rmse:1.04588
[560]	validation_0-rmse:1.04592
[561]	validation_0-rmse:1.04573
[562]	validation_0-rmse:1.04575
[563]	validation_0-rmse:1.04567
[564]	validation_0-rmse:1.04543
[565]	validation_0-rmse:1.04539
[566]	validation_0-rmse:1.04524
[567]	validation_0-rmse:1.04557
[568]	validation_0-rmse:1.04572
[569]	validation_0-rmse:1.04573
[570]	validation_0-rmse:1.04609
```

```
[571]	validation_0-rmse:1.04625
[572]	validation_0-rmse:1.04630
[573]	validation_0-rmse:1.04618
[574]	validation_0-rmse:1.04659
[575]	validation_0-rmse:1.04660
[576]	validation_0-rmse:1.04722
[577]	validation_0-rmse:1.04705
[578]	validation_0-rmse:1.04709
[579]	validation_0-rmse:1.04711
[580]	validation_0-rmse:1.04719
[581]	validation_0-rmse:1.04715
[582]	validation_0-rmse:1.04724
[583]	validation_0-rmse:1.04757
[584]	validation_0-rmse:1.04799
[585]	validation_0-rmse:1.04812
[586]	validation_0-rmse:1.04799
[587]	validation_0-rmse:1.04806
[588]	validation_0-rmse:1.04797
[589]	validation_0-rmse:1.04797
[590]	validation_0-rmse:1.04792
[591]	validation_0-rmse:1.04798
[592]	validation_0-rmse:1.04801
[593]	validation_0-rmse:1.04794
[594]	validation_0-rmse:1.04788
[595]	validation_0-rmse:1.04778
[596]	validation_0-rmse:1.04783
[597]	validation_0-rmse:1.04743
[598]	validation_0-rmse:1.04760
[599]	validation_0-rmse:1.04771
[600]	validation_0-rmse:1.04770
[601]	validation_0-rmse:1.04790
[602]	validation_0-rmse:1.04845
[603]	validation_0-rmse:1.04848
[604]	validation_0-rmse:1.04851
[605]	validation_0-rmse:1.04856
[606]	validation_0-rmse:1.04853
[607]	validation_0-rmse:1.04802
[608]	validation_0-rmse:1.04800
[609]	validation_0-rmse:1.04807
[610]	validation_0-rmse:1.04804
[611]	validation_0-rmse:1.04810
[612]	validation_0-rmse:1.04805
[613]	validation_0-rmse:1.04808
[614]	validation_0-rmse:1.04837
[615]	validation_0-rmse:1.04873
[616]	validation_0-rmse:1.04971
[617]	validation_0-rmse:1.04972
[618]	validation_0-rmse:1.04984
```

```
[619]	validation_0-rmse:1.05066
[620]	validation_0-rmse:1.05087
[621]	validation_0-rmse:1.05085
[622]	validation_0-rmse:1.05089
[623]	validation_0-rmse:1.05092
[624]	validation_0-rmse:1.05112
[625]	validation_0-rmse:1.05103
[626]	validation_0-rmse:1.05098
[627]	validation_0-rmse:1.04932
[628]	validation_0-rmse:1.04914
[629]	validation_0-rmse:1.04730
[630]	validation_0-rmse:1.04734
[631]	validation_0-rmse:1.04748
[632]	validation_0-rmse:1.04755
[633]	validation_0-rmse:1.04974
[634]	validation_0-rmse:1.04947
[635]	validation_0-rmse:1.04947
[636]	validation_0-rmse:1.04950
[637]	validation_0-rmse:1.04929
[638]	validation_0-rmse:1.04929
[639]	validation_0-rmse:1.04938
[640]	validation_0-rmse:1.04919
[641]	validation_0-rmse:1.04913
[642]	validation_0-rmse:1.04916
[643]	validation_0-rmse:1.04929
[644]	validation_0-rmse:1.04945
[645]	validation_0-rmse:1.04942
[646]	validation_0-rmse:1.04945
[647]	validation_0-rmse:1.04952
[648]	validation_0-rmse:1.04950
[649]	validation_0-rmse:1.04946
[650]	validation_0-rmse:1.04947
[651]	validation_0-rmse:1.04947
[652]	validation_0-rmse:1.05169
[653]	validation_0-rmse:1.05169
[654]	validation_0-rmse:1.05193
[655]	validation_0-rmse:1.05184
[656]	validation_0-rmse:1.05226
[657]	validation_0-rmse:1.05232
[658]	validation_0-rmse:1.05382
[659]	validation_0-rmse:1.05378
[660]	validation_0-rmse:1.05379
[661]	validation_0-rmse:1.05387
[662]	validation_0-rmse:1.05407
[663]	validation_0-rmse:1.05411
[664]	validation_0-rmse:1.05415
[665]	validation_0-rmse:1.05388
[666]	validation_0-rmse:1.05385
```

```
[667]	validation_0-rmse:1.05391
[668]	validation_0-rmse:1.05386
[669]	validation_0-rmse:1.05392
[670]	validation_0-rmse:1.05394
[671]	validation_0-rmse:1.05389
[672]	validation_0-rmse:1.05371
[673]	validation_0-rmse:1.05431
[674]	validation_0-rmse:1.05431
[675]	validation_0-rmse:1.05430
[676]	validation_0-rmse:1.05450
[677]	validation_0-rmse:1.05432
[678]	validation_0-rmse:1.05432
[679]	validation_0-rmse:1.05432
[680]	validation_0-rmse:1.05434
[681]	validation_0-rmse:1.05416
[682]	validation_0-rmse:1.05384
[683]	validation_0-rmse:1.05351
[684]	validation_0-rmse:1.05354
[685]	validation_0-rmse:1.05351
[686]	validation_0-rmse:1.05402
[687]	validation_0-rmse:1.05374
[688]	validation_0-rmse:1.05403
[689]	validation_0-rmse:1.05377
[690]	validation_0-rmse:1.05374
[691]	validation_0-rmse:1.05369
[692]	validation_0-rmse:1.05391
[693]	validation_0-rmse:1.05372
[694]	validation_0-rmse:1.05363
[695]	validation_0-rmse:1.05363
[696]	validation_0-rmse:1.05350
[697]	validation_0-rmse:1.05425
[698]	validation_0-rmse:1.05434
[699]	validation_0-rmse:1.05428
[700]	validation_0-rmse:1.05445
[701]	validation_0-rmse:1.05445
[702]	validation_0-rmse:1.05449
[703]	validation_0-rmse:1.05450
[704]	validation_0-rmse:1.05446
[705]	validation_0-rmse:1.05425
[706]	validation_0-rmse:1.05421
[707]	validation_0-rmse:1.05417
[708]	validation_0-rmse:1.05420
[709]	validation_0-rmse:1.05424
[710]	validation_0-rmse:1.05417
[711]	validation_0-rmse:1.05412
[712]	validation_0-rmse:1.05430
[713]	validation_0-rmse:1.05426
[714]	validation_0-rmse:1.05427
```

```
[715]	validation_0-rmse:1.05450
[716]	validation_0-rmse:1.05449
[717]	validation_0-rmse:1.05461
[718]	validation_0-rmse:1.05460
[719]	validation_0-rmse:1.05462
[720]	validation_0-rmse:1.05438
[721]	validation_0-rmse:1.05436
[722]	validation_0-rmse:1.05440
[723]	validation_0-rmse:1.05461
[724]	validation_0-rmse:1.05494
[725]	validation_0-rmse:1.05500
[726]	validation_0-rmse:1.05496
[727]	validation_0-rmse:1.05486
[728]	validation_0-rmse:1.05472
[729]	validation_0-rmse:1.05472
[730]	validation_0-rmse:1.05473
[731]	validation_0-rmse:1.05473
[732]	validation_0-rmse:1.05479
[733]	validation_0-rmse:1.05485
[734]	validation_0-rmse:1.05491
[735]	validation_0-rmse:1.05494
[736]	validation_0-rmse:1.05479
[737]	validation_0-rmse:1.05495
[738]	validation_0-rmse:1.05490
[739]	validation_0-rmse:1.05489
[740]	validation_0-rmse:1.05483
[741]	validation_0-rmse:1.05496
[742]	validation_0-rmse:1.05477
[743]	validation_0-rmse:1.05489
[744]	validation_0-rmse:1.05524
[745]	validation_0-rmse:1.05530
[746]	validation_0-rmse:1.05534
[747]	validation_0-rmse:1.05544
[748]	validation_0-rmse:1.05545
[749]	validation_0-rmse:1.05550
[750]	validation_0-rmse:1.05546
[751]	validation_0-rmse:1.05552
[752]	validation_0-rmse:1.05549
[753]	validation_0-rmse:1.05557
[754]	validation_0-rmse:1.05562
[755]	validation_0-rmse:1.05561
[756]	validation_0-rmse:1.05552
[757]	validation_0-rmse:1.05554
[758]	validation_0-rmse:1.05551
[759]	validation_0-rmse:1.05562
[760]	validation_0-rmse:1.05561
[761]	validation_0-rmse:1.05607
[762]	validation_0-rmse:1.05601
```

```
[763]	validation_0-rmse:1.05638
[764]	validation_0-rmse:1.05643
[765]	validation_0-rmse:1.05612
[766]	validation_0-rmse:1.05611
[767]	validation_0-rmse:1.05612
[768]	validation_0-rmse:1.05594
[769]	validation_0-rmse:1.05642
[770]	validation_0-rmse:1.05688
[771]	validation_0-rmse:1.05673
[772]	validation_0-rmse:1.05663
[773]	validation_0-rmse:1.05658
[774]	validation_0-rmse:1.05654
[775]	validation_0-rmse:1.05668
[776]	validation_0-rmse:1.05673
[777]	validation_0-rmse:1.05675
[778]	validation_0-rmse:1.05685
[779]	validation_0-rmse:1.05695
[780]	validation_0-rmse:1.05693
[781]	validation_0-rmse:1.05685
[782]	validation_0-rmse:1.05696
[783]	validation_0-rmse:1.05737
[784]	validation_0-rmse:1.05768
[785]	validation_0-rmse:1.05767
[786]	validation_0-rmse:1.05777
[787]	validation_0-rmse:1.05774
[788]	validation_0-rmse:1.05772
[789]	validation_0-rmse:1.05762
[790]	validation_0-rmse:1.05762
[791]	validation_0-rmse:1.05772
[792]	validation_0-rmse:1.05783
[793]	validation_0-rmse:1.05779
[794]	validation_0-rmse:1.05779
[795]	validation_0-rmse:1.05795
[796]	validation_0-rmse:1.05792
[797]	validation_0-rmse:1.05785
[798]	validation_0-rmse:1.05779
[799]	validation_0-rmse:1.05770
[800]	validation_0-rmse:1.05762
[801]	validation_0-rmse:1.05890
[802]	validation_0-rmse:1.05891
[803]	validation_0-rmse:1.05889
[804]	validation_0-rmse:1.05889
[805]	validation_0-rmse:1.05888
[806]	validation_0-rmse:1.05882
[807]	validation_0-rmse:1.05898
[808]	validation_0-rmse:1.05903
[809]	validation_0-rmse:1.05923
[810]	validation_0-rmse:1.05924
```

```
[811]	validation_0-rmse:1.05926
[812]	validation_0-rmse:1.05928
[813]	validation_0-rmse:1.05929
[814]	validation_0-rmse:1.05932
[815]	validation_0-rmse:1.05923
[816]	validation_0-rmse:1.05910
[817]	validation_0-rmse:1.05908
[818]	validation_0-rmse:1.05921
[819]	validation_0-rmse:1.05917
[820]	validation_0-rmse:1.05919
[821]	validation_0-rmse:1.05927
[822]	validation_0-rmse:1.05917
[823]	validation_0-rmse:1.05911
[824]	validation_0-rmse:1.05912
[825]	validation_0-rmse:1.05899
[826]	validation_0-rmse:1.05895
[827]	validation_0-rmse:1.05901
[828]	validation_0-rmse:1.05906
[829]	validation_0-rmse:1.05919
[830]	validation_0-rmse:1.05851
[831]	validation_0-rmse:1.05925
[832]	validation_0-rmse:1.05921
[833]	validation_0-rmse:1.05921
[834]	validation_0-rmse:1.05908
[835]	validation_0-rmse:1.05923
[836]	validation_0-rmse:1.06024
[837]	validation_0-rmse:1.06015
[838]	validation_0-rmse:1.06048
[839]	validation_0-rmse:1.06002
[840]	validation_0-rmse:1.06003
[841]	validation_0-rmse:1.06000
[842]	validation_0-rmse:1.05996
[843]	validation_0-rmse:1.05999
[844]	validation_0-rmse:1.05997
[845]	validation_0-rmse:1.05991
[846]	validation_0-rmse:1.05980
[847]	validation_0-rmse:1.06142
[848]	validation_0-rmse:1.06143
[849]	validation_0-rmse:1.06139
[850]	validation_0-rmse:1.06138
[851]	validation_0-rmse:1.06126
[852]	validation_0-rmse:1.06162
[853]	validation_0-rmse:1.06172
[854]	validation_0-rmse:1.06155
[855]	validation_0-rmse:1.06146
[856]	validation_0-rmse:1.06150
[857]	validation_0-rmse:1.06134
[858]	validation_0-rmse:1.06134
```

```
[859]	validation_0-rmse:1.06129
[860]	validation_0-rmse:1.06130
[861]	validation_0-rmse:1.06135
[862]	validation_0-rmse:1.06143
[863]	validation_0-rmse:1.06211
[864]	validation_0-rmse:1.06215
[865]	validation_0-rmse:1.06210
[866]	validation_0-rmse:1.06210
[867]	validation_0-rmse:1.06198
[868]	validation_0-rmse:1.06188
[869]	validation_0-rmse:1.06192
[870]	validation_0-rmse:1.06208
[871]	validation_0-rmse:1.06216
[872]	validation_0-rmse:1.06215
[873]	validation_0-rmse:1.06235
[874]	validation_0-rmse:1.06240
[875]	validation_0-rmse:1.06248
[876]	validation_0-rmse:1.06247
[877]	validation_0-rmse:1.06350
[878]	validation_0-rmse:1.06342
[879]	validation_0-rmse:1.06317
[880]	validation_0-rmse:1.06314
[881]	validation_0-rmse:1.06294
[882]	validation_0-rmse:1.06296
[883]	validation_0-rmse:1.06294
[884]	validation_0-rmse:1.06292
[885]	validation_0-rmse:1.06255
[886]	validation_0-rmse:1.06246
[887]	validation_0-rmse:1.06249
[888]	validation_0-rmse:1.06250
[889]	validation_0-rmse:1.06249
[890]	validation_0-rmse:1.06255
[891]	validation_0-rmse:1.06290
[892]	validation_0-rmse:1.06294
[893]	validation_0-rmse:1.06357
[894]	validation_0-rmse:1.06367
[895]	validation_0-rmse:1.06370
[896]	validation_0-rmse:1.06376
[897]	validation_0-rmse:1.06531
[898]	validation_0-rmse:1.06528
[899]	validation_0-rmse:1.06525
[900]	validation_0-rmse:1.06517
[901]	validation_0-rmse:1.06523
[902]	validation_0-rmse:1.06517
[903]	validation_0-rmse:1.06517
[904]	validation_0-rmse:1.06464
[905]	validation_0-rmse:1.06467
[906]	validation_0-rmse:1.06482
```

```
[907]	validation_0-rmse:1.06480
[908]	validation_0-rmse:1.06477
[909]	validation_0-rmse:1.06459
[910]	validation_0-rmse:1.06483
[911]	validation_0-rmse:1.06469
[912]	validation_0-rmse:1.06459
[913]	validation_0-rmse:1.06458
[914]	validation_0-rmse:1.06444
[915]	validation_0-rmse:1.06446
[916]	validation_0-rmse:1.06450
[917]	validation_0-rmse:1.06459
[918]	validation_0-rmse:1.06457
[919]	validation_0-rmse:1.06460
[920]	validation_0-rmse:1.06447
[921]	validation_0-rmse:1.06445
[922]	validation_0-rmse:1.06456
[923]	validation_0-rmse:1.06452
[924]	validation_0-rmse:1.06453
[925]	validation_0-rmse:1.06451
[926]	validation_0-rmse:1.06443
[927]	validation_0-rmse:1.06488
[928]	validation_0-rmse:1.06473
[929]	validation_0-rmse:1.06467
[930]	validation_0-rmse:1.06459
[931]	validation_0-rmse:1.06482
[932]	validation_0-rmse:1.06473
[933]	validation_0-rmse:1.06464
[934]	validation_0-rmse:1.06460
[935]	validation_0-rmse:1.06456
[936]	validation_0-rmse:1.06456
[937]	validation_0-rmse:1.06452
[938]	validation_0-rmse:1.06484
[939]	validation_0-rmse:1.06478
[940]	validation_0-rmse:1.06470
[941]	validation_0-rmse:1.06439
[942]	validation_0-rmse:1.06440
[943]	validation_0-rmse:1.06440
[944]	validation_0-rmse:1.06437
[945]	validation_0-rmse:1.06438
[946]	validation_0-rmse:1.06412
[947]	validation_0-rmse:1.06415
[948]	validation_0-rmse:1.06439
[949]	validation_0-rmse:1.06431
[950]	validation_0-rmse:1.06448
[951]	validation_0-rmse:1.06448
[952]	validation_0-rmse:1.06446
[953]	validation_0-rmse:1.06443
[954]	validation_0-rmse:1.06483
```

```
[955]	validation_0-rmse:1.06451
[956]	validation_0-rmse:1.06461
[957]	validation_0-rmse:1.06455
[958]	validation_0-rmse:1.06454
[959]	validation_0-rmse:1.06455
[960]	validation_0-rmse:1.06461
[961]	validation_0-rmse:1.06491
[962]	validation_0-rmse:1.06488
[963]	validation_0-rmse:1.06497
[964]	validation_0-rmse:1.06483
[965]	validation_0-rmse:1.06486
[966]	validation_0-rmse:1.06500
[967]	validation_0-rmse:1.06487
[968]	validation_0-rmse:1.06488
[969]	validation_0-rmse:1.06545
[970]	validation_0-rmse:1.06543
[971]	validation_0-rmse:1.06543
[972]	validation_0-rmse:1.06581
[973]	validation_0-rmse:1.06578
[974]	validation_0-rmse:1.06580
[975]	validation_0-rmse:1.06628
[976]	validation_0-rmse:1.06626
[977]	validation_0-rmse:1.06625
[978]	validation_0-rmse:1.06619
[979]	validation_0-rmse:1.06619
[980]	validation_0-rmse:1.06628
[981]	validation_0-rmse:1.06630
[982]	validation_0-rmse:1.06640
[983]	validation_0-rmse:1.06642
[984]	validation_0-rmse:1.06639
[985]	validation_0-rmse:1.06634
[986]	validation_0-rmse:1.06582
[987]	validation_0-rmse:1.06561
[988]	validation_0-rmse:1.06591
[989]	validation_0-rmse:1.06588
[990]	validation_0-rmse:1.06587
[991]	validation_0-rmse:1.06603
[992]	validation_0-rmse:1.06595
[993]	validation_0-rmse:1.06583
[994]	validation_0-rmse:1.06581
[995]	validation_0-rmse:1.06586
[996]	validation_0-rmse:1.06604
[997]	validation_0-rmse:1.06601
[998]	validation_0-rmse:1.06665
[999]	validation_0-rmse:1.06659
```

[13]:
```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=0.8, enable_categorical=False,
             eta=0.2, gamma=0, gpu_id=-1, importance_type=None,
             interaction_constraints='', learning_rate=0.200000003,
             max_delta_step=0, max_depth=8, min_child_weight=300, missing=nan,
             monotone_constraints='()', n_estimators=1000, n_jobs=16,
             num_parallel_tree=1, predictor='auto', random_state=42,
             reg_alpha=0.2, reg_lambda=2.0, scale_pos_weight=1, seed=42,
             subsample=0.8, tree_method='approx', validate_parameters=1,
             verbosity=None)
```

[15]:
```
print(np.sqrt(mean_squared_error(Y_valid, model_xgb.predict(X_valid))))
```

```
0.9589862849643996
```

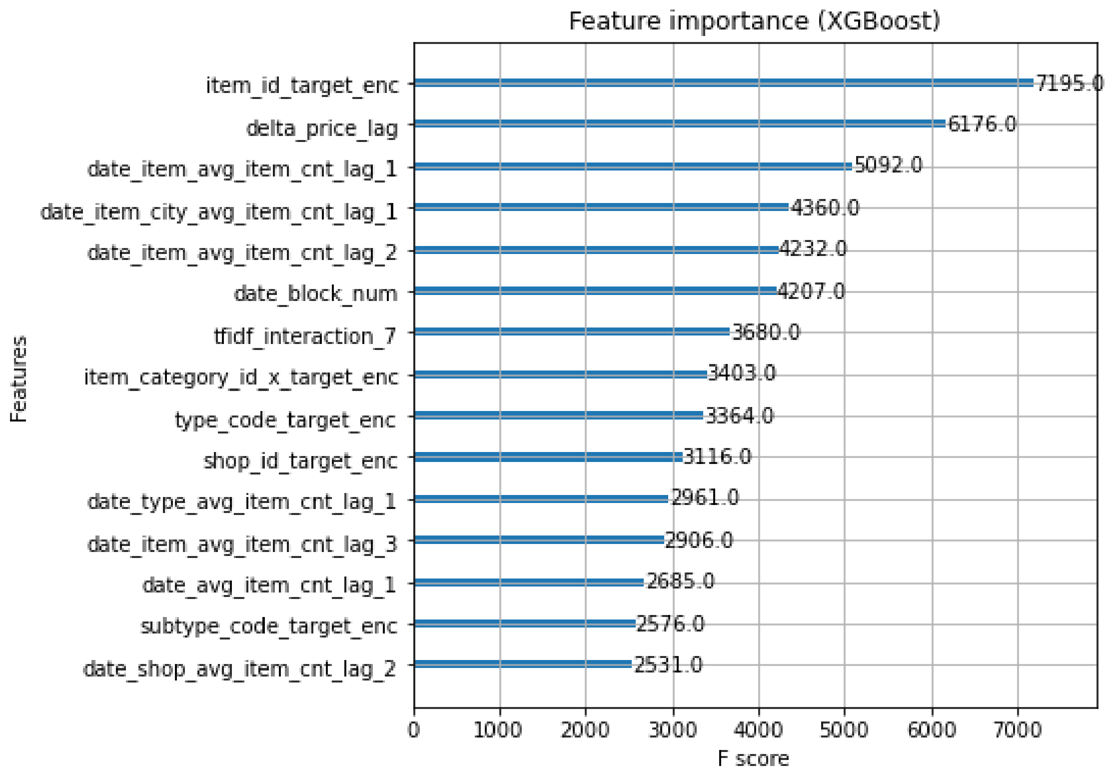
[16]:
```
joblib.dump(model_xgb, 'models/model2_xgb.pkl');
```

[17]:
```
model_xgb = joblib.load('models/model2_xgb.pkl')
```

[18]:
```
print(model_xgb.best_iteration)
```

```
8
```

[19]:
```
fig, ax = plt.subplots(figsize=(6,6))
xgb.plot_importance(model_xgb, max_num_features=15, title='Feature importance␣
↪(XGBoost)', ax=ax)
plt.show()
```



3.3.1 1.3.1 Training XGBoost Regressor on GPU

[6]:
```
model_xgb = xgb.XGBRegressor(max_depth=8, n_estimators=1000,␣
↪min_child_weight=300,
                              colsample_bytree=0.8, subsample=0.8, eta=0.2,␣
↪seed=42,
                              reg_alpha=0.2, reg_lambda=2.0,␣
↪tree_method='gpu_hist')

model_xgb.fit(X_train, Y_train, eval_metric='rmse', eval_set=[(X_valid,␣
↪Y_valid)],
              verbose=100, early_stopping_rounds=1000)  #start:10:29pm, end 1:
↪23am
```

```
[0]     validation_0-rmse:1.09784
[100]   validation_0-rmse:1.00856
[200]   validation_0-rmse:1.02441
[300]   validation_0-rmse:1.03025
[400]   validation_0-rmse:1.04731
[500]   validation_0-rmse:1.05778
[600]   validation_0-rmse:1.06427
[700]   validation_0-rmse:1.06467
```

```
[800]	validation_0-rmse:1.07077
[900]	validation_0-rmse:1.07413
[999]	validation_0-rmse:1.08529
```

[6]:
```
XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
             colsample_bynode=1, colsample_bytree=0.8, enable_categorical=False,
             eta=0.2, gamma=0, gpu_id=0, importance_type=None,
             interaction_constraints='', learning_rate=0.200000003,
             max_delta_step=0, max_depth=8, min_child_weight=300, missing=nan,
             monotone_constraints='()', n_estimators=1000, n_jobs=16,
             num_parallel_tree=1, predictor='auto', random_state=42,
             reg_alpha=0.2, reg_lambda=2.0, scale_pos_weight=1, seed=42,
             subsample=0.8, tree_method='gpu_hist', validate_parameters=1,
             verbosity=None)
```

The XGBoost method ran on CPU cost 1h 23 min, ran on GPU only cost 57 seconds.

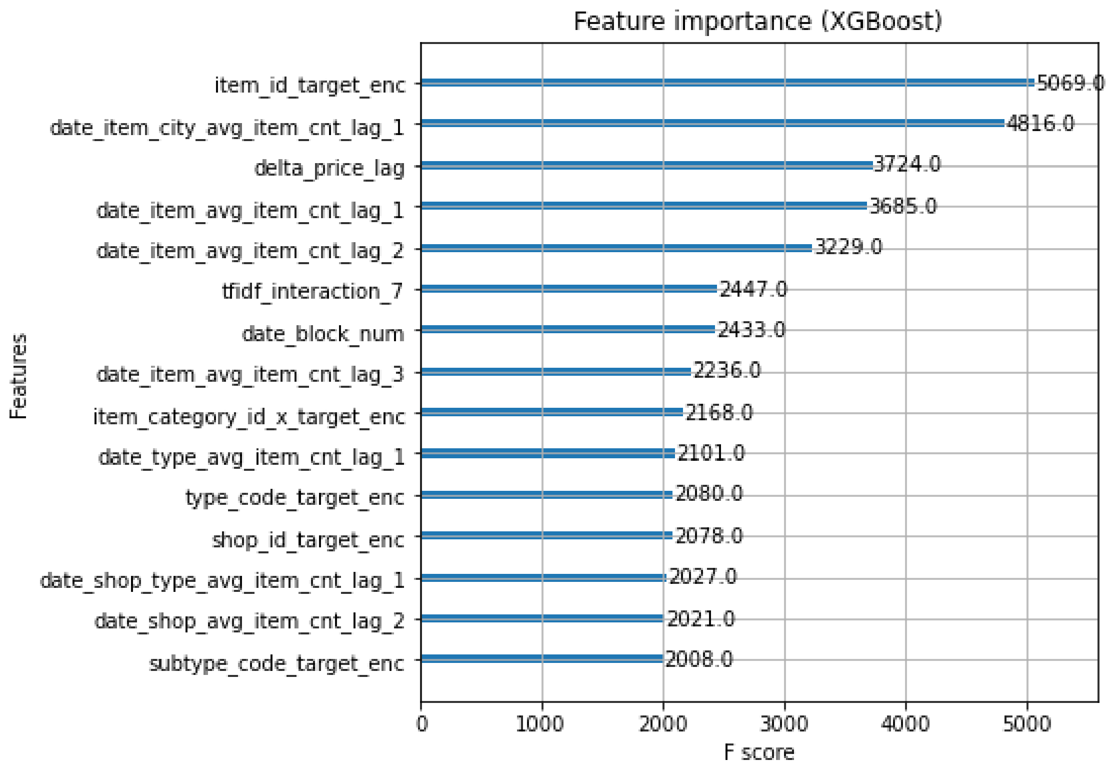
[108]:
```
print(np.sqrt(mean_squared_error(Y_valid, model_xgb.predict(X_valid))))
```

```
0.9800386356080234
```

[109]:
```
print(model_xgb.best_iteration)
```

```
5
```

[110]:
```
fig, ax = plt.subplots(figsize=(6,6))
xgb.plot_importance(model_xgb, max_num_features=15, title='Feature importance␣
↪(XGBoost)', ax=ax)
plt.show()
```



3.4 1.4 CatBoost Regressor on GPU

[7]:
```
model_cb = cb.CatBoostRegressor(task_type='GPU',iterations=1000, depth=8,␣
↪learning_rate=0.1,
                                l2_leaf_reg=9, eval_metric='RMSE')
model_cb.fit(X_train, Y_train, eval_set=(X_valid, Y_valid),
                  verbose_eval = 100, early_stopping_rounds = 1000, plot=True)
```

```
MetricVisualizer(layout=Layout(align_self='stretch', height='500px'))
```

```
0:      learn: 1.1450800        test: 1.1120357 best: 1.1120357 (0)     total:
46.3ms   remaining: 46.3s
100:    learn: 0.8194890        test: 0.9592186 best: 0.9590034 (97)    total:
4.7s     remaining: 41.9s
200:    learn: 0.7932455        test: 0.9500215 best: 0.9499622 (139)   total:
9.36s    remaining: 37.2s
300:    learn: 0.7778004        test: 0.9552388 best: 0.9491708 (223)   total:
14s      remaining: 32.5s
400:    learn: 0.7668328        test: 0.9629103 best: 0.9491708 (223)   total:
18.6s    remaining: 27.8s
500:    learn: 0.7576183        test: 0.9766797 best: 0.9491708 (223)   total:
23.3s    remaining: 23.2s
600:    learn: 0.7508175        test: 0.9793986 best: 0.9491708 (223)   total:
```

```
27.9s    remaining: 18.5s
700:    learn: 0.7446557        test: 0.9856344 best: 0.9491708 (223)   total:
32.6s    remaining: 13.9s
800:    learn: 0.7391661        test: 0.9888263 best: 0.9491708 (223)   total:
37.2s    remaining: 9.24s
900:    learn: 0.7343022        test: 0.9916049 best: 0.9491708 (223)   total:
41.8s    remaining: 4.59s
999:    learn: 0.7300689        test: 1.0012056 best: 0.9491708 (223)   total:
46.4s    remaining: 0us
bestTest = 0.9491707604
bestIteration = 223
Shrink model to first 224 iterations.
```

[7]:

```
<catboost.core.CatBoostRegressor at 0x1e443d08cd0>
```

[25]:

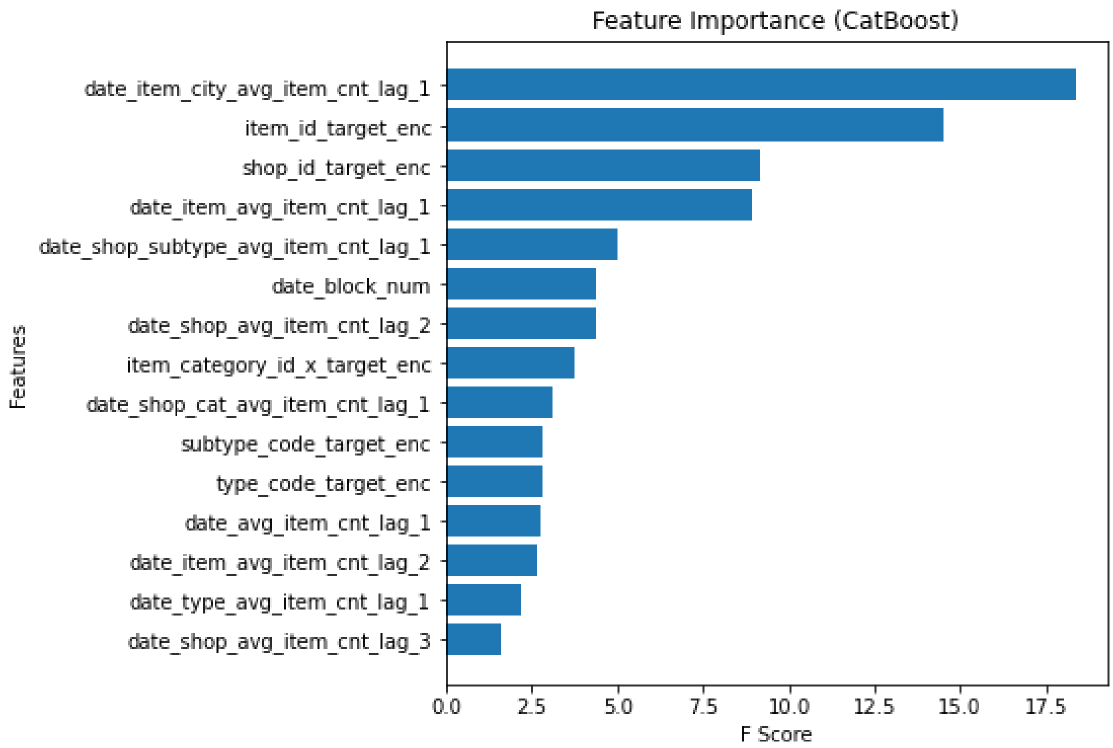
```
dict_feat_imp = {
    'feat_name': X_train.columns.values,
    'feat_imp': model_cb.feature_importances_
}
df_feat_imp = pd.DataFrame(dict_feat_imp).sort_values('feat_imp',␣
 ↪ascending=False)
df_feat_imp.head()
```

[25]:

	feat_name	feat_imp
18	date_item_city_avg_item_cnt_lag_1	18.358875
23	item_id_target_enc	14.491291
22	shop_id_target_enc	9.157742
2	date_item_avg_item_cnt_lag_1	8.950658
17	date_shop_subtype_avg_item_cnt_lag_1	5.023442

[30]:

```
max_num_features=15
fig, ax = plt.subplots(figsize=(6,6))

plt.barh(df_feat_imp.iloc[:max_num_features,0],
        df_feat_imp.iloc[:max_num_features,1])
plt.gca().invert_yaxis()
plt.title('Feature Importance (CatBoost)')
plt.xlabel('F Score')
plt.ylabel('Features')
plt.show()
```



[32]:
```
joblib.dump(model_cb, 'models/model_catboost.pkl');
```

[31]:
```
X_test =  all_data[all_data['date_block_num']==34].drop(['item_cnt_month'],␣
 ↪axis=1)
Y_test_cb = model_cb.predict(X_test).clip(0, 20)
temp = pd.DataFrame({'ID': test.index, 'item_cnt_month': Y_test_cb})
temp.to_csv('submissions/submission_' + 'catboost' + '.csv', index=False)
```

3.5 1.5 Linear models

We implement linear models `LinearRegression`, `Ridge`, `Lasso` as benchmarks.We need to implement *feature scaling* before fitting on the dataset.

3.5.1 1.5.1 LinearRegression model

[73]:
```
scaler = StandardScaler().fit(X_train)
X_train_std = scaler.transform(X_train)
X_valid_std = scaler.transform(X_valid)
```

[74]:
```
model_linreg = LinearRegression(fit_intercept=True)
model_linreg.fit(X_train_std, Y_train)
```

[74]:
```
LinearRegression()
```

[75]:
```
print(np.sqrt(mean_squared_error(Y_valid, model_linreg.predict(X_valid_std))))
```

```
0.9900095967566781
```

[76]:
```
joblib.dump(model_linreg, 'models/model3_linreg.pkl');
```

3.5.2 1.5.2 Ridge model

[77]:
```
model_ridge = Ridge(alpha=4.0, fit_intercept=True, max_iter=1000, solver='saga')
model_ridge.fit(X_train_std, Y_train)
```

[77]:
```
Ridge(alpha=4.0, max_iter=1000, solver='saga')
```

[78]:
```
print(np.sqrt(mean_squared_error(Y_valid, model_ridge.predict(X_valid_std))))
```

```
0.9900082805056044
```

[79]:
```
joblib.dump(model_ridge, 'models/model4_ridge.pkl');
```

3.5.3 1.5.3 Lasso model

[80]:
```
model_lasso = Lasso(alpha=10.0, fit_intercept=True, max_iter=1000)
model_lasso.fit(X_train_std, Y_train)
```

[80]:
```
Lasso(alpha=10.0)
```

[81]:
```
print(np.sqrt(mean_squared_error(Y_valid, model_lasso.predict(X_valid_std))))
```

```
1.1365538511881084
```

[82]:
```
joblib.dump(model_lasso, 'models/model5_lasso.pkl');
```

4 2. Ensembling Method

4.1 2.1 Ensembling: LightGBM + Linear Regression

We will implement the KFold scheme time series implemented in the previous assignment.

1. Split the train data into chunks of duration T, select first M chunks.
2. Fit N diverse models on those M chunks and predict for the chunk M+1. Then fit those models on first M+1 chunks and predict for chunk M+2 and so on, until the end. After that, use all train data to fit models and get predictions for test. Now we will have meta-features for the chunks starting from M+1 and meta-features for test.
3. Now we can use meta-features from first K chunks [M+1, M+2, …, M+K] to fit level 2 models and validate them on chunk M+K+1.Essentially we are back to step 1 with lesser amount of chunks and meta-features instead of features.

We start with **1st generation features**:

[85]:
```
model_lgbm = joblib.load('models/model1_lgbm.pkl')
model_linreg = joblib.load('models/model3_linreg.pkl')
```

[86]:
```
Y_valid_lgbm = model_lgbm.predict(X_valid)
Y_valid_lreg = model_linreg.predict(X_valid_std)
X_valid_level2 = np.c_[Y_valid_lgbm, Y_valid_lreg]
X_valid_level2[:10,:]
```

[86]:
```
array([[ 0.02086253, -0.06266107],
       [ 0.11747867, -0.04090796],
       [ 0.11699397,  0.11597089],
       [ 0.23048892,  0.40084696],
       [ 0.01993154,  0.00991157],
       [ 0.00094701, -0.08366587],
       [ 0.01659449, -0.09086691],
       [ 0.02121464, -0.10312856],
       [ 0.02121464, -0.10569445],
       [ 0.01867623, -0.09369751]])
```

Then **2nd generation feature**:

[87]:
```
X_train = all_data[all_data['date_block_num']<=32].drop(['item_cnt_month'],␣
 ↪axis=1)
Y_train = all_data[all_data['date_block_num']<=32]['item_cnt_month']
X_valid = all_data[all_data['date_block_num']==33].drop(['item_cnt_month'],␣
 ↪axis=1)
Y_valid = all_data[all_data['date_block_num']==33]['item_cnt_month']
```

[88]:
```
dates = all_data['date_block_num']
last_block = dates.max()
print('Test date_block_num is ',last_block)
dates_train = dates[dates < last_block]
dates_test = dates[dates == last_block]
```

```
Test date_block_num is  34
```

[89]:
```
periods = np.arange(26, 33, 1)
periods
```

[89]:
```
array([26, 27, 28, 29, 30, 31, 32])
```

[90]:
```
dates_train_level2 = dates_train[dates_train.isin(periods)]
Y_train_level2 = Y_train[dates_train.isin(periods)]
X_train_level2 = np.zeros([Y_train_level2.shape[0], 2])
```

[91]:
```
Y_train[dates_train.isin(periods)].shape == dates_train_level2.shape
```

[91]:
```
True
```

[ ]:
```
# cur_block_num = 26
# x = all_data[all_data['date_block_num'] < cur_block_num].
↪drop(['item_cnt_month'], axis=1)
# y = all_data[all_data['date_block_num'] < cur_block_num]['item_cnt_month'].
↪values
# x_test = all_data[all_data['date_block_num'] == cur_block_num].
↪drop(['item_cnt_month'], axis=1)
# y = all_data[all_data['date_block_num'] == cur_block_num]['item_cnt_month'].
↪values
```

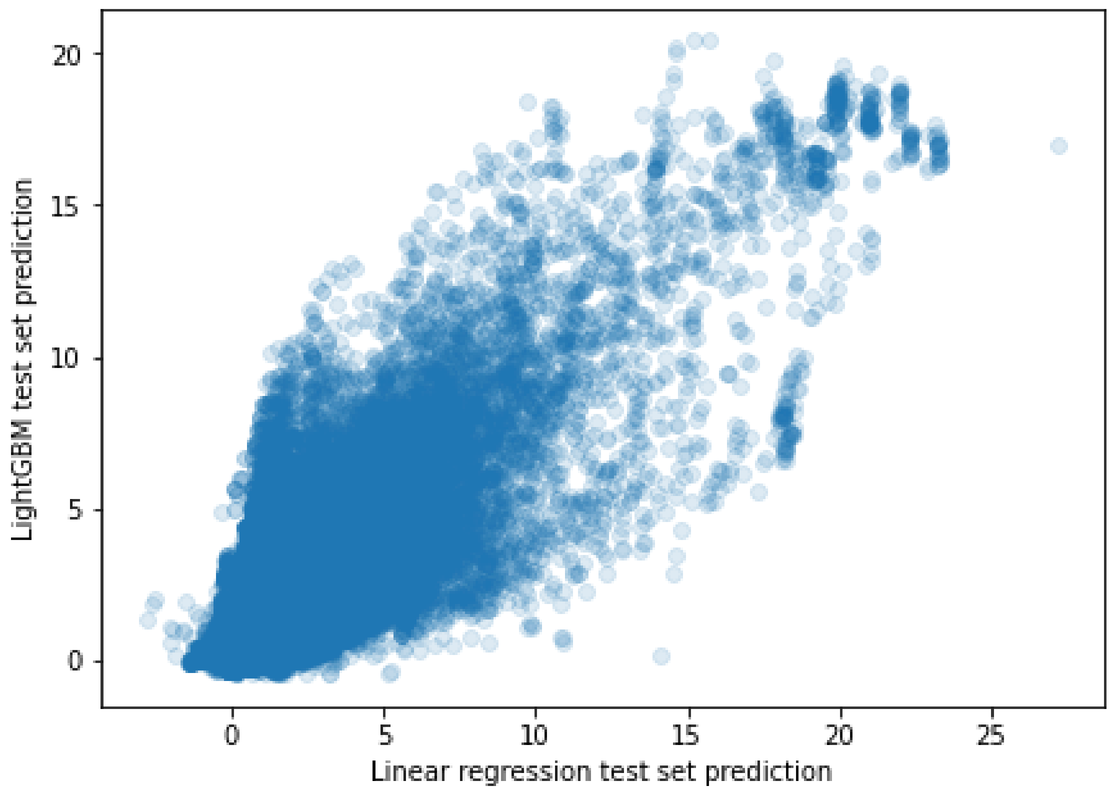
[92]:
```
#Now fill 'X_train_level2' with metafeatures

#params for lightGBM
params = {
    'boosting_type': 'dart',
    'metric': 'l2_root', # RMSE
    'verbose': 1,
    'seed': 0,
    'max_depth': 8,
    'learning_rate': 0.1,
    'reg_lambda': 2.0,
    'reg_alpha': 2.0,
    'subsample': 0.7,
    'num_leaves': 20,
    'feature_fraction': 0.8,
    'drop_rate': 0.2
}

for cur_block_num in periods:

    print('cur_block_num:', cur_block_num)

    # 1. split X_train into parts
    x = all_data[all_data['date_block_num'] < cur_block_num].
↪drop(['item_cnt_month'], axis=1)
    y = all_data[all_data['date_block_num'] < cur_block_num]['item_cnt_month'].
↪values
    x_test = all_data[all_data['date_block_num'] == cur_block_num].
↪drop(['item_cnt_month'], axis=1)
    y_test = all_data[all_data['date_block_num'] ==␣
↪cur_block_num]['item_cnt_month'].values

    #2. Fit Linear regression
    print('Fitting Linear Regression:')
    lr = LinearRegression(fit_intercept=True, normalize=True)
    scaler = StandardScaler().fit(x)
    x_std = scaler.transform(x)
```

```
    x_test_std = scaler.transform(x_test)
    lr.fit(x_std, y)
    Y_test_1 = lr.predict(x_test_std)

    # 3. Fit LightGBM and prediction
    print('LightGBM:')
    n_trees = 500
    lgb_train = lgb.Dataset(x, y)
    lgb_valid = lgb.Dataset(x_test, y_test)
    model_lgbm = lgb.train(params, lgb_train, num_boost_round=n_trees,␣
↪valid_sets=lgb_valid,
                           early_stopping_rounds=50, verbose_eval=50)
    Y_test_2 = model_lgbm.predict(x_test)

    # 4. Store predictions from 2 & 3 to 'X_train_level2'
    X_train_level2[dates_train_level2 == cur_block_num] = np.c_[Y_test_1,␣
↪Y_test_2]
    clear_output()

print(X_train_level2.mean(axis=0))
```

```
[0.31901372 0.27264671]
```

[95]:
```
np.savez('./data_processed/X_train_level2.npz',X_train_level2=X_train_level2)
```

[94]:
```
plt.figure(figsize=(7,5))
plt.scatter(X_train_level2[:,0], X_train_level2[:,1], marker='o', alpha=0.15)
plt.xlabel('Linear regression test set prediction')
plt.ylabel('LightGBM test set prediction')
plt.show()
```



Now we can do **stacking**.

[96]:
```
Y_train_level2 = Y_train[dates_train.isin(periods)]
lr_stack = LinearRegression().fit(X_train_level2, Y_train_level2)
```

[97]:
```
train_preds = lr_stack.predict(X_train_level2)
rmse_train = np.sqrt(mean_squared_error(Y_train_level2, train_preds))

valid_preds = lr_stack.predict(X_valid_level2)
rmse_valid = np.sqrt(mean_squared_error(Y_valid, valid_preds))

print('RMSE train: ', rmse_train)
print('RMSE test: ', rmse_valid)
```

```
RMSE train:  0.8091817048380416
RMSE test:  0.9788412024508074
```

5 3. Predictions

[98]:
```
X_test =  all_data[all_data['date_block_num']==34].drop(['item_cnt_month'],␣
↪axis=1)

Y_test_lgbm = model_lgbm.predict(X_test).clip(0, 20)
Y_test_xgbm = model_xgb.predict(X_test).clip(0, 20)
Y_test_lreg = model_linreg.predict(X_test).clip(0, 20)
Y_test_ridge = model_ridge.predict(X_test).clip(0, 20)
Y_test_lasso = model_lasso.predict(X_test).clip(0, 20)
```

[ ]:
```

```

[105]:
```
models = [Y_test_lgbm, Y_test_xgbm, Y_test_lreg, Y_test_ridge,
          Y_test_lasso]
names = ['LGBM', 'XGBM', 'LINREG', 'RIDGE', 'LASSO']

for model, name in zip(models, names):
    temp = pd.DataFrame({'ID': test.index, 'item_cnt_month': model})
    temp.to_csv('submissions/submission_' + str(name) + '.csv', index=False)
```

Public leaderboard **Score**:

Model	CatBoost	XGBoost	LightGBM	LinearRegression	Ridge	Lasso
Score	0.96326	0.98971	0.98313	1.07821	1.07822	1.21744

My official submission is LightGBM. Writing the data to `submission.cvs`.

[106]:
```
temp = pd.DataFrame({'ID': test.index, 'item_cnt_month': Y_test_lgbm})
temp.to_csv('submissions/submission.csv', index=False)
```