# Fetching the data

In [3]:

```
# using CurlWget

!wget --header="Host: storage.googleapis.com" --header="User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/83.0.4103.116 Safari/537.36" --header="A
ccept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/s
d-exchange;v=b3;q=0.9" --header="Accept-Language: en-US,en;q=0.9,hi;q=0.8" --header="Referer: http
s://www.kaggle.com/" "https://storage.googleapis.com/kaggle-competitions-data/kaggle-
v2/11835/224935/bundle/archive.zip?GoogleAccessId=web-data@kaggle-
161607.iam.gserviceaccount.com&Expires=1594633592&Signature=PjjHj7p8FG5sUTGGeBGB%2FE7kCUW5uhc7eUz6e
ISDnrg1vo845DVT0%2FQCyt8KwGtSZ0hf9c86kfzIseGtR94Ae77SwCGig2hy2IRCGi5Fbw1Hjmj4dQHPeOR0wYoaUtQTUM8NYq
sHICeGzblvhF4%2Fkthz2OSu6pk8kKBU3Q6ITjZV3%2Fpj%2FKIpBARH%2Ff9irshimcecf0xczZoNGnsQNH0HjrHKb9QI7L4qS
ez7l%2FnzlucEUSmWFgcIUBov7pRiddQRWvk7Gq3Gx1KxcTEASVD7%2B1KsiMxAiHoTCH7gL5P87JPstdsduhoGQi7Tyryigtns
BU2Z%2B54w%3D%3D&response-content-disposition=attachment%3B+filename%3Dreducing-commercial-aviatio
n-fatalities.zip" -c -O 'reducing-commercial-aviation-fatalities.zip'
```

```
--2020-07-10 09:46:54--  https://storage.googleapis.com/kaggle-competitions-data/kaggle-
v2/11835/224935/bundle/archive.zip?GoogleAccessId=web-data@kaggle-
161607.iam.gserviceaccount.com&Expires=1594633592&Signature=PjjHj7p8FG5sUTGGeBGB%2FE7kCUW5uhc7eUz6e
ISDnrg1vo845DVT0%2FQCyt8KwGtSZ0hf9c86kfzIseGtR94Ae77SwCGig2hy2IRCGi5Fbw1Hjmj4dQHPeOR0wYoaUtQTUM8NYq
sHICeGzblvhF4%2Fkthz2OSu6pk8kKBU3Q6ITjZV3%2Fpj%2FKIpBARH%2Ff9irshimcecf0xczZoNGnsQNH0HjrHKb9QI7L4qS
ez7l%2FnzlucEUSmWFgcIUBov7pRiddQRWvk7Gq3Gx1KxcTEASVD7%2B1KsiMxAiHoTCH7gL5P87JPstdsduhoGQi7Tyryigtns
BU2Z%2B54w%3D%3D&response-content-disposition=attachment%3B+filename%3Dreducing-commercial-
aviation-fatalities.zip
Resolving storage.googleapis.com (storage.googleapis.com)... 173.194.216.128, 108.177.11.128, 172.
217.204.128, ...
Connecting to storage.googleapis.com (storage.googleapis.com)|173.194.216.128|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2290631660 (2.1G) [application/zip]
Saving to: 'reducing-commercial-aviation-fatalities.zip'

reducing-commercial 100%[===================>]   2.13G   137MB/s    in 17s

2020-07-10 09:47:11 (130 MB/s) - 'reducing-commercial-aviation-fatalities.zip' saved
[2290631660/2290631660]
```

In [5]:

```
# unzipping the folders

!unzip /home/nehasinghsikerwar/Reducing_commercial_Aviation_fatalities/reducing-commercial-aviatio
n-fatalities.zip -d /home/nehasinghsikerwar/Reducing_commercial_Aviation_fatalities
```

```
Archive:  /home/nehasinghsikerwar/Reducing_commercial_Aviation_fatalities/reducing-commercial-
aviation-fatalities.zip
  inflating: /home/nehasinghsikerwar/Reducing_commercial_Aviation_fatalities/sample_submission.csv
  inflating: /home/nehasinghsikerwar/Reducing_commercial_Aviation_fatalities/test.csv
  inflating: /home/nehasinghsikerwar/Reducing_commercial_Aviation_fatalities/train.csv
```

# Reading the data

In [1]:

```
# importing important libraries

import warnings
warnings.filterwarnings("ignore")
import shutil
import os
import pandas as pd
import matplotlib
matplotlib.use(u'nbAgg')
```

```
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import pickle
from sklearn.manifold import TSNE
from sklearn import preprocessing
import pandas as pd
from multiprocessing import Process# this is used for multithreading
import multiprocessing
import codecs# this is used for file operations
import random as r
import xgboost
from sklearn.model_selection import RandomizedSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.calibration import CalibratedClassifierCV
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import log_loss
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
import gc
import lightgbm as lgb
```

In [2]:

```
# https://www.kaggle.com/kamalchhirang/forgot-to-shuffle-the-data-while-splitting-boom

# Memory Optimization

def memory_opt(data):

    "will change the datatypes of the columns for optimal memory usage "

    initial_memory = data.memory_usage().sum() / 1024**2
    print(' Initial Memory of the dataframe is {:.2f} '.format(initial_memory))

    for c in data.columns:
        column_type = data[c].dtype

        if column_type != object:
            min_val = data[c].min()
            max_val = data[c].max()
            if str(column_type)[:3] == 'int':
                if min_val > np.iinfo(np.int8).min and max_val < np.iinfo(np.int8).max:
                    data[c] = data[c].astype(np.int8)
                elif min_val > np.iinfo(np.int16).min and max_val < np.iinfo(np.int16).max:
                    data[c] = data[c].astype(np.int16)
                elif min_val > np.iinfo(np.int32).min and max_val < np.iinfo(np.int32).max:
                    data[c] = data[c].astype(np.int32)
                elif min_val > np.iinfo(np.int64).min and max_val < np.iinfo(np.int64).max:
                    data[c] = data[c].astype(np.int64)
            else:
                if min_val > np.finfo(np.float32).min and max_val < np.finfo(np.float32).max:
                    data[c] = data[c].astype(np.float32)
                else:
                    data[c] = data[c].astype(np.float64)
        else:
            data[c] = data[c].astype('category')

    final_memory = data.memory_usage().sum() / 1024**2
    print('after optimization, Memory is: {:.2f} '.format(final_memory))
    print('Reduced by {:.1f}%'.format(100 * (initial_memory - final_memory) / initial_memory))

    return data
```

In [ ]:

```
```

https://github.com/pandas-dev/pandas/issues/20642

I am not using float16 dtype as it has some issues as mentioned in the above link. It gives nan while calculating the mean, so I'm passing the float32 and float64 as dtype in memory optimization function in above cell

passing the float32 and float64 as dtype in memory optimization function in above cell.

In [ ]:

```
train=pd.read_csv('train.csv')
train=memory_opt(train)
print(train.shape)
```

In [5]:

```
train.head()
```

Out[5]:

| | crew | experiment | time | seat | eeg_fp1 | eeg_f7 | eeg_f8 | eeg_t4 | eeg_t6 | eeg_t5 | ... | eeg_c4 | eeg_p4 | eeg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 1 | CA | 0.011719 | 1 | -5.28545 | 26.775801 | -9.527310 | -12.793200 | 16.717800 | 33.737499 | ... | 37.368999 | 17.437599 | 19.20 |
| **1** | 1 | CA | 0.015625 | 1 | -2.42842 | 28.430901 | -9.323510 | -3.757230 | 15.969300 | 30.443600 | ... | 31.170799 | 19.399700 | 19.68 |
| **2** | 1 | CA | 0.019531 | 1 | 10.67150 | 30.420200 | 15.350700 | 24.724001 | 16.143101 | 32.142799 | ... | -12.012600 | 19.396299 | 23.17 |
| **3** | 1 | CA | 0.023438 | 1 | 11.45250 | 25.609800 | 2.433080 | 12.412500 | 20.533300 | 31.494101 | ... | 18.574100 | 23.156401 | 22.64 |
| **4** | 1 | CA | 0.027344 | 1 | 7.28321 | 25.942600 | 0.113564 | 5.748000 | 19.833599 | 28.753599 | ... | 6.555440 | 22.754700 | 22.67 |

5 rows × 28 columns

In [34]:

```
test=pd.read_csv('test.csv')
test=memory_opt(test)
print(test.shape)
```

```
 Initial Memory of the dataframe is 3837.77
after optimization, Memory is: 1764.69
Reduced by 54.0%
(17965143, 28)
```

In [4]:

```
test.head()
```

Out[4]:

| | id | crew | experiment | time | seat | eeg_fp1 | eeg_f7 | eeg_f8 | eeg_t4 | eeg_t6 | ... | eeg_f4 | eeg_c4 | eeg_p4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 0 | 1 | LOFT | 0.000000 | 0 | 17.899500 | 6.127830 | 0.994807 | -28.206200 | -47.695499 | ... | -7.044480 | -14.405100 | -4.03384 |
| **1** | 1 | 1 | LOFT | 0.000000 | 1 | 45.883202 | 94.749001 | 23.290800 | 1.392000 | 2.060940 | ... | 19.887501 | -215.179001 | 2.11832 |
| **2** | 2 | 1 | LOFT | 0.003906 | 0 | 33.120098 | 28.356501 | -7.239220 | -7.690860 | -25.833799 | ... | -7.642560 | -10.363600 | 10.95050 |
| **3** | 3 | 1 | LOFT | 0.003906 | 1 | 43.280102 | 95.887001 | 18.702299 | -1.432890 | -4.232600 | ... | 13.826600 | -214.223007 | -4.91354 |
| **4** | 4 | 1 | LOFT | 0.007812 | 0 | 7.929110 | 3.460380 | -10.860800 | -26.366699 | -25.894699 | ... | 2.045450 | -20.788799 | -3.61418 |

5 rows × 28 columns

In [8]:

```
# encoding of categorical type features (label) and changing data type to int8

dict_1 = {'A': 0, 'B': 1, 'C': 2, 'D': 3}
dict_2 = {'CA':0,'DA':1,'SS':3,'LOFT':4}
```

```
train['event'] = train['event'].apply(lambda x: dict_1[x])
train['event'] = train['event'].astype('int8')
train['experiment'] = train['experiment'].apply(lambda x: dict_2[x])
train['experiment'] = train['experiment'].astype('int8')

y_train = train['event']
y_train
```

Out[8]:

```
0          0
1          0
2          0
3          0
4          0
          ..
4867416    0
4867417    0
4867418    0
4867419    0
4867420    0
Name: event, Length: 4867421, dtype: int8
```

In [15]:

```
dict_2 = {'CA':0,'DA':1,'SS':3,'LOFT':4}

test['experiment'] = test['experiment'].apply(lambda x: dict_2[x])
test['experiment'] = test['experiment'].astype('int8')
```

# EDA

In [30]:

```
train.columns
```

Out[30]:

```
Index(['crew', 'experiment', 'time', 'seat', 'eeg_fp1', 'eeg_f7', 'eeg_f8',
       'eeg_t4', 'eeg_t6', 'eeg_t5', 'eeg_t3', 'eeg_fp2', 'eeg_o1', 'eeg_p3',
       'eeg_pz', 'eeg_f3', 'eeg_fz', 'eeg_f4', 'eeg_c4', 'eeg_p4', 'eeg_poz',
       'eeg_c3', 'eeg_cz', 'eeg_o2', 'ecg', 'r', 'gsr', 'event'],
      dtype='object')
```

In [31]:

```
test.columns
```

Out[31]:

```
Index(['id', 'crew', 'experiment', 'time', 'seat', 'eeg_fp1', 'eeg_f7',
       'eeg_f8', 'eeg_t4', 'eeg_t6', 'eeg_t5', 'eeg_t3', 'eeg_fp2', 'eeg_o1',
       'eeg_p3', 'eeg_pz', 'eeg_f3', 'eeg_fz', 'eeg_f4', 'eeg_c4', 'eeg_p4',
       'eeg_poz', 'eeg_c3', 'eeg_cz', 'eeg_o2', 'ecg', 'r', 'gsr'],
      dtype='object')
```

In [46]:

```
train.shape
```

Out[46]:

```
(4867421, 28)
```

In [47]:

```
test.shape
```

Out[47]:

```
(17965143, 28)
```

In [7]:

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4867421 entries, 0 to 4867420
Data columns (total 28 columns):
crew          int8
experiment    int8
time          float32
seat          int8
eeg_fp1       float32
eeg_f7        float32
eeg_f8        float32
eeg_t4        float32
eeg_t6        float32
eeg_t5        float32
eeg_t3        float32
eeg_fp2       float32
eeg_o1        float32
eeg_p3        float32
eeg_pz        float32
eeg_f3        float32
eeg_fz        float32
eeg_f4        float32
eeg_c4        float32
eeg_p4        float32
eeg_poz       float32
eeg_c3        float32
eeg_cz        float32
eeg_o2        float32
ecg           float32
r             float32
gsr           float32
event         int8
dtypes: float32(24), int8(4)
memory usage: 464.2 MB
```

## Univariate analysis

In [17]:

```
train.describe()
```

Out[17]:

| | crew | experiment | time | seat | eeg_fp1 | eeg_f7 | eeg_f8 | eeg_t4 | eeg_t |
|---|---|---|---|---|---|---|---|---|---|
| **count** | 4.867421e+06 | 4.867421e+06 | 4.867421e+06 | 4.867421e+06 | 4.867421e+06 | 4.867421e+06 | 4.867421e+06 | 4.867421e+06 | 4.867421e+0 |
| **mean** | 5.538783e+00 | 1.296446e+00 | 1.782358e+02 | 4.999531e-01 | 3.746335e+00 | 1.360002e+00 | 1.213644e+00 | 7.350925e-02 | 7.845481e-0 |
| **std** | 3.409353e+00 | 1.235775e+00 | 1.039592e+02 | 5.000000e-01 | 4.506762e+01 | 3.518923e+01 | 3.519242e+01 | 2.431472e+01 | 1.803932e+0 |
| **min** | 1.000000e+00 | 0.000000e+00 | 3.000000e-03 | 0.000000e+00 | -1.361360e+03 | -1.581330e+03 | -1.643950e+03 | -1.516640e+03 | -1.220510e+0 |
| **25%** | 3.000000e+00 | 0.000000e+00 | 8.808100e+01 | 0.000000e+00 | -9.200250e+00 | -8.325150e+00 | -8.767610e+00 | -7.367240e+00 | -6.102000e+0 |
| **50%** | 5.000000e+00 | 1.000000e+00 | 1.769297e+02 | 0.000000e+00 | 3.819020e-01 | 4.264100e-02 | 1.140390e-01 | 0.000000e+00 | 0.000000e+0 |
| **75%** | 7.000000e+00 | 3.000000e+00 | 2.683398e+02 | 1.000000e+00 | 1.030610e+01 | 8.753340e+00 | 9.282560e+00 | 7.437780e+00 | 6.176630e+0 |
| **max** | 1.300000e+01 | 3.000000e+00 | 3.603711e+02 | 1.000000e+00 | 1.972240e+03 | 2.048790e+03 | 2.145710e+03 | 1.731880e+03 | 9.009370e+0 |

8 rows × 28 columns

descibe() function gives us the count, mean, standard deviation, min, max and percentile values of each column in the dataset. we can see median (50 percentile) of most of the eeg features is nearly zero. From min, max and mean values somewhat it looks like gaussian distribution. We'll check the distributions with plotting.

In [21]:

```
# skewness checking

for i in train:
    print('skewness of',i,'is',train[i].skew())
```

```
skewness of crew is 0.8401397950267352
skewness of experiment is 0.43647528637257704
skewness of time is 0.023713753
skewness of seat is 0.0001877791721112034
skewness of eeg_fp1 is 4.4300733
skewness of eeg_f7 is 6.627159
skewness of eeg_f8 is 5.8071766
skewness of eeg_t4 is 0.2159868
skewness of eeg_t6 is 0.15133826
skewness of eeg_t5 is 2.92606
skewness of eeg_t3 is 4.6416698
skewness of eeg_fp2 is 5.424461
skewness of eeg_o1 is -8.095578
skewness of eeg_p3 is -0.73223907
skewness of eeg_pz is 1.9695033
skewness of eeg_f3 is 1.6009752
skewness of eeg_fz is -12.644516
skewness of eeg_f4 is -0.1869015
skewness of eeg_c4 is 0.3752349
skewness of eeg_p4 is -1.6026337
skewness of eeg_poz is 4.0142093
skewness of eeg_c3 is 0.06768693
skewness of eeg_cz is 0.5525345
skewness of eeg_o2 is 16.596567
skewness of ecg is 0.3957411
skewness of r is -0.5071985
skewness of gsr is 0.2555336
skewness of event is 0.5529129070212143
```

Highly skewed means skewness either less than -1 or more than 1. Moderately skewed means skewness either in between -1 and -0.5 or in between 1 and 0.5. Approximately symmetric means skewness in between 0.5 and -0.5.

1. Highly skewed features: eeg_fp1, eeg_f7, eeg_f8, eeg_t5, eeg_t3, eeg_fp2, eeg_o1, eeg_pz, eeg_f3, eeg_fz, eeg_p4, eeg_poz, eeg_o2
2. moderately skewed features: crew, eeg_p3, eeg_cz, r, event
3. Approximately symmetric features: experiment, time, seat, eeg_t4, eeg_t6, eeg_f4, eeg_c4, eeg_c3, ecg, gsr

In [8]:

```
# checking missing values

train.isnull().sum()
```

Out[8]:

```
crew          0
experiment    0
time          0
seat          0
eeg_fp1       0
eeg_f7        0
eeg_f8        0
eeg_t4        0
eeg_t6        0
eeg_t5        0
eeg_t3        0
eeg_fp2       0
eeg_o1        0
eeg_p3        0
eeg_pz        0
eeg_f3        0
eeg_fz        0
eeg_f4        0
eeg_c4        0
eeg_p4        0
```

```
eeg_poz        0
eeg_c3         0
eeg_cz         0
eeg_o2         0
ecg            0
r              0
gsr            0
event          0
dtype: int64
```

No missing values

In [9]:

```
train['event'].value_counts()
```

Out[9]:

```
0    2848809
2    1652686
3     235329
1     130597
Name: event, dtype: int64
```

as we can see from the count, our dataset is imbalanced dataset. Count of event 'SS' is the least and count of event 'A' = baseline is the max. We have to make the dataset more balanced before applying any model. Performance of models may vary because of data imbalance.

In [5]:

```
# https://www.kaggle.com/theoviel/starter-code-eda-and-lgbm-baseline
```

Event has lablled in following manner:

A = baseline = 0

B = SS = 1

C = CA = 2

D = DA = 3

Experiment has following values:

'CA':0

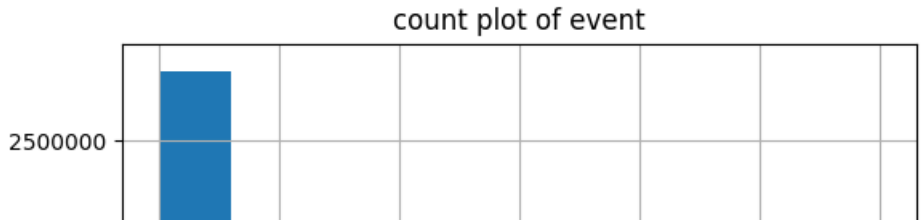
'DA':1

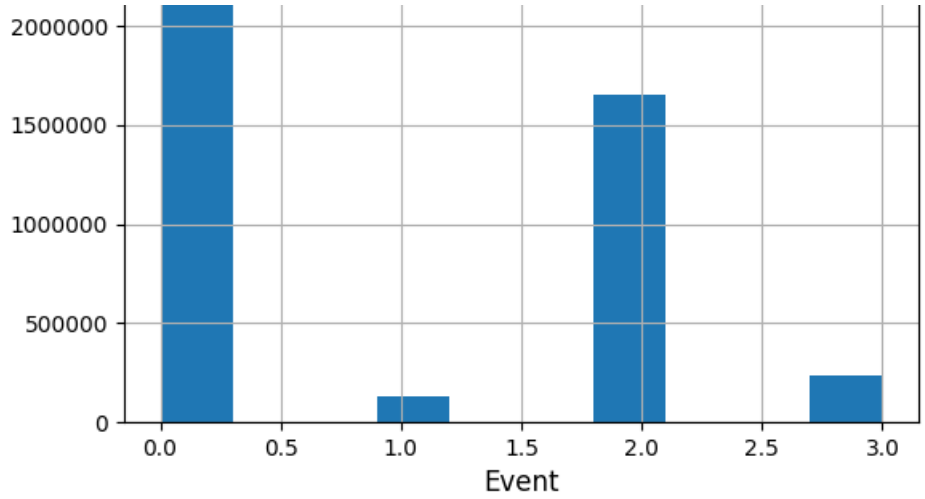
'SS':3

'LOFT':4

In [33]:

```
train['event'].hist()
plt.xlabel("Event", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.title('count plot of event')
plt.show()
```

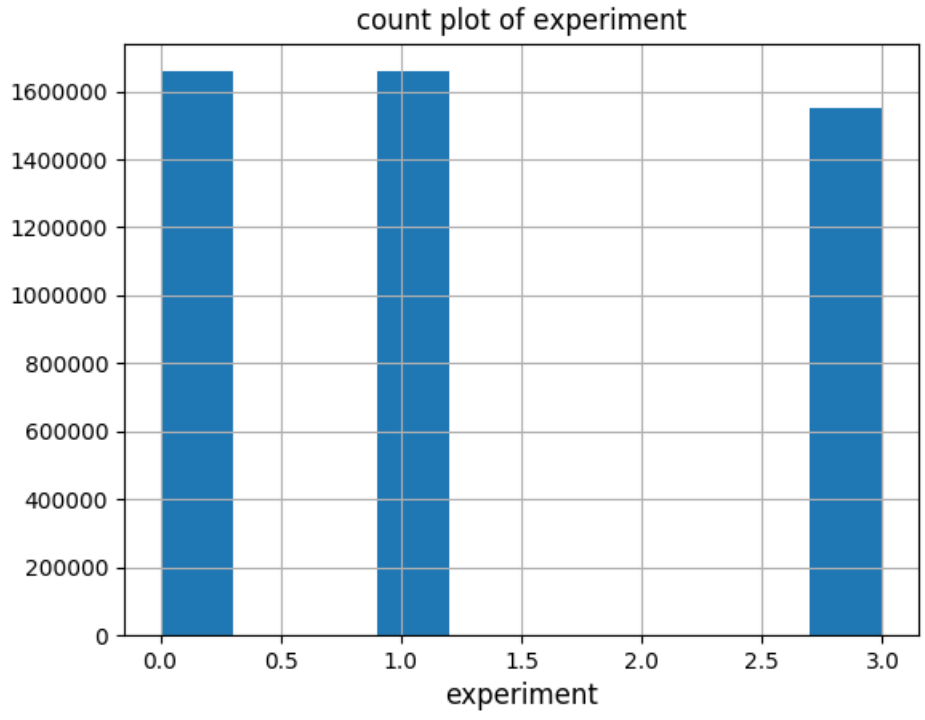





We have imbalanced dataset, as we can see for all 4 events count is different. Count of event 'SS' is the least and count of event 'A' = baseline is the max. We have to make the dataset more balanced before applying any model. Performance of models may vary because of data imbalance.

In [34]:

```
train['experiment'].hist()
plt.xlabel("experiment", fontsize=12)
plt.ylabel("Count", fontsize=12)
plt.title('count plot of experiment')
plt.show()
```




Experiment 3 (SS) happened least number of times. Count for experiment 0 and experiment 1 is almost same, means experiment CA and DA conducted equal number of times.

Experiment 4 in the test set is LOFT (Line Oriented Flight Training), which is a full flight (take off, flight, and landing) in a flight simulator.

In [61]:

```
# creating a pilot column for train dataset by taking crew and seat columns

pilot=[]
```

```
for i in range(len(train)):
    pilot.append(train['crew'][i]*10+train['seat'][i])

print(set(pilot))
len(pilot)
```

```
{130, 131, 70, 71, 40, 41, 10, 11, 80, 81, 50, 51, 20, 21, 60, 61, 30, 31}
```

Out[61]:

```
4867421
```

In [66]:

```
pil=pd.DataFrame(pilot,columns=['pilot'])

pil
```

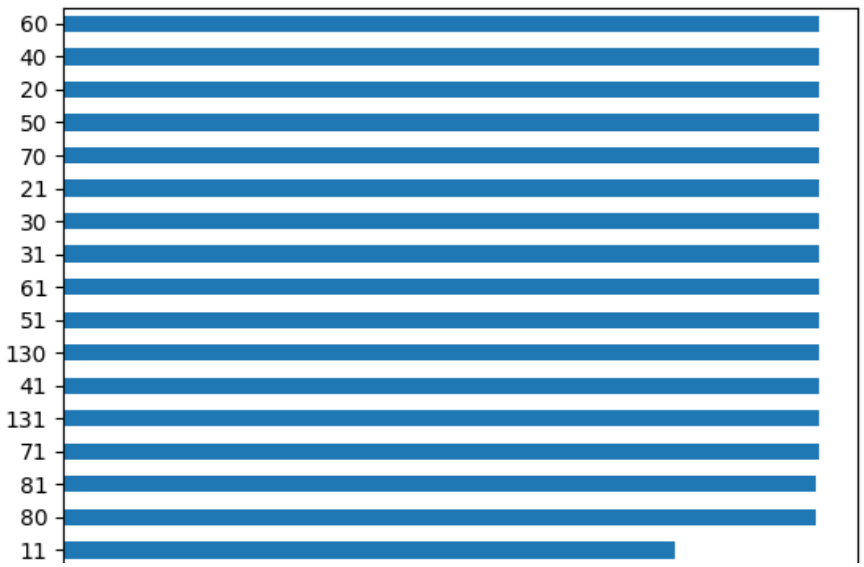
Out[66]:

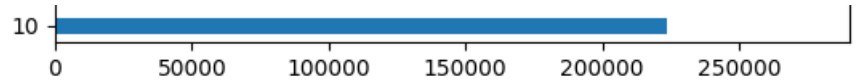
| | pilot |
|---|---|
| **0** | 11 |
| **1** | 11 |
| **2** | 11 |
| **3** | 11 |
| **4** | 11 |
| **...** | ... |
| **4867416** | 131 |
| **4867417** | 130 |
| **4867418** | 131 |
| **4867419** | 130 |
| **4867420** | 131 |

4867421 rows × 1 columns

In [15]:

```
# plotting the count plot for pilot column

pil['pilot'].value_counts().sort_values().plot(kind = 'barh')
plt.show()
```

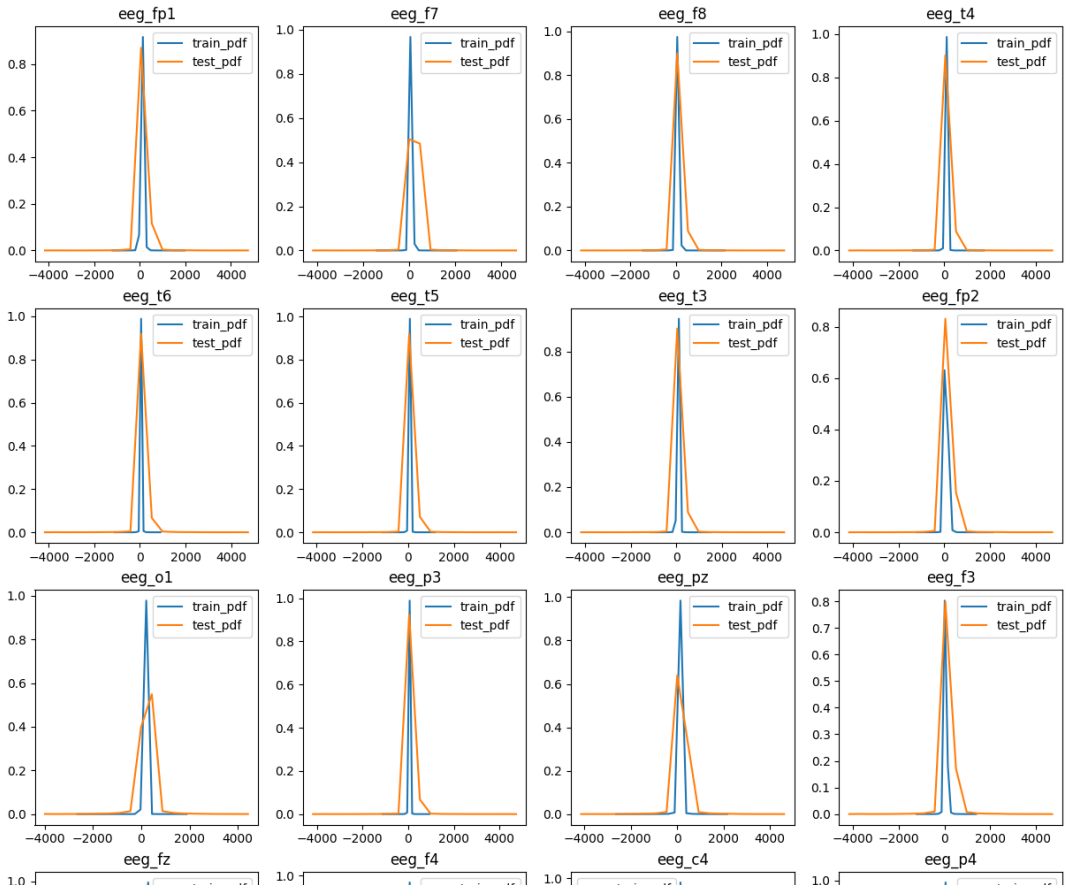


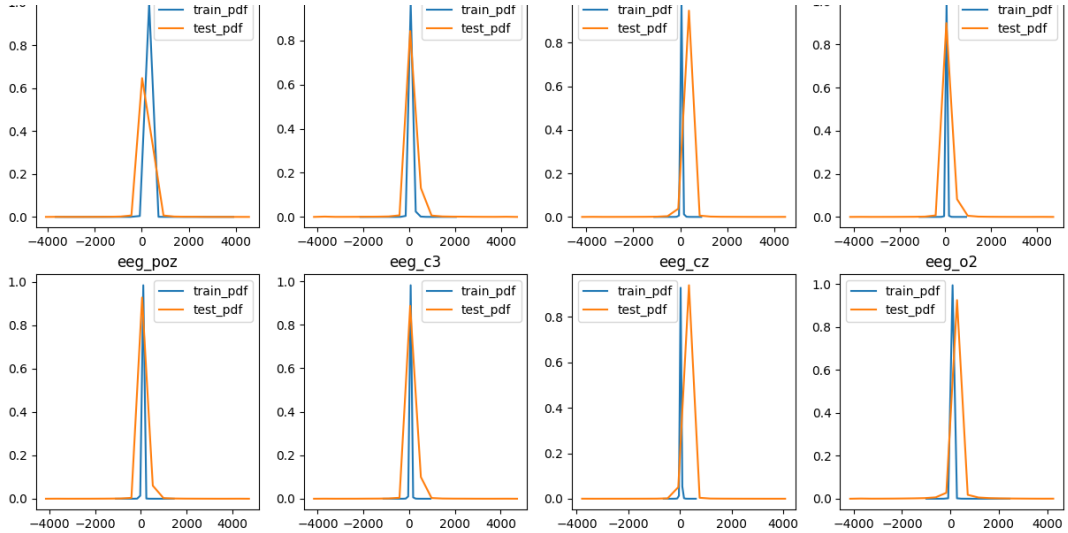



Every pilot has a different set of sensor reading values with multiple experiments performed as input, corresponding output is the event. From this plot we can see, Pilot 10 and 11 have min number of counts. Means experiment performed less with pilot 10 and 11 comparison to others. Other pilots have nearly the same counts.

In [32]:

```
eeg_features = ["eeg_fp1", "eeg_f7", "eeg_f8", "eeg_t4", "eeg_t6", "eeg_t5", "eeg_t3", "eeg_fp2", "
eeg_o1", "eeg_p3", "eeg_pz", "eeg_f3", "eeg_fz", "eeg_f4", "eeg_c4", "eeg_p4", "eeg_poz", "eeg_c3",
"eeg_cz", "eeg_o2"]

plt.figure(figsize=(15,20))
i = 0

for egg in eeg_features:
    i += 1
    plt.subplot(5, 4, i)

    counts, bin_edges = np.histogram(train[egg], bins=20, density = True)
    pdf = counts/(sum(counts))
    plt.plot(bin_edges[1:],pdf)

    counts_1, bin_edges_1 = np.histogram(test[egg], bins=20, density = True)
    pdf_1 = counts_1/(sum(counts_1))
    plt.plot(bin_edges_1[1:],pdf_1)

    plt.title(egg)
    plt.legend(['train_pdf', 'test_pdf'])
plt.show()
```

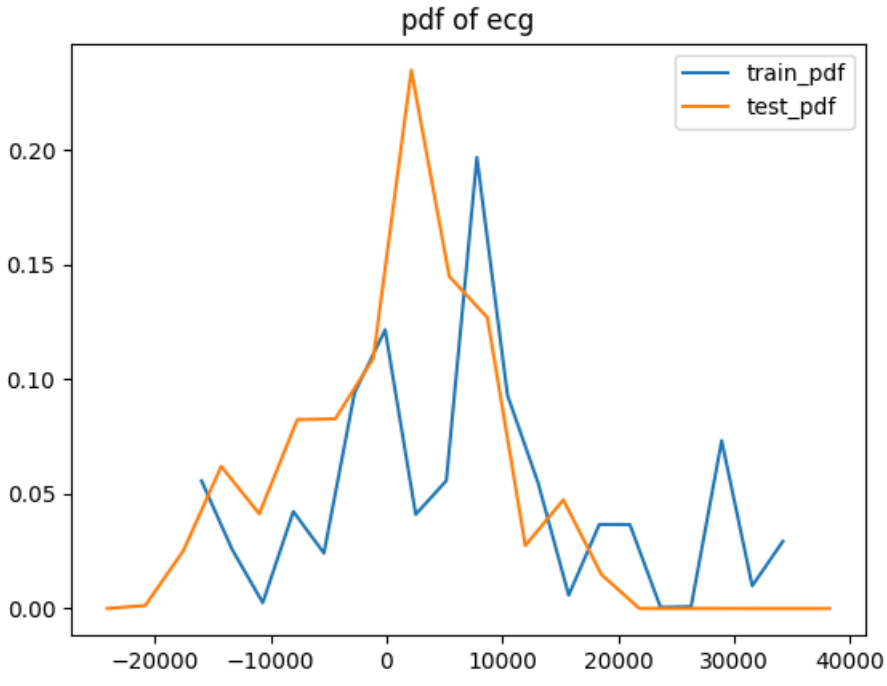





We can observe that data is looking like normally distributed. Distributions of both train and test look like centered at 0. Variance is more on the test dataset, as we see spread is more for all of the test dataset features' pdf than the train dataset features' pdf.

In [38]:

```
counts, bin_edges = np.histogram(train['ecg'], bins=20, density = True)
pdf = counts/(sum(counts))
plt.plot(bin_edges[1:],pdf)

counts_1, bin_edges_1 = np.histogram(test['ecg'], bins=20, density = True)
pdf_1 = counts_1/(sum(counts_1))
plt.plot(bin_edges_1[1:],pdf_1)

plt.title('pdf of ecg')
plt.legend(['train_pdf', 'test_pdf'])
plt.show()
```



We can see that both train and test dist plots looks similar, means having similar kind of distributions. But have too much overlapping between two. Also, from 18000 micro volt only train samples are available. So range of ecg is smaller for test dataset.

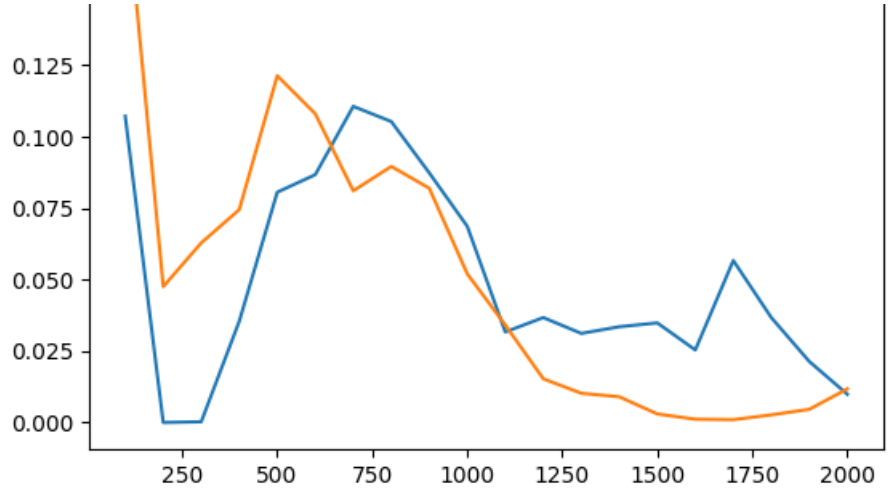
In [39]:

```
counts, bin_edges = np.histogram(train['r'], bins=20, density = True)
pdf = counts/(sum(counts))
plt.plot(bin_edges[1:],pdf)

counts_1, bin_edges_1 = np.histogram(test['r'], bins=20, density = True)
pdf_1 = counts_1/(sum(counts_1))
plt.plot(bin_edges_1[1:],pdf_1)

plt.title('pdf of r')
plt.legend(['train_pdf', 'test_pdf'])
plt.show()
```


We can see that both train and test dist plots looks kind of similar, probably having similar kind of distributions. But have too much overlapping between two. Also, samples are available for lower range of r for test dataset. So range of r is larger for test dataset.

In [40]:

```
counts, bin_edges = np.histogram(train['gsr'], bins=20, density = True)
pdf = counts/(sum(counts))
plt.plot(bin_edges[1:],pdf)

counts_1, bin_edges_1 = np.histogram(test['gsr'], bins=20, density = True)
pdf_1 = counts_1/(sum(counts_1))
plt.plot(bin_edges_1[1:],pdf_1)

plt.title('pdf of gsr')
plt.legend(['train_pdf', 'test_pdf'])
plt.show()
```

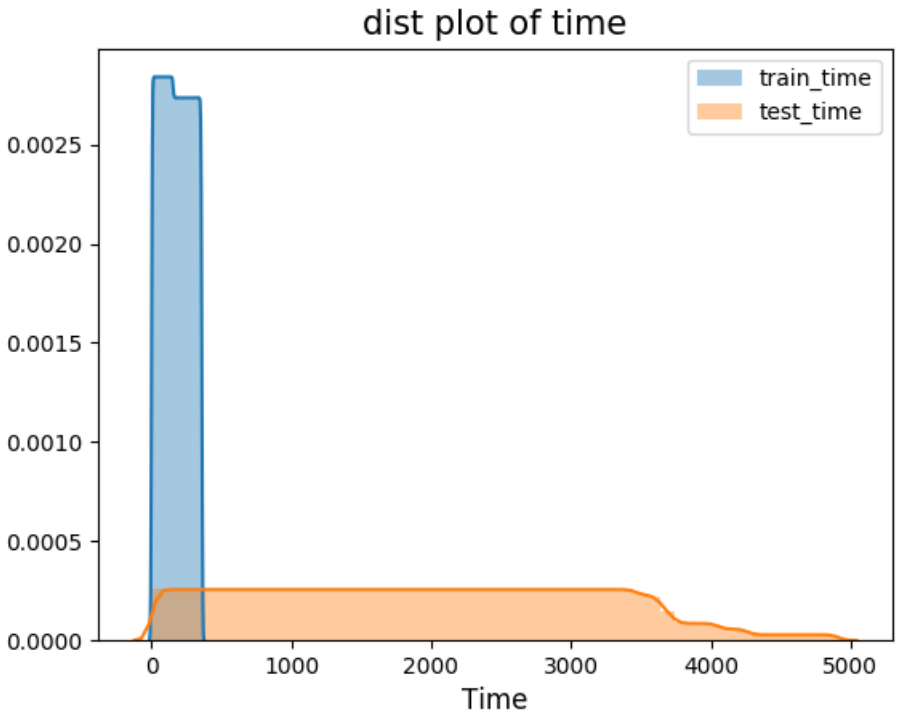



The galvanic skin response (GSR, which falls under the umbrella term of electrodermal activity, or EDA) refers to changes in sweat gland activity that are reflective of the intensity of our emotional state, otherwise known as emotional arousal.

Both train and test dist are very similar, may have similar distributions. Range is also same for train and test dataset.

In [41]:

```
sns.distplot(train['time'])
sns.distplot(test['time'])

plt.legend(['train_time', 'test_time'])
plt.xlabel("Time")
plt.title("dist plot of time")
plt.show()
```




We can see that, the test time range is much larger than of the train. Means, experiment done on train dataset is in very less time. Though we can't use time feature as flight simulator time has nothing to do with the experiment time.

## Bivariate analysis

SEAT

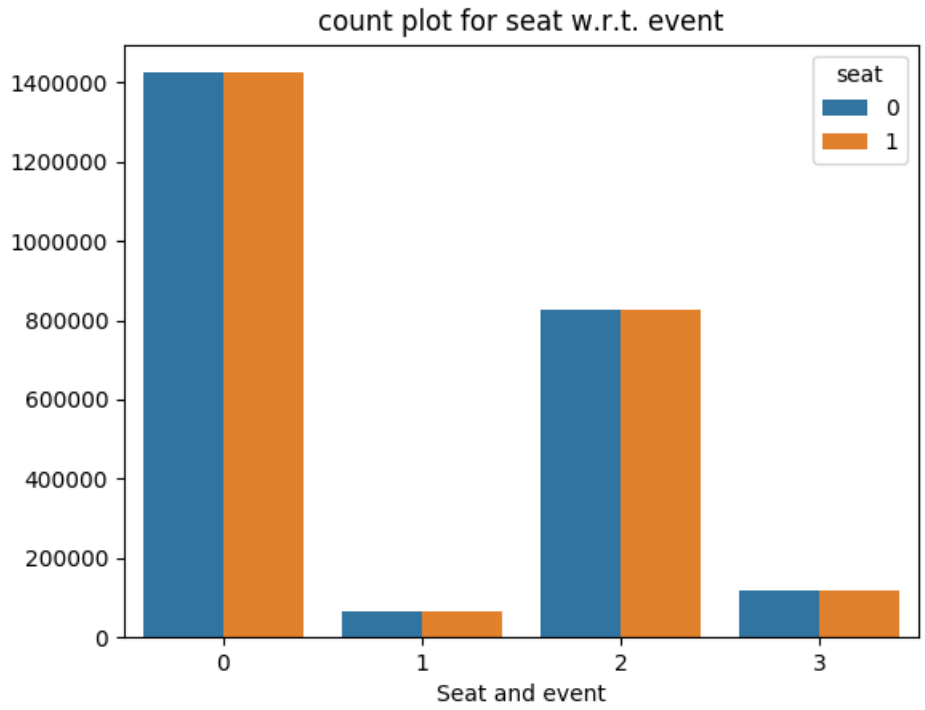
SEAT

0 : left seat

1 : right seat

In [51]:

```
sns.countplot('event', hue='seat', data=train)
plt.xlabel("Seat and event")
plt.ylabel("Count")

plt.title("count plot for seat w.r.t. event")
plt.show()
```




Seat column may not have any impact on the 'event' outcome, as we can see from the plot, count of seat 0 and seat 1 is the same for each event. We know event 0 occurred max number of times, but we can not tell it's because pilot sitting in seat 0 or 1 because both counts are equal.We will see if this feature is important or not when we see the feature importance graph later.

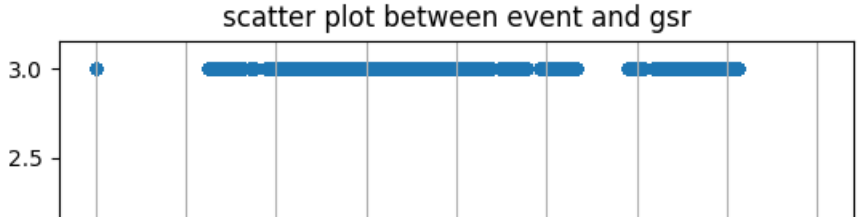
In [ ]:

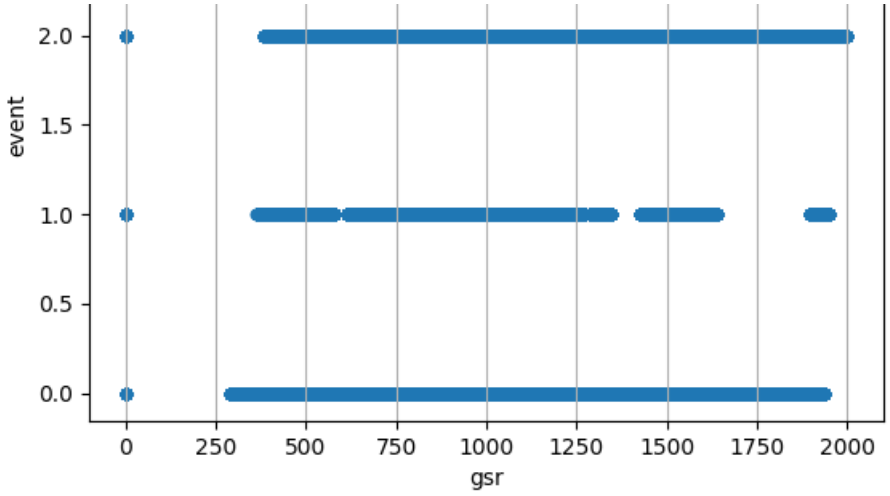
```

```

The simplest bivariate plot is the scatter plot. we need to understand how variables interact with one another. Also scatter plot tells us about the correlation. So we will plot the scatter plots and try to observe it.

In [24]:

```
train.plot(x='gsr', y='event', kind='scatter')
plt.title("scatter plot between event and gsr")
plt.show()
```

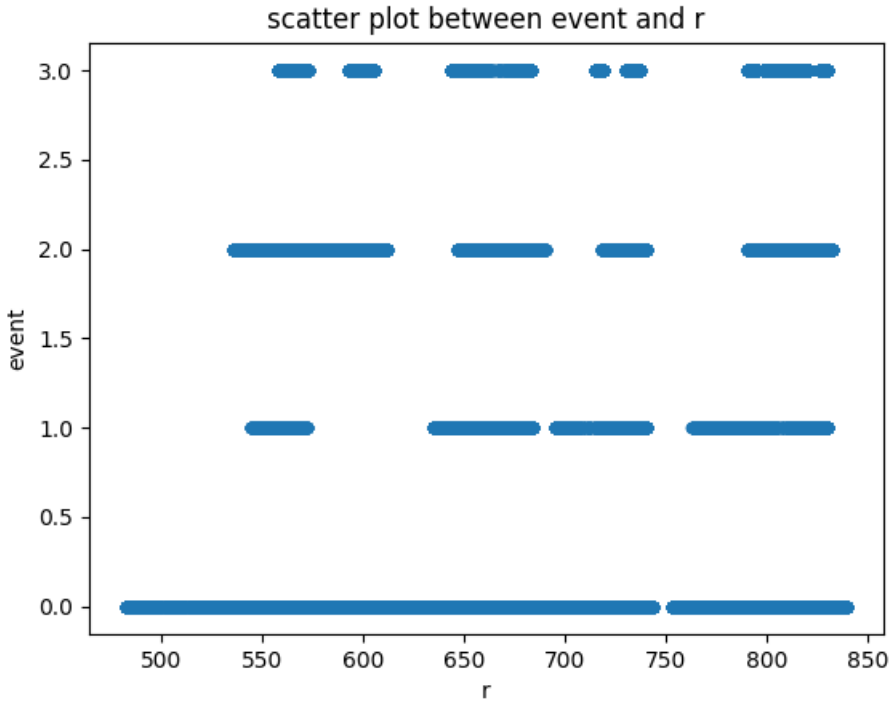





We can see that the range of gsr readings for all events is nearly same. Too much overlapping, so we can't conclude anything from this plot. Some gsr readings is zero also, it may be the outlier or wrong data. We will see such cases in feature engineering section.

In [25]:

```
train.plot(x='r', y='event', kind='scatter')
plt.title("scatter plot between event and r")

plt.show()
```

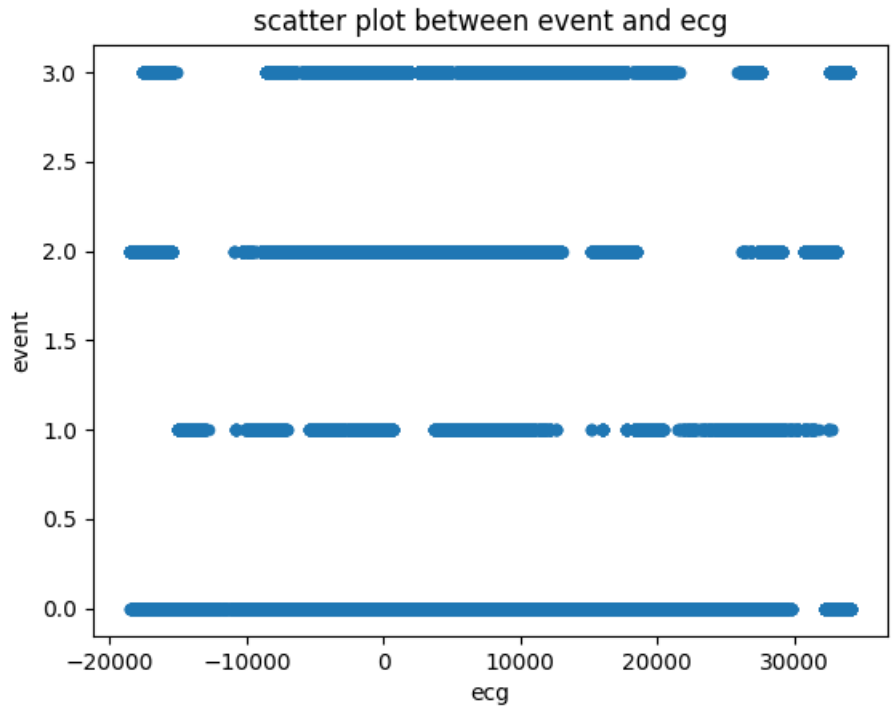



We can see that the range of r readings for event 0 is the max. If reading is near to 500-525, we can say there are more chances of event 0 occurs.

For other events range is nearly same. Too much overlapping in the range for all events.

In [26]:

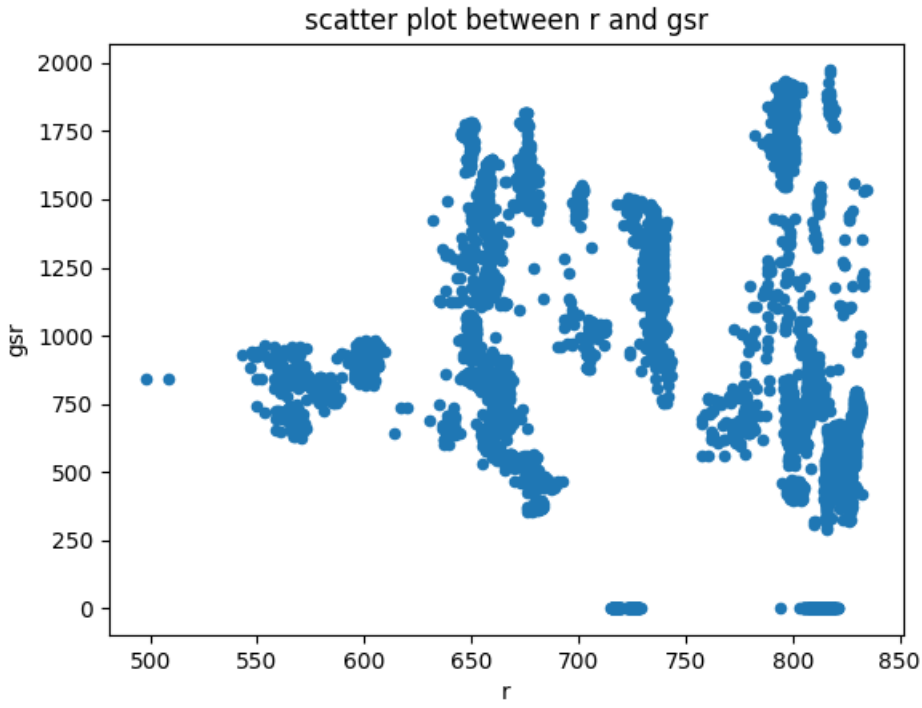
```
train.plot(x='ecg', y='event', kind='scatter')
plt.title("scatter plot between event and ecg")

plt.show()
```




There is not much to conclude with this scatter plot. We can see that the range of gsr readings for all events is nearly same. Too much overlapping, so we can't conclude anything from this plot. Some ecg readings is zero also.

In [27]:

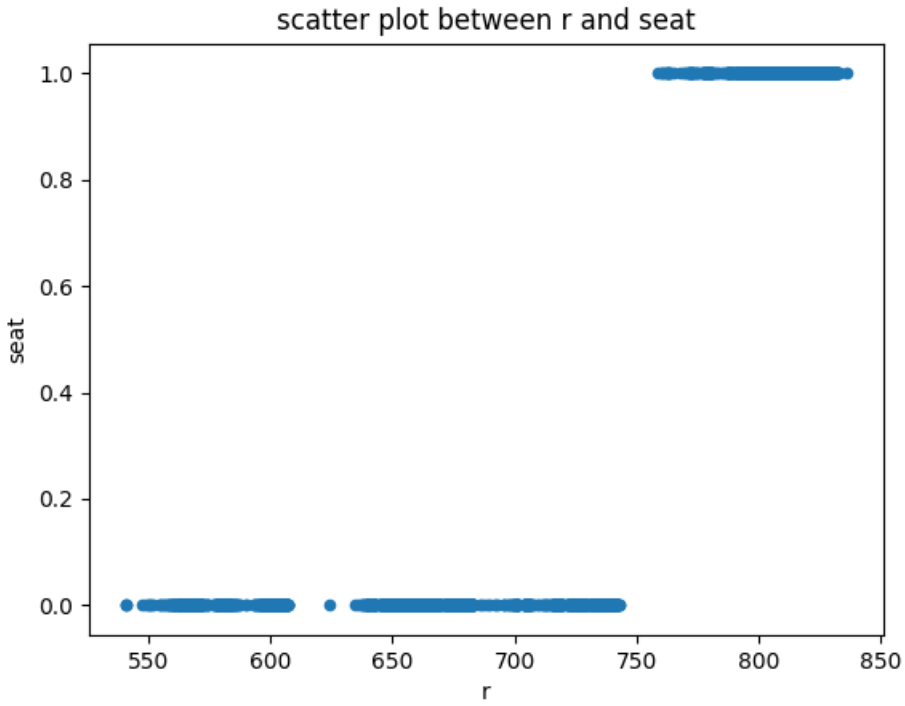
```
train.sample(5000).plot(x='r', y='gsr', kind='scatter')
plt.title("scatter plot between r and gsr")

plt.show()
```




It's looking like gsr and r are weakly correlated, but we can not say whether it's positive or negative correlation. As values of r increasing, values of gsr increasing in both positive and negative direction. We have some 0 readings on gsr scale as well. We will see correlation later in this section.

In [28]:

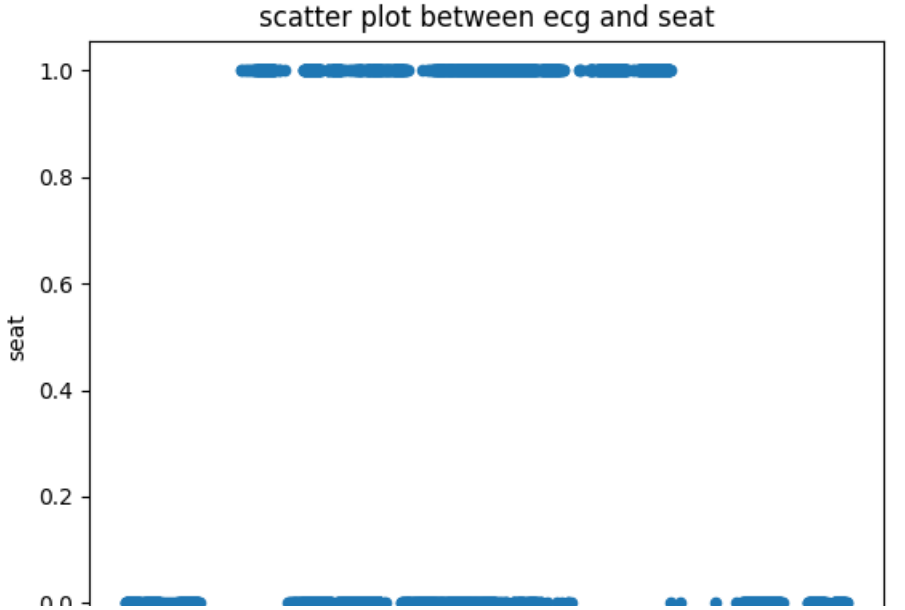
```
train.sample(5000).plot(x='r', y='seat', kind='scatter')

plt.title("scatter plot between r and seat")

plt.show()
```

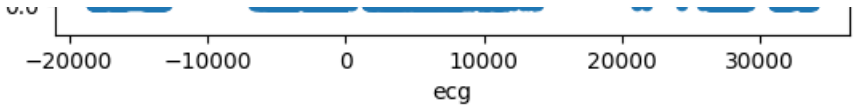



Here we can see, for seat 0 we have values lower than 750 on r-scale. And for seat 1, all values are more than 750 on r-scale. It means 'seat' and 'r' are highly correlated. And it's positive correlation, as value of seat increasing, values of r also increasing. We will see correlation (VIF) later in this section.

In [29]:

```
train.sample(5000).plot(x='ecg', y='seat', kind='scatter')
plt.title("scatter plot between ecg and seat")

plt.show()
```

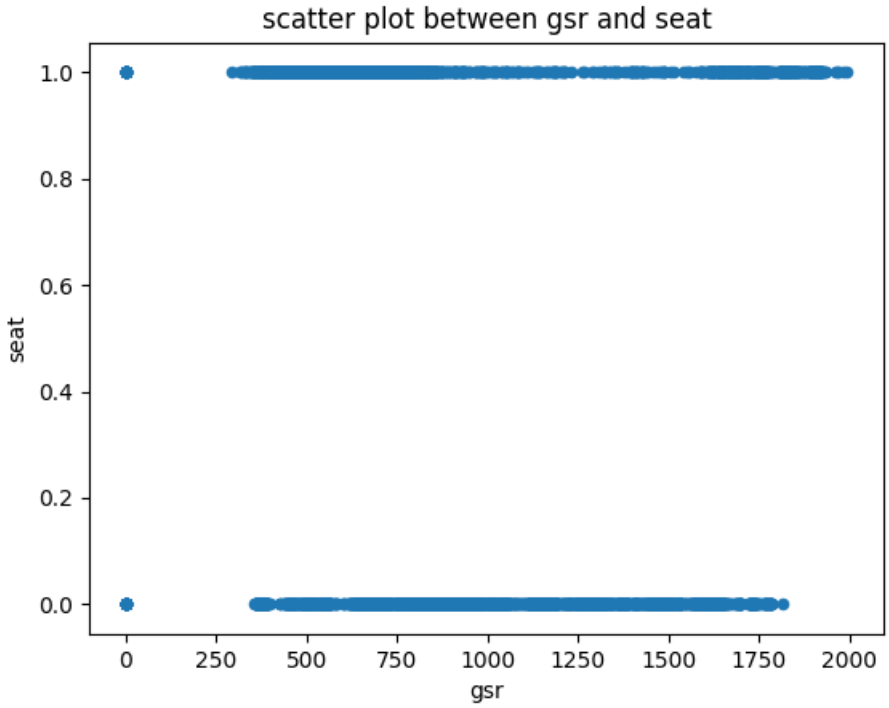





There is too much overlapping between range of ecg for seat 0 and seat 1. For seat 0 we have higher range, means pilots who were sitting in seat 0 has higher ecg means higher blood pressure.

In [30]:

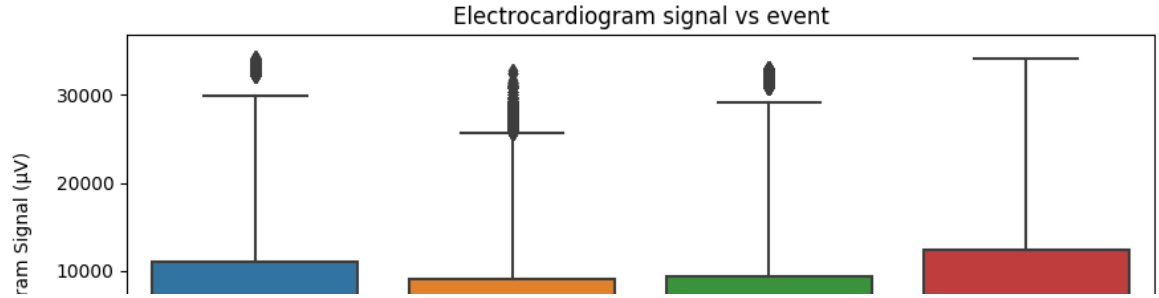
```
train.sample(5000).plot(x='gsr', y='seat', kind='scatter')
plt.title("scatter plot between gsr and seat")

plt.show()
```

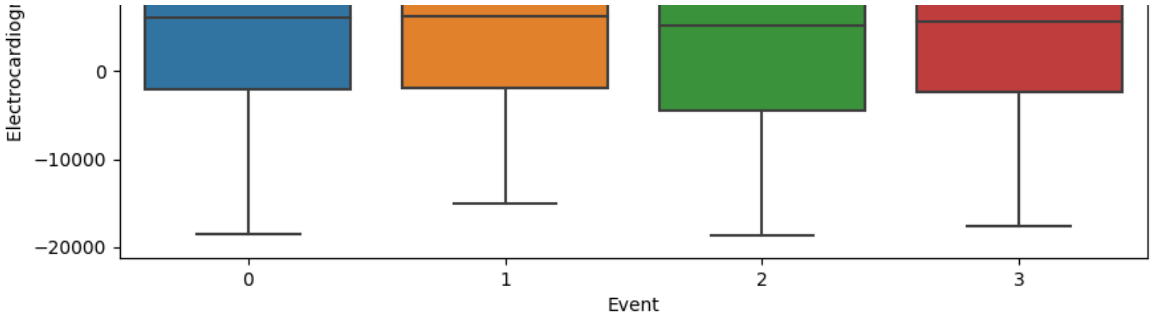



We can see that the range of gsr readings for seat 0 and seat 1 is nearly same. Too much overlapping, so we can't conclude anything from this plot. Some gsr readings is zero also, it may be the outlier or wrong data. We will see such cases in feature engineering section.

In [53]:

```
plt.figure(figsize=(10,5))

sns.boxplot(x='event', y='ecg', data=train)
plt.ylabel("Electrocardiogram Signal (µV)")
plt.xlabel("Event")
plt.title("Electrocardiogram signal vs event")
plt.show()
```

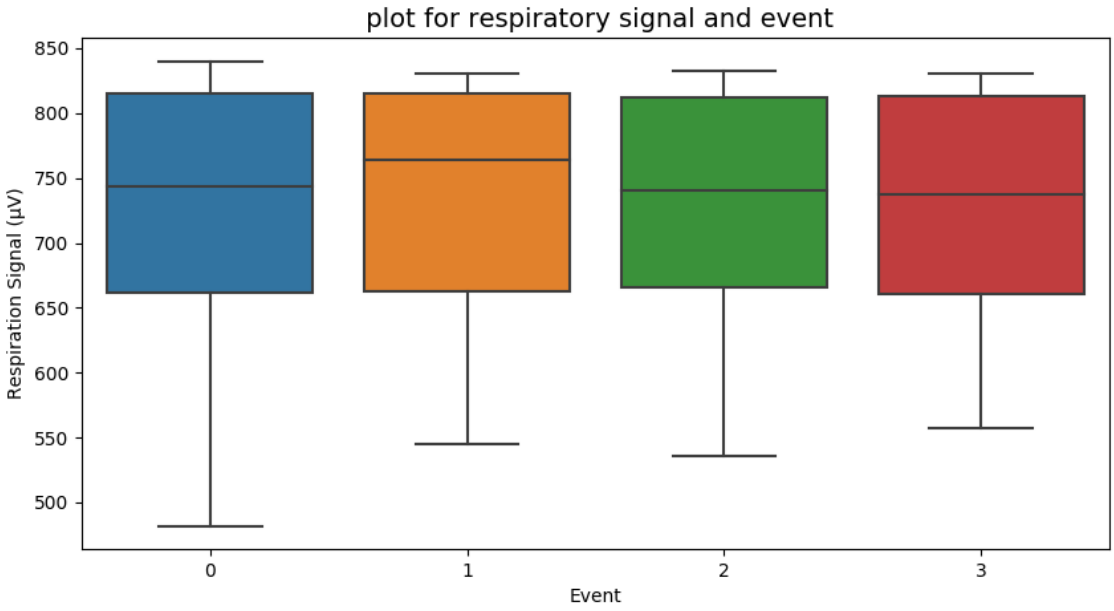





We can observe that, range is least for event 1 and max for event 3. Too much overlapping. We can't differentiate much and not much difference in mean for all the events also.

In [57]:

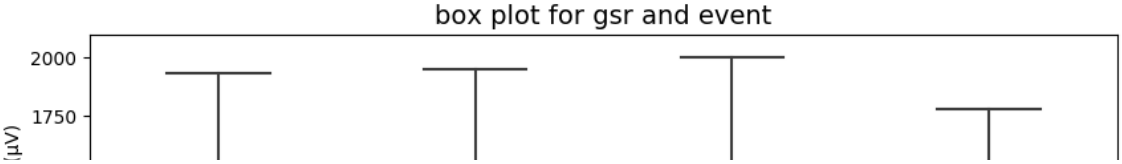
```
plt.figure(figsize=(10,5))
sns.boxplot(x='event', y='r', data=train)
plt.ylabel("Respiration Signal (µV)")
plt.xlabel("Event")
plt.title(" plot for respiratory signal and event", fontsize=14)
plt.show()
```

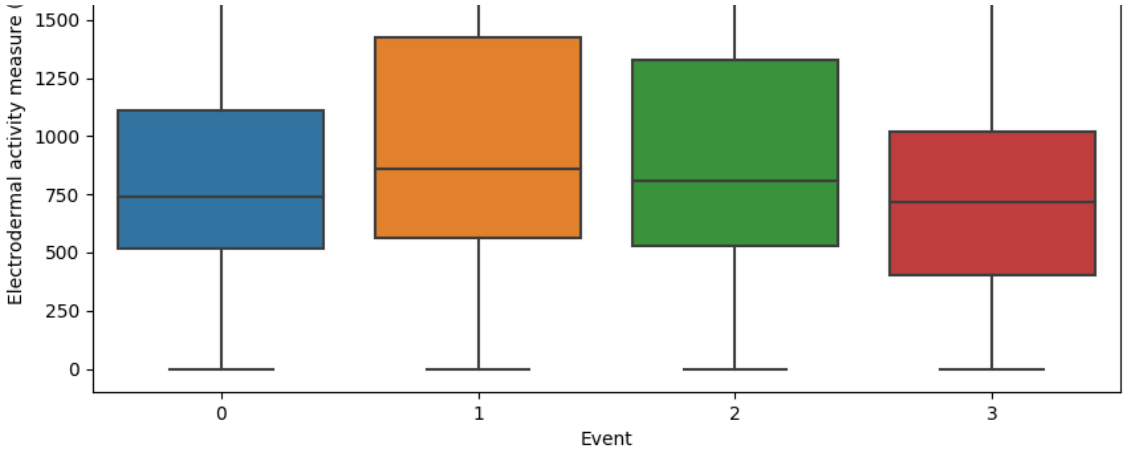



We can see that not much difference in box plots of all events. Mean is also nealry same for all the events. Event 0, 2 and 3 have nearly the same mean. Event 0 having maximum range. We can say that if r signal values lies between 450-550, event 0 (no event) may occur.

In [58]:

```
plt.figure(figsize=(10,5))
sns.boxplot(x='event', y='gsr', data=train)
plt.ylabel("Electrodermal activity measure (µV)")
plt.xlabel("Event")
plt.title("box plot for gsr and event", fontsize=14)
plt.show()
```






We can see that not much difference in box plots of all events. Range is almost same for all events and mean too. 0-1750 (μV) range is common to all the events. We have zero values in gsr readings for all events.

In [ ]:

```
```

# Feature Engineering

1. Each and every row is the collection of sensor readings of the experiment conducted on each pilot. We have not given pilot data directly. But we can find from crew and seat column.
2. There are 9 unique crews. In each crew there are two pilots, one on the left seat and one on the right seat. So total 18 pilots.
3. The train data is collected from experiments conducted on pilots in different situations. The test data is collected while a flight simulation.
4. Therefore we will not use the 'experiment', 'crew', 'seat' and 'time' features in the model, as they will not be useful for the prediction.
5. The package biosspy returns timestamped values, so we will use cubic interpolation to give values for timestamps in between.

In [16]:

```
# https://www.kaggle.com/shahaffind/reducing-commercial-aviation-fatalities-11th

from scipy.interpolate import interp1d

def interpolated_value(t, feature_ts, feature_data):
    ''' to predict values that fall within two existing data points using interpolation '''
    new = interp1d(feature_ts, feature_data, kind='cubic', fill_value="extrapolate")
    return new(t)
```

In [17]:

```
from biosppy.signals import ecg
from biosppy.signals import resp
from biosppy.signals import eda
from biosppy.signals import eeg
```

Some sensors have reading 0, i.e., some pilots have missing sensors. we will set those reading to nan, so those will be ignored during learning or predicting.

In [18]:

```
eeg_features = ["eeg_fp1", "eeg_f7", "eeg_f8", "eeg_t4", "eeg_t6", "eeg_t5", "eeg_t3", "eeg_fp2", "
eeg_o1", "eeg_p3", "eeg_pz", "eeg_f3", "eeg_fz", "eeg_f4", "eeg_c4", "eeg_p4", "eeg_poz", "eeg_c3",
"eeg_cz", "eeg_o2"]

def pilot_features(data, loca):
    ''' to add new features using biosppy.signals functions'''

    temp_df = data.loc[loca][['time', 'ecg', 'r', 'gsr'] + eeg_features].values
```

```
    temp_df = temp_df[temp_df[:,0].argsort()]

    #if any value is zero for sensors, then replace it with nan. Else create new features like 'he
art_rate', 'resp_rate', 'gsr_amp'

    if np.allclose(temp_df[:,1], 0, rtol=1e-10):
        data.loc[loca, 'ecg'] = np.nan
        print('missing egc')
    else:
        try:
            heart_sig = ecg.ecg(signal=temp_df[:,1], sampling_rate=256., show=False)
            heart_rate = heart_sig['heart_rate']
            heart_rate_ts = heart_sig['heart_rate_ts']
            data.loc[loca, 'heart_rate'] = interpolated_value(temp_df[:,0], heart_rate_ts, heart_ra
te)
        except ValueError:
            print('failed "heart_rate" extraction')

    if np.allclose(temp_df[:,2], 0, rtol=1e-10):
        data.loc[loca, 'r'] = np.nan
        print('missing r')
    else:
        try:
            resp_sig = resp.resp(signal=temp_df[:,2], sampling_rate=256., show=False)
            resp_rate = resp_sig['resp_rate']
            resp_rate_ts = resp_sig['resp_rate_ts']
            data.loc[loca, 'resp_rate'] = interpolated_value(temp_df[:,0], resp_rate_ts, resp_rate)
        except ValueError:
            print('failed "resp_rate" extraction')

    if np.allclose(temp_df[:,3], 0, rtol=1e-10):
        data.loc[loca, 'gsr'] = np.nan
        print('missing gsr')
    else:
        try:
            gsr_sig = eda.eda(signal=temp_df[:,3], sampling_rate=256., show=False)
            gsr_amp = gsr_sig['amplitudes']
            gsr_amp_ts = temp_df[gsr_sig['onsets'], 0]
            data.loc[loca, 'gsr_amp'] = interpolated_value(temp_df[:,0], gsr_amp_ts, gsr_amp)
        except IndexError:
            print('failed "gsr_amp" extraction')
        except ValueError:
            print('failed "gsr_amp" extraction')

    # creating 5 more features with 'get_power_features' function which returns 6 values, using in
terpolation on top of that.
    try:
        eeg_feat_sig = eeg.get_power_features(signal=temp_df[:,4:], sampling_rate=256.)
        eeg_ts = eeg_feat_sig['ts']
        eeg_theta = eeg_feat_sig['theta']
        eeg_alpha_low = eeg_feat_sig['alpha_low']
        eeg_alpha_high = eeg_feat_sig['alpha_high']
        eeg_beta = eeg_feat_sig['beta']
        eeg_gamma = eeg_feat_sig['gamma']
        for i, e in enumerate(eeg_features):
            data.loc[loca, e + '_theta'] = interpolated_value(temp_df[:,0], eeg_ts, eeg_theta[:,i])
            data.loc[loca, e + '_alpha_low'] = interpolated_value(temp_df[:,0], eeg_ts,
eeg_alpha_low[:,i])
            data.loc[loca, e + '_alpha_high'] = interpolated_value(temp_df[:,0], eeg_ts,
eeg_alpha_high[:,i])
            data.loc[loca, e + '_beta'] = interpolated_value(temp_df[:,0], eeg_ts, eeg_beta[:,i])
            data.loc[loca, e + '_gamma'] = interpolated_value(temp_df[:,0], eeg_ts, eeg_gamma[:,i])
    except ValueError:
        print('failed "eeg"')
```

In [13]:

```
new_train_df = train.copy()
new_train_df['heart_rate'] = np.nan
new_train_df['resp_rate'] = np.nan
new_train_df['gsr_amp'] = np.nan

for e in eeg_features:
    new_train_df[e + '_theta'] = np.nan
    new_train_df[e + '_alpha_low'] = np.nan
    new_train_df[e + '_alpha_high'] = np.nan
```

```
    new_train_df[e + '_beta'] = np.nan
    new_train_df[e + '_gamma'] = np.nan

for tupl in [(c,s,e) for c in np.unique(new_train_df['crew']) for s in np.unique(new_train_df['seat
']) for e in np.unique(new_train_df['experiment'])]:
    c, s, e = tupl
    loca = (new_train_df['crew'] == c) & (new_train_df['seat'] == s) & (new_train_df['experiment']
== e)
    print('extracting for : ', tupl)
    pilot_features(new_train_df, loca)


print(new_train_df.shape)
```

```
extracting for :  (1, 0, 0)
extracting for :  (1, 0, 1)
missing egc
extracting for :  (1, 0, 3)
failed "gsr_amp" extraction
extracting for :  (1, 1, 0)
extracting for :  (1, 1, 1)
extracting for :  (1, 1, 3)
extracting for :  (2, 0, 0)
extracting for :  (2, 0, 1)
extracting for :  (2, 0, 3)
extracting for :  (2, 1, 0)
extracting for :  (2, 1, 1)
missing gsr
extracting for :  (2, 1, 3)
failed "gsr_amp" extraction
extracting for :  (3, 0, 0)
extracting for :  (3, 0, 1)
extracting for :  (3, 0, 3)
extracting for :  (3, 1, 0)
extracting for :  (3, 1, 1)
extracting for :  (3, 1, 3)
extracting for :  (4, 0, 0)
extracting for :  (4, 0, 1)
extracting for :  (4, 0, 3)
extracting for :  (4, 1, 0)
extracting for :  (4, 1, 1)
extracting for :  (4, 1, 3)
extracting for :  (5, 0, 0)
extracting for :  (5, 0, 1)
extracting for :  (5, 0, 3)
extracting for :  (5, 1, 0)
extracting for :  (5, 1, 1)
missing gsr
extracting for :  (5, 1, 3)
extracting for :  (6, 0, 0)
extracting for :  (6, 0, 1)
failed "heart_rate" extraction
extracting for :  (6, 0, 3)
extracting for :  (6, 1, 0)
extracting for :  (6, 1, 1)
extracting for :  (6, 1, 3)
extracting for :  (7, 0, 0)
missing gsr
extracting for :  (7, 0, 1)
missing gsr
extracting for :  (7, 0, 3)
extracting for :  (7, 1, 0)
extracting for :  (7, 1, 1)
extracting for :  (7, 1, 3)
extracting for :  (8, 0, 0)
extracting for :  (8, 0, 1)
extracting for :  (8, 0, 3)
extracting for :  (8, 1, 0)
extracting for :  (8, 1, 1)
extracting for :  (8, 1, 3)
extracting for :  (13, 0, 0)
extracting for :  (13, 0, 1)
extracting for :  (13, 0, 3)
extracting for :  (13, 1, 0)
extracting for :  (13, 1, 1)
extracting for :  (13, 1, 3)
```

(486721, 131)

In [16]:

```
new_train=new_train_df.drop('experiment',axis=1)
new_train=new_train.drop('time',axis=1)

features=new_train.columns
features
```

Out[16]:

```
Index(['crew', 'seat', 'eeg_fp1', 'eeg_f7', 'eeg_f8', 'eeg_t4', 'eeg_t6',
       'eeg_t5', 'eeg_t3', 'eeg_fp2',
       ...
       'eeg_cz_theta', 'eeg_cz_alpha_low', 'eeg_cz_alpha_high', 'eeg_cz_beta',
       'eeg_cz_gamma', 'eeg_o2_theta', 'eeg_o2_alpha_low', 'eeg_o2_alpha_high',
       'eeg_o2_beta', 'eeg_o2_gamma'],
      dtype='object', length=129)
```

In [17]:

```
# saving new train dataset in csv format

new_train.to_csv('new_train.csv')
```

In [38]:

```

```

In [19]:

```
# creating different dataset for each pilot

# adding pilot features same as of train

def test_data_FE(i,list_tup):
    ''' Feature engineering for test dataset '''

    if i==0:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=10


    elif i==1:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=11


    elif i==2:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]
```

```
        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=20


    elif i==3:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=21


    elif i==4:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=30


    elif i==5:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=31


    elif i==6:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=40
```

```
    elif i==7:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=41


    elif i==8:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=50


    elif i==9:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=51


    elif i==10:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=60


    elif i==11:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
```

```
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=61


    elif i==12:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=70


    elif i==13:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=71


    elif i==14:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=80


    elif i==15:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=81


    elif i==16:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
```

```
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=130


    elif i==17:
        temp_df=test[(test['crew']==list_tup[i][0]) & (test['seat']==list_tup[i][1])]

        temp_df['heart_rate'] = np.nan
        temp_df['resp_rate'] = np.nan
        temp_df['gsr_amp'] = np.nan
        for e in eeg_features:
            temp_df[e + '_theta'] = np.nan
            temp_df[e + '_alpha_low'] = np.nan
            temp_df[e + '_alpha_high'] = np.nan
            temp_df[e + '_beta'] = np.nan
            temp_df[e + '_gamma'] = np.nan
        pilot_features(temp_df, temp_df.index.values)
        temp_df['pilot']=131


    test_id=temp_df['id']
    temp_df=temp_df.drop(['id','crew','experiment','time','seat'],axis=1)

    return temp_df,test_id
```

In [ ]:

```
```

In [3]:

```
def read_new_train_data():
    ''' reading the new created train dataset '''

    new_train=pd.read_csv('new_train.csv')
    new_train=memory_opt(new_train)
    new_train=new_train.drop('Unnamed: 0',axis=1)

    # dropping the rows with nan values as it will not going to help in model
    new_train=new_train.dropna()

    # resetting the index
    new_train.reset_index(inplace = True)

    new_train=new_train.drop('crew',axis=1)
    new_train=new_train.drop('seat',axis=1)
    new_train=new_train.drop('index',axis=1)

    return new_train
```

In [8]:

```
# creating pilot column for train dataset
new_train=read_new_train_data()
pilot=[]
for i in range(len(new_train)):
    pilot.append(new_train['crew'][i]*10+new_train['seat'][i])

print(set(pilot))
len(pilot)

pil=pd.DataFrame(pilot,columns=['pilot'])
pil.to_csv('train_pilot.csv')
```

```
{130, 131, 70, 71, 40, 41, 10, 11, 80, 81, 50, 51, 20, 21, 60, 61, 30, 31}
```

Out[8]:

```
4182820
```

In [4]:

```
# final train dataset

new_train=read_new_train_data()
pilot=pd.read_csv('train_pilot.csv')
new_train['pilot']=pilot['pilot']

new_train.head()
```

```
 Initial Memory of the dataframe is 4827.61
after optimization, Memory is: 2372.03
Reduced by 50.9%
```

Out[4]:

| | eeg_fp1 | eeg_f7 | eeg_f8 | eeg_t4 | eeg_t6 | eeg_t5 | eeg_t3 | eeg_fp2 | eeg_o1 | eeg_p3 | ... | eeg_cz_alpha_low | eeg |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -5.285 | 26.780001 | -9.5200 | -12.800000 | 16.719999 | 33.750000 | 23.719999 | -6.6950 | 29.230000 | 24.840000 | ... | 0.012816 | |
| **1** | -2.428 | 28.440001 | -9.3200 | -3.758000 | 15.970000 | 30.440001 | 21.020000 | -6.4770 | 26.639999 | 24.139999 | ... | 0.015376 | |
| **2** | 10.670 | 30.420000 | 15.3500 | 24.719999 | 16.139999 | 32.160000 | 25.440001 | -0.0887 | 28.120001 | 26.889999 | ... | 0.017788 | |
| **3** | 11.450 | 25.610001 | 2.4340 | 12.414000 | 20.530001 | 31.500000 | 19.139999 | -0.2566 | 30.660000 | 24.250000 | ... | 0.020054 | |
| **4** | 7.285 | 25.940001 | 0.1136 | 5.746000 | 19.830000 | 28.750000 | 20.580000 | -1.9530 | 31.719999 | 25.160000 | ... | 0.022180 | |

5 rows × 128 columns

## Correlation or multi-colinearity or feature selection

Here we are using 3 techniques:

1. Variance inflation factor
2. Permutation Importance
3. Recursive feature Importance

Then we can take features from any of the three techniques.

### Variance inflation factor (VIF)

In [45]:

```
# taking sample from train dataset

df=new_train.sample(60000).copy()
df=df.drop('event',axis=1)
df.head()
```

Out[45]:

| | eeg_fp1 | eeg_f7 | eeg_f8 | eeg_t4 | eeg_t6 | eeg_t5 | eeg_t3 | eeg_fp2 | eeg_o1 | eeg_p3 | ... | eeg_cz_alpha_low | eeg_cz_alpha |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **2558322** | -0.3145 | -2.758000 | 0.0533 | -4.8120 | -7.5820 | 23.330 | 5.965 | 6.370 | 2.790000 | 0.8853 | ... | 0.045752 | 0.0 |
| **3333606** | -26.2300 | -16.860001 | -8.4800 | -8.8400 | -8.0300 | -6.246 | -3.098 | -24.660 | -5.350000 | -3.2700 | ... | 0.008493 | 0.0 |
| **2462164** | -0.5244 | -0.158400 | 5.3120 | 3.5100 | 0.1854 | -7.727 | -4.780 | 11.160 | -0.569300 | -6.5300 | ... | 0.020572 | 0.0 |
| **1355480** | 5.2500 | 5.330000 | -1.4810 | 0.8716 | 4.6800 | 10.300 | 2.700 | 3.754 | 28.799999 | 8.0300 | ... | 0.007739 | 0.0 |
| **2820307** | 2.5140 | 8.870000 | -0.9280 | 2.1840 | -3.6560 | -2.982 | -0.128 | -2.559 | -0.658700 | -2.4430 | ... | 0.007391 | 0.0 |

5 rows × 127 columns

From mentioned link, below points:

1. VIF starts at 1 and has no upper limit
2. VIF = 1, no correlation between the independent variable and the other variables
3. VIF exceeding 5 or 10 indicates high multicollinearity between this independent variable and the others

In [23]:

```
# https://www.analyticsvidhya.com/blog/2020/03/what-is-multicollinearity/

from statsmodels.stats.outliers_influence import variance_inflation_factor

vif = pd.DataFrame()
vif["var"] = df.columns
vif["VIF"] = [variance_inflation_factor(df.values, i) for i in range(df.shape[1])]

vif2=vif.sort_values(by=['VIF'], ascending=False)
vif2.reset_index(inplace = True)

c=0
for i in range(len(vif2['VIF'])):
    if vif2['VIF'][i]>10:
        c=c+1
        print(vif2['var'][i], vif2['VIF'][i])

print(c)
```

```
eeg_o1_gamma 107.1301841149105
eeg_o1_beta 102.63361484068099
eeg_f8_alpha_high 95.68824964502315
eeg_p3_theta 94.33592367818268
eeg_o1_alpha_high 92.91859170270703
eeg_p4_theta 84.99213866658661
eeg_f8_beta 79.08680934153334
eeg_poz_alpha_high 74.6903149754349
eeg_c3_theta 72.0328978169125
eeg_o1_alpha_low 64.34123397271749
eeg_f7_alpha_high 59.92891825805823
eeg_fp2_alpha_high 56.57555055991812
eeg_poz_theta 52.02979122717033
eeg_t4_alpha_high 51.60083073368754
eeg_f7_alpha_low 47.03619139238127
eeg_t4_alpha_low 42.49819206002418
eeg_c4_theta 39.307740217048696
eeg_p3_alpha_low 37.77953845041639
eeg_fp1_alpha_high 36.07051833392366
eeg_poz_beta 34.75391938057621
eeg_poz_gamma 34.63431542018194
eeg_p4_alpha_low 34.3504968854365
eeg_f4_alpha_high 31.40701437266916
eeg_t4_beta 30.163996140579368
eeg_p3_alpha_high 28.239913701038144
r 27.052177830268505
eeg_poz_alpha_low 26.84833986823889
eeg_f7_gamma 26.5276681981512
eeg_f7_beta 25.933657922021442
eeg_fp2_alpha_low 25.894675399530144
eeg_p4_alpha_high 25.79600993206864
eeg_t3_alpha_high 25.638691223258245
eeg_p3_beta 24.68890857357746
eeg_fp1_alpha_low 23.962306849762914
eeg_o2_gamma 23.7847278866201
eeg_t3_alpha_low 22.869927126398608
eeg_c3_alpha_low 22.61258987600746
eeg_fp2_gamma 22.206216109341863
eeg_f8_gamma 22.17952400407939
eeg_p3_gamma 20.892376690396123
resp_rate 20.498346417021075
eeg_p4_beta 19.695836990031655
eeg_f8_alpha_low 19.602727737143596
eeg_t5_alpha_low 19.48028166717917
eeg_t5_alpha_high 18.246966704418877
eeg_f4_alpha_low 17.28338071476659
eeg_c3_alpha_high 16.70012731228095
```

```
eeg_o2_alpha_high 16.596201001953578
eeg_o2_beta 16.285002519202834
eeg_fp2_beta 16.272804382050662
eeg_cz_alpha_high 15.776469176135443
eeg_f7_theta 15.191757876576037
eeg_fp1_gamma 14.967522986207893
eeg_cz_gamma 14.038790737988707
eeg_fp1_beta 13.771247838999955
eeg_c4_alpha_low 13.750925565661166
eeg_t6_alpha_high 13.186189340202 96
eeg_f4_beta 12.776847730976733
eeg_t6_alpha_low 12.539625222217982
eeg_cz_alpha_low 12.130957460804348
eeg_t4_theta 12.111789530146101
eeg_o2_alpha_low 11.827880468558721
eeg_pz_alpha_high 11.542209194175044
eeg_c4_alpha_high 11.177808745793593
eeg_f4_gamma 10.840538724932781
eeg_f3_alpha_high 10.82348567179509
eeg_c3_beta 10.705463619271454
67
```

Here we are getting very high values of VIF for some features. So we can take threshold above which we'll remove those features. Let's take threshold as 50 and remove features one by one, which have VIF above 50.

In [13]:

```
# removing 'eeg_o1_gamma' feature

df=df.drop('eeg_o1_gamma',axis=1)

vif = pd.DataFrame()
vif["var"] = df.columns
vif["VIF"] = [variance_inflation_factor(df.values, i) for i in range(df.shape[1])]

vif2=vif.sort_values(by=['VIF'], ascending=False)
vif2.reset_index(inplace = True)

c=0
for i in range(len(vif2['VIF'])):
    if vif2['VIF'][i]>10:
        c=c+1
        print(vif2['var'][i], vif2['VIF'][i])

print(c)
```

```
eeg_o1_alpha_high 136.30814866185835
eeg_poz_alpha_low 96.27957559744205
eeg_poz_alpha_high 85.68406223864436
eeg_o1_alpha_low 79.21543885579506
eeg_f4_alpha_high 78.70921565359774
eeg_f8_alpha_high 71.12613056790958
eeg_fp1_alpha_high 66.60442217520279
eeg_p3_theta 59.468250192070336
eeg_o1_beta 57.73168014327941
eeg_f7_alpha_high 44.06739873845697
eeg_c3_theta 42.14757946704013
eeg_p3_alpha_low 38.92178662566146
eeg_poz_gamma 37.067837230380526
eeg_p4_alpha_low 36.92548014289338
eeg_p4_theta 36.794037165029074
eeg_f4_alpha_low 35.55799994973639
eeg_poz_beta 35.31593739085859
eeg_fp2_alpha_high 34.60882861758942
eeg_f8_alpha_low 34.26478942135775
eeg_p3_beta 29.627843332427556
eeg_fp1_beta 29.0393714798182
eeg_p4_alpha_high 28.901745614260708
eeg_f8_beta 28.632495129853673
eeg_fp1_alpha_low 28.488273379758585
eeg_p3_alpha_high 27.537558180389993
r 27.08497686873728
eeg_fp1_gamma 25.484179096450813
eeg_poz_theta 25.351529983618832
eeg_o2_gamma 23.94125948396026
```

```
eeg_p4_beta 23.82827078391347
eeg_c4_theta 23.365854769717092
eeg_c3_alpha_low 22.153991466145968
eeg_cz_gamma 21.641685377223965
eeg_o2_beta 21.127093754973302
eeg_cz_alpha_high 20.967181200983696
eeg_fp2_alpha_low 20.88220997024211
resp_rate 20.640339055384423
eeg_f7_beta 20.315049433138476
eeg_f7_alpha_low 20.194234709015156
eeg_t3_alpha_high 19.817353423151715
eeg_f4_gamma 19.159808309193224
eeg_t5_alpha_high 18.705083621495 79
eeg_t5_alpha_low 18.132747800252382
eeg_t3_alpha_low 16.87403486077009
eeg_c3_alpha_high 16.858877189175015
eeg_t4_beta 16.724549572773203
eeg_f4_theta 16.331040342082833
eeg_p3_gamma 16.177707177891293
eeg_t4_alpha_high 16.158732170890634
eeg_cz_alpha_low 16.039697130322118
eeg_fp2_gamma 16.014122472897927
eeg_c4_alpha_low 15.518973123051289
eeg_f4_beta 15.256512044073888
eeg_o2_alpha_high 14.988206731766072
eeg_f7_gamma 13.994682863736006
eeg_t4_alpha_low 13.952980183161847
eeg_f3_alpha_high 12.907979495658548
eeg_c3_beta 12.8097315479128
eeg_fp2_beta 12.038601162340468
eeg_t6_alpha_low 11.944418754839964
eeg_c4_alpha_high 11.599676318458547
eeg_t5_beta 11.182359826648065
eeg_pz_alpha_high 10.420408763496734
63
```

In [15]:

```
# removing 'eeg_o1_alpha_high' feature

df=df.drop('eeg_o1_alpha_high',axis=1)

vif = pd.DataFrame()
vif["var"] = df.columns
vif["VIF"] = [variance_inflation_factor(df.values, i) for i in range(df.shape[1])]

vif2=vif.sort_values(by=['VIF'], ascending=False)
vif2.reset_index(inplace = True)

c=0
for i in range(len(vif2['VIF'])):
    if vif2['VIF'][i]>10:
        c=c+1
        print(vif2['var'][i], vif2['VIF'][i])

print(c)
```

```
eeg_poz_alpha_low 96.27872896634054
eeg_poz_alpha_high 85.65597924690185
eeg_f4_alpha_high 78.56065596553296
eeg_f8_alpha_high 71.09123413651251
eeg_fp1_alpha_high 66.59886726133101
eeg_p3_theta 59.46809423472039
eeg_f7_alpha_high 44.0659074949429
eeg_c3_theta 42.1449425169879
eeg_p3_alpha_low 38.89891831725248
eeg_poz_gamma 37.05699614792764
eeg_p4_alpha_low 36.9254621392299 36
eeg_p4_theta 36.79135531236101
eeg_f4_alpha_low 35.53986159962848
eeg_poz_beta 35.31588887236004
eeg_fp2_alpha_high 34.60358177553107
eeg_f8_alpha_low 34.24555399492418
eeg_o1_alpha_low 33.583276351284496
eeg_o1_beta 32.88470870973811
```

```
eeg_p3_beta 29.608006732164217
eeg_fp1_beta 29.021491988019697
eeg_p4_alpha_high 28.89982214763633
eeg_f8_beta 28.61898475025524
eeg_fp1_alpha_low 28.488060209454225
eeg_p3_alpha_high 27.528229371586356
r 27.079470612772614
eeg_fp1_gamma 25.4785239943227
eeg_poz_theta 25.338402104153356
eeg_o2_gamma 23.93451404753896
eeg_p4_beta 23.813276650045314
eeg_c4_theta 23.364971980034703
eeg_c3_alpha_low 22.15189911787363
eeg_cz_gamma 21.637640973085478
eeg_cz_alpha_high 20.96221688323654
eeg_fp2_alpha_low 20.874003458984173
resp_rate 20.63864358882653
eeg_o2_beta 20.583709234649874
eeg_f7_beta 20.30871234653648
eeg_f7_alpha_low 20.193827544775818
eeg_t3_alpha_high 19.817350154770654
eeg_f4_gamma 19.15838233789812
eeg_t5_alpha_high 18.70349351740661
eeg_t5_alpha_low 18.132546679888126
eeg_t3_alpha_low 16.87331444401507
eeg_c3_alpha_high 16.857149540870413
eeg_t4_beta 16.724493504128812
eeg_f4_theta 16.304466153977064
eeg_p3_gamma 16.17745123624665
eeg_t4_alpha_high 16.154604334592733
eeg_cz_alpha_low 16.036968004343763
eeg_fp2_gamma 16.011793846273985
eeg_c4_alpha_low 15.51894176256181
eeg_f4_beta 15.25112624452625
eeg_o2_alpha_high 14.21636478210996
eeg_f7_gamma 13.994340301745385
eeg_t4_alpha_low 13.952932052745918
eeg_f3_alpha_high 12.901652689986808
eeg_c3_beta 12.809394848758393
eeg_fp2_beta 12.034751369509438
eeg_t6_alpha_low 11.94441678299546
eeg_c4_alpha_high 11.59810536669226
eeg_t5_beta 11.182168572919979
eeg_pz_alpha_high 10.41359033058356
62
```

In [16]:

```
# removing 'eeg_poz_alpha_low' feature

df=df.drop('eeg_poz_alpha_low',axis=1)
vif = pd.DataFrame()
vif["var"] = df.columns
vif["VIF"] = [variance_inflation_factor(df.values, i) for i in range(df.shape[1])]

vif2=vif.sort_values(by=['VIF'], ascending=False)
vif2.reset_index(inplace = True)

c=0
for i in range(len(vif2['VIF'])):
    if vif2['VIF'][i]>10:
        c=c+1
        print(vif2['var'][i], vif2['VIF'][i])

print(c)
```

```
eeg_f4_alpha_high 78.52947959585119
eeg_f8_alpha_high 71.08355045969736
eeg_fp1_alpha_high 66.52551541697667
eeg_p3_theta 58.93387494841991
eeg_f7_alpha_high 44.030986111235095
eeg_c3_theta 41.83986790850303
eeg_p3_alpha_low 37.283512518873756
eeg_poz_gamma 36.66525855543313
eeg_f4_alpha_low 35.53649769962123
```

```
eeg_poz_beta 34.74953462241956
eeg_fp2_alpha_high 34.379656546623345
eeg_p4_alpha_low 34.240828024520155
eeg_f8_alpha_low 34.18211780546573
eeg_o1_alpha_low 33.53891633073421
eeg_p4_theta 32.95825778983023
eeg_o1_beta 32.8116242926766
eeg_p3_beta 29.472164930871507
eeg_fp1_beta 29.01633381465497
eeg_f8_beta 28.596544491707526
eeg_fp1_alpha_low 28.437263990580334
eeg_p4_alpha_high 27.29269967980906
r 27.06496102995794
eeg_p3_alpha_high 26.631663339206863
eeg_fp1_gamma 25.476308210514375
eeg_p4_beta 23.683708967543303
eeg_c4_theta 23.33951590319699
eeg_o2_gamma 22.629041587422385
eeg_c3_alpha_low 22.12130441250354
eeg_poz_theta 21.792965734898907
eeg_cz_gamma 21.6197271421448
resp_rate 20.630526779854023
eeg_fp2_alpha_low 20.376807186272615
eeg_f7_beta 20.296396626449134
eeg_f7_alpha_low 20.1277559400736
eeg_t3_alpha_high 19.81706614130921
eeg_cz_alpha_high 19.78113969338499
eeg_o2_beta 19.588596235595844
eeg_f4_gamma 19.156032342321698
eeg_t5_alpha_high 18.66795276952643
eeg_t5_alpha_low 18.098300013192556
eeg_t3_alpha_low 16.857781914925198
eeg_c3_alpha_high 16.85563698377519
eeg_t4_beta 16.711499943971944
eeg_poz_alpha_high 16.48927975108958
eeg_f4_theta 16.30367823043914
eeg_p3_gamma 16.17686428714243
eeg_t4_alpha_high 16.056766141575753
eeg_fp2_gamma 15.971746634646705
eeg_cz_alpha_low 15.89991847220193
eeg_c4_alpha_low 15.496541020198487
eeg_f4_beta 15.244590417798427
eeg_o2_alpha_high 14.21068751252368
eeg_f7_gamma 13.980645573332302
eeg_t4_alpha_low 13.917086383399058
eeg_f3_alpha_high 12.895276912109976
eeg_c3_beta 12.754680533286429
eeg_fp2_beta 11.978910095457037
eeg_t6_alpha_low 11.942680289692355
eeg_c4_alpha_high 11.54457246554551
eeg_t5_beta 11.16623399288983
eeg_pz_alpha_high 10.406235434009364
61
```

In [17]:

```
# removing 'eeg_f4_alpha_high' feature

df=df.drop('eeg_f4_alpha_high',axis=1)
vif = pd.DataFrame()
vif["var"] = df.columns
vif["VIF"] = [variance_inflation_factor(df.values, i) for i in range(df.shape[1])]

vif2=vif.sort_values(by=['VIF'], ascending=False)
vif2.reset_index(inplace = True)

c=0
for i in range(len(vif2['VIF'])):
    if vif2['VIF'][i]>10:
        c=c+1
        print(vif2['var'][i], vif2['VIF'][i])

print(c)
```

```
eeg_f8_alpha_high 71.08332167947677
eeg_fp1_alpha_high 66.06990615495414
```

```
eeg_p3_theta 58.93346234640915
eeg_f7_alpha_high 44.02996481658753
eeg_c3_theta 41.829454506108696
eeg_p3_alpha_low 37.265893735390385
eeg_poz_gamma 36.61704531226379
eeg_poz_beta 34.734757024019125
eeg_p4_alpha_low 34.23190476568021
eeg_f8_alpha_low 34.18132220732475
eeg_fp2_alpha_high 33.80193110828638
eeg_o1_alpha_low 33.53838038645149
eeg_p4_theta 32.85237950162344
eeg_o1_beta 32.80886714580636
eeg_p3_beta 29.457378601201647
eeg_fp1_beta 28.95070498850506
eeg_f8_beta 28.59363033926226
eeg_fp1_alpha_low 28.239913978754352
eeg_p4_alpha_high 27.267894436246763
r 27.064381281128927
eeg_p3_alpha_high 26.614632077569272
eeg_fp1_gamma 25.475817181536144
eeg_p4_beta 23.66564308371562
eeg_c4_theta 23.32313149889845
eeg_o2_gamma 22.61540587677017
eeg_c3_alpha_low 22.098054928063807
eeg_poz_theta 21.776363637411603
eeg_cz_gamma 21.571023908436505
resp_rate 20.62962395849889
eeg_f4_alpha_low 20.317839717648052
eeg_f7_beta 20.292736828021805
eeg_f7_alpha_low 20.127755773113563
eeg_fp2_alpha_low 20.125772007544096
eeg_t3_alpha_high 19.814018702581443
eeg_cz_alpha_high 19.757754976727522
eeg_o2_beta 19.587218217686285
eeg_t5_alpha_high 18.665215297166146
eeg_t5_alpha_low 18.09800302523791
eeg_t3_alpha_low 16.85461268872238
eeg_c3_alpha_high 16.845349505716435
eeg_t4_beta 16.71054143289085
eeg_poz_alpha_high 16.482909064051796
eeg_f4_gamma 16.38204127045325
eeg_p3_gamma 16.158034548369866
eeg_t4_alpha_high 16.054709677606652
eeg_cz_alpha_low 15.89736843653048
eeg_fp2_gamma 15.818810527430756
eeg_c4_alpha_low 15.464481524876277
eeg_f4_theta 14.661903861091732
eeg_o2_alpha_high 14.203989303497956
eeg_f7_gamma 13.941753703253294
eeg_t4_alpha_low 13.913043544787651
eeg_f3_alpha_high 12.892743184580207
eeg_c3_beta 12.75310269197832
eeg_fp2_beta 11.965800612685314
eeg_t6_alpha_low 11.936963683423262
eeg_c4_alpha_high 11.531625509656253
eeg_t5_beta 11.163288473065027
eeg_pz_alpha_high 10.39614625433884
eeg_f4_beta 10.167250953403252
60
```

In [18]:

```
# removing 'eeg_f8_alpha_high' feature

df=df.drop('eeg_f8_alpha_high',axis=1)
vif = pd.DataFrame()
vif["var"] = df.columns
vif["VIF"] = [variance_inflation_factor(df.values, i) for i in range(df.shape[1])]

vif2=vif.sort_values(by=['VIF'], ascending=False)
vif2.reset_index(inplace = True)

c=0
for i in range(len(vif2['VIF'])):
    if vif2['VIF'][i]>10:
        c=c+1
```

```
        print(vif2['var'][i], vif2['VIF'][i])

print(c)
```

```
eeg_fp1_alpha_high 66.06291558961509
eeg_p3_theta 58.91529739771062
eeg_f7_alpha_high 42.635028971591254
eeg_c3_theta 41.8253019134098
eeg_p3_alpha_low 37.25498814989989
eeg_poz_gamma 36.5593914129057
eeg_poz_beta 34.72537576465631
eeg_p4_alpha_low 34.22689001022839
eeg_o1_alpha_low 33.41434336584341
eeg_p4_theta 32.85126539959785
eeg_o1_beta 32.64456837750116
eeg_p3_beta 29.407176425145366
eeg_fp1_beta 28.602009739635765
eeg_fp2_alpha_high 28.355264950964433
eeg_fp1_alpha_low 28.043689928267383
eeg_p4_alpha_high 27.240042399006636
r 27.063856933762647
eeg_p3_alpha_high 26.613921093953326
eeg_fp1_gamma 25.475149200854606
eeg_p4_beta 23.652036281849423
eeg_c4_theta 23.290618385492596
eeg_o2_gamma 22.60723728210911
eeg_c3_alpha_low 22.097577492175382
eeg_poz_theta 21.74357421917015
eeg_cz_gamma 21.421323051265272
resp_rate 20.628333431854646
eeg_f4_alpha_low 20.313327072935056
eeg_f7_beta 20.212891264750855
eeg_f7_alpha_low 19.880980607493942
eeg_cz_alpha_high 19.75724991134821
eeg_o2_beta 19.587121092144514
eeg_t3_alpha_high 19.07447613056141
eeg_t5_alpha_high 18.6123169086098
eeg_t5_alpha_low 18.088677751480727
eeg_fp2_alpha_low 17.582856187005415
eeg_c3_alpha_high 16.84316384236345
eeg_poz_alpha_high 16.4292624276529 86
eeg_f4_gamma 16.371611007155927
eeg_t4_beta 16.184841047591576
eeg_p3_gamma 16.124920207423596
eeg_t4_alpha_high 15.971361033607424
eeg_cz_alpha_low 15.887108833747947
eeg_fp2_gamma 15.722823212180646
eeg_t3_alpha_low 15.68618732916051
eeg_c4_alpha_low 15.433961880900592
eeg_f8_beta 15.10575886035589
eeg_f4_theta 14.654188764728634
eeg_o2_alpha_high 14.169364780244688
eeg_f7_gamma 13.853484700249686
eeg_t4_alpha_low 13.803979335021022
eeg_f3_alpha_high 12.89155891635641
eeg_c3_beta 12.752361288128856
eeg_t6_alpha_low 11.848898029615718
eeg_fp2_beta 11.666633198154445
eeg_c4_alpha_high 11.484001551952444
eeg_t5_beta 11.16280439315669
eeg_f8_alpha_low 10.528341562280913
eeg_pz_alpha_high 10.387205520537798
eeg_f4_beta 10.166247161595527
59
```

In [19]:

```
# removing 'eeg_fp1_alpha_high' feature

df=df.drop('eeg_fp1_alpha_high',axis=1)
vif = pd.DataFrame()
vif["var"] = df.columns
vif["VIF"] = [variance_inflation_factor(df.values, i) for i in range(df.shape[1])]

vif2=vif.sort_values(by=['VIF'], ascending=False)
```

```
vif2.reset_index(inplace = True)

c=0
for i in range(len(vif2['VIF'])):
    if vif2['VIF'][i]>10:
        c=c+1
        print(vif2['var'][i], vif2['VIF'][i])

print(c)
```

```
eeg_p3_theta 58.914891651030715
eeg_f7_alpha_high 42.13274078785371
eeg_c3_theta 41.82529422861325
eeg_p3_alpha_low 36.922602805611625
eeg_poz_gamma 36.45574693035603
eeg_poz_beta 34.67933227762848
eeg_p4_alpha_low 34.22289196979079
eeg_o1_alpha_low 33.397017909999974
eeg_p4_theta 32.72744112846342
eeg_o1_beta 32.642664320436324
eeg_p3_beta 29.37494374334942
r 27.063839248980187
eeg_p4_alpha_high 27.003073571995937
eeg_p3_alpha_high 26.606953145180956
eeg_fp1_gamma 24.9207386129968
eeg_p4_beta 23.41387746276144
eeg_c4_theta 23.16758735904565
eeg_o2_gamma 22.602483044911192
eeg_c3_alpha_low 22.037090699965617
eeg_poz_theta 21.596661238758387
eeg_cz_gamma 21.06416647562665
resp_rate 20.62532070729276
eeg_f4_alpha_low 20.291454432917405
eeg_f7_beta 20.121656956334924
eeg_fp1_beta 20.006243575833267
eeg_f7_alpha_low 19.823873385015194
eeg_cz_alpha_high 19.55553873806741
eeg_o2_beta 19.554990283789685
eeg_t3_alpha_high 19.069382220109983
eeg_t5_alpha_high 18.6094902603006
eeg_t5_alpha_low 18.078846607033906
eeg_c3_alpha_high 16.841630138445865
eeg_fp2_alpha_high 16.753300633132707
eeg_poz_alpha_high 16.405287097703887
eeg_f4_gamma 16.31964170554599
eeg_t4_beta 16.166262631819432
eeg_p3_gamma 16.12453424222673
eeg_cz_alpha_low 15.695991557695917
eeg_t3_alpha_low 15.674882448585493
eeg_t4_alpha_high 15.545910352445992
eeg_c4_alpha_low 15.354372890305768
eeg_f8_beta 15.105722423576116
eeg_f4_theta 14.652782392463525
eeg_fp2_gamma 14.60141913919837
eeg_f7_gamma 13.853475343781527
eeg_o2_alpha_high 13.814625688271242
eeg_t4_alpha_low 13.666823809350904
eeg_f3_alpha_high 12.889813077500092
eeg_c3_beta 12.74385904687886
eeg_fp2_alpha_low 12.51713801063161
eeg_t6_alpha_low 11.796536501704601
eeg_c4_alpha_high 11.481776509266375
eeg_fp2_beta 11.268803841780395
eeg_t5_beta 11.153263120236854
eeg_f8_alpha_low 10.5244479221247 48
eeg_pz_alpha_high 10.385623800950187
eeg_f4_beta 10.165962092205156
57
```

In [20]:

```
# removing 'eeg_p3_theta' feature

df=df.drop('eeg_p3_theta',axis=1)
vif = pd.DataFrame()
```

```
vif["var"] = df.columns
vif["VIF"] = [variance_inflation_factor(df.values, i) for i in range(df.shape[1])]

vif2=vif.sort_values(by=['VIF'], ascending=False)
vif2.reset_index(inplace = True)

c=0
for i in range(len(vif2['VIF'])):
    if vif2['VIF'][i]>10:
        c=c+1
        print(vif2['var'][i], vif2['VIF'][i])

print(c)
```

```
eeg_f7_alpha_high 41.63644816855291
eeg_poz_gamma 36.43954194559523
eeg_p3_alpha_low 36.142465762158 98
eeg_poz_beta 34.616169623553766
eeg_p4_alpha_low 33.754905302853956
eeg_o1_alpha_low 33.393651861963626
eeg_o1_beta 32.6425799728356
eeg_p3_beta 29.129694503110628
eeg_p4_theta 28.66985977110551
r 27.0478915808393
eeg_p4_alpha_high 27.001995356396918
eeg_p3_alpha_high 26.178561004235604
eeg_fp1_gamma 24.502761935928117
eeg_p4_beta 23.407107677411975
eeg_c4_theta 23.161345848391274
eeg_c3_theta 22.90849325116147
eeg_o2_gamma 22.583789125437544
eeg_c3_alpha_low 21.3700703385221 05
eeg_cz_gamma 21.058097859954753
resp_rate 20.60038013620279
eeg_f4_alpha_low 20.291448187873286
eeg_f7_beta 20.076663393899235
eeg_fp1_beta 19.861704969760687
eeg_f7_alpha_low 19.696710030866523
eeg_cz_alpha_high 19.552703838156397
eeg_o2_beta 19.547079995714675
eeg_t3_alpha_high 19.00874891982403
eeg_t5_alpha_high 18.53304668312037
eeg_t5_alpha_low 17.982997197440078
eeg_c3_alpha_high 16.80898738639548
eeg_fp2_alpha_high 16.711742130869013
eeg_f4_gamma 16.316198927617062
eeg_t4_beta 16.160170034903143
eeg_p3_gamma 16.116022081112458
eeg_poz_alpha_high 15.965804210868638
eeg_cz_alpha_low 15.686862688603904
eeg_t3_alpha_low 15.66594130923366
eeg_t4_alpha_high 15.423335244909925
eeg_c4_alpha_low 15.327580767006468
eeg_f8_beta 15.073832723891545
eeg_f4_theta 14.652781667554688
eeg_fp2_gamma 14.309403788058418
eeg_f7_gamma 13.8515387645942
eeg_o2_alpha_high 13.813566563891678
eeg_t4_alpha_low 13.593823789993122
eeg_f3_alpha_high 12.889784810615078
eeg_c3_beta 12.670971579283988
eeg_fp2_alpha_low 12.389756575477172
eeg_poz_theta 12.340076631868957
eeg_t6_alpha_low 11.737430382535916
eeg_c4_alpha_high 11.473824128979224
eeg_t5_beta 11.149034578984715
eeg_fp2_beta 11.130442596450525
eeg_f8_alpha_low 10.51642140919932
eeg_pz_alpha_high 10.38554389921267
eeg_f4_beta 10.160186844570694
56
```

Now all features having VIF less than 50. We'll save these features, in case we are going to use these for our model.

In [25]:

In [25]:

```
import pickle

VIF_col=df.columns.to_list()
with open("VIF_col.txt", "wb") as f:
    pickle.dump(VIF_col, f)
```

## Permutation Importance

In [15]:

```
y_t=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train.shape
```

Out[15]:

```
(4182820, 127)
```

In [17]:

```
#
https://eli5.readthedocs.io/en/latest/autodocs/sklearn.html#eli5.sklearn.permutation_importance.Pe
tionImportance
# https://www.kaggle.com/dansbecker/permutation-importance

import eli5
from eli5.sklearn import PermutationImportance
from sklearn.model_selection import train_test_split

train_X, val_X, train_y, val_y = train_test_split(new_train, y_t, random_state=1)
mod = DecisionTreeClassifier(random_state=0).fit(train_X, train_y)

perm = PermutationImportance(mod, random_state=1).fit(val_X, val_y)
t=eli5.show_weights(perm, feature_names = val_X.columns.tolist())
t
```

Out[17]:

| Weight | Feature |
|---|---|
| 0.2014 ± 0.0115 | r |
| 0.2001 ± 0.0082 | ecg |
| 0.1398 ± 0.0064 | gsr |
| 0.0906 ± 0.0033 | pilot |
| 0.0840 ± 0.0072 | eeg_f7_gamma |
| 0.0460 ± 0.0065 | gsr_amp |
| 0.0352 ± 0.0034 | eeg_t3_gamma |
| 0.0282 ± 0.0031 | eeg_t4_gamma |
| 0.0264 ± 0.0045 | eeg_fp1_theta |
| 0.0154 ± 0.0014 | eeg_c3_gamma |
| 0.0142 ± 0.0017 | eeg_fp2_gamma |
| 0.0141 ± 0.0017 | eeg_f8_gamma |
| 0.0131 ± 0.0039 | resp_rate |
| 0.0123 ± 0.0023 | eeg_t5_gamma |
| 0.0110 ± 0.0033 | eeg_c4_gamma |
| 0.0105 ± 0.0042 | eeg_f3_alpha_high |
| 0.0105 ± 0.0016 | eeg_f8_beta |
| 0.0104 ± 0.0022 | eeg_pz_theta |
| 0.0088 ± 0.0012 | eeg_p3_gamma |
| 0.0087 ± 0.0044 | eeg_o1_gamma |
| … 107 more … | |

In [31]:

```
# creating dataframe with features and their corresponding weight which we calculated from permuta
tion importance features

fea=val_X.columns.tolist()
val=perm.feature_importances_
PermImp_df=pd.DataFrame(fea,columns=['feature'])
PermImp_df['weight']=val

PermImp_df=PermImp_df.sort_values(by=['weight'], ascending=False)
PermImp_df.reset_index(inplace = True)
PermImp_df=PermImp_df.drop('index',axis=1)
PermImp_df
```

Out[31]:

| | feature | weight |
|---|---|---|
| **0** | r | 0.20144 |
| **1** | ecg | 0.20008 |
| **2** | gsr | 0.13984 |
| **3** | pilot | 0.09060 |
| **4** | eeg_f7_gamma | 0.08400 |
| **...** | ... | ... |
| **122** | eeg_t4_theta | -0.00008 |
| **123** | eeg_c4_beta | -0.00008 |
| **124** | eeg_pz | -0.00012 |
| **125** | eeg_fz_theta | -0.00076 |
| **126** | eeg_cz_alpha_low | -0.00128 |

127 rows × 2 columns

In [33]:

```
c=0
for i in PermImp_df['weight']:
    if i>0:
        c=c+1

c
```

Out[33]:

```
120
```

In [40]:

```
PermImp_df['weight'][90:]

# How many features I should keep? 119?
```

Out[40]:

```
90     0.00072
91     0.00068
92     0.00068
93     0.00064
94     0.00060
95     0.00060
96     0.00056
97     0.00056
98     0.00056
99     0.00052
100    0.00048
101    0.00048
102    0.00048
103    0.00040
104    0.00040
105    0.00040
106    0.00032
107    0.00032
108    0.00028
109    0.00020
110    0.00020
111    0.00020
112    0.00020
113    0.00020
114    0.00020
115    0.00016
116    0.00012
117    0.00012
```

```
118    0.00004
119    0.00004
120    0.00000
121   -0.00008
122   -0.00008
123   -0.00008
124   -0.00012
125   -0.00076
126   -0.00128
Name: weight, dtype: float64
```

In [37]:

```
PermImp_df.to_csv('PermImp_df.csv')
```

In [28]:

```
p=pd.read_csv('PermImp_df.csv')
p=p.drop('Unnamed: 0',axis=1)
p
```

Out[28]:

| | feature | weight |
|---|---|---|
| **0** | r | 0.20144 |
| **1** | ecg | 0.20008 |
| **2** | gsr | 0.13984 |
| **3** | pilot | 0.09060 |
| **4** | eeg_f7_gamma | 0.08400 |
| **...** | ... | ... |
| **122** | eeg_t4_theta | -0.00008 |
| **123** | eeg_c4_beta | -0.00008 |
| **124** | eeg_pz | -0.00012 |
| **125** | eeg_fz_theta | -0.00076 |
| **126** | eeg_cz_alpha_low | -0.00128 |

127 rows × 2 columns

## Recursive Feature Elimination (RFE)

In [9]:

```
# taking sample of train dataset

df=new_train.sample(30000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df.shape
```

Out[9]:

```
(30000, 127)
```

In [10]:

```
# https://www.scikit-yb.org/en/latest/api/model_selection/rfecv.html

from sklearn.model_selection import StratifiedKFold
from yellowbrick.model_selection import RFECV
from sklearn.linear_model import SGDClassifier

cv = StratifiedKFold(2)
```

```
visualizer = RFECV(SGDClassifier(n_jobs=-1), cv=cv, scoring='f1_weighted', n_jobs=-1)
visualizer.fit(df, y)
```

Out[10]:

```
RFECV(ax=<matplotlib.axes._subplots.AxesSubplot object at 0x7fa66bb0beb8>,
      cv=StratifiedKFold(n_splits=2, random_state=None, shuffle=False),
      groups=None, model=None, scoring='f1_weighted', step=1)
```

In [12]:

```
print(visualizer.support_)
visualizer.ranking_
```

```
[ True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True False False  True  True
 False False False  True  True  True False False  True  True  True  True
 False  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True False  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True  True  True  True  True  True
  True  True  True  True  True  True  True]
```

Out[12]:

```
array([ 1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1, 10,  3,  1,  1,  5,  7,  4,
        1,  1,  1,  6,  8,  1,  1,  1,  1,  9,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  2,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
        1,  1,  1,  1,  1,  1,  1,  1])
```

In [23]:

```
# taking those features which have ranking 1, means are selected in RFECV

fea=list(df.columns)
f=[]
for i in range(len(visualizer.ranking_)):
    if visualizer.ranking_[i] ==1:
        f.append(fea[i])


len(f)
```

Out[23]:

```
118
```

In [25]:

```
import pickle

RFE_col=f
with open("RFE_col.txt", "wb") as fe:
    pickle.dump(RFE_col, fe)
```

In [39]:

```

```

Summary and Conclusions :

1. Originally, we had 28 features in the train dataset and 28 features in the test dataset. We have 'event' as the

output/dependent variable in the train dataset.
2. And 'id' feature in test dataset which is not present in the train dataset. We'll use it only for the submission, not in the model prediction.
3. We have huge train and test datasets, so we used a memory optimization function to change the datatypes of columns accordingly (int8, flloat32).
4. From EDA we know most of the features are following gaussian distribution with skewness. But as decision trees/gradient boosting algorithms don't get affected by skewness, we are not going to handle this.
5. We have imbalance data, so we will make this balanced in the next section.
6. 'Experiment' feature has 'CA', 'DA', 'SS' values in train dataset, and 'LOFT' in the test dataset. So we are not going to consider this feature for our model.
7. We have 9 unique crews. And each sitting in either seat 0 or seat 1. So we can have total unique 18 pilots. In the same way, we created a pilot column. And will remove crew and seat features. As we saw in the "count plot for seat w.r.t. event", seat feature is not helping with events.
8. For the "time" feature, we can't use the "time" feature as flight simulator time has nothing to do with the experiment time.
9. Now comes to feature engineering, here we are using biosppy inbuilt functions to create new features from sensor readings. refer: "https://biosppy.readthedocs.io/en/stable/biosppy.signals.html#biosppy.signals.eeg.get_power_features".

1. And if any signal reading is zero we are replacing it with nan. And if not, we will use interpolation for the new feature values, like 'heart_rate', 'resp_rate', 'gsr_amp' (3 new features) and 5 new features for each eeg feature (20*5 = 100 new features) using 'get_power_features' function.
2. Also created a 'pilot' feature with seat and crew.
3. In this way we created 104 new features and dropping some features like seat, experiment, time, crew.
4. Now after creating all the features, we have to check for correlation. So we have 3 techniques to do that: a) VIF (variance inflation factor) b) permutation importance c) recursive feature elimination --> rfe

1. Then we checked for correlation by VIF (variance inflation factor), and set the threshold as 50. Features having VIF more than 50, will remove those recursively. And get our final features having VIF less than 50.
2. Now checked with permutaion importance, it will return features with their weights. High weight means important feature.
3. Then checked with rfe, having attribute .ranking_ which gives 1 for the selected features. In this way we can have our important features. When we have highly correlated features, we should go for rfecv.
4. We can take features from any above 3 techniques mentioned and use for the model.

In [ ]:

```
```

# Modeling

In [33]:

```
# imbalance percentage
c_0=0
c_1=0
c_2=0
c_3=0
for i in y:
    if i==0:
        c_0=c_0+1
    elif i==1:
        c_1=c_1+1
    elif i==2:
        c_2=c_2+1
    elif i==3:
        c_3=c_3+1

print('0 class percent : ',(c_0/len(y)))
print('1 class percent : ',(c_1/len(y)))
print('2 class percent : ',(c_2/len(y)))
print('3 class percent : ',(c_3/len(y)))
```

```
0 class percent :  0.5575085707728279
1 class percent :  0.028466441300366737
2 class percent :  0.37312100448979396
3 class percent :  0.0409039834370114
```

As we have imbalanced class, we will use 'class_weight' attribute as 'balanced' in the models. Or, we can use weight for each class.

In [5]:

```
# features which we got after permutation importance

features=pd.read_csv('PermImp_df.csv')

featr=[]
for i in range(len(features)):
    if (features['weight'][i]<=0):
        featr.append(features.iloc[i]['feature'])

len(featr)
```

Out[5]:

```
7
```

## Random Model

In [6]:

```
# with full train data

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
new_train.head()

# after dropping 7 columns, finally we have 120 columns
```

Out[6]:

| | eeg_fp1 | eeg_f7 | eeg_f8 | eeg_t4 | eeg_t6 | eeg_t5 | eeg_t3 | eeg_fp2 | eeg_o1 | eeg_p3 | ... | eeg_cz_theta | eeg_cz_ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -5.285 | 26.780001 | -9.5200 | -12.800000 | 16.719999 | 33.750000 | 23.719999 | -6.6950 | 29.230000 | 24.840000 | ... | 0.264202 | |
| **1** | -2.428 | 28.440001 | -9.3200 | -3.758000 | 15.970000 | 30.440001 | 21.020000 | -6.4770 | 26.639999 | 24.139999 | ... | 0.280113 | |
| **2** | 10.670 | 30.420000 | 15.3500 | 24.719999 | 16.139999 | 32.160000 | 25.440001 | -0.0887 | 28.120001 | 26.889999 | ... | 0.295369 | |
| **3** | 11.450 | 25.610001 | 2.4340 | 12.414000 | 20.530001 | 31.500000 | 19.139999 | -0.2566 | 30.660000 | 24.250000 | ... | 0.309985 | |
| **4** | 7.285 | 25.940001 | 0.1136 | 5.746000 | 19.830000 | 28.750000 | 20.580000 | -1.9530 | 31.719999 | 25.160000 | ... | 0.323974 | |

5 rows × 120 columns

In [12]:

```
# train and test split to get train and val data

train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[12]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [13]:

```
y_cv_predict = np.zeros((val_df.shape[0],4))

for i in range(val_df.shape[0]):
    rand_probs = np.random.rand(1,4)
    y_cv_predict[i] = ((rand_probs/sum(sum(rand_probs)))[0])
print("CV Data log-loss for Random Model",log_loss(y_val,y_cv_predict, eps=1e-15))
```

```
CV Data log-loss for Random Model 1.6455166263817762
```

## LGB Model

In [11]:

```
# taking sample

df=new_train.sample(90000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df.shape
df=df.drop(featr,axis=1)
df.head()

# after dropping 7 columns, finally we have 120 columns
```

Out[11]:

| | eeg_fp1 | eeg_f7 | eeg_f8 | eeg_t4 | eeg_t6 | eeg_t5 | eeg_t3 | eeg_fp2 | eeg_o1 | eeg_p3 | ... | eeg_cz_theta | eeg_cz_a |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 2.656 | 36.380001 | -44.279999 | -80.800003 | -10.1250 | 60.900002 | 1.849 | -0.071170 | 31.469999 | 31.059999 | ... | 0.036738 | |
| **1** | -10.945 | -0.986000 | 7.535000 | 2.648000 | 11.4900 | 0.320000 | -3.982 | -18.969999 | 4.010000 | 5.688000 | ... | 0.088851 | |
| **2** | 1.668 | 6.350000 | -4.580000 | -1.198000 | -11.0700 | 4.168000 | 9.445 | -0.605000 | -3.523000 | 3.912000 | ... | 0.032846 | |
| **3** | 8.484 | -16.059999 | -12.120000 | 1.365000 | 0.4553 | -0.348600 | -8.350 | 2.541000 | -2.896000 | -6.402000 | ... | 0.008405 | |
| **4** | 4.410 | 4.598000 | 3.700000 | 8.290000 | 13.7000 | 5.760000 | 3.215 | 4.938000 | 10.336000 | 0.781700 | ... | 0.067252 | |

5 rows × 120 columns

In [12]:

```
# train and test split to get train and val data

train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=
32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[12]:

```
((72000, 120), (18000, 120), 72000, 18000)
```

In [55]:

```
# lightgbm - hyperparameter tuning
# https://towardsdatascience.com/catboost-vs-light-gbm-vs-xgboost-5f93620723db

import lightgbm as lgb
from sklearn import metrics
from sklearn.model_selection import GridSearchCV

lg = lgb.LGBMClassifier(objective='multiclass', class_weight ='balanced')
param_dist = {"max_depth" : [10,25,50],"learning_rate" : [0.1, 0.5, 0.6],"n_estimators": [30,100,20
0]}

grid_search = GridSearchCV(lg, n_jobs=-1, param_grid=param_dist, cv = 3, verbose=5)
grid_search.fit(train_df, y_train)
grid_search.best_estimator_
```

```
Fitting 3 folds for each of 27 candidates, totalling 81 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed:   13.5s
[Parallel(n_jobs=-1)]: Done  56 tasks      | elapsed:  5.4min
[Parallel(n_jobs=-1)]: Done  81 out of  81 | elapsed:  7.9min finished
```

Out[55]:

```
LGBMClassifier(boosting_type='gbdt', class_weight='balanced',
               colsample_bytree=1.0, importance_type='split', learning_rate=0.5,
               max_depth=10, min_child_samples=20, min_child_weight=0.001,
               min_split_gain=0.0, n_estimators=200, n_jobs=-1, num_leaves=31,
               objective='multiclass', random_state=None, reg_alpha=0.0,
               reg_lambda=0.0, silent=True, subsample=1.0,
               subsample_for_bin=200000, subsample_freq=0)
```

In [6]:

```
# with full train data

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
new_train.head()
```

Out[6]:

| | eeg_fp1 | eeg_f7 | eeg_f8 | eeg_t4 | eeg_t6 | eeg_t5 | eeg_t3 | eeg_fp2 | eeg_o1 | eeg_p3 | ... | eeg_cz_theta | eeg_cz_ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -5.285 | 26.780001 | -9.5200 | -12.800000 | 16.719999 | 33.750000 | 23.719999 | -6.6950 | 29.230000 | 24.840000 | ... | 0.264202 | |
| **1** | -2.428 | 28.440001 | -9.3200 | -3.758000 | 15.970000 | 30.440001 | 21.020000 | -6.4770 | 26.639999 | 24.139999 | ... | 0.280113 | |
| **2** | 10.670 | 30.420000 | 15.3500 | 24.719999 | 16.139999 | 32.160000 | 25.440001 | -0.0887 | 28.120001 | 26.889999 | ... | 0.295369 | |
| **3** | 11.450 | 25.610001 | 2.4340 | 12.414000 | 20.530001 | 31.500000 | 19.139999 | -0.2566 | 30.660000 | 24.250000 | ... | 0.309985 | |
| **4** | 7.285 | 25.940001 | 0.1136 | 5.746000 | 19.830000 | 28.750000 | 20.580000 | -1.9530 | 31.719999 | 25.160000 | ... | 0.323974 | |

5 rows × 120 columns

In [7]:

```
# train and validation dataset

train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[7]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [25]:

```
# tuning n_estimators

d_train = lgb.Dataset(train_df, label=y_train)

for i in [50, 100, 200]:
    params={'class_weight':'balanced', 'learning_rate':0.5, 'max_depth':10, 'objective':'multiclass
','num_class':4,'metric': 'multi_logloss', 'n_estimators':i}
    model2 = lgb.train(params, d_train)
    y_predict=model2.predict(train_df)
    print('TRAIN LOSS is ',log_loss(y_train, y_predict),i)
    y_predict=model2.predict(val_df)
    print('CV LOSS is ',log_loss(y_val, y_predict),i)
```

```
TRAIN LOSS is  0.046452663938260655 50
CV LOSS is  0.04811346513573304 50
TRAIN LOSS is  0.04548461693615258 100
CV LOSS is  0.04999708108552035 100
TRAIN LOSS is  0.12127052650222386 200
CV LOSS is  0.13018845890431907 200
```

In [14]:

```
d_train = lgb.Dataset(train_df, label=y_train)
params={'class_weight':'balanced', 'learning_rate':0.5, 'max_depth':10,
```

```
        'n_estimators':50, 'objective':'multiclass','num_class':4,'metric': 'multi_logloss'}

model2 = lgb.train(params, d_train)

y_predict=model2.predict(train_df)
print('TRAIN LOSS is ',log_loss(y_train, y_predict))
y_predict=model2.predict(val_df)
print('CV LOSS is ',log_loss(y_val, y_predict))
```

```
TRAIN LOSS is  0.04645266393826068
CV LOSS is  0.04811346513573305
```

In [28]:

```
confusion_matrix(np.argmax(y_predict, axis=1), y_val)
```

Out[28]:

```
array([[465184,    382,    204,   7121],
       [   274,  23430,    114,      4],
       [   351,      1, 311791,      5],
       [   582,      1,     31,  27089]])
```

In [21]:

```
# test data predictions

list_tup=[]
for tup in [(x,y) for x in np.unique(test['crew']) for y in np.unique(test['seat'])]:
    list_tup.append(tup)

test_id_f=[]
pred=[]
for i in range(0,18):
    test_df,test_id=test_data_FE(i,list_tup)
    test_id_f.append(test_id)
    test_df=test_df.drop(featr,axis=1)
    y_test_pred=model2.predict(test_df)
    pred.append(y_test_pred)
    print('---------------------------------- loop : ',i+1,' ----------------------------------')
```

```
---------------------------------- loop :  1  ----------------------------------
---------------------------------- loop :  2  ----------------------------------
---------------------------------- loop :  3  ----------------------------------
---------------------------------- loop :  4  ----------------------------------
missing gsr
---------------------------------- loop :  5  ----------------------------------
---------------------------------- loop :  6  ----------------------------------
---------------------------------- loop :  7  ----------------------------------
---------------------------------- loop :  8  ----------------------------------
---------------------------------- loop :  9  ----------------------------------
failed "gsr_amp" extraction
---------------------------------- loop :  10  ----------------------------------
---------------------------------- loop :  11  ----------------------------------
---------------------------------- loop :  12  ----------------------------------
missing gsr
---------------------------------- loop :  13  ----------------------------------
---------------------------------- loop :  14  ----------------------------------
---------------------------------- loop :  15  ----------------------------------
---------------------------------- loop :  16  ----------------------------------
---------------------------------- loop :  17  ----------------------------------
---------------------------------- loop :  18  ----------------------------------
```

In [24]:

```
test_pr=[]
tst_id=[]
for i in pred:
    for j in i:
        test_pr.append(j)

for i in test_id_f:
    for j in i:
```

```
        tst_id.append(j)

len(test_pr),len(tst_id),len(test)
```

Out[24]:

```
(17965143, 17965143, 17965143)
```

In [25]:

```
sub=pd.DataFrame(test_pr,columns=['A','B','C','D'])
sub['id']=tst_id
sub=sub[['id','A','B','C','D']]
sub=sub.sort_values(by=['id'])
sub.reset_index(inplace = True)
sub=sub.drop('index',axis=1)
sub
```

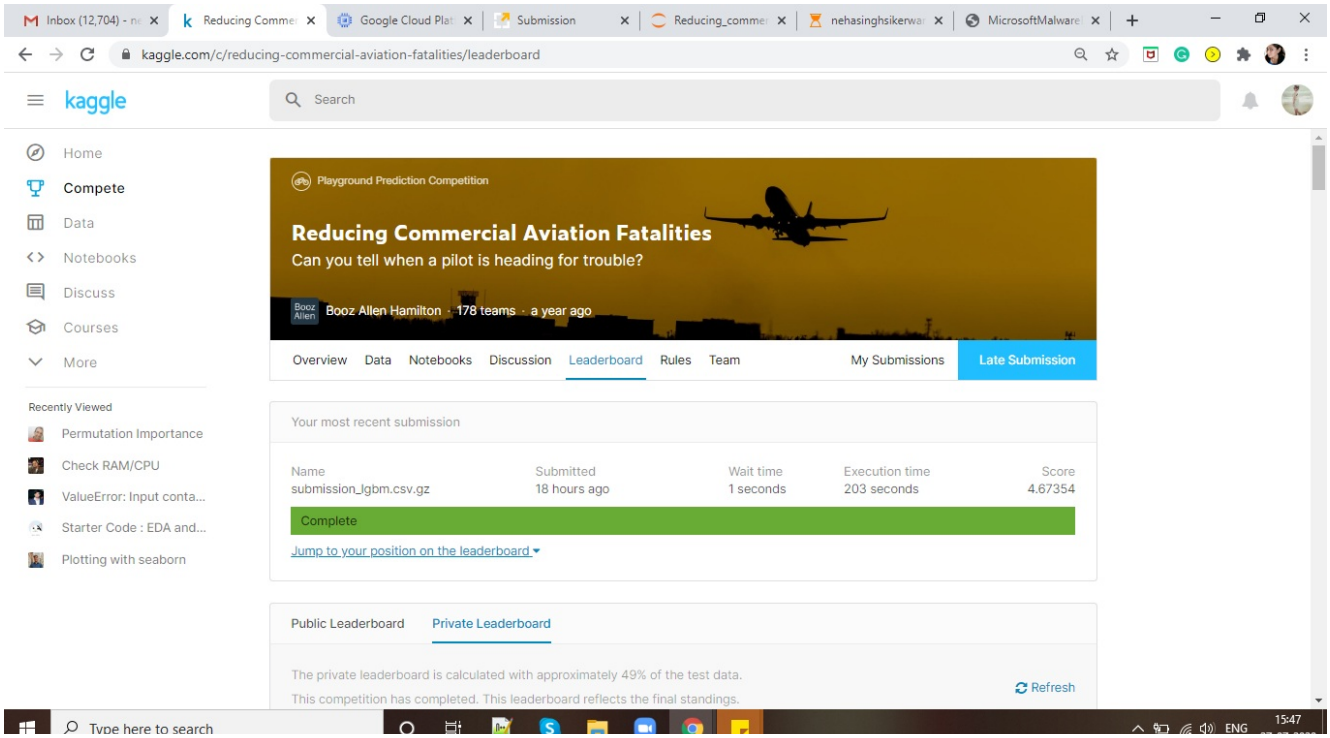
Out[25]:

| | id | A | B | C | D |
|---|---|---|---|---|---|
| **0** | 0 | 6.890736e-01 | 3.737152e-03 | 2.947467e-01 | 1.244253e-02 |
| **1** | 1 | 8.337299e-01 | 1.228325e-155 | 1.361643e-03 | 1.649085e-01 |
| **2** | 2 | 6.145191e-01 | 4.371920e-03 | 3.811090e-01 | 3.062289e-26 |
| **3** | 3 | 8.339960e-01 | 9.760696e-156 | 1.042822e-03 | 1.649611e-01 |
| **4** | 4 | 7.499924e-01 | 4.766552e-03 | 2.452411e-01 | 2.014774e-26 |
| **...** | ... | ... | ... | ... | ... |
| **17965138** | 17965138 | 0.000000e+00 | 0.000000e+00 | 4.454513e-10 | 1.000000e+00 |
| **17965139** | 17965139 | 1.453580e-07 | 0.000000e+00 | 9.999999e-01 | 0.000000e+00 |
| **17965140** | 17965140 | 0.000000e+00 | 0.000000e+00 | 1.000000e+00 | 1.877822e-22 |
| **17965141** | 17965141 | 1.453580e-07 | 0.000000e+00 | 9.999999e-01 | 0.000000e+00 |
| **17965142** | 17965142 | 0.000000e+00 | 0.000000e+00 | 1.000000e+00 | 1.877822e-22 |

17965143 rows × 5 columns

In [2]:

```
sub.to_csv('submission_lgbm.csv.gz',index=False, compression='gzip')
```



In [ ]:

```
```

## XGBoost

In [13]:

```
# hyperparameter tuning with train data sample

df=new_train.sample(90000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df=df.drop(featr,axis=1)
df.head()

train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=
32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[13]:

```
((72000, 120), (18000, 120), 72000, 18000)
```

In [35]:

```
from xgboost.sklearn import XGBClassifier
from scipy.stats import uniform, randint

x_cfl=XGBClassifier(weight=[0.5,10,0.7,7], objective='multi:softprob',tree_method='hist', max_delta
_step=2, n_jobs=-1, eval_metric='mlogloss')
prams={'learning_rate':uniform(0.03, 0.4),
     'n_estimators':randint(50,200),
     'max_depth':randint(5,30)}

random_cfl=RandomizedSearchCV(x_cfl,param_distributions=prams,verbose=10,n_jobs=-1)
random_cfl.fit(train_df, y_train)
print (random_cfl.best_params_)
```

```
{'learning_rate': 0.38241502061564765, 'n_estimators': 106, 'max_depth': 11}
```

In [9]:

```
# training with full train data

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
new_train.shape

train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[9]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [14]:

```
from xgboost.sklearn import XGBClassifier

x_cfl=XGBClassifier(objective='multi:softprob',eval_metric='mlogloss', max_delta_step=5, tree_metho
d='hist', n_estimators=106, max_depth=11, learning_rate=0.38, n_jobs=-1)
x_cfl.fit(train_df,y_train,verbose=True)
predict_y = x_cfl.predict_proba(train_df)
```

```
print(log_loss(y_train, predict_y))
predict_y = x_cfl.predict_proba(val_df)
print("The cross validation log loss is:",log_loss(y_val, predict_y))
```

```
0.00024006044310750398
The cross validation log loss is: 0.0008236892036873947
```

In [31]:

```
# test data predictions

list_tup=[]
for tup in [(x,y) for x in np.unique(test['crew']) for y in np.unique(test['seat'])]:
    list_tup.append(tup)

test_id_f=[]
pred=[]
for i in range(0,18):
    test_df,test_id=test_data_FE(i,list_tup)
    test_id_f.append(test_id)
    test_df=test_df.drop(featr,axis=1)
    y_test_pred=x_cfl.predict_proba(test_df)
    pred.append(y_test_pred)
    print('---------------------------------- loop : ',i+1,' ----------------------------------')
```

```
---------------------------------- loop :  1  ----------------------------------
---------------------------------- loop :  2  ----------------------------------
---------------------------------- loop :  3  ----------------------------------
---------------------------------- loop :  4  ----------------------------------
missing gsr
---------------------------------- loop :  5  ----------------------------------
---------------------------------- loop :  6  ----------------------------------
---------------------------------- loop :  7  ----------------------------------
---------------------------------- loop :  8  ----------------------------------
---------------------------------- loop :  9  ----------------------------------
failed "gsr_amp" extraction
---------------------------------- loop :  10  ----------------------------------
---------------------------------- loop :  11  ----------------------------------
---------------------------------- loop :  12  ----------------------------------
missing gsr
---------------------------------- loop :  13  ----------------------------------
---------------------------------- loop :  14  ----------------------------------
---------------------------------- loop :  15  ----------------------------------
---------------------------------- loop :  16  ----------------------------------
---------------------------------- loop :  17  ----------------------------------
---------------------------------- loop :  18  ----------------------------------
```

In [32]:

```
test_pr=[]
tst_id=[]
for i in pred:
    for j in i:
        test_pr.append(j)

for i in test_id_f:
    for j in i:
        tst_id.append(j)

sub=pd.DataFrame(test_pr,columns=['A','B','C','D'])
sub['id']=tst_id
sub=sub[['id','A','B','C','D']]
sub=sub.sort_values(by=['id'])
sub.reset_index(inplace = True)
sub=sub.drop('index',axis=1)
sub
```

Out[32]:

| | id | A | B | C | D |
|---|---|---|---|---|---|
| **0** | 0 | 0.999687 | 0.000033 | 0.000189 | 0.000091 |

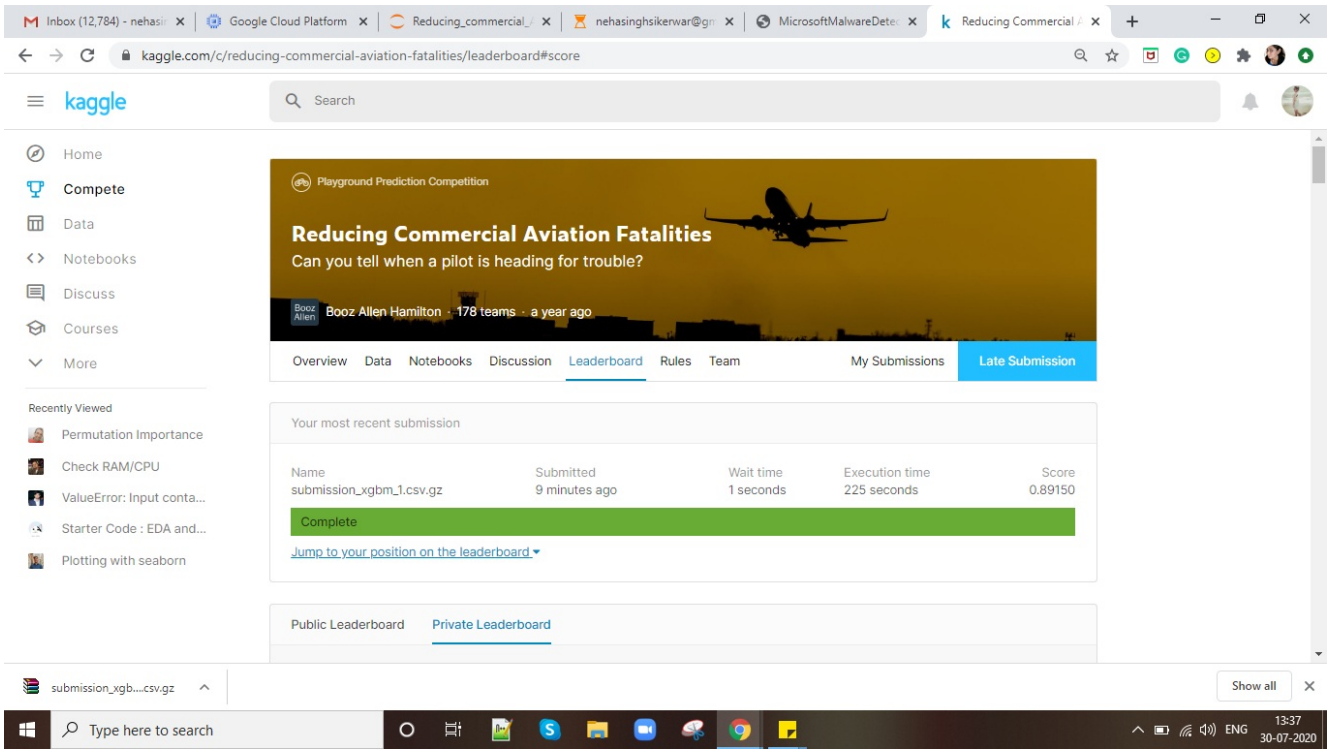
| | id | A | B | C | D |
|---|---|---|---|---|---|
| **1** | 1 | 0.863346 | 0.008624 | 0.116180 | 0.011851 |
| **2** | 2 | 0.999373 | 0.000046 | 0.000121 | 0.000461 |
| **3** | 3 | 0.781599 | 0.010629 | 0.196469 | 0.011303 |
| **4** | 4 | 0.999479 | 0.000041 | 0.000165 | 0.000314 |
| **...** | ... | ... | ... | ... | ... |
| **17965138** | 17965138 | 0.936770 | 0.060522 | 0.002649 | 0.000059 |
| **17965139** | 17965139 | 0.999461 | 0.000012 | 0.000456 | 0.000071 |
| **17965140** | 17965140 | 0.911677 | 0.084435 | 0.003824 | 0.000063 |
| **17965141** | 17965141 | 0.999461 | 0.000012 | 0.000456 | 0.000071 |
| **17965142** | 17965142 | 0.899560 | 0.095381 | 0.004958 | 0.000101 |

17965143 rows × 5 columns

In [33]:

```
sub.to_csv('submission_xgbm_1.csv.gz',index=False, compression='gzip')
```

In [ ]:




In [ ]:

## Random Forest

In [19]:

```
df=new_train.sample(90000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df=df.drop(featr,axis=1)
df.shape
train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=
```

```
32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[19]:

```
((72000, 120), (18000, 120), 72000, 18000)
```

In [9]:

```
from sklearn.model_selection import RandomizedSearchCV
from sklearn.ensemble import RandomForestClassifier

n_estimators = [int(x) for x in np.linspace(start = 50, stop = 250, num = 5)]
max_depth = [int(x) for x in np.linspace(5, 35, num = 4)]

prams={'n_estimators': n_estimators,
       'max_depth': max_depth}

rf = RandomForestClassifier(class_weight='balanced',n_jobs=-1)
rf_r = RandomizedSearchCV(estimator = rf, param_distributions = prams, verbose=2, random_state=42,
n_jobs = -1)
rf_r.fit(train_df, y_train)
print (rf_r.best_params_)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done  25 tasks      | elapsed:  5.2min
[Parallel(n_jobs=-1)]: Done  50 out of  50 | elapsed: 11.2min finished

{'n_estimators': 100, 'max_depth': 25}
```

In [7]:

```
# training with full train data

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[7]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [20]:

```
r_cfl=RandomForestClassifier(n_estimators=100, max_depth=25, random_state=42, n_jobs=-1)
r_cfl.fit(train_df,y_train)
sig_clf = CalibratedClassifierCV(r_cfl)
sig_clf.fit(train_df,y_train)

predict_y = sig_clf.predict_proba(train_df)
print ("The train log loss is:",log_loss(y_train, predict_y))
predict_y = sig_clf.predict_proba(val_df)
print("The cross validation log loss is:",log_loss(y_val, predict_y))
```

```
The train log loss is: 0.0212187718371411
The cross validation log loss is: 0.11121686718047841
```

We can see from CV loss, the performance of random forest is poor than the XGBoost. So we'll move to next model.

In [ ]:

## CatBoost

In [7]:

```
df=new_train.sample(90000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df=df.drop(featr,axis=1)
df.shape
train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=
32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[7]:

```
((72000, 120), (18000, 120), 72000, 18000)
```

In [8]:

```
from sklearn.model_selection import GridSearchCV
from catboost import CatBoostClassifier

params = {'depth': [4, 7, 10],
          'learning_rate' : [0.1,0.2,0.3],'iterations': [300]}

cb = CatBoostClassifier(loss_function='MultiClass')
cb_model = GridSearchCV(cb, params,n_jobs=-1)
cb_model.fit(train_df, y_train)
print (cb_model.best_params_)
```

```
0: learn: 1.0340681 total: 747ms remaining: 3m 43s
1: learn: 0.8532669 total: 1.41s remaining: 3m 29s
2: learn: 0.7087264 total: 2.07s remaining: 3m 25s
3: learn: 0.6285890 total: 2.76s remaining: 3m 24s
4: learn: 0.5707169 total: 3.44s remaining: 3m 23s
5: learn: 0.5282556 total: 4.14s remaining: 3m 22s
6: learn: 0.4807023 total: 4.78s remaining: 3m 20s
7: learn: 0.4397033 total: 5.47s remaining: 3m 19s
8: learn: 0.4104687 total: 6.13s remaining: 3m 18s
9: learn: 0.3857413 total: 6.82s remaining: 3m 17s
10: learn: 0.3606784 total: 7.49s remaining: 3m 16s
11: learn: 0.3458833 total: 8.17s remaining: 3m 16s
12: learn: 0.3332622 total: 8.81s remaining: 3m 14s
13: learn: 0.3200908 total: 9.49s remaining: 3m 13s
14: learn: 0.3028210 total: 10.2s remaining: 3m 13s
15: learn: 0.2893339 total: 10.8s remaining: 3m 12s
16: learn: 0.2763684 total: 11.5s remaining: 3m 12s
17: learn: 0.2652153 total: 12.2s remaining: 3m 11s
18: learn: 0.2580110 total: 12.9s remaining: 3m 10s
19: learn: 0.2547928 total: 13.5s remaining: 3m 9s
20: learn: 0.2469142 total: 14.2s remaining: 3m 9s
21: learn: 0.2417753 total: 14.9s remaining: 3m 8s
22: learn: 0.2363829 total: 15.6s remaining: 3m 7s
23: learn: 0.2336592 total: 16.3s remaining: 3m 6s
24: learn: 0.2274228 total: 16.9s remaining: 3m 6s
25: learn: 0.2223686 total: 17.6s remaining: 3m 5s
26: learn: 0.2195998 total: 18.3s remaining: 3m 4s
27: learn: 0.2121736 total: 19s remaining: 3m 4s
28: learn: 0.2083079 total: 19.6s remaining: 3m 3s
29: learn: 0.2040546 total: 20.3s remaining: 3m 2s
30: learn: 0.2013786 total: 20.9s remaining: 3m 1s
31: learn: 0.1979558 total: 21.7s remaining: 3m 1s
32: learn: 0.1932677 total: 22.4s remaining: 3m 1s
33: learn: 0.1884381 total: 23.1s remaining: 3m
34: learn: 0.1845642 total: 23.7s remaining: 2m 59s
35: learn: 0.1819596 total: 24.4s remaining: 2m 58s
36: learn: 0.1785046 total: 25.1s remaining: 2m 58s
37: learn: 0.1740233 total: 25.8s remaining: 2m 57s
38: learn: 0.1715933 total: 26.5s remaining: 2m 57s
39: learn: 0.1695901 total: 27.1s remaining: 2m 56s
40: learn: 0.1678085 total: 27.8s remaining: 2m 55s
```

```
41: learn: 0.1652125 total: 28.4s remaining: 2m 54s
42: learn: 0.1624344 total: 29.1s remaining: 2m 53s
43: learn: 0.1605123 total: 29.7s remaining: 2m 52s
44: learn: 0.1578122 total: 30.4s remaining: 2m 52s
45: learn: 0.1551998 total: 31s remaining: 2m 51s
46: learn: 0.1540020 total: 31.7s remaining: 2m 50s
47: learn: 0.1522264 total: 32.3s remaining: 2m 49s
48: learn: 0.1507860 total: 32.9s remaining: 2m 48s
49: learn: 0.1480864 total: 33.6s remaining: 2m 48s
50: learn: 0.1456744 total: 34.3s remaining: 2m 47s
51: learn: 0.1440167 total: 35s remaining: 2m 46s
52: learn: 0.1418960 total: 35.7s remaining: 2m 46s
53: learn: 0.1396485 total: 36.4s remaining: 2m 45s
54: learn: 0.1386712 total: 37s remaining: 2m 44s
55: learn: 0.1359190 total: 37.7s remaining: 2m 44s
56: learn: 0.1346345 total: 38.3s remaining: 2m 43s
57: learn: 0.1331502 total: 39s remaining: 2m 42s
58: learn: 0.1301322 total: 39.7s remaining: 2m 42s
59: learn: 0.1291297 total: 40.3s remaining: 2m 41s
60: learn: 0.1275830 total: 41s remaining: 2m 40s
61: learn: 0.1259073 total: 41.6s remaining: 2m 39s
62: learn: 0.1247600 total: 42.3s remaining: 2m 39s
63: learn: 0.1238441 total: 43s remaining: 2m 38s
64: learn: 0.1231180 total: 43.6s remaining: 2m 37s
65: learn: 0.1222836 total: 44.3s remaining: 2m 36s
66: learn: 0.1208058 total: 44.9s remaining: 2m 36s
67: learn: 0.1200368 total: 45.6s remaining: 2m 35s
68: learn: 0.1192979 total: 46.2s remaining: 2m 34s
69: learn: 0.1175874 total: 46.9s remaining: 2m 34s
70: learn: 0.1164743 total: 47.6s remaining: 2m 33s
71: learn: 0.1155759 total: 48.2s remaining: 2m 32s
72: learn: 0.1150818 total: 48.9s remaining: 2m 31s
73: learn: 0.1132850 total: 49.5s remaining: 2m 31s
74: learn: 0.1119571 total: 50.2s remaining: 2m 30s
75: learn: 0.1097258 total: 50.9s remaining: 2m 29s
76: learn: 0.1088560 total: 51.5s remaining: 2m 29s
77: learn: 0.1076653 total: 52.2s remaining: 2m 28s
78: learn: 0.1068773 total: 52.8s remaining: 2m 27s
79: learn: 0.1063708 total: 53.4s remaining: 2m 26s
80: learn: 0.1055739 total: 54.1s remaining: 2m 26s
81: learn: 0.1047789 total: 54.7s remaining: 2m 25s
82: learn: 0.1042176 total: 55.3s remaining: 2m 24s
83: learn: 0.1031500 total: 56s remaining: 2m 23s
84: learn: 0.1026080 total: 56.6s remaining: 2m 23s
85: learn: 0.1009007 total: 57.3s remaining: 2m 22s
86: learn: 0.1004171 total: 57.9s remaining: 2m 21s
87: learn: 0.0998086 total: 58.5s remaining: 2m 21s
88: learn: 0.0987988 total: 59.2s remaining: 2m 20s
89: learn: 0.0975964 total: 59.9s remaining: 2m 19s
90: learn: 0.0961524 total: 1m remaining: 2m 19s
91: learn: 0.0947028 total: 1m 1s remaining: 2m 18s
92: learn: 0.0932812 total: 1m 1s remaining: 2m 17s
93: learn: 0.0925233 total: 1m 2s remaining: 2m 16s
94: learn: 0.0918372 total: 1m 3s remaining: 2m 16s
95: learn: 0.0914026 total: 1m 3s remaining: 2m 15s
96: learn: 0.0905584 total: 1m 4s remaining: 2m 14s
97: learn: 0.0899190 total: 1m 5s remaining: 2m 14s
98: learn: 0.0886446 total: 1m 5s remaining: 2m 13s
99: learn: 0.0873952 total: 1m 6s remaining: 2m 12s
100: learn: 0.0864936 total: 1m 6s remaining: 2m 11s
101: learn: 0.0862639 total: 1m 7s remaining: 2m 11s
102: learn: 0.0854115 total: 1m 8s remaining: 2m 10s
103: learn: 0.0847002 total: 1m 8s remaining: 2m 9s
104: learn: 0.0840274 total: 1m 9s remaining: 2m 9s
105: learn: 0.0836000 total: 1m 10s remaining: 2m 8s
106: learn: 0.0833532 total: 1m 10s remaining: 2m 7s
107: learn: 0.0830171 total: 1m 11s remaining: 2m 6s
108: learn: 0.0822878 total: 1m 12s remaining: 2m 6s
109: learn: 0.0817931 total: 1m 12s remaining: 2m 5s
110: learn: 0.0812729 total: 1m 13s remaining: 2m 4s
111: learn: 0.0806310 total: 1m 13s remaining: 2m 4s
112: learn: 0.0800741 total: 1m 14s remaining: 2m 3s
113: learn: 0.0785756 total: 1m 15s remaining: 2m 2s
114: learn: 0.0781286 total: 1m 15s remaining: 2m 2s
115: learn: 0.0776335 total: 1m 16s remaining: 2m 1s
116: learn: 0.0768372 total: 1m 17s remaining: 2m
117: learn: 0.0765166 total: 1m 17s remaining: 2m
```

```
118: learn: 0.0758007 total: 1m 18s remaining: 1m 59s
119: learn: 0.0751407 total: 1m 19s remaining: 1m 58s
120: learn: 0.0745464 total: 1m 19s remaining: 1m 58s
121: learn: 0.0739734 total: 1m 20s remaining: 1m 57s
122: learn: 0.0730527 total: 1m 21s remaining: 1m 56s
123: learn: 0.0726487 total: 1m 21s remaining: 1m 56s
124: learn: 0.0722848 total: 1m 22s remaining: 1m 55s
125: learn: 0.0717871 total: 1m 23s remaining: 1m 54s
126: learn: 0.0714934 total: 1m 23s remaining: 1m 54s
127: learn: 0.0710840 total: 1m 24s remaining: 1m 53s
128: learn: 0.0705561 total: 1m 25s remaining: 1m 52s
129: learn: 0.0702921 total: 1m 25s remaining: 1m 52s
130: learn: 0.0700226 total: 1m 26s remaining: 1m 51s
131: learn: 0.0696257 total: 1m 26s remaining: 1m 50s
132: learn: 0.0691343 total: 1m 27s remaining: 1m 50s
133: learn: 0.0687620 total: 1m 28s remaining: 1m 49s
134: learn: 0.0681676 total: 1m 28s remaining: 1m 48s
135: learn: 0.0676759 total: 1m 29s remaining: 1m 48s
136: learn: 0.0669970 total: 1m 30s remaining: 1m 47s
137: learn: 0.0665534 total: 1m 30s remaining: 1m 46s
138: learn: 0.0659905 total: 1m 31s remaining: 1m 46s
139: learn: 0.0656169 total: 1m 32s remaining: 1m 45s
140: learn: 0.0653730 total: 1m 32s remaining: 1m 44s
141: learn: 0.0650452 total: 1m 33s remaining: 1m 44s
142: learn: 0.0645727 total: 1m 34s remaining: 1m 43s
143: learn: 0.0640592 total: 1m 34s remaining: 1m 42s
144: learn: 0.0633457 total: 1m 35s remaining: 1m 42s
145: learn: 0.0630824 total: 1m 36s remaining: 1m 41s
146: learn: 0.0626512 total: 1m 36s remaining: 1m 40s
147: learn: 0.0618874 total: 1m 37s remaining: 1m 40s
148: learn: 0.0616020 total: 1m 38s remaining: 1m 39s
149: learn: 0.0612850 total: 1m 38s remaining: 1m 38s
150: learn: 0.0609057 total: 1m 39s remaining: 1m 38s
151: learn: 0.0605559 total: 1m 40s remaining: 1m 37s
152: learn: 0.0600621 total: 1m 40s remaining: 1m 36s
153: learn: 0.0596850 total: 1m 41s remaining: 1m 36s
154: learn: 0.0593327 total: 1m 42s remaining: 1m 35s
155: learn: 0.0591647 total: 1m 42s remaining: 1m 34s
156: learn: 0.0587758 total: 1m 43s remaining: 1m 34s
157: learn: 0.0584597 total: 1m 43s remaining: 1m 33s
158: learn: 0.0582247 total: 1m 44s remaining: 1m 32s
159: learn: 0.0578855 total: 1m 45s remaining: 1m 32s
160: learn: 0.0577799 total: 1m 45s remaining: 1m 31s
161: learn: 0.0574852 total: 1m 46s remaining: 1m 30s
162: learn: 0.0570603 total: 1m 47s remaining: 1m 30s
163: learn: 0.0567026 total: 1m 47s remaining: 1m 29s
164: learn: 0.0558255 total: 1m 48s remaining: 1m 28s
165: learn: 0.0552977 total: 1m 49s remaining: 1m 28s
166: learn: 0.0548887 total: 1m 49s remaining: 1m 27s
167: learn: 0.0546301 total: 1m 50s remaining: 1m 26s
168: learn: 0.0538252 total: 1m 51s remaining: 1m 26s
169: learn: 0.0537069 total: 1m 51s remaining: 1m 25s
170: learn: 0.0533152 total: 1m 52s remaining: 1m 24s
171: learn: 0.0532004 total: 1m 52s remaining: 1m 24s
172: learn: 0.0528917 total: 1m 53s remaining: 1m 23s
173: learn: 0.0525943 total: 1m 54s remaining: 1m 22s
174: learn: 0.0520462 total: 1m 54s remaining: 1m 22s
175: learn: 0.0515281 total: 1m 55s remaining: 1m 21s
176: learn: 0.0512162 total: 1m 56s remaining: 1m 20s
177: learn: 0.0508970 total: 1m 56s remaining: 1m 20s
178: learn: 0.0502966 total: 1m 57s remaining: 1m 19s
179: learn: 0.0501152 total: 1m 58s remaining: 1m 18s
180: learn: 0.0497826 total: 1m 58s remaining: 1m 18s
181: learn: 0.0495112 total: 1m 59s remaining: 1m 17s
182: learn: 0.0493067 total: 1m 59s remaining: 1m 16s
183: learn: 0.0489943 total: 2m remaining: 1m 15s
184: learn: 0.0489001 total: 2m 1s remaining: 1m 15s
185: learn: 0.0486642 total: 2m 1s remaining: 1m 14s
186: learn: 0.0484738 total: 2m 2s remaining: 1m 13s
187: learn: 0.0482327 total: 2m 2s remaining: 1m 13s
188: learn: 0.0478552 total: 2m 3s remaining: 1m 12s
189: learn: 0.0477370 total: 2m 4s remaining: 1m 11s
190: learn: 0.0475783 total: 2m 4s remaining: 1m 11s
191: learn: 0.0474093 total: 2m 5s remaining: 1m 10s
192: learn: 0.0469785 total: 2m 6s remaining: 1m 9s
193: learn: 0.0468157 total: 2m 6s remaining: 1m 9s
194: learn: 0.0466890 total: 2m 7s remaining: 1m 8s
```

```
195: learn: 0.0463802 total: 2m 7s remaining: 1m 7s
196: learn: 0.0460866 total: 2m 8s remaining: 1m 7s
197: learn: 0.0458820 total: 2m 9s remaining: 1m 6s
198: learn: 0.0457354 total: 2m 9s remaining: 1m 5s
199: learn: 0.0456289 total: 2m 10s remaining: 1m 5s
200: learn: 0.0454257 total: 2m 11s remaining: 1m 4s
201: learn: 0.0452607 total: 2m 11s remaining: 1m 3s
202: learn: 0.0450315 total: 2m 12s remaining: 1m 3s
203: learn: 0.0448892 total: 2m 13s remaining: 1m 2s
204: learn: 0.0447462 total: 2m 13s remaining: 1m 2s
205: learn: 0.0444723 total: 2m 14s remaining: 1m 1s
206: learn: 0.0442571 total: 2m 15s remaining: 1m
207: learn: 0.0440199 total: 2m 15s remaining: 1m
208: learn: 0.0437212 total: 2m 16s remaining: 59.5s
209: learn: 0.0433217 total: 2m 17s remaining: 58.9s
210: learn: 0.0431359 total: 2m 18s remaining: 58.2s
211: learn: 0.0430021 total: 2m 18s remaining: 57.6s
212: learn: 0.0428896 total: 2m 19s remaining: 56.9s
213: learn: 0.0427329 total: 2m 19s remaining: 56.2s
214: learn: 0.0424798 total: 2m 20s remaining: 55.6s
215: learn: 0.0418443 total: 2m 21s remaining: 55s
216: learn: 0.0416568 total: 2m 21s remaining: 54.3s
217: learn: 0.0415419 total: 2m 22s remaining: 53.7s
218: learn: 0.0413645 total: 2m 23s remaining: 53s
219: learn: 0.0410642 total: 2m 23s remaining: 52.4s
220: learn: 0.0408001 total: 2m 24s remaining: 51.7s
221: learn: 0.0406167 total: 2m 25s remaining: 51s
222: learn: 0.0400951 total: 2m 25s remaining: 50.4s
223: learn: 0.0399889 total: 2m 26s remaining: 49.7s
224: learn: 0.0399040 total: 2m 27s remaining: 49.1s
225: learn: 0.0397859 total: 2m 27s remaining: 48.4s
226: learn: 0.0395386 total: 2m 28s remaining: 47.8s
227: learn: 0.0393600 total: 2m 29s remaining: 47.1s
228: learn: 0.0391394 total: 2m 29s remaining: 46.4s
229: learn: 0.0390178 total: 2m 30s remaining: 45.8s
230: learn: 0.0383856 total: 2m 31s remaining: 45.1s
231: learn: 0.0380752 total: 2m 31s remaining: 44.5s
232: learn: 0.0379591 total: 2m 32s remaining: 43.8s
233: learn: 0.0376731 total: 2m 33s remaining: 43.2s
234: learn: 0.0375237 total: 2m 33s remaining: 42.5s
235: learn: 0.0373767 total: 2m 34s remaining: 41.9s
236: learn: 0.0372553 total: 2m 35s remaining: 41.2s
237: learn: 0.0369487 total: 2m 35s remaining: 40.6s
238: learn: 0.0368533 total: 2m 36s remaining: 39.9s
239: learn: 0.0367489 total: 2m 37s remaining: 39.3s
240: learn: 0.0365693 total: 2m 37s remaining: 38.6s
241: learn: 0.0363909 total: 2m 38s remaining: 38s
242: learn: 0.0363222 total: 2m 39s remaining: 37.3s
243: learn: 0.0360775 total: 2m 39s remaining: 36.6s
244: learn: 0.0359338 total: 2m 40s remaining: 36s
245: learn: 0.0358486 total: 2m 40s remaining: 35.3s
246: learn: 0.0356533 total: 2m 41s remaining: 34.7s
247: learn: 0.0354500 total: 2m 42s remaining: 34s
248: learn: 0.0352592 total: 2m 42s remaining: 33.4s
249: learn: 0.0352140 total: 2m 43s remaining: 32.7s
250: learn: 0.0351544 total: 2m 44s remaining: 32.1s
251: learn: 0.0348104 total: 2m 44s remaining: 31.4s
252: learn: 0.0345677 total: 2m 45s remaining: 30.8s
253: learn: 0.0344107 total: 2m 46s remaining: 30.1s
254: learn: 0.0342395 total: 2m 46s remaining: 29.5s
255: learn: 0.0340383 total: 2m 47s remaining: 28.8s
256: learn: 0.0338879 total: 2m 48s remaining: 28.2s
257: learn: 0.0337561 total: 2m 48s remaining: 27.5s
258: learn: 0.0336204 total: 2m 49s remaining: 26.8s
259: learn: 0.0335238 total: 2m 50s remaining: 26.2s
260: learn: 0.0332780 total: 2m 50s remaining: 25.5s
261: learn: 0.0331537 total: 2m 51s remaining: 24.9s
262: learn: 0.0330251 total: 2m 52s remaining: 24.2s
263: learn: 0.0328728 total: 2m 52s remaining: 23.6s
264: learn: 0.0327671 total: 2m 53s remaining: 22.9s
265: learn: 0.0326428 total: 2m 54s remaining: 22.3s
266: learn: 0.0325795 total: 2m 54s remaining: 21.6s
267: learn: 0.0324696 total: 2m 55s remaining: 21s
268: learn: 0.0322839 total: 2m 56s remaining: 20.3s
269: learn: 0.0321324 total: 2m 56s remaining: 19.7s
270: learn: 0.0319474 total: 2m 57s remaining: 19s
271: learn: 0.0318978 total: 2m 58s remaining: 18.4s
```

```
272: learn: 0.0316608 total: 2m 58s remaining: 17.7s
273: learn: 0.0315594 total: 2m 59s remaining: 17s
274: learn: 0.0313788 total: 3m remaining: 16.4s
275: learn: 0.0312432 total: 3m remaining: 15.7s
276: learn: 0.0311299 total: 3m 1s remaining: 15.1s
277: learn: 0.0308407 total: 3m 2s remaining: 14.4s
278: learn: 0.0306169 total: 3m 3s remaining: 13.8s
279: learn: 0.0305443 total: 3m 3s remaining: 13.1s
280: learn: 0.0304559 total: 3m 4s remaining: 12.5s
281: learn: 0.0303941 total: 3m 5s remaining: 11.8s
282: learn: 0.0303080 total: 3m 5s remaining: 11.2s
283: learn: 0.0301237 total: 3m 6s remaining: 10.5s
284: learn: 0.0300263 total: 3m 7s remaining: 9.85s
285: learn: 0.0299049 total: 3m 7s remaining: 9.19s
286: learn: 0.0297793 total: 3m 8s remaining: 8.54s
287: learn: 0.0295841 total: 3m 9s remaining: 7.88s
288: learn: 0.0294255 total: 3m 9s remaining: 7.22s
289: learn: 0.0292444 total: 3m 10s remaining: 6.57s
290: learn: 0.0290769 total: 3m 11s remaining: 5.91s
291: learn: 0.0288859 total: 3m 11s remaining: 5.25s
292: learn: 0.0287597 total: 3m 12s remaining: 4.6s
293: learn: 0.0287075 total: 3m 13s remaining: 3.94s
294: learn: 0.0286383 total: 3m 13s remaining: 3.28s
295: learn: 0.0285102 total: 3m 14s remaining: 2.63s
296: learn: 0.0284418 total: 3m 15s remaining: 1.97s
297: learn: 0.0283365 total: 3m 15s remaining: 1.31s
298: learn: 0.0282683 total: 3m 16s remaining: 657ms
299: learn: 0.0281645 total: 3m 17s remaining: 0us
{'learning_rate': 0.3, 'iterations': 300, 'depth': 10}
```

In [25]:

```
# use_best_model=True, eval_set=eval_dataset to find best model iterations

from catboost import Pool
train_dataset = Pool(data=train_df,
                     label=y_train)

eval_dataset = Pool(data=val_df,
                    label=y_val)

clf = CatBoostClassifier(loss_function='MultiClass', depth=10, iterations=1000, l2_leaf_reg= 1, lea
rning_rate= 0.38)
clf.fit(train_dataset, use_best_model=True, eval_set=eval_dataset)
print("Count of trees in model = {}".format(clf.tree_count_))


y_predict=clf.predict_proba(train_df)
print('TRAIN LOSS is ',log_loss(y_train, y_predict))
y_predict=clf.predict_proba(val_df)
print('CV LOSS is ',log_loss(y_val, y_predict))
```

```
0: learn: 0.9584757 test: 0.9537488 best: 0.9537488 (0) total: 610ms remaining: 10m 9s
1: learn: 0.7440640 test: 0.7410102 best: 0.7410102 (1) total: 1.23s remaining: 10m 12s
2: learn: 0.6120653 test: 0.6079750 best: 0.6079750 (2) total: 1.84s remaining: 10m 12s
3: learn: 0.5360013 test: 0.5341823 best: 0.5341823 (3) total: 2.48s remaining: 10m 16s
4: learn: 0.4817079 test: 0.4807530 best: 0.4807530 (4) total: 3.06s remaining: 10m 9s
5: learn: 0.4405264 test: 0.4403896 best: 0.4403896 (5) total: 3.68s remaining: 10m 9s
6: learn: 0.4079496 test: 0.4082326 best: 0.4082326 (6) total: 4.25s remaining: 10m 2s
7: learn: 0.3757881 test: 0.3767762 best: 0.3767762 (7) total: 4.85s remaining: 10m 1s
8: learn: 0.3525693 test: 0.3540374 best: 0.3540374 (8) total: 5.43s remaining: 9m 58s
9: learn: 0.3296255 test: 0.3322832 best: 0.3322832 (9) total: 6.05s remaining: 9m 59s
10: learn: 0.3062696 test: 0.3105304 best: 0.3105304 (10) total: 6.69s remaining: 10m 1s
11: learn: 0.2922034 test: 0.2964209 best: 0.2964209 (11) total: 7.29s remaining: 9m 59s
12: learn: 0.2825968 test: 0.2875000 best: 0.2875000 (12) total: 7.88s remaining: 9m 58s
13: learn: 0.2714159 test: 0.2775350 best: 0.2775350 (13) total: 8.49s remaining: 9m 58s
14: learn: 0.2564127 test: 0.2624707 best: 0.2624707 (14) total: 9.14s remaining: 10m
15: learn: 0.2460939 test: 0.2540584 best: 0.2540584 (15) total: 9.78s remaining: 10m 1s
16: learn: 0.2377272 test: 0.2471330 best: 0.2471330 (16) total: 10.4s remaining: 10m
17: learn: 0.2300314 test: 0.2400716 best: 0.2400716 (17) total: 11s remaining: 10m
18: learn: 0.2243074 test: 0.2353800 best: 0.2353800 (18) total: 11.6s remaining: 10m
19: learn: 0.2182641 test: 0.2304641 best: 0.2304641 (19) total: 12.3s remaining: 10m 1s
20: learn: 0.2126187 test: 0.2264208 best: 0.2264208 (20) total: 12.9s remaining: 10m 1s
21: learn: 0.2075004 test: 0.2221811 best: 0.2221811 (21) total: 13.5s remaining: 10m 1s
22: learn: 0.2015226 test: 0.2169849 best: 0.2169849 (22) total: 14.2s remaining: 10m 2s
23: learn: 0.1979621 test: 0.2137725 best: 0.2137725 (23) total: 14.8s remaining: 10m
```

```
23: learn: 0.1979621 test: 0.2137725 best: 0.2137725 (23) total: 14.8s remaining: 10m
24: learn: 0.1903498 test: 0.2067229 best: 0.2067229 (24) total: 15.4s remaining: 10m
25: learn: 0.1873977 test: 0.2047189 best: 0.2047189 (25) total: 16s remaining: 9m 59s
26: learn: 0.1839042 test: 0.2024309 best: 0.2024309 (26) total: 16.6s remaining: 9m 59s
27: learn: 0.1762022 test: 0.1952341 best: 0.1952341 (27) total: 17.3s remaining: 10m
28: learn: 0.1744327 test: 0.1941682 best: 0.1941682 (28) total: 17.9s remaining: 9m 59s
29: learn: 0.1706046 test: 0.1917344 best: 0.1917344 (29) total: 18.5s remaining: 9m 59s
30: learn: 0.1666473 test: 0.1880766 best: 0.1880766 (30) total: 19.2s remaining: 9m 59s
31: learn: 0.1625266 test: 0.1845458 best: 0.1845458 (31) total: 19.8s remaining: 9m 59s
32: learn: 0.1609568 test: 0.1835506 best: 0.1835506 (32) total: 20.4s remaining: 9m 57s
33: learn: 0.1585640 test: 0.1816986 best: 0.1816986 (33) total: 21s remaining: 9m 57s
34: learn: 0.1528394 test: 0.1760550 best: 0.1760550 (34) total: 21.7s remaining: 9m 57s
35: learn: 0.1478680 test: 0.1712399 best: 0.1712399 (35) total: 22.3s remaining: 9m 57s
36: learn: 0.1454497 test: 0.1697605 best: 0.1697605 (36) total: 23s remaining: 9m 57s
37: learn: 0.1441945 test: 0.1691008 best: 0.1691008 (37) total: 23.6s remaining: 9m 56s
38: learn: 0.1427037 test: 0.1679732 best: 0.1679732 (38) total: 24.1s remaining: 9m 54s
39: learn: 0.1378663 test: 0.1629275 best: 0.1629275 (39) total: 24.8s remaining: 9m 54s
40: learn: 0.1361971 test: 0.1616576 best: 0.1616576 (40) total: 25.4s remaining: 9m 53s
41: learn: 0.1340635 test: 0.1601605 best: 0.1601605 (41) total: 26s remaining: 9m 53s
42: learn: 0.1306208 test: 0.1567786 best: 0.1567786 (42) total: 26.6s remaining: 9m 52s
43: learn: 0.1286337 test: 0.1554842 best: 0.1554842 (43) total: 27.3s remaining: 9m 52s
44: learn: 0.1245325 test: 0.1513022 best: 0.1513022 (44) total: 27.9s remaining: 9m 52s
45: learn: 0.1209484 test: 0.1479929 best: 0.1479929 (45) total: 28.6s remaining: 9m 52s
46: learn: 0.1174480 test: 0.1448467 best: 0.1448467 (46) total: 29.2s remaining: 9m 52s
47: learn: 0.1147345 test: 0.1425338 best: 0.1425338 (47) total: 29.9s remaining: 9m 52s
48: learn: 0.1132578 test: 0.1415757 best: 0.1415757 (48) total: 30.5s remaining: 9m 51s
49: learn: 0.1118296 test: 0.1401237 best: 0.1401237 (49) total: 31.1s remaining: 9m 50s
50: learn: 0.1102600 test: 0.1392489 best: 0.1392489 (50) total: 31.7s remaining: 9m 50s
51: learn: 0.1084975 test: 0.1378044 best: 0.1378044 (51) total: 32.4s remaining: 9m 50s
52: learn: 0.1064121 test: 0.1357870 best: 0.1357870 (52) total: 33s remaining: 9m 49s
53: learn: 0.1025375 test: 0.1316326 best: 0.1316326 (53) total: 33.7s remaining: 9m 49s
54: learn: 0.1002445 test: 0.1298740 best: 0.1298740 (54) total: 34.3s remaining: 9m 49s
55: learn: 0.0987099 test: 0.1289888 best: 0.1289888 (55) total: 35s remaining: 9m 49s
56: learn: 0.0973070 test: 0.1280988 best: 0.1280988 (56) total: 35.6s remaining: 9m 48s
57: learn: 0.0966449 test: 0.1275215 best: 0.1275215 (57) total: 36.2s remaining: 9m 47s
58: learn: 0.0953791 test: 0.1267580 best: 0.1267580 (58) total: 36.8s remaining: 9m 47s
59: learn: 0.0946365 test: 0.1262256 best: 0.1262256 (59) total: 37.4s remaining: 9m 46s
60: learn: 0.0937397 test: 0.1257294 best: 0.1257294 (60) total: 38s remaining: 9m 45s
61: learn: 0.0924773 test: 0.1247620 best: 0.1247620 (61) total: 38.7s remaining: 9m 44s
62: learn: 0.0910397 test: 0.1239134 best: 0.1239134 (62) total: 39.3s remaining: 9m 44s
63: learn: 0.0898876 test: 0.1232168 best: 0.1232168 (63) total: 40s remaining: 9m 44s
64: learn: 0.0884311 test: 0.1223364 best: 0.1223364 (64) total: 40.6s remaining: 9m 44s
65: learn: 0.0877708 test: 0.1220017 best: 0.1220017 (65) total: 41.3s remaining: 9m 43s
66: learn: 0.0870907 test: 0.1216985 best: 0.1216985 (66) total: 41.9s remaining: 9m 43s
67: learn: 0.0858302 test: 0.1206465 best: 0.1206465 (67) total: 42.5s remaining: 9m 42s
68: learn: 0.0839809 test: 0.1195320 best: 0.1195320 (68) total: 43.2s remaining: 9m 42s
69: learn: 0.0827110 test: 0.1185754 best: 0.1185754 (69) total: 43.8s remaining: 9m 42s
70: learn: 0.0817687 test: 0.1176947 best: 0.1176947 (70) total: 44.4s remaining: 9m 41s
71: learn: 0.0804398 test: 0.1164170 best: 0.1164170 (71) total: 45.1s remaining: 9m 41s
72: learn: 0.0793474 test: 0.1152673 best: 0.1152673 (72) total: 45.7s remaining: 9m 40s
73: learn: 0.0781354 test: 0.1140471 best: 0.1140471 (73) total: 46.3s remaining: 9m 39s
74: learn: 0.0767910 test: 0.1127490 best: 0.1127490 (74) total: 47s remaining: 9m 39s
75: learn: 0.0750222 test: 0.1116319 best: 0.1116319 (75) total: 47.6s remaining: 9m 39s
76: learn: 0.0742574 test: 0.1112448 best: 0.1112448 (76) total: 48.3s remaining: 9m 38s
77: learn: 0.0727896 test: 0.1101493 best: 0.1101493 (77) total: 48.9s remaining: 9m 38s
78: learn: 0.0717496 test: 0.1092938 best: 0.1092938 (78) total: 49.6s remaining: 9m 38s
79: learn: 0.0712171 test: 0.1090868 best: 0.1090868 (79) total: 50.2s remaining: 9m 37s
80: learn: 0.0705194 test: 0.1085532 best: 0.1085532 (80) total: 50.8s remaining: 9m 36s
81: learn: 0.0697817 test: 0.1082444 best: 0.1082444 (81) total: 51.4s remaining: 9m 35s
82: learn: 0.0689116 test: 0.1075944 best: 0.1075944 (82) total: 52.1s remaining: 9m 35s
83: learn: 0.0680716 test: 0.1067832 best: 0.1067832 (83) total: 52.7s remaining: 9m 34s
84: learn: 0.0673015 test: 0.1062977 best: 0.1062977 (84) total: 53.4s remaining: 9m 34s
85: learn: 0.0654145 test: 0.1044055 best: 0.1044055 (85) total: 54s remaining: 9m 34s
86: learn: 0.0644853 test: 0.1034652 best: 0.1034652 (86) total: 54.7s remaining: 9m 34s
87: learn: 0.0634747 test: 0.1027154 best: 0.1027154 (87) total: 55.3s remaining: 9m 33s
88: learn: 0.0623420 test: 0.1017073 best: 0.1017073 (88) total: 56s remaining: 9m 33s
89: learn: 0.0612073 test: 0.1008066 best: 0.1008066 (89) total: 56.6s remaining: 9m 32s
90: learn: 0.0605494 test: 0.1003502 best: 0.1003502 (90) total: 57.3s remaining: 9m 31s
91: learn: 0.0599380 test: 0.0998478 best: 0.0998478 (91) total: 57.9s remaining: 9m 31s
92: learn: 0.0595426 test: 0.0995534 best: 0.0995534 (92) total: 58.4s remaining: 9m 30s
93: learn: 0.0586162 test: 0.0988422 best: 0.0988422 (93) total: 59.1s remaining: 9m 29s
94: learn: 0.0573965 test: 0.0976816 best: 0.0976816 (94) total: 59.8s remaining: 9m 29s
95: learn: 0.0569804 test: 0.0975225 best: 0.0975225 (95) total: 1m remaining: 9m 28s
96: learn: 0.0564981 test: 0.0969933 best: 0.0969933 (96) total: 1m remaining: 9m 27s
97: learn: 0.0558232 test: 0.0963281 best: 0.0963281 (97) total: 1m 1s remaining: 9m 26s
98: learn: 0.0552799 test: 0.0959379 best: 0.0959379 (98) total: 1m 2s remaining: 9m 26s
99: learn: 0.0546133 test: 0.0953935 best: 0.0953935 (99) total: 1m 2s remaining: 9m 25s
100: learn: 0.0538024 test: 0.0949640 best: 0.0949640 (100) total: 1m 3s remaining: 9m 25s
```

```
100: learn: 0.0538924 test: 0.0949649 best: 0.0949649 (100) total: 1m 3s remaining: 9m 25s
101: learn: 0.0531300 test: 0.0944047 best: 0.0944047 (101) total: 1m 4s remaining: 9m 24s
102: learn: 0.0525556 test: 0.0940218 best: 0.0940218 (102) total: 1m 4s remaining: 9m 24s
103: learn: 0.0521115 test: 0.0937419 best: 0.0937419 (103) total: 1m 5s remaining: 9m 23s
104: learn: 0.0517264 test: 0.0934660 best: 0.0934660 (104) total: 1m 6s remaining: 9m 23s
105: learn: 0.0513875 test: 0.0933140 best: 0.0933140 (105) total: 1m 6s remaining: 9m 22s
106: learn: 0.0505160 test: 0.0923104 best: 0.0923104 (106) total: 1m 7s remaining: 9m 21s
107: learn: 0.0499760 test: 0.0921362 best: 0.0921362 (107) total: 1m 7s remaining: 9m 21s
108: learn: 0.0497678 test: 0.0919589 best: 0.0919589 (108) total: 1m 8s remaining: 9m 20s
109: learn: 0.0491979 test: 0.0914891 best: 0.0914891 (109) total: 1m 9s remaining: 9m 19s
110: learn: 0.0487668 test: 0.0911101 best: 0.0911101 (110) total: 1m 9s remaining: 9m 19s
111: learn: 0.0484753 test: 0.0908721 best: 0.0908721 (111) total: 1m 10s remaining: 9m 18s
112: learn: 0.0480841 test: 0.0903918 best: 0.0903918 (112) total: 1m 11s remaining: 9m 18s
113: learn: 0.0473233 test: 0.0899587 best: 0.0899587 (113) total: 1m 11s remaining: 9m 17s
114: learn: 0.0468212 test: 0.0893913 best: 0.0893913 (114) total: 1m 12s remaining: 9m 17s
115: learn: 0.0463210 test: 0.0890784 best: 0.0890784 (115) total: 1m 13s remaining: 9m 16s
116: learn: 0.0455009 test: 0.0883547 best: 0.0883547 (116) total: 1m 13s remaining: 9m 16s
117: learn: 0.0450405 test: 0.0879591 best: 0.0879591 (117) total: 1m 14s remaining: 9m 15s
118: learn: 0.0445375 test: 0.0873049 best: 0.0873049 (118) total: 1m 15s remaining: 9m 15s
119: learn: 0.0439869 test: 0.0874121 best: 0.0873049 (118) total: 1m 15s remaining: 9m 15s
120: learn: 0.0437974 test: 0.0873730 best: 0.0873049 (118) total: 1m 16s remaining: 9m 14s
121: learn: 0.0435110 test: 0.0871647 best: 0.0871647 (121) total: 1m 16s remaining: 9m 13s
122: learn: 0.0433087 test: 0.0868980 best: 0.0868980 (122) total: 1m 17s remaining: 9m 12s
123: learn: 0.0428145 test: 0.0865727 best: 0.0865727 (123) total: 1m 18s remaining: 9m 12s
124: learn: 0.0426268 test: 0.0864511 best: 0.0864511 (124) total: 1m 18s remaining: 9m 11s
125: learn: 0.0422355 test: 0.0861990 best: 0.0861990 (125) total: 1m 19s remaining: 9m 11s
126: learn: 0.0419915 test: 0.0860416 best: 0.0860416 (126) total: 1m 20s remaining: 9m 10s
127: learn: 0.0415267 test: 0.0857301 best: 0.0857301 (127) total: 1m 20s remaining: 9m 9s
128: learn: 0.0412354 test: 0.0854841 best: 0.0854841 (128) total: 1m 21s remaining: 9m 9s
129: learn: 0.0408743 test: 0.0852415 best: 0.0852415 (129) total: 1m 21s remaining: 9m 8s
130: learn: 0.0406220 test: 0.0852034 best: 0.0852034 (130) total: 1m 22s remaining: 9m 8s
131: learn: 0.0403189 test: 0.0849571 best: 0.0849571 (131) total: 1m 23s remaining: 9m 7s
132: learn: 0.0398223 test: 0.0845497 best: 0.0845497 (132) total: 1m 23s remaining: 9m 6s
133: learn: 0.0394431 test: 0.0841908 best: 0.0841908 (133) total: 1m 24s remaining: 9m 6s
134: learn: 0.0390372 test: 0.0837474 best: 0.0837474 (134) total: 1m 25s remaining: 9m 5s
135: learn: 0.0387228 test: 0.0834852 best: 0.0834852 (135) total: 1m 25s remaining: 9m 5s
136: learn: 0.0382570 test: 0.0829674 best: 0.0829674 (136) total: 1m 26s remaining: 9m 4s
137: learn: 0.0374625 test: 0.0817955 best: 0.0817955 (137) total: 1m 27s remaining: 9m 4s
138: learn: 0.0371008 test: 0.0815543 best: 0.0815543 (138) total: 1m 27s remaining: 9m 3s
139: learn: 0.0367487 test: 0.0813198 best: 0.0813198 (139) total: 1m 28s remaining: 9m 3s
140: learn: 0.0362147 test: 0.0807060 best: 0.0807060 (140) total: 1m 29s remaining: 9m 2s
141: learn: 0.0358073 test: 0.0805022 best: 0.0805022 (141) total: 1m 29s remaining: 9m 2s
142: learn: 0.0351532 test: 0.0798362 best: 0.0798362 (142) total: 1m 30s remaining: 9m 1s
143: learn: 0.0346042 test: 0.0794018 best: 0.0794018 (143) total: 1m 31s remaining: 9m 1s
144: learn: 0.0343475 test: 0.0791861 best: 0.0791861 (144) total: 1m 31s remaining: 9m
145: learn: 0.0340705 test: 0.0788052 best: 0.0788052 (145) total: 1m 32s remaining: 8m 59s
146: learn: 0.0338347 test: 0.0786451 best: 0.0786451 (146) total: 1m 32s remaining: 8m 59s
147: learn: 0.0336490 test: 0.0784341 best: 0.0784341 (147) total: 1m 33s remaining: 8m 58s
148: learn: 0.0333084 test: 0.0782616 best: 0.0782616 (148) total: 1m 34s remaining: 8m 57s
149: learn: 0.0331176 test: 0.0782690 best: 0.0782616 (148) total: 1m 34s remaining: 8m 57s
150: learn: 0.0328443 test: 0.0781074 best: 0.0781074 (150) total: 1m 35s remaining: 8m 56s
151: learn: 0.0325646 test: 0.0777683 best: 0.0777683 (151) total: 1m 36s remaining: 8m 55s
152: learn: 0.0323519 test: 0.0776092 best: 0.0776092 (152) total: 1m 36s remaining: 8m 55s
153: learn: 0.0321802 test: 0.0776483 best: 0.0776092 (152) total: 1m 37s remaining: 8m 54s
154: learn: 0.0319109 test: 0.0775099 best: 0.0775099 (154) total: 1m 37s remaining: 8m 54s
155: learn: 0.0317105 test: 0.0774074 best: 0.0774074 (155) total: 1m 38s remaining: 8m 53s
156: learn: 0.0311133 test: 0.0766579 best: 0.0766579 (156) total: 1m 39s remaining: 8m 53s
157: learn: 0.0309784 test: 0.0766161 best: 0.0766161 (157) total: 1m 39s remaining: 8m 52s
158: learn: 0.0305732 test: 0.0761414 best: 0.0761414 (158) total: 1m 40s remaining: 8m 52s
159: learn: 0.0300851 test: 0.0758191 best: 0.0758191 (159) total: 1m 41s remaining: 8m 51s
160: learn: 0.0299859 test: 0.0757375 best: 0.0757375 (160) total: 1m 41s remaining: 8m 50s
161: learn: 0.0297064 test: 0.0754042 best: 0.0754042 (161) total: 1m 42s remaining: 8m 50s
162: learn: 0.0295228 test: 0.0752234 best: 0.0752234 (162) total: 1m 43s remaining: 8m 49s
163: learn: 0.0291508 test: 0.0747067 best: 0.0747067 (163) total: 1m 43s remaining: 8m 49s
164: learn: 0.0286822 test: 0.0741455 best: 0.0741455 (164) total: 1m 44s remaining: 8m 48s
165: learn: 0.0282226 test: 0.0736422 best: 0.0736422 (165) total: 1m 45s remaining: 8m 48s
166: learn: 0.0278648 test: 0.0732618 best: 0.0732618 (166) total: 1m 45s remaining: 8m 47s
167: learn: 0.0274787 test: 0.0729289 best: 0.0729289 (167) total: 1m 46s remaining: 8m 46s
168: learn: 0.0272977 test: 0.0727825 best: 0.0727825 (168) total: 1m 47s remaining: 8m 46s
169: learn: 0.0270698 test: 0.0726483 best: 0.0726483 (169) total: 1m 47s remaining: 8m 45s
170: learn: 0.0267947 test: 0.0723678 best: 0.0723678 (170) total: 1m 48s remaining: 8m 45s
171: learn: 0.0265947 test: 0.0722262 best: 0.0722262 (171) total: 1m 49s remaining: 8m 44s
172: learn: 0.0263196 test: 0.0718105 best: 0.0718105 (172) total: 1m 49s remaining: 8m 44s
173: learn: 0.0261661 test: 0.0717194 best: 0.0717194 (173) total: 1m 50s remaining: 8m 43s
174: learn: 0.0260412 test: 0.0715558 best: 0.0715558 (174) total: 1m 50s remaining: 8m 43s
175: learn: 0.0258110 test: 0.0714301 best: 0.0714301 (175) total: 1m 51s remaining: 8m 42s
176: learn: 0.0255501 test: 0.0713389 best: 0.0713389 (176) total: 1m 52s remaining: 8m 42s
```

```
177: learn: 0.0254511 test: 0.0712751 best: 0.0712751 (177) total: 1m 52s remaining: 8m 41s
178: learn: 0.0252943 test: 0.0712935 best: 0.0712751 (177) total: 1m 53s remaining: 8m 40s
179: learn: 0.0251330 test: 0.0711361 best: 0.0711361 (179) total: 1m 54s remaining: 8m 40s
180: learn: 0.0250119 test: 0.0710821 best: 0.0710821 (180) total: 1m 54s remaining: 8m 39s
181: learn: 0.0247958 test: 0.0708784 best: 0.0708784 (181) total: 1m 55s remaining: 8m 39s
182: learn: 0.0246709 test: 0.0708446 best: 0.0708446 (182) total: 1m 56s remaining: 8m 38s
183: learn: 0.0244895 test: 0.0706759 best: 0.0706759 (183) total: 1m 56s remaining: 8m 38s
184: learn: 0.0241298 test: 0.0702615 best: 0.0702615 (184) total: 1m 57s remaining: 8m 37s
185: learn: 0.0240132 test: 0.0701423 best: 0.0701423 (185) total: 1m 58s remaining: 8m 36s
186: learn: 0.0238177 test: 0.0700804 best: 0.0700804 (186) total: 1m 58s remaining: 8m 36s
187: learn: 0.0234758 test: 0.0696181 best: 0.0696181 (187) total: 1m 59s remaining: 8m 36s
188: learn: 0.0233196 test: 0.0694372 best: 0.0694372 (188) total: 2m remaining: 8m 35s
189: learn: 0.0231235 test: 0.0692066 best: 0.0692066 (189) total: 2m remaining: 8m 35s
190: learn: 0.0230673 test: 0.0691611 best: 0.0691611 (190) total: 2m 1s remaining: 8m 34s
191: learn: 0.0229215 test: 0.0691450 best: 0.0691450 (191) total: 2m 2s remaining: 8m 33s
192: learn: 0.0227670 test: 0.0691204 best: 0.0691204 (192) total: 2m 2s remaining: 8m 33s
193: learn: 0.0225930 test: 0.0689760 best: 0.0689760 (193) total: 2m 3s remaining: 8m 32s
194: learn: 0.0224311 test: 0.0687059 best: 0.0687059 (194) total: 2m 4s remaining: 8m 32s
195: learn: 0.0221749 test: 0.0685009 best: 0.0685009 (195) total: 2m 4s remaining: 8m 31s
196: learn: 0.0220717 test: 0.0684481 best: 0.0684481 (196) total: 2m 5s remaining: 8m 31s
197: learn: 0.0217830 test: 0.0681019 best: 0.0681019 (197) total: 2m 6s remaining: 8m 30s
198: learn: 0.0216800 test: 0.0679841 best: 0.0679841 (198) total: 2m 6s remaining: 8m 29s
199: learn: 0.0215276 test: 0.0679237 best: 0.0679237 (199) total: 2m 7s remaining: 8m 29s
200: learn: 0.0212219 test: 0.0678534 best: 0.0678534 (200) total: 2m 8s remaining: 8m 28s
201: learn: 0.0210993 test: 0.0678678 best: 0.0678534 (200) total: 2m 8s remaining: 8m 28s
202: learn: 0.0208729 test: 0.0676189 best: 0.0676189 (202) total: 2m 9s remaining: 8m 27s
203: learn: 0.0205970 test: 0.0673284 best: 0.0673284 (203) total: 2m 9s remaining: 8m 27s
204: learn: 0.0202796 test: 0.0668085 best: 0.0668085 (204) total: 2m 10s remaining: 8m 26s
205: learn: 0.0201467 test: 0.0667683 best: 0.0667683 (205) total: 2m 11s remaining: 8m 26s
206: learn: 0.0200917 test: 0.0667000 best: 0.0667000 (206) total: 2m 11s remaining: 8m 25s
207: learn: 0.0197744 test: 0.0663019 best: 0.0663019 (207) total: 2m 12s remaining: 8m 24s
208: learn: 0.0197217 test: 0.0662747 best: 0.0662747 (208) total: 2m 13s remaining: 8m 24s
209: learn: 0.0195236 test: 0.0661506 best: 0.0661506 (209) total: 2m 13s remaining: 8m 23s
210: learn: 0.0194399 test: 0.0661020 best: 0.0661020 (210) total: 2m 14s remaining: 8m 23s
211: learn: 0.0193422 test: 0.0659870 best: 0.0659870 (211) total: 2m 15s remaining: 8m 22s
212: learn: 0.0192492 test: 0.0659660 best: 0.0659660 (212) total: 2m 15s remaining: 8m 22s
213: learn: 0.0190766 test: 0.0658018 best: 0.0658018 (213) total: 2m 16s remaining: 8m 21s
214: learn: 0.0188675 test: 0.0655087 best: 0.0655087 (214) total: 2m 17s remaining: 8m 21s
215: learn: 0.0187467 test: 0.0653450 best: 0.0653450 (215) total: 2m 17s remaining: 8m 20s
216: learn: 0.0186031 test: 0.0652615 best: 0.0652615 (216) total: 2m 18s remaining: 8m 19s
217: learn: 0.0185201 test: 0.0652359 best: 0.0652359 (217) total: 2m 19s remaining: 8m 19s
218: learn: 0.0184233 test: 0.0651678 best: 0.0651678 (218) total: 2m 19s remaining: 8m 18s
219: learn: 0.0183141 test: 0.0650356 best: 0.0650356 (219) total: 2m 20s remaining: 8m 18s
220: learn: 0.0182013 test: 0.0649354 best: 0.0649354 (220) total: 2m 21s remaining: 8m 17s
221: learn: 0.0180337 test: 0.0648155 best: 0.0648155 (221) total: 2m 21s remaining: 8m 16s
222: learn: 0.0178172 test: 0.0646900 best: 0.0646900 (222) total: 2m 22s remaining: 8m 16s
223: learn: 0.0176752 test: 0.0646098 best: 0.0646098 (223) total: 2m 23s remaining: 8m 16s
224: learn: 0.0175299 test: 0.0643203 best: 0.0643203 (224) total: 2m 23s remaining: 8m 15s
225: learn: 0.0174339 test: 0.0643083 best: 0.0643083 (225) total: 2m 24s remaining: 8m 15s
226: learn: 0.0173100 test: 0.0641870 best: 0.0641870 (226) total: 2m 25s remaining: 8m 14s
227: learn: 0.0171512 test: 0.0641196 best: 0.0641196 (227) total: 2m 25s remaining: 8m 14s
228: learn: 0.0170949 test: 0.0640359 best: 0.0640359 (228) total: 2m 26s remaining: 8m 13s
229: learn: 0.0168972 test: 0.0638588 best: 0.0638588 (229) total: 2m 27s remaining: 8m 13s
230: learn: 0.0166481 test: 0.0634870 best: 0.0634870 (230) total: 2m 27s remaining: 8m 12s
231: learn: 0.0164905 test: 0.0633898 best: 0.0633898 (231) total: 2m 28s remaining: 8m 12s
232: learn: 0.0163979 test: 0.0633716 best: 0.0633716 (232) total: 2m 29s remaining: 8m 11s
233: learn: 0.0162981 test: 0.0632637 best: 0.0632637 (233) total: 2m 29s remaining: 8m 10s
234: learn: 0.0161982 test: 0.0632670 best: 0.0632637 (233) total: 2m 30s remaining: 8m 10s
235: learn: 0.0160580 test: 0.0631291 best: 0.0631291 (235) total: 2m 31s remaining: 8m 9s
236: learn: 0.0159265 test: 0.0629414 best: 0.0629414 (236) total: 2m 31s remaining: 8m 9s
237: learn: 0.0157826 test: 0.0628848 best: 0.0628848 (237) total: 2m 32s remaining: 8m 8s
238: learn: 0.0157240 test: 0.0628295 best: 0.0628295 (238) total: 2m 33s remaining: 8m 7s
239: learn: 0.0156159 test: 0.0627075 best: 0.0627075 (239) total: 2m 33s remaining: 8m 7s
240: learn: 0.0154397 test: 0.0625578 best: 0.0625578 (240) total: 2m 34s remaining: 8m 6s
241: learn: 0.0153609 test: 0.0625756 best: 0.0625578 (240) total: 2m 35s remaining: 8m 6s
242: learn: 0.0152036 test: 0.0624757 best: 0.0624757 (242) total: 2m 36s remaining: 8m 5s
243: learn: 0.0149678 test: 0.0621234 best: 0.0621234 (243) total: 2m 36s remaining: 8m 5s
244: learn: 0.0148529 test: 0.0620187 best: 0.0620187 (244) total: 2m 37s remaining: 8m 4s
245: learn: 0.0148024 test: 0.0620552 best: 0.0620187 (244) total: 2m 37s remaining: 8m 4s
246: learn: 0.0147080 test: 0.0620301 best: 0.0620187 (244) total: 2m 38s remaining: 8m 3s
247: learn: 0.0146668 test: 0.0619730 best: 0.0619730 (247) total: 2m 39s remaining: 8m 2s
248: learn: 0.0145612 test: 0.0617879 best: 0.0617879 (248) total: 2m 39s remaining: 8m 2s
249: learn: 0.0144871 test: 0.0617749 best: 0.0617749 (249) total: 2m 40s remaining: 8m 1s
250: learn: 0.0144092 test: 0.0617716 best: 0.0617716 (250) total: 2m 41s remaining: 8m
251: learn: 0.0143379 test: 0.0616985 best: 0.0616985 (251) total: 2m 41s remaining: 8m
252: learn: 0.0142404 test: 0.0615193 best: 0.0615193 (252) total: 2m 42s remaining: 7m 59s
253: learn: 0.0141584 test: 0.0614794 best: 0.0614794 (253) total: 2m 43s remaining: 7m 59s
```

```
254: learn: 0.0140647 test: 0.0614802 best: 0.0614794 (253) total: 2m 43s remaining: 7m 58s
255: learn: 0.0140222 test: 0.0614509 best: 0.0614509 (255) total: 2m 44s remaining: 7m 57s
256: learn: 0.0139396 test: 0.0614486 best: 0.0614486 (256) total: 2m 45s remaining: 7m 57s
257: learn: 0.0138385 test: 0.0613383 best: 0.0613383 (257) total: 2m 45s remaining: 7m 56s
258: learn: 0.0137911 test: 0.0613245 best: 0.0613245 (258) total: 2m 46s remaining: 7m 56s
259: learn: 0.0136847 test: 0.0610857 best: 0.0610857 (259) total: 2m 47s remaining: 7m 55s
260: learn: 0.0136319 test: 0.0610590 best: 0.0610590 (260) total: 2m 47s remaining: 7m 54s
261: learn: 0.0135590 test: 0.0610626 best: 0.0610590 (260) total: 2m 48s remaining: 7m 54s
262: learn: 0.0134784 test: 0.0610557 best: 0.0610557 (262) total: 2m 48s remaining: 7m 53s
263: learn: 0.0134240 test: 0.0610180 best: 0.0610180 (263) total: 2m 49s remaining: 7m 52s
264: learn: 0.0133686 test: 0.0609367 best: 0.0609367 (264) total: 2m 50s remaining: 7m 52s
265: learn: 0.0132956 test: 0.0608427 best: 0.0608427 (265) total: 2m 50s remaining: 7m 51s
266: learn: 0.0132058 test: 0.0607731 best: 0.0607731 (266) total: 2m 51s remaining: 7m 51s
267: learn: 0.0130898 test: 0.0607898 best: 0.0607731 (266) total: 2m 52s remaining: 7m 50s
268: learn: 0.0130002 test: 0.0607730 best: 0.0607730 (268) total: 2m 52s remaining: 7m 50s
269: learn: 0.0129253 test: 0.0607072 best: 0.0607072 (269) total: 2m 53s remaining: 7m 49s
270: learn: 0.0129010 test: 0.0607223 best: 0.0607072 (269) total: 2m 54s remaining: 7m 48s
271: learn: 0.0127133 test: 0.0604012 best: 0.0604012 (271) total: 2m 54s remaining: 7m 47s
272: learn: 0.0126198 test: 0.0604126 best: 0.0604012 (271) total: 2m 55s remaining: 7m 47s
273: learn: 0.0125567 test: 0.0604222 best: 0.0604012 (271) total: 2m 56s remaining: 7m 46s
274: learn: 0.0124684 test: 0.0603266 best: 0.0603266 (274) total: 2m 56s remaining: 7m 46s
275: learn: 0.0123535 test: 0.0602337 best: 0.0602337 (275) total: 2m 57s remaining: 7m 45s
276: learn: 0.0121678 test: 0.0599941 best: 0.0599941 (276) total: 2m 58s remaining: 7m 44s
277: learn: 0.0121005 test: 0.0599898 best: 0.0599898 (277) total: 2m 58s remaining: 7m 44s
278: learn: 0.0120119 test: 0.0598824 best: 0.0598824 (278) total: 2m 59s remaining: 7m 43s
279: learn: 0.0119543 test: 0.0598048 best: 0.0598048 (279) total: 3m remaining: 7m 43s
280: learn: 0.0118387 test: 0.0596703 best: 0.0596703 (280) total: 3m remaining: 7m 42s
281: learn: 0.0117763 test: 0.0596231 best: 0.0596231 (281) total: 3m 1s remaining: 7m 42s
282: learn: 0.0116021 test: 0.0592997 best: 0.0592997 (282) total: 3m 2s remaining: 7m 41s
283: learn: 0.0114781 test: 0.0590340 best: 0.0590340 (283) total: 3m 2s remaining: 7m 40s
284: learn: 0.0113930 test: 0.0589800 best: 0.0589800 (284) total: 3m 3s remaining: 7m 40s
285: learn: 0.0113476 test: 0.0589868 best: 0.0589800 (284) total: 3m 4s remaining: 7m 39s
286: learn: 0.0113038 test: 0.0589487 best: 0.0589487 (286) total: 3m 4s remaining: 7m 38s
287: learn: 0.0112282 test: 0.0589755 best: 0.0589487 (286) total: 3m 5s remaining: 7m 38s
288: learn: 0.0111878 test: 0.0589547 best: 0.0589487 (286) total: 3m 6s remaining: 7m 37s
289: learn: 0.0110700 test: 0.0588063 best: 0.0588063 (289) total: 3m 6s remaining: 7m 37s
290: learn: 0.0110299 test: 0.0586892 best: 0.0586892 (290) total: 3m 7s remaining: 7m 36s
291: learn: 0.0109431 test: 0.0585201 best: 0.0585201 (291) total: 3m 7s remaining: 7m 35s
292: learn: 0.0108810 test: 0.0585814 best: 0.0585201 (291) total: 3m 8s remaining: 7m 35s
293: learn: 0.0108109 test: 0.0584368 best: 0.0584368 (293) total: 3m 9s remaining: 7m 34s
294: learn: 0.0107509 test: 0.0583276 best: 0.0583276 (294) total: 3m 9s remaining: 7m 34s
295: learn: 0.0107198 test: 0.0583320 best: 0.0583276 (294) total: 3m 10s remaining: 7m 33s
296: learn: 0.0106614 test: 0.0583251 best: 0.0583251 (296) total: 3m 11s remaining: 7m 32s
297: learn: 0.0105276 test: 0.0581333 best: 0.0581333 (297) total: 3m 11s remaining: 7m 32s
298: learn: 0.0104985 test: 0.0581320 best: 0.0581320 (298) total: 3m 12s remaining: 7m 31s
299: learn: 0.0104344 test: 0.0580818 best: 0.0580818 (299) total: 3m 13s remaining: 7m 30s
300: learn: 0.0104033 test: 0.0580843 best: 0.0580818 (299) total: 3m 13s remaining: 7m 30s
301: learn: 0.0103482 test: 0.0580742 best: 0.0580742 (301) total: 3m 14s remaining: 7m 29s
302: learn: 0.0101617 test: 0.0576983 best: 0.0576983 (302) total: 3m 15s remaining: 7m 29s
303: learn: 0.0101330 test: 0.0576889 best: 0.0576889 (303) total: 3m 15s remaining: 7m 28s
304: learn: 0.0100928 test: 0.0576664 best: 0.0576664 (304) total: 3m 16s remaining: 7m 27s
305: learn: 0.0100146 test: 0.0576088 best: 0.0576088 (305) total: 3m 17s remaining: 7m 27s
306: learn: 0.0099318 test: 0.0575603 best: 0.0575603 (306) total: 3m 17s remaining: 7m 26s
307: learn: 0.0098831 test: 0.0575316 best: 0.0575316 (307) total: 3m 18s remaining: 7m 26s
308: learn: 0.0098554 test: 0.0574939 best: 0.0574939 (308) total: 3m 19s remaining: 7m 25s
309: learn: 0.0097235 test: 0.0572644 best: 0.0572644 (309) total: 3m 19s remaining: 7m 24s
310: learn: 0.0096595 test: 0.0572417 best: 0.0572417 (310) total: 3m 20s remaining: 7m 24s
311: learn: 0.0096371 test: 0.0572234 best: 0.0572234 (311) total: 3m 21s remaining: 7m 23s
312: learn: 0.0096124 test: 0.0572054 best: 0.0572054 (312) total: 3m 21s remaining: 7m 23s
313: learn: 0.0095638 test: 0.0571633 best: 0.0571633 (313) total: 3m 22s remaining: 7m 22s
314: learn: 0.0095041 test: 0.0570358 best: 0.0570358 (314) total: 3m 23s remaining: 7m 21s
315: learn: 0.0094845 test: 0.0570424 best: 0.0570358 (314) total: 3m 23s remaining: 7m 21s
316: learn: 0.0094583 test: 0.0570515 best: 0.0570358 (314) total: 3m 24s remaining: 7m 20s
317: learn: 0.0094238 test: 0.0570047 best: 0.0570047 (317) total: 3m 25s remaining: 7m 19s
318: learn: 0.0094057 test: 0.0570031 best: 0.0570031 (318) total: 3m 25s remaining: 7m 19s
319: learn: 0.0092940 test: 0.0569075 best: 0.0569075 (319) total: 3m 26s remaining: 7m 18s
320: learn: 0.0092611 test: 0.0568882 best: 0.0568882 (320) total: 3m 27s remaining: 7m 17s
321: learn: 0.0092067 test: 0.0568397 best: 0.0568397 (321) total: 3m 27s remaining: 7m 17s
322: learn: 0.0091618 test: 0.0567782 best: 0.0567782 (322) total: 3m 28s remaining: 7m 16s
323: learn: 0.0091230 test: 0.0566870 best: 0.0566870 (323) total: 3m 28s remaining: 7m 15s
324: learn: 0.0090518 test: 0.0566020 best: 0.0566020 (324) total: 3m 29s remaining: 7m 15s
325: learn: 0.0090231 test: 0.0565223 best: 0.0565223 (325) total: 3m 30s remaining: 7m 14s
326: learn: 0.0088752 test: 0.0562044 best: 0.0562044 (326) total: 3m 30s remaining: 7m 14s
327: learn: 0.0088202 test: 0.0562023 best: 0.0562023 (327) total: 3m 31s remaining: 7m 13s
328: learn: 0.0087937 test: 0.0562366 best: 0.0562023 (327) total: 3m 32s remaining: 7m 13s
329: learn: 0.0087313 test: 0.0560993 best: 0.0560993 (329) total: 3m 32s remaining: 7m 12s
330: learn: 0.0086914 test: 0.0560492 best: 0.0560492 (330) total: 3m 33s remaining: 7m 11s
```

```
331: learn: 0.0086113 test: 0.0558365 best: 0.0558365 (331) total: 3m 34s remaining: 7m 11s
332: learn: 0.0085336 test: 0.0557925 best: 0.0557925 (332) total: 3m 34s remaining: 7m 10s
333: learn: 0.0084996 test: 0.0557801 best: 0.0557801 (333) total: 3m 35s remaining: 7m 9s
334: learn: 0.0084626 test: 0.0558131 best: 0.0557801 (333) total: 3m 36s remaining: 7m 9s
335: learn: 0.0084287 test: 0.0557099 best: 0.0557099 (335) total: 3m 36s remaining: 7m 8s
336: learn: 0.0084014 test: 0.0556751 best: 0.0556751 (336) total: 3m 37s remaining: 7m 7s
337: learn: 0.0083312 test: 0.0556030 best: 0.0556030 (337) total: 3m 38s remaining: 7m 7s
338: learn: 0.0082795 test: 0.0555411 best: 0.0555411 (338) total: 3m 38s remaining: 7m 6s
339: learn: 0.0082582 test: 0.0555533 best: 0.0555411 (338) total: 3m 39s remaining: 7m 6s
340: learn: 0.0082115 test: 0.0555892 best: 0.0555411 (338) total: 3m 40s remaining: 7m 5s
341: learn: 0.0081965 test: 0.0555629 best: 0.0555411 (338) total: 3m 40s remaining: 7m 4s
342: learn: 0.0081593 test: 0.0555521 best: 0.0555411 (338) total: 3m 41s remaining: 7m 4s
343: learn: 0.0081411 test: 0.0555444 best: 0.0555411 (338) total: 3m 42s remaining: 7m 3s
344: learn: 0.0080903 test: 0.0554906 best: 0.0554906 (344) total: 3m 42s remaining: 7m 2s
345: learn: 0.0080554 test: 0.0553965 best: 0.0553965 (345) total: 3m 43s remaining: 7m 2s
346: learn: 0.0080044 test: 0.0553786 best: 0.0553786 (346) total: 3m 44s remaining: 7m 1s
347: learn: 0.0079687 test: 0.0553471 best: 0.0553471 (347) total: 3m 44s remaining: 7m 1s
348: learn: 0.0079458 test: 0.0553326 best: 0.0553326 (348) total: 3m 45s remaining: 7m
349: learn: 0.0078998 test: 0.0552830 best: 0.0552830 (349) total: 3m 46s remaining: 6m 59s
350: learn: 0.0078626 test: 0.0552011 best: 0.0552011 (350) total: 3m 46s remaining: 6m 59s
351: learn: 0.0077498 test: 0.0549867 best: 0.0549867 (351) total: 3m 47s remaining: 6m 58s
352: learn: 0.0076818 test: 0.0548152 best: 0.0548152 (352) total: 3m 48s remaining: 6m 58s
353: learn: 0.0076557 test: 0.0548121 best: 0.0548121 (353) total: 3m 48s remaining: 6m 57s
354: learn: 0.0075909 test: 0.0547081 best: 0.0547081 (354) total: 3m 49s remaining: 6m 56s
355: learn: 0.0075621 test: 0.0547203 best: 0.0547081 (354) total: 3m 50s remaining: 6m 56s
356: learn: 0.0075210 test: 0.0546217 best: 0.0546217 (356) total: 3m 50s remaining: 6m 55s
357: learn: 0.0074971 test: 0.0546608 best: 0.0546217 (356) total: 3m 51s remaining: 6m 54s
358: learn: 0.0074480 test: 0.0545575 best: 0.0545575 (358) total: 3m 52s remaining: 6m 54s
359: learn: 0.0074183 test: 0.0545786 best: 0.0545575 (358) total: 3m 52s remaining: 6m 53s
360: learn: 0.0073657 test: 0.0546278 best: 0.0545575 (358) total: 3m 53s remaining: 6m 53s
361: learn: 0.0073058 test: 0.0544653 best: 0.0544653 (361) total: 3m 54s remaining: 6m 52s
362: learn: 0.0072518 test: 0.0543783 best: 0.0543783 (362) total: 3m 54s remaining: 6m 51s
363: learn: 0.0072085 test: 0.0544343 best: 0.0543783 (362) total: 3m 55s remaining: 6m 51s
364: learn: 0.0071883 test: 0.0544471 best: 0.0543783 (362) total: 3m 56s remaining: 6m 50s
365: learn: 0.0071232 test: 0.0542729 best: 0.0542729 (365) total: 3m 56s remaining: 6m 50s
366: learn: 0.0070661 test: 0.0542017 best: 0.0542017 (366) total: 3m 57s remaining: 6m 49s
367: learn: 0.0070114 test: 0.0542037 best: 0.0542017 (366) total: 3m 58s remaining: 6m 48s
368: learn: 0.0069807 test: 0.0542718 best: 0.0542017 (366) total: 3m 58s remaining: 6m 48s
369: learn: 0.0069279 test: 0.0542371 best: 0.0542017 (366) total: 3m 59s remaining: 6m 47s
370: learn: 0.0068760 test: 0.0542099 best: 0.0542017 (366) total: 4m remaining: 6m 47s
371: learn: 0.0068550 test: 0.0542363 best: 0.0542017 (366) total: 4m remaining: 6m 46s
372: learn: 0.0068226 test: 0.0542223 best: 0.0542017 (366) total: 4m 1s remaining: 6m 45s
373: learn: 0.0067961 test: 0.0541701 best: 0.0541701 (373) total: 4m 2s remaining: 6m 45s
374: learn: 0.0067360 test: 0.0540487 best: 0.0540487 (374) total: 4m 2s remaining: 6m 44s
375: learn: 0.0066524 test: 0.0538636 best: 0.0538636 (375) total: 4m 3s remaining: 6m 43s
376: learn: 0.0066363 test: 0.0539009 best: 0.0538636 (375) total: 4m 3s remaining: 6m 43s
377: learn: 0.0066255 test: 0.0538780 best: 0.0538636 (375) total: 4m 4s remaining: 6m 42s
378: learn: 0.0066100 test: 0.0538634 best: 0.0538634 (378) total: 4m 5s remaining: 6m 41s
379: learn: 0.0065498 test: 0.0538058 best: 0.0538058 (379) total: 4m 5s remaining: 6m 41s
380: learn: 0.0065308 test: 0.0537858 best: 0.0537858 (380) total: 4m 6s remaining: 6m 40s
381: learn: 0.0064981 test: 0.0537863 best: 0.0537858 (380) total: 4m 7s remaining: 6m 39s
382: learn: 0.0064775 test: 0.0537894 best: 0.0537858 (380) total: 4m 7s remaining: 6m 39s
383: learn: 0.0064375 test: 0.0537670 best: 0.0537670 (383) total: 4m 8s remaining: 6m 38s
384: learn: 0.0063734 test: 0.0536059 best: 0.0536059 (384) total: 4m 9s remaining: 6m 38s
385: learn: 0.0063523 test: 0.0535573 best: 0.0535573 (385) total: 4m 9s remaining: 6m 37s
386: learn: 0.0063081 test: 0.0534886 best: 0.0534886 (386) total: 4m 10s remaining: 6m 36s
387: learn: 0.0062664 test: 0.0534528 best: 0.0534528 (387) total: 4m 11s remaining: 6m 36s
388: learn: 0.0062397 test: 0.0534253 best: 0.0534253 (388) total: 4m 11s remaining: 6m 35s
389: learn: 0.0062285 test: 0.0534074 best: 0.0534074 (389) total: 4m 12s remaining: 6m 34s
390: learn: 0.0062170 test: 0.0534332 best: 0.0534074 (389) total: 4m 13s remaining: 6m 34s
391: learn: 0.0061940 test: 0.0534133 best: 0.0534074 (389) total: 4m 13s remaining: 6m 33s
392: learn: 0.0061809 test: 0.0534016 best: 0.0534016 (392) total: 4m 14s remaining: 6m 32s
393: learn: 0.0061547 test: 0.0533288 best: 0.0533288 (393) total: 4m 14s remaining: 6m 32s
394: learn: 0.0061372 test: 0.0533172 best: 0.0533172 (394) total: 4m 15s remaining: 6m 31s
395: learn: 0.0061113 test: 0.0532897 best: 0.0532897 (395) total: 4m 16s remaining: 6m 30s
396: learn: 0.0060752 test: 0.0533081 best: 0.0532897 (395) total: 4m 16s remaining: 6m 30s
397: learn: 0.0060534 test: 0.0533743 best: 0.0532897 (395) total: 4m 17s remaining: 6m 29s
398: learn: 0.0060323 test: 0.0534306 best: 0.0532897 (395) total: 4m 18s remaining: 6m 28s
399: learn: 0.0060068 test: 0.0534590 best: 0.0532897 (395) total: 4m 18s remaining: 6m 28s
400: learn: 0.0059810 test: 0.0534103 best: 0.0532897 (395) total: 4m 19s remaining: 6m 27s
401: learn: 0.0059575 test: 0.0533558 best: 0.0532897 (395) total: 4m 20s remaining: 6m 27s
402: learn: 0.0059341 test: 0.0533878 best: 0.0532897 (395) total: 4m 20s remaining: 6m 26s
403: learn: 0.0058977 test: 0.0533563 best: 0.0532897 (395) total: 4m 21s remaining: 6m 25s
404: learn: 0.0058758 test: 0.0533856 best: 0.0532897 (395) total: 4m 22s remaining: 6m 25s
405: learn: 0.0058407 test: 0.0533138 best: 0.0532897 (395) total: 4m 22s remaining: 6m 24s
406: learn: 0.0058130 test: 0.0533004 best: 0.0532897 (395) total: 4m 23s remaining: 6m 23s
407: learn: 0.0057875 test: 0.0532644 best: 0.0532644 (407) total: 4m 24s remaining: 6m 23s
```

```
408: learn: 0.0057689 test: 0.0532676 best: 0.0532644 (407) total: 4m 24s remaining: 6m 22s
409: learn: 0.0057315 test: 0.0532343 best: 0.0532343 (409) total: 4m 25s remaining: 6m 22s
410: learn: 0.0057119 test: 0.0532602 best: 0.0532343 (409) total: 4m 26s remaining: 6m 21s
411: learn: 0.0056954 test: 0.0532827 best: 0.0532343 (409) total: 4m 26s remaining: 6m 20s
412: learn: 0.0056647 test: 0.0532476 best: 0.0532343 (409) total: 4m 27s remaining: 6m 20s
413: learn: 0.0056391 test: 0.0531542 best: 0.0531542 (413) total: 4m 28s remaining: 6m 19s
414: learn: 0.0056198 test: 0.0530965 best: 0.0530965 (414) total: 4m 28s remaining: 6m 18s
415: learn: 0.0056026 test: 0.0530928 best: 0.0530928 (415) total: 4m 29s remaining: 6m 18s
416: learn: 0.0055622 test: 0.0530585 best: 0.0530585 (416) total: 4m 30s remaining: 6m 17s
417: learn: 0.0055532 test: 0.0530511 best: 0.0530511 (417) total: 4m 30s remaining: 6m 16s
418: learn: 0.0055338 test: 0.0530450 best: 0.0530450 (418) total: 4m 31s remaining: 6m 16s
419: learn: 0.0055237 test: 0.0530416 best: 0.0530416 (419) total: 4m 32s remaining: 6m 15s
420: learn: 0.0054910 test: 0.0529676 best: 0.0529676 (420) total: 4m 32s remaining: 6m 15s
421: learn: 0.0054610 test: 0.0529737 best: 0.0529676 (420) total: 4m 33s remaining: 6m 14s
422: learn: 0.0054300 test: 0.0529990 best: 0.0529676 (420) total: 4m 34s remaining: 6m 13s
423: learn: 0.0054042 test: 0.0529838 best: 0.0529676 (420) total: 4m 34s remaining: 6m 13s
424: learn: 0.0053789 test: 0.0529521 best: 0.0529521 (424) total: 4m 35s remaining: 6m 12s
425: learn: 0.0053597 test: 0.0529270 best: 0.0529270 (425) total: 4m 36s remaining: 6m 11s
426: learn: 0.0053462 test: 0.0529013 best: 0.0529013 (426) total: 4m 36s remaining: 6m 11s
427: learn: 0.0053293 test: 0.0529051 best: 0.0529013 (426) total: 4m 37s remaining: 6m 10s
428: learn: 0.0052940 test: 0.0528948 best: 0.0528948 (428) total: 4m 37s remaining: 6m 9s
429: learn: 0.0052801 test: 0.0529092 best: 0.0528948 (428) total: 4m 38s remaining: 6m 9s
430: learn: 0.0052520 test: 0.0528365 best: 0.0528365 (430) total: 4m 39s remaining: 6m 8s
431: learn: 0.0052179 test: 0.0527678 best: 0.0527678 (431) total: 4m 39s remaining: 6m 8s
432: learn: 0.0051921 test: 0.0527827 best: 0.0527678 (431) total: 4m 40s remaining: 6m 7s
433: learn: 0.0051845 test: 0.0527965 best: 0.0527678 (431) total: 4m 41s remaining: 6m 6s
434: learn: 0.0051676 test: 0.0528030 best: 0.0527678 (431) total: 4m 41s remaining: 6m 6s
435: learn: 0.0051451 test: 0.0527271 best: 0.0527271 (435) total: 4m 42s remaining: 6m 5s
436: learn: 0.0051344 test: 0.0527538 best: 0.0527271 (435) total: 4m 43s remaining: 6m 4s
437: learn: 0.0050570 test: 0.0525956 best: 0.0525956 (437) total: 4m 43s remaining: 6m 4s
438: learn: 0.0050203 test: 0.0526427 best: 0.0525956 (437) total: 4m 44s remaining: 6m 3s
439: learn: 0.0049952 test: 0.0526431 best: 0.0525956 (437) total: 4m 45s remaining: 6m 2s
440: learn: 0.0049780 test: 0.0526235 best: 0.0525956 (437) total: 4m 45s remaining: 6m 2s
441: learn: 0.0049606 test: 0.0526131 best: 0.0525956 (437) total: 4m 46s remaining: 6m 1s
442: learn: 0.0049445 test: 0.0525894 best: 0.0525894 (442) total: 4m 47s remaining: 6m
443: learn: 0.0049299 test: 0.0526072 best: 0.0525894 (442) total: 4m 47s remaining: 6m
444: learn: 0.0049209 test: 0.0526068 best: 0.0525894 (442) total: 4m 48s remaining: 5m 59s
445: learn: 0.0048905 test: 0.0525873 best: 0.0525873 (445) total: 4m 48s remaining: 5m 58s
446: learn: 0.0048819 test: 0.0526108 best: 0.0525873 (445) total: 4m 49s remaining: 5m 58s
447: learn: 0.0048633 test: 0.0526156 best: 0.0525873 (445) total: 4m 50s remaining: 5m 57s
448: learn: 0.0048442 test: 0.0526256 best: 0.0525873 (445) total: 4m 50s remaining: 5m 56s
449: learn: 0.0048012 test: 0.0524364 best: 0.0524364 (449) total: 4m 51s remaining: 5m 56s
450: learn: 0.0047810 test: 0.0523907 best: 0.0523907 (450) total: 4m 52s remaining: 5m 55s
451: learn: 0.0047657 test: 0.0524014 best: 0.0523907 (450) total: 4m 52s remaining: 5m 55s
452: learn: 0.0047584 test: 0.0524690 best: 0.0523907 (450) total: 4m 53s remaining: 5m 54s
453: learn: 0.0047476 test: 0.0524911 best: 0.0523907 (450) total: 4m 54s remaining: 5m 53s
454: learn: 0.0047350 test: 0.0524598 best: 0.0523907 (450) total: 4m 54s remaining: 5m 53s
455: learn: 0.0047202 test: 0.0524052 best: 0.0523907 (450) total: 4m 55s remaining: 5m 52s
456: learn: 0.0046943 test: 0.0523630 best: 0.0523630 (456) total: 4m 56s remaining: 5m 51s
457: learn: 0.0046828 test: 0.0523730 best: 0.0523630 (456) total: 4m 56s remaining: 5m 51s
458: learn: 0.0046674 test: 0.0523951 best: 0.0523630 (456) total: 4m 57s remaining: 5m 50s
459: learn: 0.0046439 test: 0.0523574 best: 0.0523574 (459) total: 4m 58s remaining: 5m 49s
460: learn: 0.0046348 test: 0.0523509 best: 0.0523509 (460) total: 4m 58s remaining: 5m 49s
461: learn: 0.0046262 test: 0.0523575 best: 0.0523509 (460) total: 4m 59s remaining: 5m 48s
462: learn: 0.0046163 test: 0.0523237 best: 0.0523237 (462) total: 4m 59s remaining: 5m 47s
463: learn: 0.0046073 test: 0.0522926 best: 0.0522926 (463) total: 5m remaining: 5m 47s
464: learn: 0.0045846 test: 0.0523041 best: 0.0522926 (463) total: 5m 1s remaining: 5m 46s
465: learn: 0.0045715 test: 0.0522551 best: 0.0522551 (465) total: 5m 1s remaining: 5m 45s
466: learn: 0.0045569 test: 0.0522284 best: 0.0522284 (466) total: 5m 2s remaining: 5m 45s
467: learn: 0.0045471 test: 0.0522142 best: 0.0522142 (467) total: 5m 3s remaining: 5m 44s
468: learn: 0.0045163 test: 0.0521577 best: 0.0521577 (468) total: 5m 3s remaining: 5m 43s
469: learn: 0.0044815 test: 0.0521000 best: 0.0521000 (469) total: 5m 4s remaining: 5m 43s
470: learn: 0.0044576 test: 0.0520974 best: 0.0520974 (470) total: 5m 5s remaining: 5m 42s
471: learn: 0.0044469 test: 0.0521330 best: 0.0520974 (470) total: 5m 5s remaining: 5m 42s
472: learn: 0.0044218 test: 0.0521391 best: 0.0520974 (470) total: 5m 6s remaining: 5m 41s
473: learn: 0.0044028 test: 0.0521423 best: 0.0520974 (470) total: 5m 7s remaining: 5m 40s
474: learn: 0.0043937 test: 0.0521751 best: 0.0520974 (470) total: 5m 7s remaining: 5m 40s
475: learn: 0.0043861 test: 0.0521804 best: 0.0520974 (470) total: 5m 8s remaining: 5m 39s
476: learn: 0.0043676 test: 0.0521246 best: 0.0520974 (470) total: 5m 9s remaining: 5m 38s
477: learn: 0.0043510 test: 0.0521375 best: 0.0520974 (470) total: 5m 9s remaining: 5m 38s
478: learn: 0.0043444 test: 0.0521572 best: 0.0520974 (470) total: 5m 10s remaining: 5m 37s
479: learn: 0.0043314 test: 0.0521255 best: 0.0520974 (470) total: 5m 10s remaining: 5m 36s
480: learn: 0.0043234 test: 0.0521400 best: 0.0520974 (470) total: 5m 11s remaining: 5m 36s
481: learn: 0.0043022 test: 0.0520948 best: 0.0520948 (481) total: 5m 12s remaining: 5m 35s
482: learn: 0.0042939 test: 0.0521042 best: 0.0520948 (481) total: 5m 12s remaining: 5m 34s
483: learn: 0.0042740 test: 0.0520741 best: 0.0520741 (483) total: 5m 13s remaining: 5m 34s
484: learn: 0.0042437 test: 0.0519388 best: 0.0519388 (484) total: 5m 13s remaining: 5m 33s
```

```
485: learn: 0.0042067 test: 0.0518938 best: 0.0518938 (485) total: 5m 14s remaining: 5m 32s
486: learn: 0.0041998 test: 0.0519033 best: 0.0518938 (485) total: 5m 15s remaining: 5m 32s
487: learn: 0.0041900 test: 0.0519072 best: 0.0518938 (485) total: 5m 15s remaining: 5m 31s
488: learn: 0.0041691 test: 0.0519059 best: 0.0518938 (485) total: 5m 16s remaining: 5m 30s
489: learn: 0.0041436 test: 0.0518722 best: 0.0518722 (489) total: 5m 17s remaining: 5m 30s
490: learn: 0.0041327 test: 0.0518823 best: 0.0518722 (489) total: 5m 17s remaining: 5m 29s
491: learn: 0.0041212 test: 0.0519371 best: 0.0518722 (489) total: 5m 18s remaining: 5m 28s
492: learn: 0.0041034 test: 0.0519383 best: 0.0518722 (489) total: 5m 18s remaining: 5m 28s
493: learn: 0.0040671 test: 0.0518140 best: 0.0518140 (493) total: 5m 19s remaining: 5m 27s
494: learn: 0.0040520 test: 0.0518052 best: 0.0518052 (494) total: 5m 20s remaining: 5m 26s
495: learn: 0.0040424 test: 0.0518304 best: 0.0518052 (494) total: 5m 20s remaining: 5m 26s
496: learn: 0.0040193 test: 0.0516971 best: 0.0516971 (496) total: 5m 21s remaining: 5m 25s
497: learn: 0.0040019 test: 0.0517034 best: 0.0516971 (496) total: 5m 22s remaining: 5m 24s
498: learn: 0.0039955 test: 0.0517369 best: 0.0516971 (496) total: 5m 22s remaining: 5m 23s
499: learn: 0.0039761 test: 0.0516556 best: 0.0516556 (499) total: 5m 23s remaining: 5m 23s
500: learn: 0.0039562 test: 0.0516591 best: 0.0516556 (499) total: 5m 23s remaining: 5m 22s
501: learn: 0.0039346 test: 0.0516963 best: 0.0516556 (499) total: 5m 24s remaining: 5m 22s
502: learn: 0.0039287 test: 0.0517069 best: 0.0516556 (499) total: 5m 25s remaining: 5m 21s
503: learn: 0.0038932 test: 0.0516857 best: 0.0516556 (499) total: 5m 25s remaining: 5m 20s
504: learn: 0.0038810 test: 0.0516833 best: 0.0516556 (499) total: 5m 26s remaining: 5m 20s
505: learn: 0.0038543 test: 0.0515733 best: 0.0515733 (505) total: 5m 27s remaining: 5m 19s
506: learn: 0.0038456 test: 0.0516040 best: 0.0515733 (505) total: 5m 27s remaining: 5m 18s
507: learn: 0.0038290 test: 0.0515864 best: 0.0515733 (505) total: 5m 28s remaining: 5m 18s
508: learn: 0.0038107 test: 0.0515483 best: 0.0515483 (508) total: 5m 29s remaining: 5m 17s
509: learn: 0.0037990 test: 0.0515478 best: 0.0515478 (509) total: 5m 29s remaining: 5m 16s
510: learn: 0.0037838 test: 0.0515060 best: 0.0515060 (510) total: 5m 30s remaining: 5m 16s
511: learn: 0.0037718 test: 0.0514941 best: 0.0514941 (511) total: 5m 31s remaining: 5m 15s
512: learn: 0.0037499 test: 0.0514420 best: 0.0514420 (512) total: 5m 31s remaining: 5m 14s
513: learn: 0.0037368 test: 0.0514811 best: 0.0514420 (512) total: 5m 32s remaining: 5m 14s
514: learn: 0.0037254 test: 0.0514498 best: 0.0514420 (512) total: 5m 32s remaining: 5m 13s
515: learn: 0.0037043 test: 0.0513580 best: 0.0513580 (515) total: 5m 33s remaining: 5m 12s
516: learn: 0.0036926 test: 0.0513859 best: 0.0513580 (515) total: 5m 34s remaining: 5m 12s
517: learn: 0.0036846 test: 0.0513856 best: 0.0513580 (515) total: 5m 34s remaining: 5m 11s
518: learn: 0.0036773 test: 0.0513891 best: 0.0513580 (515) total: 5m 35s remaining: 5m 10s
519: learn: 0.0036580 test: 0.0513088 best: 0.0513088 (519) total: 5m 36s remaining: 5m 10s
520: learn: 0.0036468 test: 0.0512835 best: 0.0512835 (520) total: 5m 36s remaining: 5m 9s
521: learn: 0.0036358 test: 0.0513158 best: 0.0512835 (520) total: 5m 37s remaining: 5m 8s
522: learn: 0.0036214 test: 0.0513021 best: 0.0512835 (520) total: 5m 38s remaining: 5m 8s
523: learn: 0.0036091 test: 0.0513576 best: 0.0512835 (520) total: 5m 38s remaining: 5m 7s
524: learn: 0.0036014 test: 0.0513245 best: 0.0512835 (520) total: 5m 39s remaining: 5m 7s
525: learn: 0.0035855 test: 0.0512831 best: 0.0512831 (525) total: 5m 40s remaining: 5m 6s
526: learn: 0.0035711 test: 0.0512668 best: 0.0512668 (526) total: 5m 40s remaining: 5m 5s
527: learn: 0.0035548 test: 0.0511886 best: 0.0511886 (527) total: 5m 41s remaining: 5m 5s
528: learn: 0.0035491 test: 0.0511913 best: 0.0511886 (527) total: 5m 41s remaining: 5m 4s
529: learn: 0.0035385 test: 0.0511540 best: 0.0511540 (529) total: 5m 42s remaining: 5m 3s
530: learn: 0.0035275 test: 0.0511664 best: 0.0511540 (529) total: 5m 43s remaining: 5m 3s
531: learn: 0.0035181 test: 0.0511866 best: 0.0511540 (529) total: 5m 43s remaining: 5m 2s
532: learn: 0.0035117 test: 0.0512036 best: 0.0511540 (529) total: 5m 44s remaining: 5m 1s
533: learn: 0.0035059 test: 0.0512141 best: 0.0511540 (529) total: 5m 45s remaining: 5m 1s
534: learn: 0.0034955 test: 0.0512089 best: 0.0511540 (529) total: 5m 45s remaining: 5m
535: learn: 0.0034857 test: 0.0511991 best: 0.0511540 (529) total: 5m 46s remaining: 4m 59s
536: learn: 0.0034734 test: 0.0511986 best: 0.0511540 (529) total: 5m 47s remaining: 4m 59s
537: learn: 0.0034612 test: 0.0511756 best: 0.0511540 (529) total: 5m 47s remaining: 4m 58s
538: learn: 0.0034483 test: 0.0511912 best: 0.0511540 (529) total: 5m 48s remaining: 4m 57s
539: learn: 0.0034352 test: 0.0511484 best: 0.0511484 (539) total: 5m 49s remaining: 4m 57s
540: learn: 0.0034274 test: 0.0511362 best: 0.0511362 (540) total: 5m 49s remaining: 4m 56s
541: learn: 0.0034008 test: 0.0511690 best: 0.0511362 (540) total: 5m 50s remaining: 4m 56s
542: learn: 0.0033913 test: 0.0512138 best: 0.0511362 (540) total: 5m 50s remaining: 4m 55s
543: learn: 0.0033763 test: 0.0511926 best: 0.0511362 (540) total: 5m 51s remaining: 4m 54s
544: learn: 0.0033703 test: 0.0512322 best: 0.0511362 (540) total: 5m 52s remaining: 4m 54s
545: learn: 0.0033547 test: 0.0512151 best: 0.0511362 (540) total: 5m 52s remaining: 4m 53s
546: learn: 0.0033501 test: 0.0512429 best: 0.0511362 (540) total: 5m 53s remaining: 4m 52s
547: learn: 0.0033416 test: 0.0512243 best: 0.0511362 (540) total: 5m 54s remaining: 4m 52s
548: learn: 0.0033280 test: 0.0511987 best: 0.0511362 (540) total: 5m 54s remaining: 4m 51s
549: learn: 0.0033213 test: 0.0512154 best: 0.0511362 (540) total: 5m 55s remaining: 4m 50s
550: learn: 0.0033175 test: 0.0512159 best: 0.0511362 (540) total: 5m 55s remaining: 4m 50s
551: learn: 0.0033078 test: 0.0512358 best: 0.0511362 (540) total: 5m 56s remaining: 4m 49s
552: learn: 0.0033009 test: 0.0512414 best: 0.0511362 (540) total: 5m 57s remaining: 4m 48s
553: learn: 0.0032783 test: 0.0512246 best: 0.0511362 (540) total: 5m 57s remaining: 4m 48s
554: learn: 0.0032718 test: 0.0512497 best: 0.0511362 (540) total: 5m 58s remaining: 4m 47s
555: learn: 0.0032671 test: 0.0512423 best: 0.0511362 (540) total: 5m 58s remaining: 4m 46s
556: learn: 0.0032616 test: 0.0512067 best: 0.0511362 (540) total: 5m 59s remaining: 4m 45s
557: learn: 0.0032344 test: 0.0510983 best: 0.0510983 (557) total: 6m remaining: 4m 45s
558: learn: 0.0032271 test: 0.0510683 best: 0.0510683 (558) total: 6m remaining: 4m 44s
559: learn: 0.0032186 test: 0.0510111 best: 0.0510111 (559) total: 6m 1s remaining: 4m 44s
560: learn: 0.0031903 test: 0.0509418 best: 0.0509418 (560) total: 6m 2s remaining: 4m 43s
561: learn: 0.0031796 test: 0.0508809 best: 0.0508809 (561) total: 6m 2s remaining: 4m 42s
```

```
562: learn: 0.0031669 test: 0.0508925 best: 0.0508809 (561) total: 6m 3s remaining: 4m 42s
563: learn: 0.0031597 test: 0.0509185 best: 0.0508809 (561) total: 6m 4s remaining: 4m 41s
564: learn: 0.0031518 test: 0.0509325 best: 0.0508809 (561) total: 6m 4s remaining: 4m 40s
565: learn: 0.0031415 test: 0.0509077 best: 0.0508809 (561) total: 6m 5s remaining: 4m 40s
566: learn: 0.0031321 test: 0.0509190 best: 0.0508809 (561) total: 6m 5s remaining: 4m 39s
567: learn: 0.0031239 test: 0.0508904 best: 0.0508809 (561) total: 6m 6s remaining: 4m 38s
568: learn: 0.0031113 test: 0.0509107 best: 0.0508809 (561) total: 6m 6s remaining: 4m 37s
569: learn: 0.0031031 test: 0.0508785 best: 0.0508785 (569) total: 6m 7s remaining: 4m 37s
570: learn: 0.0030968 test: 0.0508555 best: 0.0508555 (570) total: 6m 8s remaining: 4m 36s
571: learn: 0.0030884 test: 0.0508230 best: 0.0508230 (571) total: 6m 8s remaining: 4m 35s
572: learn: 0.0030834 test: 0.0508513 best: 0.0508230 (571) total: 6m 9s remaining: 4m 35s
573: learn: 0.0030663 test: 0.0507930 best: 0.0507930 (573) total: 6m 10s remaining: 4m 34s
574: learn: 0.0030585 test: 0.0508065 best: 0.0507930 (573) total: 6m 10s remaining: 4m 33s
575: learn: 0.0030479 test: 0.0508279 best: 0.0507930 (573) total: 6m 11s remaining: 4m 33s
576: learn: 0.0030401 test: 0.0507948 best: 0.0507930 (573) total: 6m 11s remaining: 4m 32s
577: learn: 0.0030240 test: 0.0507838 best: 0.0507838 (577) total: 6m 12s remaining: 4m 32s
578: learn: 0.0030186 test: 0.0507626 best: 0.0507626 (578) total: 6m 13s remaining: 4m 31s
579: learn: 0.0029883 test: 0.0506285 best: 0.0506285 (579) total: 6m 13s remaining: 4m 30s
580: learn: 0.0029809 test: 0.0506545 best: 0.0506285 (579) total: 6m 14s remaining: 4m 30s
581: learn: 0.0029713 test: 0.0506522 best: 0.0506285 (579) total: 6m 15s remaining: 4m 29s
582: learn: 0.0029638 test: 0.0506471 best: 0.0506285 (579) total: 6m 15s remaining: 4m 28s
583: learn: 0.0029502 test: 0.0506778 best: 0.0506285 (579) total: 6m 16s remaining: 4m 28s
584: learn: 0.0029411 test: 0.0506745 best: 0.0506285 (579) total: 6m 16s remaining: 4m 27s
585: learn: 0.0029352 test: 0.0506870 best: 0.0506285 (579) total: 6m 17s remaining: 4m 26s
586: learn: 0.0029274 test: 0.0507060 best: 0.0506285 (579) total: 6m 18s remaining: 4m 26s
587: learn: 0.0029224 test: 0.0506863 best: 0.0506285 (579) total: 6m 18s remaining: 4m 25s
588: learn: 0.0029162 test: 0.0506740 best: 0.0506285 (579) total: 6m 19s remaining: 4m 24s
589: learn: 0.0029018 test: 0.0506164 best: 0.0506164 (589) total: 6m 20s remaining: 4m 24s
590: learn: 0.0028951 test: 0.0505829 best: 0.0505829 (590) total: 6m 20s remaining: 4m 23s
591: learn: 0.0028913 test: 0.0505808 best: 0.0505808 (591) total: 6m 21s remaining: 4m 22s
592: learn: 0.0028852 test: 0.0506138 best: 0.0505808 (591) total: 6m 21s remaining: 4m 22s
593: learn: 0.0028738 test: 0.0506010 best: 0.0505808 (591) total: 6m 22s remaining: 4m 21s
594: learn: 0.0028652 test: 0.0505800 best: 0.0505800 (594) total: 6m 23s remaining: 4m 20s
595: learn: 0.0028583 test: 0.0506131 best: 0.0505800 (594) total: 6m 23s remaining: 4m 20s
596: learn: 0.0028355 test: 0.0504966 best: 0.0504966 (596) total: 6m 24s remaining: 4m 19s
597: learn: 0.0028263 test: 0.0504717 best: 0.0504717 (597) total: 6m 25s remaining: 4m 18s
598: learn: 0.0028199 test: 0.0504958 best: 0.0504717 (597) total: 6m 25s remaining: 4m 18s
599: learn: 0.0028082 test: 0.0504573 best: 0.0504573 (599) total: 6m 26s remaining: 4m 17s
600: learn: 0.0028019 test: 0.0504295 best: 0.0504295 (600) total: 6m 26s remaining: 4m 16s
601: learn: 0.0027985 test: 0.0504415 best: 0.0504295 (600) total: 6m 27s remaining: 4m 16s
602: learn: 0.0027904 test: 0.0504665 best: 0.0504295 (600) total: 6m 28s remaining: 4m 15s
603: learn: 0.0027780 test: 0.0504736 best: 0.0504295 (600) total: 6m 28s remaining: 4m 14s
604: learn: 0.0027741 test: 0.0504893 best: 0.0504295 (600) total: 6m 29s remaining: 4m 14s
605: learn: 0.0027622 test: 0.0504594 best: 0.0504295 (600) total: 6m 30s remaining: 4m 13s
606: learn: 0.0027558 test: 0.0504273 best: 0.0504273 (606) total: 6m 30s remaining: 4m 12s
607: learn: 0.0027494 test: 0.0504353 best: 0.0504273 (606) total: 6m 31s remaining: 4m 12s
608: learn: 0.0027437 test: 0.0504443 best: 0.0504273 (606) total: 6m 32s remaining: 4m 11s
609: learn: 0.0027380 test: 0.0504296 best: 0.0504273 (606) total: 6m 32s remaining: 4m 11s
610: learn: 0.0027249 test: 0.0504169 best: 0.0504169 (610) total: 6m 33s remaining: 4m 10s
611: learn: 0.0027192 test: 0.0504482 best: 0.0504169 (610) total: 6m 33s remaining: 4m 9s
612: learn: 0.0027134 test: 0.0504221 best: 0.0504169 (610) total: 6m 34s remaining: 4m 9s
613: learn: 0.0027065 test: 0.0504422 best: 0.0504169 (610) total: 6m 35s remaining: 4m 8s
614: learn: 0.0026988 test: 0.0504524 best: 0.0504169 (610) total: 6m 35s remaining: 4m 7s
615: learn: 0.0026918 test: 0.0504398 best: 0.0504169 (610) total: 6m 36s remaining: 4m 7s
616: learn: 0.0026689 test: 0.0503457 best: 0.0503457 (616) total: 6m 37s remaining: 4m 6s
617: learn: 0.0026628 test: 0.0503331 best: 0.0503331 (617) total: 6m 37s remaining: 4m 5s
618: learn: 0.0026537 test: 0.0503383 best: 0.0503331 (617) total: 6m 38s remaining: 4m 5s
619: learn: 0.0026454 test: 0.0503425 best: 0.0503331 (617) total: 6m 39s remaining: 4m 4s
620: learn: 0.0026381 test: 0.0503269 best: 0.0503269 (620) total: 6m 39s remaining: 4m 3s
621: learn: 0.0026346 test: 0.0503331 best: 0.0503269 (620) total: 6m 40s remaining: 4m 3s
622: learn: 0.0026297 test: 0.0502901 best: 0.0502901 (622) total: 6m 40s remaining: 4m 2s
623: learn: 0.0026219 test: 0.0502745 best: 0.0502745 (623) total: 6m 41s remaining: 4m 1s
624: learn: 0.0026183 test: 0.0502812 best: 0.0502745 (623) total: 6m 42s remaining: 4m 1s
625: learn: 0.0026110 test: 0.0502775 best: 0.0502745 (623) total: 6m 42s remaining: 4m
626: learn: 0.0026046 test: 0.0503135 best: 0.0502745 (623) total: 6m 43s remaining: 3m 59s
627: learn: 0.0025898 test: 0.0502451 best: 0.0502451 (627) total: 6m 43s remaining: 3m 59s
628: learn: 0.0025826 test: 0.0502300 best: 0.0502300 (628) total: 6m 44s remaining: 3m 58s
629: learn: 0.0025649 test: 0.0501382 best: 0.0501382 (629) total: 6m 45s remaining: 3m 58s
630: learn: 0.0025566 test: 0.0501313 best: 0.0501313 (630) total: 6m 45s remaining: 3m 57s
631: learn: 0.0025508 test: 0.0501056 best: 0.0501056 (631) total: 6m 46s remaining: 3m 56s
632: learn: 0.0025427 test: 0.0501341 best: 0.0501056 (631) total: 6m 47s remaining: 3m 56s
633: learn: 0.0025237 test: 0.0500273 best: 0.0500273 (633) total: 6m 47s remaining: 3m 55s
634: learn: 0.0025179 test: 0.0500569 best: 0.0500273 (633) total: 6m 48s remaining: 3m 54s
635: learn: 0.0025124 test: 0.0500321 best: 0.0500273 (633) total: 6m 49s remaining: 3m 54s
636: learn: 0.0025108 test: 0.0500436 best: 0.0500273 (633) total: 6m 49s remaining: 3m 53s
637: learn: 0.0025053 test: 0.0500059 best: 0.0500059 (637) total: 6m 50s remaining: 3m 52s
638: learn: 0.0025001 test: 0.0499758 best: 0.0499758 (638) total: 6m 50s remaining: 3m 52s
```

```
639: learn: 0.0024950 test: 0.0500042 best: 0.0499758 (638) total: 6m 51s remaining: 3m 51s
640: learn: 0.0024841 test: 0.0499770 best: 0.0499758 (638) total: 6m 52s remaining: 3m 50s
641: learn: 0.0024740 test: 0.0499719 best: 0.0499719 (641) total: 6m 52s remaining: 3m 50s
642: learn: 0.0024653 test: 0.0499502 best: 0.0499502 (642) total: 6m 53s remaining: 3m 49s
643: learn: 0.0024603 test: 0.0499830 best: 0.0499502 (642) total: 6m 54s remaining: 3m 48s
644: learn: 0.0024564 test: 0.0500237 best: 0.0499502 (642) total: 6m 54s remaining: 3m 48s
645: learn: 0.0024447 test: 0.0500155 best: 0.0499502 (642) total: 6m 55s remaining: 3m 47s
646: learn: 0.0024378 test: 0.0499902 best: 0.0499502 (642) total: 6m 56s remaining: 3m 47s
647: learn: 0.0024322 test: 0.0499946 best: 0.0499502 (642) total: 6m 56s remaining: 3m 46s
648: learn: 0.0024243 test: 0.0500009 best: 0.0499502 (642) total: 6m 57s remaining: 3m 45s
649: learn: 0.0024175 test: 0.0499952 best: 0.0499502 (642) total: 6m 57s remaining: 3m 45s
650: learn: 0.0024107 test: 0.0500161 best: 0.0499502 (642) total: 6m 58s remaining: 3m 44s
651: learn: 0.0024062 test: 0.0500198 best: 0.0499502 (642) total: 6m 59s remaining: 3m 43s
652: learn: 0.0024003 test: 0.0500310 best: 0.0499502 (642) total: 6m 59s remaining: 3m 43s
653: learn: 0.0023973 test: 0.0500398 best: 0.0499502 (642) total: 7m remaining: 3m 42s
654: learn: 0.0023941 test: 0.0500746 best: 0.0499502 (642) total: 7m 1s remaining: 3m 41s
655: learn: 0.0023901 test: 0.0500560 best: 0.0499502 (642) total: 7m 1s remaining: 3m 41s
656: learn: 0.0023869 test: 0.0500360 best: 0.0499502 (642) total: 7m 2s remaining: 3m 40s
657: learn: 0.0023761 test: 0.0500563 best: 0.0499502 (642) total: 7m 3s remaining: 3m 39s
658: learn: 0.0023616 test: 0.0500359 best: 0.0499502 (642) total: 7m 3s remaining: 3m 39s
659: learn: 0.0023546 test: 0.0500140 best: 0.0499502 (642) total: 7m 4s remaining: 3m 38s
660: learn: 0.0023507 test: 0.0500302 best: 0.0499502 (642) total: 7m 5s remaining: 3m 38s
661: learn: 0.0023435 test: 0.0500378 best: 0.0499502 (642) total: 7m 5s remaining: 3m 37s
662: learn: 0.0023400 test: 0.0500491 best: 0.0499502 (642) total: 7m 6s remaining: 3m 36s
663: learn: 0.0023338 test: 0.0500363 best: 0.0499502 (642) total: 7m 7s remaining: 3m 36s
664: learn: 0.0023297 test: 0.0500464 best: 0.0499502 (642) total: 7m 7s remaining: 3m 35s
665: learn: 0.0023267 test: 0.0501013 best: 0.0499502 (642) total: 7m 8s remaining: 3m 34s
666: learn: 0.0023160 test: 0.0500957 best: 0.0499502 (642) total: 7m 8s remaining: 3m 34s
667: learn: 0.0023085 test: 0.0500740 best: 0.0499502 (642) total: 7m 9s remaining: 3m 33s
668: learn: 0.0023050 test: 0.0500924 best: 0.0499502 (642) total: 7m 10s remaining: 3m 32s
669: learn: 0.0023006 test: 0.0501583 best: 0.0499502 (642) total: 7m 10s remaining: 3m 32s
670: learn: 0.0022861 test: 0.0500853 best: 0.0499502 (642) total: 7m 11s remaining: 3m 31s
671: learn: 0.0022798 test: 0.0500911 best: 0.0499502 (642) total: 7m 12s remaining: 3m 31s
672: learn: 0.0022743 test: 0.0501052 best: 0.0499502 (642) total: 7m 12s remaining: 3m 30s
673: learn: 0.0022617 test: 0.0500133 best: 0.0499502 (642) total: 7m 13s remaining: 3m 29s
674: learn: 0.0022525 test: 0.0499268 best: 0.0499268 (674) total: 7m 14s remaining: 3m 29s
675: learn: 0.0022502 test: 0.0499601 best: 0.0499268 (674) total: 7m 14s remaining: 3m 28s
676: learn: 0.0022462 test: 0.0499515 best: 0.0499268 (674) total: 7m 15s remaining: 3m 27s
677: learn: 0.0022411 test: 0.0499539 best: 0.0499268 (674) total: 7m 16s remaining: 3m 27s
678: learn: 0.0022348 test: 0.0499840 best: 0.0499268 (674) total: 7m 16s remaining: 3m 26s
679: learn: 0.0022292 test: 0.0499542 best: 0.0499268 (674) total: 7m 17s remaining: 3m 25s
680: learn: 0.0022224 test: 0.0499300 best: 0.0499268 (674) total: 7m 18s remaining: 3m 25s
681: learn: 0.0022178 test: 0.0498892 best: 0.0498892 (681) total: 7m 18s remaining: 3m 24s
682: learn: 0.0022134 test: 0.0498546 best: 0.0498546 (682) total: 7m 19s remaining: 3m 23s
683: learn: 0.0022063 test: 0.0498441 best: 0.0498441 (683) total: 7m 20s remaining: 3m 23s
684: learn: 0.0022017 test: 0.0498755 best: 0.0498441 (683) total: 7m 20s remaining: 3m 22s
685: learn: 0.0021976 test: 0.0498446 best: 0.0498441 (683) total: 7m 21s remaining: 3m 21s
686: learn: 0.0021942 test: 0.0498456 best: 0.0498441 (683) total: 7m 21s remaining: 3m 21s
687: learn: 0.0021770 test: 0.0497674 best: 0.0497674 (687) total: 7m 22s remaining: 3m 20s
688: learn: 0.0021675 test: 0.0498013 best: 0.0497674 (687) total: 7m 23s remaining: 3m 20s
689: learn: 0.0021650 test: 0.0498003 best: 0.0497674 (687) total: 7m 23s remaining: 3m 19s
690: learn: 0.0021593 test: 0.0498219 best: 0.0497674 (687) total: 7m 24s remaining: 3m 18s
691: learn: 0.0021537 test: 0.0498196 best: 0.0497674 (687) total: 7m 25s remaining: 3m 18s
692: learn: 0.0021421 test: 0.0498458 best: 0.0497674 (687) total: 7m 25s remaining: 3m 17s
693: learn: 0.0021353 test: 0.0498582 best: 0.0497674 (687) total: 7m 26s remaining: 3m 16s
694: learn: 0.0021240 test: 0.0498175 best: 0.0497674 (687) total: 7m 27s remaining: 3m 16s
695: learn: 0.0021214 test: 0.0498061 best: 0.0497674 (687) total: 7m 27s remaining: 3m 15s
696: learn: 0.0021179 test: 0.0498133 best: 0.0497674 (687) total: 7m 28s remaining: 3m 14s
697: learn: 0.0021067 test: 0.0497395 best: 0.0497395 (697) total: 7m 29s remaining: 3m 14s
698: learn: 0.0021030 test: 0.0497518 best: 0.0497395 (697) total: 7m 29s remaining: 3m 13s
699: learn: 0.0020984 test: 0.0497564 best: 0.0497395 (697) total: 7m 30s remaining: 3m 13s
700: learn: 0.0020916 test: 0.0497794 best: 0.0497395 (697) total: 7m 31s remaining: 3m 12s
701: learn: 0.0020878 test: 0.0497774 best: 0.0497395 (697) total: 7m 31s remaining: 3m 11s
702: learn: 0.0020748 test: 0.0497630 best: 0.0497395 (697) total: 7m 32s remaining: 3m 11s
703: learn: 0.0020695 test: 0.0497744 best: 0.0497395 (697) total: 7m 33s remaining: 3m 10s
704: learn: 0.0020658 test: 0.0497729 best: 0.0497395 (697) total: 7m 33s remaining: 3m 9s
705: learn: 0.0020601 test: 0.0497729 best: 0.0497395 (697) total: 7m 34s remaining: 3m 9s
706: learn: 0.0020568 test: 0.0497933 best: 0.0497395 (697) total: 7m 35s remaining: 3m 8s
707: learn: 0.0020528 test: 0.0497811 best: 0.0497395 (697) total: 7m 35s remaining: 3m 7s
708: learn: 0.0020465 test: 0.0498213 best: 0.0497395 (697) total: 7m 36s remaining: 3m 7s
709: learn: 0.0020399 test: 0.0497653 best: 0.0497395 (697) total: 7m 36s remaining: 3m 6s
710: learn: 0.0020343 test: 0.0497465 best: 0.0497395 (697) total: 7m 37s remaining: 3m 6s
711: learn: 0.0020322 test: 0.0497607 best: 0.0497395 (697) total: 7m 38s remaining: 3m 5s
712: learn: 0.0020299 test: 0.0497652 best: 0.0497395 (697) total: 7m 38s remaining: 3m 4s
713: learn: 0.0020243 test: 0.0497482 best: 0.0497395 (697) total: 7m 39s remaining: 3m 4s
714: learn: 0.0020199 test: 0.0497423 best: 0.0497395 (697) total: 7m 40s remaining: 3m 3s
715: learn: 0.0020147 test: 0.0497476 best: 0.0497395 (697) total: 7m 40s remaining: 3m 2s
```

```
716: learn: 0.0020031 test: 0.0496885 best: 0.0496885 (716) total: 7m 41s remaining: 3m 2s
717: learn: 0.0019955 test: 0.0496378 best: 0.0496378 (717) total: 7m 42s remaining: 3m 1s
718: learn: 0.0019928 test: 0.0496433 best: 0.0496378 (717) total: 7m 42s remaining: 3m
719: learn: 0.0019891 test: 0.0496167 best: 0.0496167 (719) total: 7m 43s remaining: 3m
720: learn: 0.0019850 test: 0.0496083 best: 0.0496083 (720) total: 7m 44s remaining: 2m 59s
721: learn: 0.0019789 test: 0.0496292 best: 0.0496083 (720) total: 7m 44s remaining: 2m 58s
722: learn: 0.0019761 test: 0.0496513 best: 0.0496083 (720) total: 7m 45s remaining: 2m 58s
723: learn: 0.0019663 test: 0.0496307 best: 0.0496083 (720) total: 7m 46s remaining: 2m 57s
724: learn: 0.0019634 test: 0.0496392 best: 0.0496083 (720) total: 7m 46s remaining: 2m 57s
725: learn: 0.0019607 test: 0.0496342 best: 0.0496083 (720) total: 7m 47s remaining: 2m 56s
726: learn: 0.0019569 test: 0.0496546 best: 0.0496083 (720) total: 7m 47s remaining: 2m 55s
727: learn: 0.0019538 test: 0.0496592 best: 0.0496083 (720) total: 7m 48s remaining: 2m 55s
728: learn: 0.0019438 test: 0.0496196 best: 0.0496083 (720) total: 7m 49s remaining: 2m 54s
729: learn: 0.0019402 test: 0.0496101 best: 0.0496083 (720) total: 7m 49s remaining: 2m 53s
730: learn: 0.0019354 test: 0.0496298 best: 0.0496083 (720) total: 7m 50s remaining: 2m 53s
731: learn: 0.0019332 test: 0.0496113 best: 0.0496083 (720) total: 7m 51s remaining: 2m 52s
732: learn: 0.0019266 test: 0.0495689 best: 0.0495689 (732) total: 7m 51s remaining: 2m 51s
733: learn: 0.0019224 test: 0.0495770 best: 0.0495689 (732) total: 7m 52s remaining: 2m 51s
734: learn: 0.0019178 test: 0.0496132 best: 0.0495689 (732) total: 7m 53s remaining: 2m 50s
735: learn: 0.0019117 test: 0.0496537 best: 0.0495689 (732) total: 7m 53s remaining: 2m 49s
736: learn: 0.0019071 test: 0.0496555 best: 0.0495689 (732) total: 7m 54s remaining: 2m 49s
737: learn: 0.0019018 test: 0.0496365 best: 0.0495689 (732) total: 7m 55s remaining: 2m 48s
738: learn: 0.0018970 test: 0.0496029 best: 0.0495689 (732) total: 7m 55s remaining: 2m 48s
739: learn: 0.0018916 test: 0.0495940 best: 0.0495689 (732) total: 7m 56s remaining: 2m 47s
740: learn: 0.0018842 test: 0.0495728 best: 0.0495689 (732) total: 7m 57s remaining: 2m 46s
741: learn: 0.0018805 test: 0.0496153 best: 0.0495689 (732) total: 7m 57s remaining: 2m 46s
742: learn: 0.0018763 test: 0.0495931 best: 0.0495689 (732) total: 7m 58s remaining: 2m 45s
743: learn: 0.0018663 test: 0.0495205 best: 0.0495205 (743) total: 7m 59s remaining: 2m 44s
744: learn: 0.0018629 test: 0.0495476 best: 0.0495205 (743) total: 7m 59s remaining: 2m 44s
745: learn: 0.0018602 test: 0.0495983 best: 0.0495205 (743) total: 8m remaining: 2m 43s
746: learn: 0.0018521 test: 0.0496375 best: 0.0495205 (743) total: 8m remaining: 2m 42s
747: learn: 0.0018448 test: 0.0495958 best: 0.0495205 (743) total: 8m 1s remaining: 2m 42s
748: learn: 0.0018402 test: 0.0495768 best: 0.0495205 (743) total: 8m 2s remaining: 2m 41s
749: learn: 0.0018380 test: 0.0496052 best: 0.0495205 (743) total: 8m 2s remaining: 2m 40s
750: learn: 0.0018325 test: 0.0495983 best: 0.0495205 (743) total: 8m 3s remaining: 2m 40s
751: learn: 0.0018252 test: 0.0496285 best: 0.0495205 (743) total: 8m 4s remaining: 2m 39s
752: learn: 0.0018225 test: 0.0496317 best: 0.0495205 (743) total: 8m 4s remaining: 2m 39s
753: learn: 0.0018165 test: 0.0496617 best: 0.0495205 (743) total: 8m 5s remaining: 2m 38s
754: learn: 0.0018145 test: 0.0496630 best: 0.0495205 (743) total: 8m 6s remaining: 2m 37s
755: learn: 0.0018115 test: 0.0496739 best: 0.0495205 (743) total: 8m 6s remaining: 2m 37s
756: learn: 0.0018051 test: 0.0496290 best: 0.0495205 (743) total: 8m 7s remaining: 2m 36s
757: learn: 0.0017986 test: 0.0495851 best: 0.0495205 (743) total: 8m 7s remaining: 2m 35s
758: learn: 0.0017960 test: 0.0495857 best: 0.0495205 (743) total: 8m 8s remaining: 2m 35s
759: learn: 0.0017933 test: 0.0495953 best: 0.0495205 (743) total: 8m 9s remaining: 2m 34s
760: learn: 0.0017913 test: 0.0496139 best: 0.0495205 (743) total: 8m 9s remaining: 2m 33s
761: learn: 0.0017755 test: 0.0495015 best: 0.0495015 (761) total: 8m 10s remaining: 2m 33s
762: learn: 0.0017737 test: 0.0494730 best: 0.0494730 (762) total: 8m 11s remaining: 2m 32s
763: learn: 0.0017704 test: 0.0494565 best: 0.0494565 (763) total: 8m 11s remaining: 2m 31s
764: learn: 0.0017679 test: 0.0494664 best: 0.0494565 (763) total: 8m 12s remaining: 2m 31s
765: learn: 0.0017636 test: 0.0494663 best: 0.0494565 (763) total: 8m 13s remaining: 2m 30s
766: learn: 0.0017603 test: 0.0494890 best: 0.0494565 (763) total: 8m 13s remaining: 2m 29s
767: learn: 0.0017562 test: 0.0494355 best: 0.0494355 (767) total: 8m 14s remaining: 2m 29s
768: learn: 0.0017525 test: 0.0494071 best: 0.0494071 (768) total: 8m 14s remaining: 2m 28s
769: learn: 0.0017492 test: 0.0493623 best: 0.0493623 (769) total: 8m 15s remaining: 2m 28s
770: learn: 0.0017477 test: 0.0493731 best: 0.0493623 (769) total: 8m 16s remaining: 2m 27s
771: learn: 0.0017456 test: 0.0493886 best: 0.0493623 (769) total: 8m 16s remaining: 2m 26s
772: learn: 0.0017420 test: 0.0493992 best: 0.0493623 (769) total: 8m 17s remaining: 2m 26s
773: learn: 0.0017399 test: 0.0493961 best: 0.0493623 (769) total: 8m 18s remaining: 2m 25s
774: learn: 0.0017368 test: 0.0493903 best: 0.0493623 (769) total: 8m 18s remaining: 2m 24s
775: learn: 0.0017341 test: 0.0494071 best: 0.0493623 (769) total: 8m 19s remaining: 2m 24s
776: learn: 0.0017316 test: 0.0494083 best: 0.0493623 (769) total: 8m 20s remaining: 2m 23s
777: learn: 0.0017303 test: 0.0493975 best: 0.0493623 (769) total: 8m 20s remaining: 2m 22s
778: learn: 0.0017243 test: 0.0493958 best: 0.0493623 (769) total: 8m 21s remaining: 2m 22s
779: learn: 0.0017218 test: 0.0493731 best: 0.0493623 (769) total: 8m 21s remaining: 2m 21s
780: learn: 0.0017193 test: 0.0493898 best: 0.0493623 (769) total: 8m 22s remaining: 2m 20s
781: learn: 0.0017151 test: 0.0493629 best: 0.0493623 (769) total: 8m 23s remaining: 2m 20s
782: learn: 0.0017135 test: 0.0493779 best: 0.0493623 (769) total: 8m 23s remaining: 2m 19s
783: learn: 0.0017099 test: 0.0493733 best: 0.0493623 (769) total: 8m 24s remaining: 2m 18s
784: learn: 0.0017053 test: 0.0493611 best: 0.0493611 (784) total: 8m 25s remaining: 2m 18s
785: learn: 0.0017027 test: 0.0493625 best: 0.0493611 (784) total: 8m 25s remaining: 2m 17s
786: learn: 0.0017001 test: 0.0493432 best: 0.0493432 (786) total: 8m 26s remaining: 2m 17s
787: learn: 0.0016970 test: 0.0493629 best: 0.0493432 (786) total: 8m 27s remaining: 2m 16s
788: learn: 0.0016895 test: 0.0493817 best: 0.0493432 (786) total: 8m 27s remaining: 2m 15s
789: learn: 0.0016872 test: 0.0493814 best: 0.0493432 (786) total: 8m 28s remaining: 2m 15s
790: learn: 0.0016838 test: 0.0493852 best: 0.0493432 (786) total: 8m 29s remaining: 2m 14s
791: learn: 0.0016829 test: 0.0493835 best: 0.0493432 (786) total: 8m 29s remaining: 2m 13s
792: learn: 0.0016792 test: 0.0493959 best: 0.0493432 (786) total: 8m 30s remaining: 2m 13s
```

```
793: learn: 0.0016751 test: 0.0493988 best: 0.0493432 (786) total: 8m 31s remaining: 2m 12s
794: learn: 0.0016697 test: 0.0493935 best: 0.0493432 (786) total: 8m 31s remaining: 2m 11s
795: learn: 0.0016664 test: 0.0494014 best: 0.0493432 (786) total: 8m 32s remaining: 2m 11s
796: learn: 0.0016624 test: 0.0493856 best: 0.0493432 (786) total: 8m 33s remaining: 2m 10s
797: learn: 0.0016543 test: 0.0493775 best: 0.0493432 (786) total: 8m 33s remaining: 2m 10s
798: learn: 0.0016507 test: 0.0493699 best: 0.0493432 (786) total: 8m 34s remaining: 2m 9s
799: learn: 0.0016455 test: 0.0493364 best: 0.0493364 (799) total: 8m 35s remaining: 2m 8s
800: learn: 0.0016429 test: 0.0493745 best: 0.0493364 (799) total: 8m 35s remaining: 2m 8s
801: learn: 0.0016399 test: 0.0493750 best: 0.0493364 (799) total: 8m 36s remaining: 2m 7s
802: learn: 0.0016310 test: 0.0493034 best: 0.0493034 (802) total: 8m 37s remaining: 2m 6s
803: learn: 0.0016281 test: 0.0493316 best: 0.0493034 (802) total: 8m 37s remaining: 2m 6s
804: learn: 0.0016239 test: 0.0493663 best: 0.0493034 (802) total: 8m 38s remaining: 2m 5s
805: learn: 0.0016212 test: 0.0493583 best: 0.0493034 (802) total: 8m 38s remaining: 2m 4s
806: learn: 0.0016184 test: 0.0493541 best: 0.0493034 (802) total: 8m 39s remaining: 2m 4s
807: learn: 0.0016146 test: 0.0493599 best: 0.0493034 (802) total: 8m 40s remaining: 2m 3s
808: learn: 0.0016111 test: 0.0493465 best: 0.0493034 (802) total: 8m 40s remaining: 2m 2s
809: learn: 0.0016066 test: 0.0493566 best: 0.0493034 (802) total: 8m 41s remaining: 2m 2s
810: learn: 0.0016040 test: 0.0493769 best: 0.0493034 (802) total: 8m 42s remaining: 2m 1s
811: learn: 0.0016003 test: 0.0493618 best: 0.0493034 (802) total: 8m 42s remaining: 2m 1s
812: learn: 0.0015987 test: 0.0494045 best: 0.0493034 (802) total: 8m 43s remaining: 2m
813: learn: 0.0015966 test: 0.0494127 best: 0.0493034 (802) total: 8m 44s remaining: 1m 59s
814: learn: 0.0015938 test: 0.0494281 best: 0.0493034 (802) total: 8m 44s remaining: 1m 59s
815: learn: 0.0015912 test: 0.0494097 best: 0.0493034 (802) total: 8m 45s remaining: 1m 58s
816: learn: 0.0015877 test: 0.0494170 best: 0.0493034 (802) total: 8m 46s remaining: 1m 57s
817: learn: 0.0015849 test: 0.0494015 best: 0.0493034 (802) total: 8m 46s remaining: 1m 57s
818: learn: 0.0015828 test: 0.0494109 best: 0.0493034 (802) total: 8m 47s remaining: 1m 56s
819: learn: 0.0015810 test: 0.0494089 best: 0.0493034 (802) total: 8m 48s remaining: 1m 55s
820: learn: 0.0015787 test: 0.0494053 best: 0.0493034 (802) total: 8m 48s remaining: 1m 55s
821: learn: 0.0015752 test: 0.0493957 best: 0.0493034 (802) total: 8m 49s remaining: 1m 54s
822: learn: 0.0015734 test: 0.0493872 best: 0.0493034 (802) total: 8m 49s remaining: 1m 53s
823: learn: 0.0015665 test: 0.0493358 best: 0.0493034 (802) total: 8m 50s remaining: 1m 53s
824: learn: 0.0015649 test: 0.0493659 best: 0.0493034 (802) total: 8m 51s remaining: 1m 52s
825: learn: 0.0015598 test: 0.0493094 best: 0.0493034 (802) total: 8m 52s remaining: 1m 52s
826: learn: 0.0015572 test: 0.0493294 best: 0.0493034 (802) total: 8m 52s remaining: 1m 51s
827: learn: 0.0015549 test: 0.0493437 best: 0.0493034 (802) total: 8m 53s remaining: 1m 50s
828: learn: 0.0015530 test: 0.0493585 best: 0.0493034 (802) total: 8m 54s remaining: 1m 50s
829: learn: 0.0015490 test: 0.0493419 best: 0.0493034 (802) total: 8m 54s remaining: 1m 49s
830: learn: 0.0015441 test: 0.0493388 best: 0.0493034 (802) total: 8m 55s remaining: 1m 48s
831: learn: 0.0015411 test: 0.0493307 best: 0.0493034 (802) total: 8m 56s remaining: 1m 48s
832: learn: 0.0015371 test: 0.0493298 best: 0.0493034 (802) total: 8m 56s remaining: 1m 47s
833: learn: 0.0015338 test: 0.0493136 best: 0.0493034 (802) total: 8m 57s remaining: 1m 46s
834: learn: 0.0015326 test: 0.0493309 best: 0.0493034 (802) total: 8m 57s remaining: 1m 46s
835: learn: 0.0015303 test: 0.0493400 best: 0.0493034 (802) total: 8m 58s remaining: 1m 45s
836: learn: 0.0015260 test: 0.0493176 best: 0.0493034 (802) total: 8m 59s remaining: 1m 45s
837: learn: 0.0015238 test: 0.0493188 best: 0.0493034 (802) total: 8m 59s remaining: 1m 44s
838: learn: 0.0015182 test: 0.0492540 best: 0.0492540 (838) total: 9m remaining: 1m 43s
839: learn: 0.0015118 test: 0.0492356 best: 0.0492356 (839) total: 9m 1s remaining: 1m 43s
840: learn: 0.0015095 test: 0.0492285 best: 0.0492285 (840) total: 9m 1s remaining: 1m 42s
841: learn: 0.0015056 test: 0.0492326 best: 0.0492285 (840) total: 9m 2s remaining: 1m 41s
842: learn: 0.0015041 test: 0.0492104 best: 0.0492104 (842) total: 9m 3s remaining: 1m 41s
843: learn: 0.0014998 test: 0.0492379 best: 0.0492104 (842) total: 9m 3s remaining: 1m 40s
844: learn: 0.0014966 test: 0.0492469 best: 0.0492104 (842) total: 9m 4s remaining: 1m 39s
845: learn: 0.0014946 test: 0.0492261 best: 0.0492104 (842) total: 9m 5s remaining: 1m 39s
846: learn: 0.0014912 test: 0.0492699 best: 0.0492104 (842) total: 9m 5s remaining: 1m 38s
847: learn: 0.0014818 test: 0.0492264 best: 0.0492104 (842) total: 9m 6s remaining: 1m 37s
848: learn: 0.0014794 test: 0.0492280 best: 0.0492104 (842) total: 9m 7s remaining: 1m 37s
849: learn: 0.0014776 test: 0.0491850 best: 0.0491850 (849) total: 9m 7s remaining: 1m 36s
850: learn: 0.0014761 test: 0.0491762 best: 0.0491762 (850) total: 9m 8s remaining: 1m 36s
851: learn: 0.0014742 test: 0.0491648 best: 0.0491648 (851) total: 9m 9s remaining: 1m 35s
852: learn: 0.0014664 test: 0.0491192 best: 0.0491192 (852) total: 9m 9s remaining: 1m 34s
853: learn: 0.0014644 test: 0.0491368 best: 0.0491192 (852) total: 9m 10s remaining: 1m 34s
854: learn: 0.0014627 test: 0.0491572 best: 0.0491192 (852) total: 9m 10s remaining: 1m 33s
855: learn: 0.0014593 test: 0.0491057 best: 0.0491057 (855) total: 9m 11s remaining: 1m 32s
856: learn: 0.0014575 test: 0.0491200 best: 0.0491057 (855) total: 9m 12s remaining: 1m 32s
857: learn: 0.0014545 test: 0.0491072 best: 0.0491057 (855) total: 9m 12s remaining: 1m 31s
858: learn: 0.0014523 test: 0.0491383 best: 0.0491057 (855) total: 9m 13s remaining: 1m 30s
859: learn: 0.0014507 test: 0.0491476 best: 0.0491057 (855) total: 9m 14s remaining: 1m 30s
860: learn: 0.0014459 test: 0.0491026 best: 0.0491026 (860) total: 9m 14s remaining: 1m 29s
861: learn: 0.0014441 test: 0.0491285 best: 0.0491026 (860) total: 9m 15s remaining: 1m 28s
862: learn: 0.0014428 test: 0.0491497 best: 0.0491026 (860) total: 9m 16s remaining: 1m 28s
863: learn: 0.0014404 test: 0.0491272 best: 0.0491026 (860) total: 9m 16s remaining: 1m 27s
864: learn: 0.0014382 test: 0.0491470 best: 0.0491026 (860) total: 9m 17s remaining: 1m 26s
865: learn: 0.0014365 test: 0.0491386 best: 0.0491026 (860) total: 9m 18s remaining: 1m 26s
866: learn: 0.0014340 test: 0.0491386 best: 0.0491026 (860) total: 9m 18s remaining: 1m 25s
867: learn: 0.0014296 test: 0.0491118 best: 0.0491026 (860) total: 9m 19s remaining: 1m 25s
868: learn: 0.0014276 test: 0.0491006 best: 0.0491006 (868) total: 9m 19s remaining: 1m 24s
869: learn: 0.0014200 test: 0.0490927 best: 0.0490927 (869) total: 9m 20s remaining: 1m 23s
```

```
870: learn: 0.0014167 test: 0.0491358 best: 0.0490927 (869) total: 9m 21s remaining: 1m 23s
871: learn: 0.0014151 test: 0.0491595 best: 0.0490927 (869) total: 9m 21s remaining: 1m 22s
872: learn: 0.0014110 test: 0.0491414 best: 0.0490927 (869) total: 9m 22s remaining: 1m 21s
873: learn: 0.0014098 test: 0.0491677 best: 0.0490927 (869) total: 9m 23s remaining: 1m 21s
874: learn: 0.0014086 test: 0.0491564 best: 0.0490927 (869) total: 9m 23s remaining: 1m 20s
875: learn: 0.0014063 test: 0.0491485 best: 0.0490927 (869) total: 9m 24s remaining: 1m 19s
876: learn: 0.0014028 test: 0.0491551 best: 0.0490927 (869) total: 9m 25s remaining: 1m 19s
877: learn: 0.0014005 test: 0.0491517 best: 0.0490927 (869) total: 9m 25s remaining: 1m 18s
878: learn: 0.0013986 test: 0.0491863 best: 0.0490927 (869) total: 9m 26s remaining: 1m 17s
879: learn: 0.0013969 test: 0.0491572 best: 0.0490927 (869) total: 9m 26s remaining: 1m 17s
880: learn: 0.0013943 test: 0.0491345 best: 0.0490927 (869) total: 9m 27s remaining: 1m 16s
881: learn: 0.0013919 test: 0.0491409 best: 0.0490927 (869) total: 9m 28s remaining: 1m 16s
882: learn: 0.0013882 test: 0.0491818 best: 0.0490927 (869) total: 9m 28s remaining: 1m 15s
883: learn: 0.0013855 test: 0.0491422 best: 0.0490927 (869) total: 9m 29s remaining: 1m 14s
884: learn: 0.0013830 test: 0.0491620 best: 0.0490927 (869) total: 9m 30s remaining: 1m 14s
885: learn: 0.0013803 test: 0.0491556 best: 0.0490927 (869) total: 9m 30s remaining: 1m 13s
886: learn: 0.0013779 test: 0.0491762 best: 0.0490927 (869) total: 9m 31s remaining: 1m 12s
887: learn: 0.0013751 test: 0.0491982 best: 0.0490927 (869) total: 9m 32s remaining: 1m 12s
888: learn: 0.0013728 test: 0.0491836 best: 0.0490927 (869) total: 9m 32s remaining: 1m 11s
889: learn: 0.0013709 test: 0.0492286 best: 0.0490927 (869) total: 9m 33s remaining: 1m 10s
890: learn: 0.0013692 test: 0.0492312 best: 0.0490927 (869) total: 9m 34s remaining: 1m 10s
891: learn: 0.0013670 test: 0.0492498 best: 0.0490927 (869) total: 9m 34s remaining: 1m 9s
892: learn: 0.0013660 test: 0.0492712 best: 0.0490927 (869) total: 9m 35s remaining: 1m 8s
893: learn: 0.0013630 test: 0.0492651 best: 0.0490927 (869) total: 9m 36s remaining: 1m 8s
894: learn: 0.0013599 test: 0.0492596 best: 0.0490927 (869) total: 9m 36s remaining: 1m 7s
895: learn: 0.0013553 test: 0.0492404 best: 0.0490927 (869) total: 9m 37s remaining: 1m 7s
896: learn: 0.0013519 test: 0.0492163 best: 0.0490927 (869) total: 9m 38s remaining: 1m 6s
897: learn: 0.0013482 test: 0.0491781 best: 0.0490927 (869) total: 9m 38s remaining: 1m 5s
898: learn: 0.0013464 test: 0.0491950 best: 0.0490927 (869) total: 9m 39s remaining: 1m 5s
899: learn: 0.0013450 test: 0.0491686 best: 0.0490927 (869) total: 9m 40s remaining: 1m 4s
900: learn: 0.0013412 test: 0.0491832 best: 0.0490927 (869) total: 9m 40s remaining: 1m 3s
901: learn: 0.0013390 test: 0.0492374 best: 0.0490927 (869) total: 9m 41s remaining: 1m 3s
902: learn: 0.0013380 test: 0.0492367 best: 0.0490927 (869) total: 9m 41s remaining: 1m 2s
903: learn: 0.0013355 test: 0.0492486 best: 0.0490927 (869) total: 9m 42s remaining: 1m 1s
904: learn: 0.0013339 test: 0.0492539 best: 0.0490927 (869) total: 9m 43s remaining: 1m 1s
905: learn: 0.0013323 test: 0.0492444 best: 0.0490927 (869) total: 9m 43s remaining: 1m
906: learn: 0.0013304 test: 0.0492389 best: 0.0490927 (869) total: 9m 44s remaining: 59.9s
907: learn: 0.0013284 test: 0.0492461 best: 0.0490927 (869) total: 9m 45s remaining: 59.3s
908: learn: 0.0013271 test: 0.0492389 best: 0.0490927 (869) total: 9m 45s remaining: 58.6s
909: learn: 0.0013244 test: 0.0492409 best: 0.0490927 (869) total: 9m 46s remaining: 58s
910: learn: 0.0013225 test: 0.0492227 best: 0.0490927 (869) total: 9m 47s remaining: 57.3s
911: learn: 0.0013195 test: 0.0492503 best: 0.0490927 (869) total: 9m 47s remaining: 56.7s
912: learn: 0.0013183 test: 0.0492741 best: 0.0490927 (869) total: 9m 48s remaining: 56.1s
913: learn: 0.0013153 test: 0.0492817 best: 0.0490927 (869) total: 9m 48s remaining: 55.4s
914: learn: 0.0013117 test: 0.0492965 best: 0.0490927 (869) total: 9m 49s remaining: 54.8s
915: learn: 0.0013063 test: 0.0492809 best: 0.0490927 (869) total: 9m 50s remaining: 54.1s
916: learn: 0.0013021 test: 0.0492581 best: 0.0490927 (869) total: 9m 50s remaining: 53.5s
917: learn: 0.0013007 test: 0.0492484 best: 0.0490927 (869) total: 9m 51s remaining: 52.8s
918: learn: 0.0012943 test: 0.0491832 best: 0.0490927 (869) total: 9m 52s remaining: 52.2s
919: learn: 0.0012934 test: 0.0491901 best: 0.0490927 (869) total: 9m 52s remaining: 51.5s
920: learn: 0.0012896 test: 0.0491772 best: 0.0490927 (869) total: 9m 53s remaining: 50.9s
921: learn: 0.0012872 test: 0.0491821 best: 0.0490927 (869) total: 9m 54s remaining: 50.3s
922: learn: 0.0012834 test: 0.0491298 best: 0.0490927 (869) total: 9m 54s remaining: 49.6s
923: learn: 0.0012806 test: 0.0491501 best: 0.0490927 (869) total: 9m 55s remaining: 49s
924: learn: 0.0012769 test: 0.0491376 best: 0.0490927 (869) total: 9m 56s remaining: 48.3s
925: learn: 0.0012752 test: 0.0491591 best: 0.0490927 (869) total: 9m 56s remaining: 47.7s
926: learn: 0.0012739 test: 0.0491485 best: 0.0490927 (869) total: 9m 57s remaining: 47s
927: learn: 0.0012713 test: 0.0491511 best: 0.0490927 (869) total: 9m 58s remaining: 46.4s
928: learn: 0.0012694 test: 0.0491429 best: 0.0490927 (869) total: 9m 58s remaining: 45.8s
929: learn: 0.0012665 test: 0.0491053 best: 0.0490927 (869) total: 9m 59s remaining: 45.1s
930: learn: 0.0012634 test: 0.0490777 best: 0.0490777 (930) total: 10m remaining: 44.5s
931: learn: 0.0012616 test: 0.0491010 best: 0.0490777 (930) total: 10m remaining: 43.8s
932: learn: 0.0012591 test: 0.0490780 best: 0.0490777 (930) total: 10m 1s remaining: 43.2s
933: learn: 0.0012569 test: 0.0491001 best: 0.0490777 (930) total: 10m 2s remaining: 42.5s
934: learn: 0.0012541 test: 0.0491078 best: 0.0490777 (930) total: 10m 2s remaining: 41.9s
935: learn: 0.0012533 test: 0.0491128 best: 0.0490777 (930) total: 10m 3s remaining: 41.3s
936: learn: 0.0012505 test: 0.0490735 best: 0.0490735 (936) total: 10m 4s remaining: 40.6s
937: learn: 0.0012483 test: 0.0490800 best: 0.0490735 (936) total: 10m 4s remaining: 40s
938: learn: 0.0012421 test: 0.0490214 best: 0.0490214 (938) total: 10m 5s remaining: 39.3s
939: learn: 0.0012389 test: 0.0490221 best: 0.0490214 (938) total: 10m 6s remaining: 38.7s
940: learn: 0.0012364 test: 0.0490668 best: 0.0490214 (938) total: 10m 6s remaining: 38s
941: learn: 0.0012350 test: 0.0490629 best: 0.0490214 (938) total: 10m 7s remaining: 37.4s
942: learn: 0.0012333 test: 0.0490809 best: 0.0490214 (938) total: 10m 8s remaining: 36.8s
943: learn: 0.0012321 test: 0.0490954 best: 0.0490214 (938) total: 10m 8s remaining: 36.1s
944: learn: 0.0012300 test: 0.0490956 best: 0.0490214 (938) total: 10m 9s remaining: 35.5s
945: learn: 0.0012251 test: 0.0490374 best: 0.0490214 (938) total: 10m 9s remaining: 34.8s
946: learn: 0.0012228 test: 0.0490545 best: 0.0490214 (938) total: 10m 10s remaining: 34.2s
```

```
947: learn: 0.0012215 test: 0.0490718 best: 0.0490214 (938) total: 10m 11s remaining: 33.5s
948: learn: 0.0012186 test: 0.0491123 best: 0.0490214 (938) total: 10m 11s remaining: 32.9s
949: learn: 0.0012167 test: 0.0490995 best: 0.0490214 (938) total: 10m 12s remaining: 32.2s
950: learn: 0.0012067 test: 0.0490355 best: 0.0490214 (938) total: 10m 13s remaining: 31.6s
951: learn: 0.0012046 test: 0.0490249 best: 0.0490214 (938) total: 10m 13s remaining: 30.9s
952: learn: 0.0012025 test: 0.0490709 best: 0.0490214 (938) total: 10m 14s remaining: 30.3s
953: learn: 0.0012012 test: 0.0490747 best: 0.0490214 (938) total: 10m 15s remaining: 29.7s
954: learn: 0.0011999 test: 0.0490951 best: 0.0490214 (938) total: 10m 15s remaining: 29s
955: learn: 0.0011954 test: 0.0490270 best: 0.0490214 (938) total: 10m 16s remaining: 28.4s
956: learn: 0.0011927 test: 0.0490625 best: 0.0490214 (938) total: 10m 17s remaining: 27.7s
957: learn: 0.0011914 test: 0.0490582 best: 0.0490214 (938) total: 10m 17s remaining: 27.1s
958: learn: 0.0011899 test: 0.0491039 best: 0.0490214 (938) total: 10m 18s remaining: 26.4s
959: learn: 0.0011853 test: 0.0491161 best: 0.0490214 (938) total: 10m 18s remaining: 25.8s
960: learn: 0.0011838 test: 0.0491091 best: 0.0490214 (938) total: 10m 19s remaining: 25.1s
961: learn: 0.0011812 test: 0.0491357 best: 0.0490214 (938) total: 10m 20s remaining: 24.5s
962: learn: 0.0011772 test: 0.0491121 best: 0.0490214 (938) total: 10m 20s remaining: 23.9s
963: learn: 0.0011759 test: 0.0491173 best: 0.0490214 (938) total: 10m 21s remaining: 23.2s
964: learn: 0.0011749 test: 0.0491206 best: 0.0490214 (938) total: 10m 22s remaining: 22.6s
965: learn: 0.0011738 test: 0.0490944 best: 0.0490214 (938) total: 10m 22s remaining: 21.9s
966: learn: 0.0011722 test: 0.0490988 best: 0.0490214 (938) total: 10m 23s remaining: 21.3s
967: learn: 0.0011705 test: 0.0490479 best: 0.0490214 (938) total: 10m 23s remaining: 20.6s
968: learn: 0.0011683 test: 0.0490665 best: 0.0490214 (938) total: 10m 24s remaining: 20s
969: learn: 0.0011671 test: 0.0490799 best: 0.0490214 (938) total: 10m 25s remaining: 19.3s
970: learn: 0.0011656 test: 0.0490898 best: 0.0490214 (938) total: 10m 25s remaining: 18.7s
971: learn: 0.0011633 test: 0.0491024 best: 0.0490214 (938) total: 10m 26s remaining: 18s
972: learn: 0.0011611 test: 0.0491070 best: 0.0490214 (938) total: 10m 27s remaining: 17.4s
973: learn: 0.0011596 test: 0.0491150 best: 0.0490214 (938) total: 10m 27s remaining: 16.8s
974: learn: 0.0011565 test: 0.0490995 best: 0.0490214 (938) total: 10m 28s remaining: 16.1s
975: learn: 0.0011545 test: 0.0491064 best: 0.0490214 (938) total: 10m 29s remaining: 15.5s
976: learn: 0.0011530 test: 0.0491034 best: 0.0490214 (938) total: 10m 29s remaining: 14.8s
977: learn: 0.0011471 test: 0.0490384 best: 0.0490214 (938) total: 10m 30s remaining: 14.2s
978: learn: 0.0011452 test: 0.0490610 best: 0.0490214 (938) total: 10m 30s remaining: 13.5s
979: learn: 0.0011435 test: 0.0490474 best: 0.0490214 (938) total: 10m 31s remaining: 12.9s
980: learn: 0.0011411 test: 0.0490415 best: 0.0490214 (938) total: 10m 32s remaining: 12.2s
981: learn: 0.0011380 test: 0.0490215 best: 0.0490214 (938) total: 10m 32s remaining: 11.6s
982: learn: 0.0011366 test: 0.0490391 best: 0.0490214 (938) total: 10m 33s remaining: 11s
983: learn: 0.0011345 test: 0.0490424 best: 0.0490214 (938) total: 10m 34s remaining: 10.3s
984: learn: 0.0011323 test: 0.0490094 best: 0.0490094 (984) total: 10m 34s remaining: 9.67s
985: learn: 0.0011312 test: 0.0490009 best: 0.0490009 (985) total: 10m 35s remaining: 9.02s
986: learn: 0.0011295 test: 0.0490500 best: 0.0490009 (985) total: 10m 36s remaining: 8.38s
987: learn: 0.0011274 test: 0.0490237 best: 0.0490009 (985) total: 10m 36s remaining: 7.73s
988: learn: 0.0011255 test: 0.0490311 best: 0.0490009 (985) total: 10m 37s remaining: 7.09s
989: learn: 0.0011242 test: 0.0490616 best: 0.0490009 (985) total: 10m 37s remaining: 6.44s
990: learn: 0.0011229 test: 0.0490522 best: 0.0490009 (985) total: 10m 38s remaining: 5.8s
991: learn: 0.0011211 test: 0.0490251 best: 0.0490009 (985) total: 10m 39s remaining: 5.16s
992: learn: 0.0011190 test: 0.0490224 best: 0.0490009 (985) total: 10m 39s remaining: 4.51s
993: learn: 0.0011179 test: 0.0490206 best: 0.0490009 (985) total: 10m 40s remaining: 3.87s
994: learn: 0.0011159 test: 0.0490702 best: 0.0490009 (985) total: 10m 41s remaining: 3.22s
995: learn: 0.0011146 test: 0.0490780 best: 0.0490009 (985) total: 10m 41s remaining: 2.58s
996: learn: 0.0011132 test: 0.0491140 best: 0.0490009 (985) total: 10m 42s remaining: 1.93s
997: learn: 0.0011059 test: 0.0490805 best: 0.0490009 (985) total: 10m 43s remaining: 1.29s
998: learn: 0.0011034 test: 0.0490240 best: 0.0490009 (985) total: 10m 43s remaining: 644ms
999: learn: 0.0011017 test: 0.0490495 best: 0.0490009 (985) total: 10m 44s remaining: 0us

bestTest = 0.04900087896
bestIteration = 985

Shrink model to first 986 iterations.
Count of trees in model = 986
TRAIN LOSS is  0.0011312425089442042
CV LOSS is  0.04900087895905964
```

In [13]:

```
# training with full train data

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[13]:

((3346256, 120), (836564, 120), 3346256, 836564)

((3346256, 120), (836564, 120), 3346256, 836564)

In [ ]:

```
from catboost import CatBoostClassifier
from catboost import Pool
train_dataset = Pool(data=train_df,
                     label=y_train)

eval_dataset = Pool(data=val_df,
                    label=y_val)

clf = CatBoostClassifier(loss_function='MultiClass', depth=10, iterations=1000, l2_leaf_reg= 1, lea
rning_rate= 0.38)
clf.fit(train_dataset, use_best_model=True, eval_set=eval_dataset)
print("Count of trees in model = {}".format(clf.tree_count_))
```

```
0: learn: 0.9843333 test: 0.9840206 best: 0.9840206 (0) total: 11.4s remaining: 3h 9m 1s
1: learn: 0.7734221 test: 0.7732115 best: 0.7732115 (1) total: 22.4s remaining: 3h 6m 26s
2: learn: 0.6014114 test: 0.6013031 best: 0.6013031 (2) total: 37.7s remaining: 3h 28m 54s
3: learn: 0.5213668 test: 0.5211523 best: 0.5211523 (3) total: 52.5s remaining: 3h 37m 50s
4: learn: 0.4714117 test: 0.4710449 best: 0.4710449 (4) total: 1m 4s remaining: 3h 32m 29s
5: learn: 0.4219616 test: 0.4216771 best: 0.4216771 (5) total: 1m 17s remaining: 3h 34m 53s
6: learn: 0.3775904 test: 0.3774239 best: 0.3774239 (6) total: 1m 32s remaining: 3h 37m 34s
7: learn: 0.3521778 test: 0.3521122 best: 0.3521122 (7) total: 1m 43s remaining: 3h 34m 49s
8: learn: 0.3261198 test: 0.3261392 best: 0.3261392 (8) total: 1m 57s remaining: 3h 35m 49s
9: learn: 0.3023050 test: 0.3022509 best: 0.3022509 (9) total: 2m 11s remaining: 3h 36m 31s
10: learn: 0.2863865 test: 0.2864168 best: 0.2864168 (10) total: 2m 22s remaining: 3h 34m 8s
11: learn: 0.2718743 test: 0.2719093 best: 0.2719093 (11) total: 2m 35s remaining: 3h 32m 53s
12: learn: 0.2558200 test: 0.2559923 best: 0.2559923 (12) total: 2m 49s remaining: 3h 34m 17s
13: learn: 0.2426807 test: 0.2427943 best: 0.2427943 (13) total: 3m 2s remaining: 3h 33m 44s
14: learn: 0.2331724 test: 0.2332384 best: 0.2332384 (14) total: 3m 14s remaining: 3h 33m 3s
15: learn: 0.2219549 test: 0.2220957 best: 0.2220957 (15) total: 3m 28s remaining: 3h 33m 12s
16: learn: 0.2106215 test: 0.2106793 best: 0.2106793 (16) total: 3m 41s remaining: 3h 33m 2s
17: learn: 0.2037639 test: 0.2039201 best: 0.2039201 (17) total: 3m 52s remaining: 3h 31m 36s
18: learn: 0.1983782 test: 0.1985488 best: 0.1985488 (18) total: 4m 5s remaining: 3h 31m 2s
19: learn: 0.1898226 test: 0.1901266 best: 0.1901266 (19) total: 4m 18s remaining: 3h 31m 11s
20: learn: 0.1828189 test: 0.1830237 best: 0.1830237 (20) total: 4m 32s remaining: 3h 31m 24s
21: learn: 0.1785932 test: 0.1787509 best: 0.1787509 (21) total: 4m 44s remaining: 3h 30m 43s
22: learn: 0.1711984 test: 0.1714165 best: 0.1714165 (22) total: 4m 58s remaining: 3h 31m 33s
23: learn: 0.1631174 test: 0.1633787 best: 0.1633787 (23) total: 5m 13s remaining: 3h 32m 22s
24: learn: 0.1580737 test: 0.1583655 best: 0.1583655 (24) total: 5m 25s remaining: 3h 31m 19s
25: learn: 0.1521473 test: 0.1524651 best: 0.1524651 (25) total: 5m 38s remaining: 3h 31m 26s
26: learn: 0.1473180 test: 0.1476742 best: 0.1476742 (26) total: 5m 51s remaining: 3h 31m 1s
27: learn: 0.1441374 test: 0.1445128 best: 0.1445128 (27) total: 6m 2s remaining: 3h 29m 40s
28: learn: 0.1404018 test: 0.1407556 best: 0.1407556 (28) total: 6m 14s remaining: 3h 29m 2s
29: learn: 0.1365464 test: 0.1369745 best: 0.1369745 (29) total: 6m 27s remaining: 3h 29m 3s
30: learn: 0.1334354 test: 0.1338312 best: 0.1338312 (30) total: 6m 41s remaining: 3h 29m 8s
31: learn: 0.1311847 test: 0.1316128 best: 0.1316128 (31) total: 6m 52s remaining: 3h 28m 4s
32: learn: 0.1283847 test: 0.1288157 best: 0.1288157 (32) total: 7m 4s remaining: 3h 27m 22s
33: learn: 0.1252334 test: 0.1256851 best: 0.1256851 (33) total: 7m 17s remaining: 3h 27m 13s
34: learn: 0.1212450 test: 0.1217345 best: 0.1217345 (34) total: 7m 30s remaining: 3h 27m 7s
35: learn: 0.1179135 test: 0.1183962 best: 0.1183962 (35) total: 7m 44s remaining: 3h 27m 19s
36: learn: 0.1145977 test: 0.1151185 best: 0.1151185 (36) total: 7m 58s remaining: 3h 27m 34s
37: learn: 0.1123735 test: 0.1129187 best: 0.1129187 (37) total: 8m 11s remaining: 3h 27m 16s
38: learn: 0.1095214 test: 0.1100657 best: 0.1100657 (38) total: 8m 24s remaining: 3h 27m 13s
39: learn: 0.1068032 test: 0.1073022 best: 0.1073022 (39) total: 8m 39s remaining: 3h 27m 37s
40: learn: 0.1047267 test: 0.1052558 best: 0.1052558 (40) total: 8m 50s remaining: 3h 26m 52s
41: learn: 0.1023042 test: 0.1028193 best: 0.1028193 (41) total: 9m 2s remaining: 3h 26m 24s
42: learn: 0.1004105 test: 0.1009241 best: 0.1009241 (42) total: 9m 15s remaining: 3h 26m 6s
43: learn: 0.0977978 test: 0.0982836 best: 0.0982836 (43) total: 9m 28s remaining: 3h 25m 59s
44: learn: 0.0945281 test: 0.0950491 best: 0.0950491 (44) total: 9m 43s remaining: 3h 26m 16s
45: learn: 0.0914119 test: 0.0918820 best: 0.0918820 (45) total: 9m 57s remaining: 3h 26m 35s
46: learn: 0.0896470 test: 0.0901668 best: 0.0901668 (46) total: 10m 9s remaining: 3h 26m 6s
47: learn: 0.0883524 test: 0.0888931 best: 0.0888931 (47) total: 10m 21s remaining: 3h 25m 25s
48: learn: 0.0857668 test: 0.0863364 best: 0.0863364 (48) total: 10m 34s remaining: 3h 25m 23s
49: learn: 0.0838281 test: 0.0844327 best: 0.0844327 (49) total: 10m 48s remaining: 3h 25m 15s
50: learn: 0.0822441 test: 0.0828523 best: 0.0828523 (50) total: 11m remaining: 3h 24m 51s
51: learn: 0.0804755 test: 0.0810758 best: 0.0810758 (51) total: 11m 13s remaining: 3h 24m 38s
52: learn: 0.0793950 test: 0.0799945 best: 0.0799945 (52) total: 11m 24s remaining: 3h 23m 55s
53: learn: 0.0783172 test: 0.0789500 best: 0.0789500 (53) total: 11m 36s remaining: 3h 23m 22s
54: learn: 0.0767186 test: 0.0773581 best: 0.0773581 (54) total: 11m 51s remaining: 3h 23m 41s
55: learn: 0.0753406 test: 0.0759764 best: 0.0759764 (55) total: 12m 3s remaining: 3h 23m 20s
56: learn: 0.0741043 test: 0.0747335 best: 0.0747335 (56) total: 12m 15s remaining: 3h 22m 49s
57: learn: 0.0729079 test: 0.0735389 best: 0.0735389 (57) total: 12m 28s remaining: 3h 22m 28s
58: learn: 0.0714995 test: 0.0721115 best: 0.0721115 (58) total: 12m 41s remaining: 3h 22m 26s
```

```
59: learn: 0.0699677 test: 0.0705588 best: 0.0705588 (59) total: 12m 54s remaining: 3h 22m 15s
60: learn: 0.0685815 test: 0.0691467 best: 0.0691467 (60) total: 13m 7s remaining: 3h 22m 6s
61: learn: 0.0676093 test: 0.0681861 best: 0.0681861 (61) total: 13m 19s remaining: 3h 21m 33s
62: learn: 0.0662451 test: 0.0668339 best: 0.0668339 (62) total: 13m 32s remaining: 3h 21m 23s
63: learn: 0.0646238 test: 0.0652447 best: 0.0652447 (63) total: 13m 46s remaining: 3h 21m 23s
64: learn: 0.0630641 test: 0.0637013 best: 0.0637013 (64) total: 13m 59s remaining: 3h 21m 14s
65: learn: 0.0621514 test: 0.0627861 best: 0.0627861 (65) total: 14m 11s remaining: 3h 20m 50s
66: learn: 0.0612233 test: 0.0618590 best: 0.0618590 (66) total: 14m 24s remaining: 3h 20m 38s
67: learn: 0.0601404 test: 0.0607586 best: 0.0607586 (67) total: 14m 37s remaining: 3h 20m 21s
68: learn: 0.0590671 test: 0.0596590 best: 0.0596590 (68) total: 14m 49s remaining: 3h 20m
69: learn: 0.0582572 test: 0.0588261 best: 0.0588261 (69) total: 15m 1s remaining: 3h 19m 32s
70: learn: 0.0567770 test: 0.0573285 best: 0.0573285 (70) total: 15m 16s remaining: 3h 19m 46s
71: learn: 0.0558494 test: 0.0564162 best: 0.0564162 (71) total: 15m 30s remaining: 3h 19m 47s
72: learn: 0.0548243 test: 0.0554090 best: 0.0554090 (72) total: 15m 44s remaining: 3h 19m 50s
73: learn: 0.0542734 test: 0.0548787 best: 0.0548787 (73) total: 15m 55s remaining: 3h 19m 17s
74: learn: 0.0529792 test: 0.0535985 best: 0.0535985 (74) total: 16m 9s remaining: 3h 19m 17s
75: learn: 0.0522454 test: 0.0528540 best: 0.0528540 (75) total: 16m 22s remaining: 3h 18m 59s
76: learn: 0.0513031 test: 0.0519355 best: 0.0519355 (76) total: 16m 35s remaining: 3h 18m 55s
77: learn: 0.0508501 test: 0.0514900 best: 0.0514900 (77) total: 16m 46s remaining: 3h 18m 20s
78: learn: 0.0493733 test: 0.0500143 best: 0.0500143 (78) total: 17m 2s remaining: 3h 18m 36s
79: learn: 0.0481304 test: 0.0487646 best: 0.0487646 (79) total: 17m 15s remaining: 3h 18m 28s
80: learn: 0.0475780 test: 0.0482073 best: 0.0482073 (80) total: 17m 26s remaining: 3h 17m 55s
81: learn: 0.0464645 test: 0.0470883 best: 0.0470883 (81) total: 17m 41s remaining: 3h 18m 2s
82: learn: 0.0455226 test: 0.0461216 best: 0.0461216 (82) total: 17m 54s remaining: 3h 17m 55s
83: learn: 0.0445165 test: 0.0451406 best: 0.0451406 (83) total: 18m 7s remaining: 3h 17m 38s
84: learn: 0.0438598 test: 0.0444809 best: 0.0444809 (84) total: 18m 20s remaining: 3h 17m 23s
85: learn: 0.0429913 test: 0.0436022 best: 0.0436022 (85) total: 18m 34s remaining: 3h 17m 20s
86: learn: 0.0422322 test: 0.0428266 best: 0.0428266 (86) total: 18m 46s remaining: 3h 17m 6s
87: learn: 0.0411930 test: 0.0418022 best: 0.0418022 (87) total: 19m remaining: 3h 17m
88: learn: 0.0402833 test: 0.0408807 best: 0.0408807 (88) total: 19m 14s remaining: 3h 16m 59s
89: learn: 0.0397350 test: 0.0403320 best: 0.0403320 (89) total: 19m 26s remaining: 3h 16m 39s
90: learn: 0.0391661 test: 0.0397690 best: 0.0397690 (90) total: 19m 39s remaining: 3h 16m 24s
91: learn: 0.0384924 test: 0.0390992 best: 0.0390992 (91) total: 19m 52s remaining: 3h 16m 12s
92: learn: 0.0379406 test: 0.0385450 best: 0.0385450 (92) total: 20m 5s remaining: 3h 16m
93: learn: 0.0374794 test: 0.0380801 best: 0.0380801 (93) total: 20m 18s remaining: 3h 15m 45s
94: learn: 0.0366272 test: 0.0372037 best: 0.0372037 (94) total: 20m 32s remaining: 3h 15m 45s
95: learn: 0.0361479 test: 0.0367130 best: 0.0367130 (95) total: 20m 44s remaining: 3h 15m 19s
96: learn: 0.0355927 test: 0.0361720 best: 0.0361720 (96) total: 20m 57s remaining: 3h 15m 7s
97: learn: 0.0348699 test: 0.0354484 best: 0.0354484 (97) total: 21m 10s remaining: 3h 14m 55s
98: learn: 0.0342456 test: 0.0348273 best: 0.0348273 (98) total: 21m 23s remaining: 3h 14m 43s
99: learn: 0.0336126 test: 0.0341923 best: 0.0341923 (99) total: 21m 36s remaining: 3h 14m 29s
100: learn: 0.0334320 test: 0.0340051 best: 0.0340051 (100) total: 21m 47s remaining: 3h 13m 55s
101: learn: 0.0329296 test: 0.0335119 best: 0.0335119 (101) total: 22m remaining: 3h 13m 46s
102: learn: 0.0322233 test: 0.0327925 best: 0.0327925 (102) total: 22m 13s remaining: 3h 13m 36s
103: learn: 0.0317841 test: 0.0323362 best: 0.0323362 (103) total: 22m 26s remaining: 3h 13m 23s
104: learn: 0.0313335 test: 0.0318825 best: 0.0318825 (104) total: 22m 39s remaining: 3h 13m 12s
105: learn: 0.0307496 test: 0.0312990 best: 0.0312990 (105) total: 22m 53s remaining: 3h 13m 3s
106: learn: 0.0299917 test: 0.0305308 best: 0.0305308 (106) total: 23m 7s remaining: 3h 12m 58s
107: learn: 0.0295283 test: 0.0300770 best: 0.0300770 (107) total: 23m 20s remaining: 3h 12m 47s
108: learn: 0.0290408 test: 0.0295912 best: 0.0295912 (108) total: 23m 34s remaining: 3h 12m 42s
109: learn: 0.0287618 test: 0.0293020 best: 0.0293020 (109) total: 23m 46s remaining: 3h 12m 19s
110: learn: 0.0285165 test: 0.0290646 best: 0.0290646 (110) total: 23m 57s remaining: 3h 11m 55s
111: learn: 0.0279289 test: 0.0284806 best: 0.0284806 (111) total: 24m 11s remaining: 3h 11m 44s
112: learn: 0.0274882 test: 0.0280156 best: 0.0280156 (112) total: 24m 23s remaining: 3h 11m 31s
113: learn: 0.0270008 test: 0.0275107 best: 0.0275107 (113) total: 24m 37s remaining: 3h 11m 19s
114: learn: 0.0264535 test: 0.0269535 best: 0.0269535 (114) total: 24m 51s remaining: 3h 11m 18s
115: learn: 0.0260887 test: 0.0265682 best: 0.0265682 (115) total: 25m 4s remaining: 3h 11m 3s
116: learn: 0.0257081 test: 0.0261803 best: 0.0261803 (116) total: 25m 16s remaining: 3h 10m 45s
117: learn: 0.0252360 test: 0.0256921 best: 0.0256921 (117) total: 25m 30s remaining: 3h 10m 37s
118: learn: 0.0250719 test: 0.0255286 best: 0.0255286 (118) total: 25m 41s remaining: 3h 10m 12s
119: learn: 0.0247212 test: 0.0251737 best: 0.0251737 (119) total: 25m 54s remaining: 3h 10m 1s
120: learn: 0.0241619 test: 0.0246053 best: 0.0246053 (120) total: 26m 9s remaining: 3h 9m 58s
121: learn: 0.0237815 test: 0.0242161 best: 0.0242161 (121) total: 26m 20s remaining: 3h 9m 35s
122: learn: 0.0235090 test: 0.0239337 best: 0.0239337 (122) total: 26m 33s remaining: 3h 9m 23s
123: learn: 0.0231137 test: 0.0235321 best: 0.0235321 (123) total: 26m 47s remaining: 3h 9m 19s
124: learn: 0.0228193 test: 0.0232390 best: 0.0232390 (124) total: 27m remaining: 3h 9m 3s
125: learn: 0.0224585 test: 0.0228745 best: 0.0228745 (125) total: 27m 12s remaining: 3h 8m 46s
126: learn: 0.0221488 test: 0.0225673 best: 0.0225673 (126) total: 27m 26s remaining: 3h 8m 35s
127: learn: 0.0217950 test: 0.0222059 best: 0.0222059 (127) total: 27m 38s remaining: 3h 8m 18s
128: learn: 0.0216812 test: 0.0220939 best: 0.0220939 (128) total: 27m 50s remaining: 3h 7m 57s
129: learn: 0.0214013 test: 0.0218172 best: 0.0218172 (129) total: 28m 3s remaining: 3h 7m 44s
130: learn: 0.0208847 test: 0.0212983 best: 0.0212983 (130) total: 28m 17s remaining: 3h 7m 42s
131: learn: 0.0206939 test: 0.0211032 best: 0.0211032 (131) total: 28m 30s remaining: 3h 7m 30s
132: learn: 0.0205928 test: 0.0210021 best: 0.0210021 (132) total: 28m 41s remaining: 3h 7m 2s
133: learn: 0.0203553 test: 0.0207652 best: 0.0207652 (133) total: 28m 52s remaining: 3h 6m 39s
134: learn: 0.0198600 test: 0.0202674 best: 0.0202674 (134) total: 29m 7s remaining: 3h 6m 39s
135: learn: 0.0194911 test: 0.0198965 best: 0.0198965 (135) total: 29m 21s remaining: 3h 6m 30s
```

```
136: learn: 0.0191098 test: 0.0195108 best: 0.0195108 (136) total: 29m 35s remaining: 3h 6m 24s
137: learn: 0.0189768 test: 0.0193795 best: 0.0193795 (137) total: 29m 46s remaining: 3h 6m
138: learn: 0.0186981 test: 0.0190949 best: 0.0190949 (138) total: 30m remaining: 3h 5m 49s
139: learn: 0.0185039 test: 0.0188907 best: 0.0188907 (139) total: 30m 12s remaining: 3h 5m 34s
140: learn: 0.0183572 test: 0.0187436 best: 0.0187436 (140) total: 30m 25s remaining: 3h 5m 19s
141: learn: 0.0181844 test: 0.0185749 best: 0.0185749 (141) total: 30m 37s remaining: 3h 5m 4s
142: learn: 0.0178187 test: 0.0182059 best: 0.0182059 (142) total: 30m 52s remaining: 3h 4m 59s
143: learn: 0.0174944 test: 0.0178816 best: 0.0178816 (143) total: 31m 4s remaining: 3h 4m 46s
144: learn: 0.0172837 test: 0.0176752 best: 0.0176752 (144) total: 31m 19s remaining: 3h 4m 39s
145: learn: 0.0169923 test: 0.0173742 best: 0.0173742 (145) total: 31m 31s remaining: 3h 4m 24s
146: learn: 0.0168364 test: 0.0172143 best: 0.0172143 (146) total: 31m 44s remaining: 3h 4m 10s
147: learn: 0.0166061 test: 0.0169822 best: 0.0169822 (147) total: 31m 58s remaining: 3h 4m 2s
148: learn: 0.0163892 test: 0.0167562 best: 0.0167562 (148) total: 32m 11s remaining: 3h 3m 50s
149: learn: 0.0162522 test: 0.0166229 best: 0.0166229 (149) total: 32m 24s remaining: 3h 3m 37s
150: learn: 0.0161392 test: 0.0165126 best: 0.0165126 (150) total: 32m 35s remaining: 3h 3m 13s
151: learn: 0.0159774 test: 0.0163509 best: 0.0163509 (151) total: 32m 47s remaining: 3h 2m 59s
152: learn: 0.0156766 test: 0.0160437 best: 0.0160437 (152) total: 33m 2s remaining: 3h 2m 56s
153: learn: 0.0153750 test: 0.0157407 best: 0.0157407 (153) total: 33m 17s remaining: 3h 2m 50s
154: learn: 0.0152385 test: 0.0156024 best: 0.0156024 (154) total: 33m 29s remaining: 3h 2m 32s
155: learn: 0.0151600 test: 0.0155291 best: 0.0155291 (155) total: 33m 40s remaining: 3h 2m 13s
156: learn: 0.0149580 test: 0.0153327 best: 0.0153327 (156) total: 33m 53s remaining: 3h 2m
157: learn: 0.0148439 test: 0.0152201 best: 0.0152201 (157) total: 34m 5s remaining: 3h 1m 40s
158: learn: 0.0145697 test: 0.0149440 best: 0.0149440 (158) total: 34m 19s remaining: 3h 1m 34s
159: learn: 0.0143165 test: 0.0146844 best: 0.0146844 (159) total: 34m 33s remaining: 3h 1m 28s
160: learn: 0.0141414 test: 0.0145088 best: 0.0145088 (160) total: 34m 46s remaining: 3h 1m 15s
161: learn: 0.0139365 test: 0.0142989 best: 0.0142989 (161) total: 34m 58s remaining: 3h 55s
162: learn: 0.0138100 test: 0.0141708 best: 0.0141708 (162) total: 35m 11s remaining: 3h 43s
163: learn: 0.0136862 test: 0.0140376 best: 0.0140376 (163) total: 35m 24s remaining: 3h 28s
164: learn: 0.0135524 test: 0.0139092 best: 0.0139092 (164) total: 35m 36s remaining: 3h 14s
165: learn: 0.0133953 test: 0.0137475 best: 0.0137475 (165) total: 35m 49s remaining: 3h 1s
166: learn: 0.0132939 test: 0.0136501 best: 0.0136501 (166) total: 36m 1s remaining: 2h 59m 40s
167: learn: 0.0132229 test: 0.0135803 best: 0.0135803 (167) total: 36m 12s remaining: 2h 59m 20s
168: learn: 0.0131486 test: 0.0135072 best: 0.0135072 (168) total: 36m 24s remaining: 2h 59m 1s
169: learn: 0.0130113 test: 0.0133672 best: 0.0133672 (169) total: 36m 36s remaining: 2h 58m 44s
170: learn: 0.0129171 test: 0.0132737 best: 0.0132737 (170) total: 36m 50s remaining: 2h 58m 38s
171: learn: 0.0127523 test: 0.0131106 best: 0.0131106 (171) total: 37m 3s remaining: 2h 58m 24s
172: learn: 0.0126848 test: 0.0130446 best: 0.0130446 (172) total: 37m 14s remaining: 2h 58m 2s
173: learn: 0.0124770 test: 0.0128289 best: 0.0128289 (173) total: 37m 28s remaining: 2h 57m 52s
174: learn: 0.0123636 test: 0.0127180 best: 0.0127180 (174) total: 37m 39s remaining: 2h 57m 33s
175: learn: 0.0122650 test: 0.0126173 best: 0.0126173 (175) total: 37m 52s remaining: 2h 57m 18s
176: learn: 0.0122348 test: 0.0125863 best: 0.0125863 (176) total: 38m 2s remaining: 2h 56m 53s
177: learn: 0.0120921 test: 0.0124443 best: 0.0124443 (177) total: 38m 13s remaining: 2h 56m 32s
178: learn: 0.0120044 test: 0.0123523 best: 0.0123523 (178) total: 38m 25s remaining: 2h 56m 15s
179: learn: 0.0119371 test: 0.0122863 best: 0.0122863 (179) total: 38m 37s remaining: 2h 55m 55s
180: learn: 0.0118423 test: 0.0121908 best: 0.0121908 (180) total: 38m 47s remaining: 2h 55m 33s
181: learn: 0.0116914 test: 0.0120376 best: 0.0120376 (181) total: 38m 59s remaining: 2h 55m 15s
182: learn: 0.0115260 test: 0.0118707 best: 0.0118707 (182) total: 39m 13s remaining: 2h 55m 5s
183: learn: 0.0114236 test: 0.0117709 best: 0.0117709 (183) total: 39m 24s remaining: 2h 54m 47s
184: learn: 0.0113282 test: 0.0116757 best: 0.0116757 (184) total: 39m 36s remaining: 2h 54m 30s
185: learn: 0.0112011 test: 0.0115465 best: 0.0115465 (185) total: 39m 48s remaining: 2h 54m 14s
186: learn: 0.0111042 test: 0.0114487 best: 0.0114487 (186) total: 40m 1s remaining: 2h 53m 59s
187: learn: 0.0110618 test: 0.0114029 best: 0.0114029 (187) total: 40m 12s remaining: 2h 53m 41s
188: learn: 0.0109535 test: 0.0112973 best: 0.0112973 (188) total: 40m 25s remaining: 2h 53m 27s
189: learn: 0.0108619 test: 0.0112078 best: 0.0112078 (189) total: 40m 37s remaining: 2h 53m 9s
190: learn: 0.0107441 test: 0.0110857 best: 0.0110857 (190) total: 40m 49s remaining: 2h 52m 56s
191: learn: 0.0107182 test: 0.0110601 best: 0.0110601 (191) total: 41m remaining: 2h 52m 35s
192: learn: 0.0105936 test: 0.0109388 best: 0.0109388 (192) total: 41m 13s remaining: 2h 52m 23s
193: learn: 0.0105047 test: 0.0108487 best: 0.0108487 (193) total: 41m 25s remaining: 2h 52m 5s
194: learn: 0.0103795 test: 0.0107144 best: 0.0107144 (194) total: 41m 38s remaining: 2h 51m 55s
195: learn: 0.0103414 test: 0.0106807 best: 0.0106807 (195) total: 41m 49s remaining: 2h 51m 35s
196: learn: 0.0102458 test: 0.0105820 best: 0.0105820 (196) total: 42m 1s remaining: 2h 51m 19s
197: learn: 0.0101144 test: 0.0104492 best: 0.0104492 (197) total: 42m 13s remaining: 2h 51m 3s
198: learn: 0.0100312 test: 0.0103646 best: 0.0103646 (198) total: 42m 26s remaining: 2h 50m 48s
199: learn: 0.0099210 test: 0.0102550 best: 0.0102550 (199) total: 42m 39s remaining: 2h 50m 37s
200: learn: 0.0098176 test: 0.0101522 best: 0.0101522 (200) total: 42m 51s remaining: 2h 50m 22s
201: learn: 0.0097473 test: 0.0100890 best: 0.0100890 (201) total: 43m 5s remaining: 2h 50m 13s
202: learn: 0.0095934 test: 0.0099342 best: 0.0099342 (202) total: 43m 18s remaining: 2h 50m 3s
203: learn: 0.0095079 test: 0.0098468 best: 0.0098468 (203) total: 43m 31s remaining: 2h 49m 50s
204: learn: 0.0094738 test: 0.0098223 best: 0.0098223 (204) total: 43m 43s remaining: 2h 49m 32s
205: learn: 0.0093541 test: 0.0096990 best: 0.0096990 (205) total: 43m 56s remaining: 2h 49m 22s
206: learn: 0.0093050 test: 0.0096511 best: 0.0096511 (206) total: 44m 7s remaining: 2h 49m 3s
207: learn: 0.0091984 test: 0.0095364 best: 0.0095364 (207) total: 44m 21s remaining: 2h 48m 52s
208: learn: 0.0091497 test: 0.0094866 best: 0.0094866 (208) total: 44m 32s remaining: 2h 48m 34s
209: learn: 0.0090562 test: 0.0093953 best: 0.0093953 (209) total: 44m 45s remaining: 2h 48m 21s
210: learn: 0.0089332 test: 0.0092693 best: 0.0092693 (210) total: 44m 57s remaining: 2h 48m 8s
211: learn: 0.0088898 test: 0.0092237 best: 0.0092237 (211) total: 45m 9s remaining: 2h 47m 52s
212: learn: 0.0088541 test: 0.0091875 best: 0.0091875 (212) total: 45m 21s remaining: 2h 47m 33s
```

```
212: learn: 0.0088041 test: 0.0091079 best: 0.0091079 (212) total: 45m 21s remaining: 2h 47m 33s
213: learn: 0.0087499 test: 0.0090798 best: 0.0090798 (213) total: 45m 33s remaining: 2h 47m 20s
214: learn: 0.0087194 test: 0.0090560 best: 0.0090560 (214) total: 45m 45s remaining: 2h 47m 3s
215: learn: 0.0086480 test: 0.0089771 best: 0.0089771 (215) total: 45m 57s remaining: 2h 46m 49s
216: learn: 0.0086072 test: 0.0089348 best: 0.0089348 (216) total: 46m 9s remaining: 2h 46m 33s
217: learn: 0.0084943 test: 0.0088164 best: 0.0088164 (217) total: 46m 23s remaining: 2h 46m 23s
218: learn: 0.0084460 test: 0.0087698 best: 0.0087698 (218) total: 46m 35s remaining: 2h 46m 10s
219: learn: 0.0083554 test: 0.0086797 best: 0.0086797 (219) total: 46m 48s remaining: 2h 45m 59s
220: learn: 0.0081951 test: 0.0085214 best: 0.0085214 (220) total: 47m 2s remaining: 2h 45m 49s
221: learn: 0.0081747 test: 0.0085033 best: 0.0085033 (221) total: 47m 14s remaining: 2h 45m 34s
222: learn: 0.0081505 test: 0.0084848 best: 0.0084848 (222) total: 47m 27s remaining: 2h 45m 21s
223: learn: 0.0080987 test: 0.0084315 best: 0.0084315 (223) total: 47m 39s remaining: 2h 45m 5s
224: learn: 0.0080626 test: 0.0083951 best: 0.0083951 (224) total: 47m 50s remaining: 2h 44m 48s
225: learn: 0.0080116 test: 0.0083446 best: 0.0083446 (225) total: 48m 3s remaining: 2h 44m 35s
226: learn: 0.0079724 test: 0.0083078 best: 0.0083078 (226) total: 48m 15s remaining: 2h 44m 21s
227: learn: 0.0079379 test: 0.0082732 best: 0.0082732 (227) total: 48m 27s remaining: 2h 44m 4s
228: learn: 0.0078629 test: 0.0081992 best: 0.0081992 (228) total: 48m 40s remaining: 2h 43m 52s
229: learn: 0.0078300 test: 0.0081702 best: 0.0081702 (229) total: 48m 52s remaining: 2h 43m 36s
230: learn: 0.0077726 test: 0.0081153 best: 0.0081153 (230) total: 49m 4s remaining: 2h 43m 21s
231: learn: 0.0077405 test: 0.0080883 best: 0.0080883 (231) total: 49m 16s remaining: 2h 43m 8s
232: learn: 0.0076385 test: 0.0079851 best: 0.0079851 (232) total: 49m 29s remaining: 2h 42m 56s
233: learn: 0.0076046 test: 0.0079535 best: 0.0079535 (233) total: 49m 42s remaining: 2h 42m 44s
234: learn: 0.0075832 test: 0.0079313 best: 0.0079313 (234) total: 49m 53s remaining: 2h 42m 23s
235: learn: 0.0075472 test: 0.0078966 best: 0.0078966 (235) total: 50m 5s remaining: 2h 42m 9s
236: learn: 0.0075256 test: 0.0078775 best: 0.0078775 (236) total: 50m 17s remaining: 2h 41m 55s
237: learn: 0.0075020 test: 0.0078599 best: 0.0078599 (237) total: 50m 29s remaining: 2h 41m 40s
238: learn: 0.0074264 test: 0.0077835 best: 0.0077835 (238) total: 50m 42s remaining: 2h 41m 26s
239: learn: 0.0073438 test: 0.0077010 best: 0.0077010 (239) total: 50m 55s remaining: 2h 41m 14s
240: learn: 0.0073258 test: 0.0076858 best: 0.0076858 (240) total: 51m 5s remaining: 2h 40m 55s
241: learn: 0.0072934 test: 0.0076533 best: 0.0076533 (241) total: 51m 17s remaining: 2h 40m 38s
242: learn: 0.0072604 test: 0.0076208 best: 0.0076208 (242) total: 51m 28s remaining: 2h 40m 21s
243: learn: 0.0071679 test: 0.0075248 best: 0.0075248 (243) total: 51m 41s remaining: 2h 40m 8s
244: learn: 0.0071331 test: 0.0074862 best: 0.0074862 (244) total: 51m 52s remaining: 2h 39m 51s
245: learn: 0.0071220 test: 0.0074764 best: 0.0074764 (245) total: 52m 3s remaining: 2h 39m 32s
246: learn: 0.0070835 test: 0.0074385 best: 0.0074385 (246) total: 52m 14s remaining: 2h 39m 17s
247: learn: 0.0069928 test: 0.0073473 best: 0.0073473 (247) total: 52m 29s remaining: 2h 39m 8s
248: learn: 0.0069611 test: 0.0073143 best: 0.0073143 (248) total: 52m 40s remaining: 2h 38m 51s
249: learn: 0.0068918 test: 0.0072406 best: 0.0072406 (249) total: 52m 53s remaining: 2h 38m 39s
250: learn: 0.0068714 test: 0.0072228 best: 0.0072228 (250) total: 53m 3s remaining: 2h 38m 20s
251: learn: 0.0068396 test: 0.0071926 best: 0.0071926 (251) total: 53m 16s remaining: 2h 38m 7s
252: learn: 0.0068149 test: 0.0071684 best: 0.0071684 (252) total: 53m 26s remaining: 2h 37m 48s
253: learn: 0.0067743 test: 0.0071320 best: 0.0071320 (253) total: 53m 39s remaining: 2h 37m 34s
254: learn: 0.0066949 test: 0.0070504 best: 0.0070504 (254) total: 53m 51s remaining: 2h 37m 21s
255: learn: 0.0066334 test: 0.0069885 best: 0.0069885 (255) total: 54m 3s remaining: 2h 37m 7s
256: learn: 0.0066216 test: 0.0069788 best: 0.0069788 (256) total: 54m 14s remaining: 2h 36m 47s
257: learn: 0.0065612 test: 0.0069153 best: 0.0069153 (257) total: 54m 26s remaining: 2h 36m 33s
258: learn: 0.0065400 test: 0.0068966 best: 0.0068966 (258) total: 54m 37s remaining: 2h 36m 17s
259: learn: 0.0064938 test: 0.0068514 best: 0.0068514 (259) total: 54m 50s remaining: 2h 36m 4s
260: learn: 0.0064612 test: 0.0068196 best: 0.0068196 (260) total: 55m remaining: 2h 35m 46s
261: learn: 0.0064504 test: 0.0068114 best: 0.0068114 (261) total: 55m 10s remaining: 2h 35m 25s
262: learn: 0.0064299 test: 0.0067919 best: 0.0067919 (262) total: 55m 22s remaining: 2h 35m 9s
263: learn: 0.0063544 test: 0.0067143 best: 0.0067143 (263) total: 55m 34s remaining: 2h 34m 57s
264: learn: 0.0063230 test: 0.0066881 best: 0.0066881 (264) total: 55m 47s remaining: 2h 34m 45s
265: learn: 0.0062857 test: 0.0066510 best: 0.0066510 (265) total: 56m remaining: 2h 34m 31s
266: learn: 0.0062651 test: 0.0066301 best: 0.0066301 (266) total: 56m 11s remaining: 2h 34m 17s
267: learn: 0.0062299 test: 0.0065937 best: 0.0065937 (267) total: 56m 24s remaining: 2h 34m 3s
268: learn: 0.0062100 test: 0.0065761 best: 0.0065761 (268) total: 56m 36s remaining: 2h 33m 50s
269: learn: 0.0061710 test: 0.0065370 best: 0.0065370 (269) total: 56m 48s remaining: 2h 33m 34s
270: learn: 0.0061552 test: 0.0065240 best: 0.0065240 (270) total: 56m 59s remaining: 2h 33m 17s
271: learn: 0.0061286 test: 0.0064979 best: 0.0064979 (271) total: 57m 10s remaining: 2h 33m 1s
272: learn: 0.0060784 test: 0.0064482 best: 0.0064482 (272) total: 57m 24s remaining: 2h 32m 52s
273: learn: 0.0060522 test: 0.0064227 best: 0.0064227 (273) total: 57m 35s remaining: 2h 32m 36s
274: learn: 0.0060343 test: 0.0064079 best: 0.0064079 (274) total: 57m 47s remaining: 2h 32m 22s
275: learn: 0.0059820 test: 0.0063535 best: 0.0063535 (275) total: 57m 59s remaining: 2h 32m 6s
276: learn: 0.0059326 test: 0.0063083 best: 0.0063083 (276) total: 58m 12s remaining: 2h 31m 55s
277: learn: 0.0058708 test: 0.0062482 best: 0.0062482 (277) total: 58m 25s remaining: 2h 31m 44s
278: learn: 0.0058528 test: 0.0062334 best: 0.0062334 (278) total: 58m 38s remaining: 2h 31m 32s
279: learn: 0.0058397 test: 0.0062222 best: 0.0062222 (279) total: 58m 49s remaining: 2h 31m 16s
280: learn: 0.0058020 test: 0.0061825 best: 0.0061825 (280) total: 59m 2s remaining: 2h 31m 4s
281: learn: 0.0057931 test: 0.0061743 best: 0.0061743 (281) total: 59m 13s remaining: 2h 30m 47s
282: learn: 0.0057561 test: 0.0061351 best: 0.0061351 (282) total: 59m 24s remaining: 2h 30m 30s
283: learn: 0.0057192 test: 0.0061008 best: 0.0061008 (283) total: 59m 37s remaining: 2h 30m 20s
284: learn: 0.0056439 test: 0.0060204 best: 0.0060204 (284) total: 59m 51s remaining: 2h 30m 10s
285: learn: 0.0056306 test: 0.0060081 best: 0.0060081 (285) total: 1h 2s remaining: 2h 29m 52s
286: learn: 0.0056129 test: 0.0059940 best: 0.0059940 (286) total: 1h 15s remaining: 2h 29m 41s
287: learn: 0.0055870 test: 0.0059666 best: 0.0059666 (287) total: 1h 27s remaining: 2h 29m 26s
288: learn: 0.0055237 test: 0.0059007 best: 0.0059007 (288) total: 1h 41s remaining: 2h 29m 18s
289: learn: 0.0054775 test: 0.0058521 best: 0.0058521 (289) total: 1h 53s remaining: 2h 29m 5s
```

```
289: learn: 0.0054779 test: 0.0058521 best: 0.0058521 (289) total: 1h 59s remaining: 2h 29m 5s
290: learn: 0.0054548 test: 0.0058292 best: 0.0058292 (290) total: 1h 1m 7s remaining: 2h 28m 54s
291: learn: 0.0054325 test: 0.0058076 best: 0.0058076 (291) total: 1h 1m 18s remaining: 2h 28m 40s
292: learn: 0.0054189 test: 0.0057994 best: 0.0057994 (292) total: 1h 1m 30s remaining: 2h 28m 24s
293: learn: 0.0053753 test: 0.0057565 best: 0.0057565 (293) total: 1h 1m 41s remaining: 2h 28m 9s
294: learn: 0.0053129 test: 0.0056911 best: 0.0056911 (294) total: 1h 1m 54s remaining: 2h 27m 56s
295: learn: 0.0052790 test: 0.0056578 best: 0.0056578 (295) total: 1h 2m 6s remaining: 2h 27m 42s
296: learn: 0.0052564 test: 0.0056372 best: 0.0056372 (296) total: 1h 2m 19s remaining: 2h 27m 31s
297: learn: 0.0052427 test: 0.0056238 best: 0.0056238 (297) total: 1h 2m 30s remaining: 2h 27m 15s
298: learn: 0.0052246 test: 0.0056067 best: 0.0056067 (298) total: 1h 2m 42s remaining: 2h 27m 1s
299: learn: 0.0052056 test: 0.0055852 best: 0.0055852 (299) total: 1h 2m 54s remaining: 2h 26m 47s
300: learn: 0.0051801 test: 0.0055617 best: 0.0055617 (300) total: 1h 3m 6s remaining: 2h 26m 32s
301: learn: 0.0051493 test: 0.0055337 best: 0.0055337 (301) total: 1h 3m 17s remaining: 2h 26m 17s
302: learn: 0.0051338 test: 0.0055202 best: 0.0055202 (302) total: 1h 3m 29s remaining: 2h 26m 3s
303: learn: 0.0051004 test: 0.0054856 best: 0.0054856 (303) total: 1h 3m 42s remaining: 2h 25m 51s
304: learn: 0.0050724 test: 0.0054571 best: 0.0054571 (304) total: 1h 3m 53s remaining: 2h 25m 35s
305: learn: 0.0050499 test: 0.0054377 best: 0.0054377 (305) total: 1h 4m 6s remaining: 2h 25m 23s
306: learn: 0.0050284 test: 0.0054160 best: 0.0054160 (306) total: 1h 4m 18s remaining: 2h 25m 9s
307: learn: 0.0050125 test: 0.0054028 best: 0.0054028 (307) total: 1h 4m 30s remaining: 2h 24m 56s
308: learn: 0.0049956 test: 0.0053887 best: 0.0053887 (308) total: 1h 4m 44s remaining: 2h 24m 47s
309: learn: 0.0049727 test: 0.0053656 best: 0.0053656 (309) total: 1h 4m 56s remaining: 2h 24m 33s
310: learn: 0.0049506 test: 0.0053466 best: 0.0053466 (310) total: 1h 5m 8s remaining: 2h 24m 19s
311: learn: 0.0049424 test: 0.0053415 best: 0.0053415 (311) total: 1h 5m 19s remaining: 2h 24m 2s
312: learn: 0.0049153 test: 0.0053112 best: 0.0053112 (312) total: 1h 5m 30s remaining: 2h 23m 47s
313: learn: 0.0048890 test: 0.0052885 best: 0.0052885 (313) total: 1h 5m 43s remaining: 2h 23m 34s
314: learn: 0.0048636 test: 0.0052636 best: 0.0052636 (314) total: 1h 5m 55s remaining: 2h 23m 22s
315: learn: 0.0048493 test: 0.0052495 best: 0.0052495 (315) total: 1h 6m 6s remaining: 2h 23m 6s
316: learn: 0.0048112 test: 0.0052127 best: 0.0052127 (316) total: 1h 6m 20s remaining: 2h 22m 55s
317: learn: 0.0047997 test: 0.0052055 best: 0.0052055 (317) total: 1h 6m 32s remaining: 2h 22m 43s
318: learn: 0.0047692 test: 0.0051730 best: 0.0051730 (318) total: 1h 6m 44s remaining: 2h 22m 28s
319: learn: 0.0047537 test: 0.0051570 best: 0.0051570 (319) total: 1h 6m 57s remaining: 2h 22m 16s
320: learn: 0.0047450 test: 0.0051480 best: 0.0051480 (320) total: 1h 7m 9s remaining: 2h 22m 2s
321: learn: 0.0047135 test: 0.0051154 best: 0.0051154 (321) total: 1h 7m 20s remaining: 2h 21m 47s
322: learn: 0.0047029 test: 0.0051080 best: 0.0051080 (322) total: 1h 7m 31s remaining: 2h 21m 32s
323: learn: 0.0046903 test: 0.0050977 best: 0.0050977 (323) total: 1h 7m 42s remaining: 2h 21m 16s
324: learn: 0.0046740 test: 0.0050846 best: 0.0050846 (324) total: 1h 7m 54s remaining: 2h 21m 3s
325: learn: 0.0046252 test: 0.0050334 best: 0.0050334 (325) total: 1h 8m 8s remaining: 2h 20m 53s
326: learn: 0.0045934 test: 0.0049997 best: 0.0049997 (326) total: 1h 8m 22s remaining: 2h 20m 42s
327: learn: 0.0045858 test: 0.0049939 best: 0.0049939 (327) total: 1h 8m 33s remaining: 2h 20m 27s
328: learn: 0.0045673 test: 0.0049764 best: 0.0049764 (328) total: 1h 8m 45s remaining: 2h 20m 14s
329: learn: 0.0045424 test: 0.0049512 best: 0.0049512 (329) total: 1h 8m 57s remaining: 2h 20m
330: learn: 0.0045299 test: 0.0049408 best: 0.0049408 (330) total: 1h 9m 9s remaining: 2h 19m 46s
331: learn: 0.0045025 test: 0.0049142 best: 0.0049142 (331) total: 1h 9m 22s remaining: 2h 19m 34s
332: learn: 0.0044746 test: 0.0048886 best: 0.0048886 (332) total: 1h 9m 35s remaining: 2h 19m 23s
333: learn: 0.0044559 test: 0.0048720 best: 0.0048720 (333) total: 1h 9m 49s remaining: 2h 19m 13s
334: learn: 0.0044437 test: 0.0048594 best: 0.0048594 (334) total: 1h 10m remaining: 2h 18m 58s
335: learn: 0.0043854 test: 0.0047989 best: 0.0047989 (335) total: 1h 10m 14s remaining: 2h 18m 48
s
336: learn: 0.0043507 test: 0.0047647 best: 0.0047647 (336) total: 1h 10m 27s remaining: 2h 18m 37
s
337: learn: 0.0043271 test: 0.0047443 best: 0.0047443 (337) total: 1h 10m 41s remaining: 2h 18m 26
s
338: learn: 0.0042743 test: 0.0046872 best: 0.0046872 (338) total: 1h 10m 55s remaining: 2h 18m 18
s
339: learn: 0.0042550 test: 0.0046714 best: 0.0046714 (339) total: 1h 11m 9s remaining: 2h 18m 7s
340: learn: 0.0042360 test: 0.0046532 best: 0.0046532 (340) total: 1h 11m 21s remaining: 2h 17m 55
s
341: learn: 0.0042309 test: 0.0046483 best: 0.0046483 (341) total: 1h 11m 32s remaining: 2h 17m 38
s
342: learn: 0.0042128 test: 0.0046321 best: 0.0046321 (342) total: 1h 11m 45s remaining: 2h 17m 27
s
343: learn: 0.0042032 test: 0.0046242 best: 0.0046242 (343) total: 1h 11m 56s remaining: 2h 17m 10
s
344: learn: 0.0041903 test: 0.0046120 best: 0.0046120 (344) total: 1h 12m 6s remaining: 2h 16m 54s
345: learn: 0.0041662 test: 0.0045880 best: 0.0045880 (345) total: 1h 12m 18s remaining: 2h 16m 41
s
346: learn: 0.0041503 test: 0.0045725 best: 0.0045725 (346) total: 1h 12m 32s remaining: 2h 16m 30
s
347: learn: 0.0041428 test: 0.0045697 best: 0.0045697 (347) total: 1h 12m 43s remaining: 2h 16m 15
s
348: learn: 0.0041310 test: 0.0045580 best: 0.0045580 (348) total: 1h 12m 55s remaining: 2h 16m 2s
349: learn: 0.0041227 test: 0.0045522 best: 0.0045522 (349) total: 1h 13m 7s remaining: 2h 15m 48s
350: learn: 0.0040880 test: 0.0045156 best: 0.0045156 (350) total: 1h 13m 21s remaining: 2h 15m 39
s
351: learn: 0.0040520 test: 0.0044805 best: 0.0044805 (351) total: 1h 13m 35s remaining: 2h 15m 28
s
352: learn: 0.0040472 test: 0.0044781 best: 0.0044781 (352) total: 1h 13m 46s remaining: 2h 15m 12
s
```

s
353: learn: 0.0040345 test: 0.0044659 best: 0.0044659 (353) total: 1h 13m 57s remaining: 2h 14m 58
s
354: learn: 0.0040009 test: 0.0044313 best: 0.0044313 (354) total: 1h 14m 10s remaining: 2h 14m 45
s
355: learn: 0.0039788 test: 0.0044085 best: 0.0044085 (355) total: 1h 14m 22s remaining: 2h 14m 32
s
356: learn: 0.0039685 test: 0.0044007 best: 0.0044007 (356) total: 1h 14m 35s remaining: 2h 14m 20
s
357: learn: 0.0039631 test: 0.0043966 best: 0.0043966 (357) total: 1h 14m 46s remaining: 2h 14m 5s
358: learn: 0.0039529 test: 0.0043889 best: 0.0043889 (358) total: 1h 14m 59s remaining: 2h 13m 53
s
359: learn: 0.0038986 test: 0.0043340 best: 0.0043340 (359) total: 1h 15m 13s remaining: 2h 13m 43
s
360: learn: 0.0038873 test: 0.0043252 best: 0.0043252 (360) total: 1h 15m 25s remaining: 2h 13m 30
s
361: learn: 0.0038742 test: 0.0043120 best: 0.0043120 (361) total: 1h 15m 36s remaining: 2h 13m 16
s
362: learn: 0.0038649 test: 0.0043037 best: 0.0043037 (362) total: 1h 15m 48s remaining: 2h 13m 1s
363: learn: 0.0038380 test: 0.0042772 best: 0.0042772 (363) total: 1h 16m remaining: 2h 12m 47s
364: learn: 0.0038168 test: 0.0042570 best: 0.0042570 (364) total: 1h 16m 13s remaining: 2h 12m 37
s
365: learn: 0.0037720 test: 0.0042096 best: 0.0042096 (365) total: 1h 16m 27s remaining: 2h 12m 26
s
366: learn: 0.0037603 test: 0.0041994 best: 0.0041994 (366) total: 1h 16m 38s remaining: 2h 12m 12
s
367: learn: 0.0037418 test: 0.0041800 best: 0.0041800 (367) total: 1h 16m 49s remaining: 2h 11m 55
s
368: learn: 0.0037294 test: 0.0041671 best: 0.0041671 (368) total: 1h 17m 1s remaining: 2h 11m 43s
369: learn: 0.0037196 test: 0.0041577 best: 0.0041577 (369) total: 1h 17m 12s remaining: 2h 11m 28
s
370: learn: 0.0037058 test: 0.0041434 best: 0.0041434 (370) total: 1h 17m 25s remaining: 2h 11m 15
s
371: learn: 0.0036861 test: 0.0041209 best: 0.0041209 (371) total: 1h 17m 37s remaining: 2h 11m 2s
372: learn: 0.0036622 test: 0.0040979 best: 0.0040979 (372) total: 1h 17m 50s remaining: 2h 10m 51
s
373: learn: 0.0036457 test: 0.0040810 best: 0.0040810 (373) total: 1h 18m 2s remaining: 2h 10m 38s
374: learn: 0.0036331 test: 0.0040691 best: 0.0040691 (374) total: 1h 18m 14s remaining: 2h 10m 24
s
375: learn: 0.0036302 test: 0.0040668 best: 0.0040668 (375) total: 1h 18m 24s remaining: 2h 10m 7s
376: learn: 0.0036225 test: 0.0040584 best: 0.0040584 (376) total: 1h 18m 36s remaining: 2h 9m 53s
377: learn: 0.0035954 test: 0.0040312 best: 0.0040312 (377) total: 1h 18m 47s remaining: 2h 9m 39s
378: learn: 0.0035751 test: 0.0040118 best: 0.0040118 (378) total: 1h 19m remaining: 2h 9m 27s
379: learn: 0.0035444 test: 0.0039790 best: 0.0039790 (379) total: 1h 19m 14s remaining: 2h 9m 18s
380: learn: 0.0035338 test: 0.0039688 best: 0.0039688 (380) total: 1h 19m 25s remaining: 2h 9m 1s
381: learn: 0.0035256 test: 0.0039631 best: 0.0039631 (381) total: 1h 19m 35s remaining: 2h 8m 46s
382: learn: 0.0035106 test: 0.0039495 best: 0.0039495 (382) total: 1h 19m 47s remaining: 2h 8m 32s
383: learn: 0.0035045 test: 0.0039436 best: 0.0039436 (383) total: 1h 19m 57s remaining: 2h 8m 15s
384: learn: 0.0034954 test: 0.0039354 best: 0.0039354 (384) total: 1h 20m 8s remaining: 2h 8m 1s
385: learn: 0.0034845 test: 0.0039257 best: 0.0039257 (385) total: 1h 20m 21s remaining: 2h 7m 48s
386: learn: 0.0034554 test: 0.0038952 best: 0.0038952 (386) total: 1h 20m 34s remaining: 2h 7m 38s
387: learn: 0.0034338 test: 0.0038739 best: 0.0038739 (387) total: 1h 20m 46s remaining: 2h 7m 23s
388: learn: 0.0034250 test: 0.0038672 best: 0.0038672 (388) total: 1h 20m 59s remaining: 2h 7m 12s
389: learn: 0.0034018 test: 0.0038450 best: 0.0038450 (389) total: 1h 21m 13s remaining: 2h 7m 2s
390: learn: 0.0033840 test: 0.0038272 best: 0.0038272 (390) total: 1h 21m 26s remaining: 2h 6m 51s
391: learn: 0.0033691 test: 0.0038120 best: 0.0038120 (391) total: 1h 21m 39s remaining: 2h 6m 39s
392: learn: 0.0033332 test: 0.0037732 best: 0.0037732 (392) total: 1h 21m 53s remaining: 2h 6m 29s
393: learn: 0.0033191 test: 0.0037629 best: 0.0037629 (393) total: 1h 22m 7s remaining: 2h 6m 18s
394: learn: 0.0033062 test: 0.0037533 best: 0.0037533 (394) total: 1h 22m 20s remaining: 2h 6m 7s
395: learn: 0.0032977 test: 0.0037496 best: 0.0037496 (395) total: 1h 22m 32s remaining: 2h 5m 53s
396: learn: 0.0032894 test: 0.0037421 best: 0.0037421 (396) total: 1h 22m 43s remaining: 2h 5m 38s
397: learn: 0.0032825 test: 0.0037378 best: 0.0037378 (397) total: 1h 22m 54s remaining: 2h 5m 24s
398: learn: 0.0032738 test: 0.0037309 best: 0.0037309 (398) total: 1h 23m 7s remaining: 2h 5m 12s
399: learn: 0.0032696 test: 0.0037272 best: 0.0037272 (399) total: 1h 23m 17s remaining: 2h 4m 56s
400: learn: 0.0032538 test: 0.0037117 best: 0.0037117 (400) total: 1h 23m 30s remaining: 2h 4m 44s
401: learn: 0.0032398 test: 0.0036977 best: 0.0036977 (401) total: 1h 23m 43s remaining: 2h 4m 33s
402: learn: 0.0032339 test: 0.0036920 best: 0.0036920 (402) total: 1h 23m 54s remaining: 2h 4m 18s
403: learn: 0.0032277 test: 0.0036881 best: 0.0036881 (403) total: 1h 24m 7s remaining: 2h 4m 5s
404: learn: 0.0032071 test: 0.0036669 best: 0.0036669 (404) total: 1h 24m 19s remaining: 2h 3m 53s
405: learn: 0.0031947 test: 0.0036539 best: 0.0036539 (405) total: 1h 24m 32s remaining: 2h 3m 40s
406: learn: 0.0031877 test: 0.0036487 best: 0.0036487 (406) total: 1h 24m 45s remaining: 2h 3m 28s
407: learn: 0.0031715 test: 0.0036301 best: 0.0036301 (407) total: 1h 24m 58s remaining: 2h 3m 17s
408: learn: 0.0031647 test: 0.0036253 best: 0.0036253 (408) total: 1h 25m 11s remaining: 2h 3m 5s
409: learn: 0.0031562 test: 0.0036184 best: 0.0036184 (409) total: 1h 25m 22s remaining: 2h 2m 51s
410: learn: 0.0031461 test: 0.0036080 best: 0.0036080 (410) total: 1h 25m 35s remaining: 2h 2m 40s
411: learn: 0.0031254 test: 0.0035839 best: 0.0035839 (411) total: 1h 25m 48s remaining: 2h 2m 27s
412: learn: 0.0031171 test: 0.0035759 best: 0.0035759 (412) total: 1h 25m 59s remaining: 2h 2m 13s
413: learn: 0.0030996 test: 0.0035582 best: 0.0035582 (413) total: 1h 26m 11s remaining: 2h 2m

413: learn: 0.0030996 test: 0.0035582 best: 0.0035582 (413) total: 1h 26m 11s remaining: 2h 2m
414: learn: 0.0030783 test: 0.0035377 best: 0.0035377 (414) total: 1h 26m 22s remaining: 2h 1m 45s
415: learn: 0.0030733 test: 0.0035352 best: 0.0035352 (415) total: 1h 26m 34s remaining: 2h 1m 32s
416: learn: 0.0030555 test: 0.0035169 best: 0.0035169 (416) total: 1h 26m 46s remaining: 2h 1m 19s
417: learn: 0.0030426 test: 0.0035043 best: 0.0035043 (417) total: 1h 26m 58s remaining: 2h 1m 6s
418: learn: 0.0030374 test: 0.0035010 best: 0.0035010 (418) total: 1h 27m 10s remaining: 2h 52s
419: learn: 0.0030225 test: 0.0034867 best: 0.0034867 (419) total: 1h 27m 23s remaining: 2h 40s
420: learn: 0.0030043 test: 0.0034673 best: 0.0034673 (420) total: 1h 27m 36s remaining: 2h 29s
421: learn: 0.0030001 test: 0.0034654 best: 0.0034654 (421) total: 1h 27m 48s remaining: 2h 15s
422: learn: 0.0029885 test: 0.0034557 best: 0.0034557 (422) total: 1h 28m 1s remaining: 2h 4s
423: learn: 0.0029711 test: 0.0034416 best: 0.0034416 (423) total: 1h 28m 13s remaining: 1h 59m 50
s
424: learn: 0.0029691 test: 0.0034406 best: 0.0034406 (424) total: 1h 28m 22s remaining: 1h 59m 34
s
425: learn: 0.0029518 test: 0.0034257 best: 0.0034257 (425) total: 1h 28m 36s remaining: 1h 59m 23
s
426: learn: 0.0029484 test: 0.0034243 best: 0.0034243 (426) total: 1h 28m 47s remaining: 1h 59m 9s
427: learn: 0.0029422 test: 0.0034206 best: 0.0034206 (427) total: 1h 28m 59s remaining: 1h 58m 55
s
428: learn: 0.0029212 test: 0.0034002 best: 0.0034002 (428) total: 1h 29m 12s remaining: 1h 58m 43
s
429: learn: 0.0029128 test: 0.0033924 best: 0.0033924 (429) total: 1h 29m 24s remaining: 1h 58m 30
s
430: learn: 0.0029002 test: 0.0033832 best: 0.0033832 (430) total: 1h 29m 38s remaining: 1h 58m 20
s
431: learn: 0.0028899 test: 0.0033743 best: 0.0033743 (431) total: 1h 29m 51s remaining: 1h 58m 8s
432: learn: 0.0028757 test: 0.0033582 best: 0.0033582 (432) total: 1h 30m 2s remaining: 1h 57m 54s
433: learn: 0.0028676 test: 0.0033494 best: 0.0033494 (433) total: 1h 30m 14s remaining: 1h 57m 41
s
434: learn: 0.0028589 test: 0.0033389 best: 0.0033389 (434) total: 1h 30m 27s remaining: 1h 57m 29
s
435: learn: 0.0028286 test: 0.0033059 best: 0.0033059 (435) total: 1h 30m 41s remaining: 1h 57m 19
s
436: learn: 0.0028218 test: 0.0032996 best: 0.0032996 (436) total: 1h 30m 53s remaining: 1h 57m 6s
437: learn: 0.0028122 test: 0.0032934 best: 0.0032934 (437) total: 1h 31m 6s remaining: 1h 56m 54s
438: learn: 0.0027908 test: 0.0032711 best: 0.0032711 (438) total: 1h 31m 20s remaining: 1h 56m 43
s
439: learn: 0.0027755 test: 0.0032571 best: 0.0032571 (439) total: 1h 31m 34s remaining: 1h 56m 32
s
440: learn: 0.0027698 test: 0.0032529 best: 0.0032529 (440) total: 1h 31m 45s remaining: 1h 56m 18
s
441: learn: 0.0027658 test: 0.0032495 best: 0.0032495 (441) total: 1h 31m 56s remaining: 1h 56m 4s
442: learn: 0.0027560 test: 0.0032393 best: 0.0032393 (442) total: 1h 32m 8s remaining: 1h 55m 51s
443: learn: 0.0027461 test: 0.0032305 best: 0.0032305 (443) total: 1h 32m 21s remaining: 1h 55m 39
s
444: learn: 0.0027342 test: 0.0032186 best: 0.0032186 (444) total: 1h 32m 33s remaining: 1h 55m 26
s
445: learn: 0.0027297 test: 0.0032164 best: 0.0032164 (445) total: 1h 32m 46s remaining: 1h 55m 14
s
446: learn: 0.0027228 test: 0.0032098 best: 0.0032098 (446) total: 1h 32m 58s remaining: 1h 55m 1s
447: learn: 0.0027174 test: 0.0032051 best: 0.0032051 (447) total: 1h 33m 10s remaining: 1h 54m 48
s
448: learn: 0.0026890 test: 0.0031775 best: 0.0031775 (448) total: 1h 33m 24s remaining: 1h 54m 37
s
449: learn: 0.0026848 test: 0.0031719 best: 0.0031719 (449) total: 1h 33m 34s remaining: 1h 54m 22
s
450: learn: 0.0026762 test: 0.0031652 best: 0.0031652 (450) total: 1h 33m 46s remaining: 1h 54m 9s
451: learn: 0.0026657 test: 0.0031549 best: 0.0031549 (451) total: 1h 33m 59s remaining: 1h 53m 57
s
452: learn: 0.0026598 test: 0.0031500 best: 0.0031500 (452) total: 1h 34m 11s remaining: 1h 53m 44
s
453: learn: 0.0026547 test: 0.0031460 best: 0.0031460 (453) total: 1h 34m 23s remaining: 1h 53m 31
s
454: learn: 0.0026378 test: 0.0031289 best: 0.0031289 (454) total: 1h 34m 36s remaining: 1h 53m 19
s
455: learn: 0.0026325 test: 0.0031232 best: 0.0031232 (455) total: 1h 34m 48s remaining: 1h 53m 6s
456: learn: 0.0026278 test: 0.0031207 best: 0.0031207 (456) total: 1h 35m remaining: 1h 52m 53s
457: learn: 0.0026182 test: 0.0031114 best: 0.0031114 (457) total: 1h 35m 13s remaining: 1h 52m 41
s
458: learn: 0.0026110 test: 0.0031061 best: 0.0031061 (458) total: 1h 35m 24s remaining: 1h 52m 27
s
459: learn: 0.0026039 test: 0.0030983 best: 0.0030983 (459) total: 1h 35m 36s remaining: 1h 52m 14
s
460: learn: 0.0026011 test: 0.0030970 best: 0.0030970 (460) total: 1h 35m 48s remaining: 1h 52m 1s
461: learn: 0.0025871 test: 0.0030812 best: 0.0030812 (461) total: 1h 36m remaining: 1h 51m 48s
462: learn: 0.0025661 test: 0.0030584 best: 0.0030584 (462) total: 1h 36m 13s remaining: 1h 51m 36
s
463: learn: 0.0025550 test: 0.0030488 best: 0.0030488 (463) total: 1h 36m 26s remaining: 1h 51m 24

463: learn: 0.0025559 test: 0.0030488 best: 0.0030488 (463) total: 1h 36m 26s remaining: 1h 51m 24
s
464: learn: 0.0025492 test: 0.0030434 best: 0.0030434 (464) total: 1h 36m 38s remaining: 1h 51m 11
s
465: learn: 0.0025441 test: 0.0030386 best: 0.0030386 (465) total: 1h 36m 49s remaining: 1h 50m 57
s
466: learn: 0.0025358 test: 0.0030321 best: 0.0030321 (466) total: 1h 37m 1s remaining: 1h 50m 44s
467: learn: 0.0025306 test: 0.0030270 best: 0.0030270 (467) total: 1h 37m 13s remaining: 1h 50m 31
s
468: learn: 0.0025181 test: 0.0030138 best: 0.0030138 (468) total: 1h 37m 26s remaining: 1h 50m 19
s
469: learn: 0.0025009 test: 0.0029962 best: 0.0029962 (469) total: 1h 37m 39s remaining: 1h 50m 6s
470: learn: 0.0024874 test: 0.0029840 best: 0.0029840 (470) total: 1h 37m 52s remaining: 1h 49m 55
s
471: learn: 0.0024817 test: 0.0029793 best: 0.0029793 (471) total: 1h 38m 3s remaining: 1h 49m 41s
472: learn: 0.0024758 test: 0.0029753 best: 0.0029753 (472) total: 1h 38m 16s remaining: 1h 49m 30
s
473: learn: 0.0024667 test: 0.0029673 best: 0.0029673 (473) total: 1h 38m 29s remaining: 1h 49m 18
s
474: learn: 0.0024574 test: 0.0029589 best: 0.0029589 (474) total: 1h 38m 41s remaining: 1h 49m 4s
475: learn: 0.0024510 test: 0.0029542 best: 0.0029542 (475) total: 1h 38m 53s remaining: 1h 48m 52
s
476: learn: 0.0024406 test: 0.0029423 best: 0.0029423 (476) total: 1h 39m 5s remaining: 1h 48m 39s
477: learn: 0.0024373 test: 0.0029403 best: 0.0029403 (477) total: 1h 39m 16s remaining: 1h 48m 25
s
478: learn: 0.0024300 test: 0.0029350 best: 0.0029350 (478) total: 1h 39m 29s remaining: 1h 48m 12
s
479: learn: 0.0024234 test: 0.0029310 best: 0.0029310 (479) total: 1h 39m 42s remaining: 1h 48m 1s
480: learn: 0.0024177 test: 0.0029261 best: 0.0029261 (480) total: 1h 39m 55s remaining: 1h 47m 48
s
481: learn: 0.0024057 test: 0.0029129 best: 0.0029129 (481) total: 1h 40m 8s remaining: 1h 47m 37s
482: learn: 0.0023971 test: 0.0029051 best: 0.0029051 (482) total: 1h 40m 21s remaining: 1h 47m 25
s
483: learn: 0.0023820 test: 0.0028891 best: 0.0028891 (483) total: 1h 40m 33s remaining: 1h 47m 12
s
484: learn: 0.0023788 test: 0.0028883 best: 0.0028883 (484) total: 1h 40m 44s remaining: 1h 46m 57
s
485: learn: 0.0023698 test: 0.0028806 best: 0.0028806 (485) total: 1h 40m 56s remaining: 1h 46m 45
s
486: learn: 0.0023630 test: 0.0028737 best: 0.0028737 (486) total: 1h 41m 7s remaining: 1h 46m 31s
487: learn: 0.0023600 test: 0.0028727 best: 0.0028727 (487) total: 1h 41m 19s remaining: 1h 46m 18
s
488: learn: 0.0023517 test: 0.0028673 best: 0.0028673 (488) total: 1h 41m 32s remaining: 1h 46m 7s
489: learn: 0.0023471 test: 0.0028658 best: 0.0028658 (489) total: 1h 41m 45s remaining: 1h 45m 54
s
490: learn: 0.0023410 test: 0.0028611 best: 0.0028611 (490) total: 1h 41m 57s remaining: 1h 45m 41
s
491: learn: 0.0023285 test: 0.0028504 best: 0.0028504 (491) total: 1h 42m 11s remaining: 1h 45m 31
s
492: learn: 0.0023248 test: 0.0028484 best: 0.0028484 (492) total: 1h 42m 24s remaining: 1h 45m 19
s
493: learn: 0.0023180 test: 0.0028422 best: 0.0028422 (493) total: 1h 42m 38s remaining: 1h 45m 7s
494: learn: 0.0023116 test: 0.0028380 best: 0.0028380 (494) total: 1h 42m 50s remaining: 1h 44m 55
s
495: learn: 0.0023075 test: 0.0028355 best: 0.0028355 (495) total: 1h 43m 2s remaining: 1h 44m 42s
496: learn: 0.0022981 test: 0.0028257 best: 0.0028257 (496) total: 1h 43m 14s remaining: 1h 44m 28
s
497: learn: 0.0022910 test: 0.0028198 best: 0.0028198 (497) total: 1h 43m 27s remaining: 1h 44m 17
s
498: learn: 0.0022673 test: 0.0027931 best: 0.0027931 (498) total: 1h 43m 41s remaining: 1h 44m 6s
499: learn: 0.0022502 test: 0.0027751 best: 0.0027751 (499) total: 1h 43m 54s remaining: 1h 43m 54
s
500: learn: 0.0022473 test: 0.0027737 best: 0.0027737 (500) total: 1h 44m 5s remaining: 1h 43m 40s
501: learn: 0.0022407 test: 0.0027693 best: 0.0027693 (501) total: 1h 44m 17s remaining: 1h 43m 27
s
502: learn: 0.0022326 test: 0.0027620 best: 0.0027620 (502) total: 1h 44m 28s remaining: 1h 43m 13
s
503: learn: 0.0022269 test: 0.0027563 best: 0.0027563 (503) total: 1h 44m 39s remaining: 1h 43m
504: learn: 0.0022166 test: 0.0027494 best: 0.0027494 (504) total: 1h 44m 53s remaining: 1h 42m 49
s
505: learn: 0.0022111 test: 0.0027448 best: 0.0027448 (505) total: 1h 45m 6s remaining: 1h 42m 36s
506: learn: 0.0021983 test: 0.0027341 best: 0.0027341 (506) total: 1h 45m 20s remaining: 1h 42m 25
s
507: learn: 0.0021912 test: 0.0027295 best: 0.0027295 (507) total: 1h 45m 33s remaining: 1h 42m 13
s
508: learn: 0.0021872 test: 0.0027263 best: 0.0027263 (508) total: 1h 45m 44s remaining: 1h 42m
509: learn: 0.0021840 test: 0.0027235 best: 0.0027235 (509) total: 1h 45m 56s remaining: 1h 41m 47

s
510: learn: 0.0021771 test: 0.0027177 best: 0.0027177 (510) total: 1h 46m 7s remaining: 1h 41m 33s
511: learn: 0.0021672 test: 0.0027093 best: 0.0027093 (511) total: 1h 46m 20s remaining: 1h 41m 21
s
512: learn: 0.0021485 test: 0.0026900 best: 0.0026900 (512) total: 1h 46m 34s remaining: 1h 41m 10
s
513: learn: 0.0021452 test: 0.0026880 best: 0.0026880 (513) total: 1h 46m 46s remaining: 1h 40m 57
s
514: learn: 0.0021301 test: 0.0026720 best: 0.0026720 (514) total: 1h 47m remaining: 1h 40m 46s
515: learn: 0.0021278 test: 0.0026715 best: 0.0026715 (515) total: 1h 47m 12s remaining: 1h 40m 33
s
516: learn: 0.0021255 test: 0.0026705 best: 0.0026705 (516) total: 1h 47m 24s remaining: 1h 40m 20
s
517: learn: 0.0021229 test: 0.0026674 best: 0.0026674 (517) total: 1h 47m 35s remaining: 1h 40m 7s
518: learn: 0.0021190 test: 0.0026646 best: 0.0026646 (518) total: 1h 47m 48s remaining: 1h 39m 54
s
519: learn: 0.0021159 test: 0.0026622 best: 0.0026622 (519) total: 1h 47m 59s remaining: 1h 39m 41
s
520: learn: 0.0021092 test: 0.0026558 best: 0.0026558 (520) total: 1h 48m 12s remaining: 1h 39m 28
s
521: learn: 0.0021003 test: 0.0026470 best: 0.0026470 (521) total: 1h 48m 26s remaining: 1h 39m 17
s
522: learn: 0.0020853 test: 0.0026325 best: 0.0026325 (522) total: 1h 48m 39s remaining: 1h 39m 6s
523: learn: 0.0020830 test: 0.0026311 best: 0.0026311 (523) total: 1h 48m 50s remaining: 1h 38m 52
s
524: learn: 0.0020809 test: 0.0026297 best: 0.0026297 (524) total: 1h 49m 1s remaining: 1h 38m 38s
525: learn: 0.0020767 test: 0.0026258 best: 0.0026258 (525) total: 1h 49m 13s remaining: 1h 38m 26
s
526: learn: 0.0020660 test: 0.0026146 best: 0.0026146 (526) total: 1h 49m 27s remaining: 1h 38m 14
s
527: learn: 0.0020588 test: 0.0026093 best: 0.0026093 (527) total: 1h 49m 39s remaining: 1h 38m 1s
528: learn: 0.0020519 test: 0.0026047 best: 0.0026047 (528) total: 1h 49m 52s remaining: 1h 37m 49
s
529: learn: 0.0020458 test: 0.0025987 best: 0.0025987 (529) total: 1h 50m 5s remaining: 1h 37m 38s
530: learn: 0.0020400 test: 0.0025932 best: 0.0025932 (530) total: 1h 50m 19s remaining: 1h 37m 26
s
531: learn: 0.0020324 test: 0.0025858 best: 0.0025858 (531) total: 1h 50m 31s remaining: 1h 37m 13
s
532: learn: 0.0020266 test: 0.0025815 best: 0.0025815 (532) total: 1h 50m 43s remaining: 1h 37m 1s
533: learn: 0.0020235 test: 0.0025803 best: 0.0025803 (533) total: 1h 50m 56s remaining: 1h 36m 48
s
534: learn: 0.0020154 test: 0.0025703 best: 0.0025703 (534) total: 1h 51m 9s remaining: 1h 36m 37s
535: learn: 0.0020104 test: 0.0025663 best: 0.0025663 (535) total: 1h 51m 22s remaining: 1h 36m 25
s
536: learn: 0.0020073 test: 0.0025642 best: 0.0025642 (536) total: 1h 51m 33s remaining: 1h 36m 11
s
537: learn: 0.0019997 test: 0.0025583 best: 0.0025583 (537) total: 1h 51m 47s remaining: 1h 35m 59
s
538: learn: 0.0019961 test: 0.0025550 best: 0.0025550 (538) total: 1h 51m 58s remaining: 1h 35m 46
s
539: learn: 0.0019931 test: 0.0025533 best: 0.0025533 (539) total: 1h 52m 11s remaining: 1h 35m 34
s
540: learn: 0.0019798 test: 0.0025410 best: 0.0025410 (540) total: 1h 52m 25s remaining: 1h 35m 23
s
541: learn: 0.0019717 test: 0.0025334 best: 0.0025334 (541) total: 1h 52m 38s remaining: 1h 35m 10
s
542: learn: 0.0019693 test: 0.0025324 best: 0.0025324 (542) total: 1h 52m 50s remaining: 1h 34m 57
s
543: learn: 0.0019668 test: 0.0025303 best: 0.0025303 (543) total: 1h 53m 1s remaining: 1h 34m 44s
544: learn: 0.0019626 test: 0.0025285 best: 0.0025285 (544) total: 1h 53m 13s remaining: 1h 34m 31
s
545: learn: 0.0019572 test: 0.0025234 best: 0.0025234 (545) total: 1h 53m 25s remaining: 1h 34m 18
s
546: learn: 0.0019528 test: 0.0025181 best: 0.0025181 (546) total: 1h 53m 36s remaining: 1h 34m 5s
547: learn: 0.0019491 test: 0.0025139 best: 0.0025139 (547) total: 1h 53m 48s remaining: 1h 33m 51
s
548: learn: 0.0019461 test: 0.0025128 best: 0.0025128 (548) total: 1h 54m remaining: 1h 33m 39s
549: learn: 0.0019410 test: 0.0025081 best: 0.0025081 (549) total: 1h 54m 12s remaining: 1h 33m 26
s
550: learn: 0.0019348 test: 0.0025034 best: 0.0025034 (550) total: 1h 54m 24s remaining: 1h 33m 14
s
551: learn: 0.0019281 test: 0.0024990 best: 0.0024990 (551) total: 1h 54m 39s remaining: 1h 33m 3s
552: learn: 0.0019248 test: 0.0024966 best: 0.0024966 (552) total: 1h 54m 51s remaining: 1h 32m 50
s
553: learn: 0.0019071 test: 0.0024766 best: 0.0024766 (553) total: 1h 55m 6s remaining: 1h 32m 40s
554: learn: 0.0019003 test: 0.0024713 best: 0.0024713 (554) total: 1h 55m 18s remaining: 1h 32m 27
s

555: learn: 0.0018955 test: 0.0024665 best: 0.0024665 (555) total: 1h 55m 31s remaining: 1h 32m 15
s
556: learn: 0.0018908 test: 0.0024627 best: 0.0024627 (556) total: 1h 55m 43s remaining: 1h 32m 2s
557: learn: 0.0018861 test: 0.0024581 best: 0.0024581 (557) total: 1h 55m 55s remaining: 1h 31m 49
s
558: learn: 0.0018827 test: 0.0024555 best: 0.0024555 (558) total: 1h 56m 7s remaining: 1h 31m 36s
559: learn: 0.0018771 test: 0.0024490 best: 0.0024490 (559) total: 1h 56m 17s remaining: 1h 31m 22
s
560: learn: 0.0018662 test: 0.0024389 best: 0.0024389 (560) total: 1h 56m 30s remaining: 1h 31m 10
s
561: learn: 0.0018622 test: 0.0024346 best: 0.0024346 (561) total: 1h 56m 41s remaining: 1h 30m 56
s
562: learn: 0.0018594 test: 0.0024321 best: 0.0024321 (562) total: 1h 56m 53s remaining: 1h 30m 43
s
563: learn: 0.0018559 test: 0.0024319 best: 0.0024319 (563) total: 1h 57m 6s remaining: 1h 30m 31s
564: learn: 0.0018524 test: 0.0024284 best: 0.0024284 (564) total: 1h 57m 18s remaining: 1h 30m 19
s
565: learn: 0.0018482 test: 0.0024247 best: 0.0024247 (565) total: 1h 57m 31s remaining: 1h 30m 6s
566: learn: 0.0018392 test: 0.0024164 best: 0.0024164 (566) total: 1h 57m 45s remaining: 1h 29m 55
s
567: learn: 0.0018352 test: 0.0024137 best: 0.0024137 (567) total: 1h 57m 58s remaining: 1h 29m 43
s
568: learn: 0.0018322 test: 0.0024109 best: 0.0024109 (568) total: 1h 58m 9s remaining: 1h 29m 29s
569: learn: 0.0018268 test: 0.0024062 best: 0.0024062 (569) total: 1h 58m 21s remaining: 1h 29m 17
s
570: learn: 0.0018249 test: 0.0024055 best: 0.0024055 (570) total: 1h 58m 32s remaining: 1h 29m 3s
571: learn: 0.0018206 test: 0.0024025 best: 0.0024025 (571) total: 1h 58m 44s remaining: 1h 28m 50
s
572: learn: 0.0018155 test: 0.0023983 best: 0.0023983 (572) total: 1h 58m 57s remaining: 1h 28m 38
s
573: learn: 0.0018125 test: 0.0023949 best: 0.0023949 (573) total: 1h 59m 9s remaining: 1h 28m 25s
574: learn: 0.0018094 test: 0.0023928 best: 0.0023928 (574) total: 1h 59m 20s remaining: 1h 28m 12
s
575: learn: 0.0018035 test: 0.0023881 best: 0.0023881 (575) total: 1h 59m 33s remaining: 1h 28m
576: learn: 0.0017987 test: 0.0023838 best: 0.0023838 (576) total: 1h 59m 45s remaining: 1h 27m 47
s
577: learn: 0.0017968 test: 0.0023821 best: 0.0023821 (577) total: 1h 59m 56s remaining: 1h 27m 33
s
578: learn: 0.0017900 test: 0.0023759 best: 0.0023759 (578) total: 2h 8s remaining: 1h 27m 21s
579: learn: 0.0017871 test: 0.0023737 best: 0.0023737 (579) total: 2h 21s remaining: 1h 27m 9s
580: learn: 0.0017792 test: 0.0023658 best: 0.0023658 (580) total: 2h 34s remaining: 1h 26m 57s
581: learn: 0.0017611 test: 0.0023432 best: 0.0023432 (581) total: 2h 48s remaining: 1h 26m 45s
582: learn: 0.0017593 test: 0.0023421 best: 0.0023421 (582) total: 2h 1m remaining: 1h 26m 32s
583: learn: 0.0017571 test: 0.0023410 best: 0.0023410 (583) total: 2h 1m 10s remaining: 1h 26m 19s
584: learn: 0.0017507 test: 0.0023357 best: 0.0023357 (584) total: 2h 1m 24s remaining: 1h 26m 7s
585: learn: 0.0017442 test: 0.0023302 best: 0.0023302 (585) total: 2h 1m 36s remaining: 1h 25m 55s
586: learn: 0.0017402 test: 0.0023290 best: 0.0023290 (586) total: 2h 1m 50s remaining: 1h 25m 43s
587: learn: 0.0017347 test: 0.0023265 best: 0.0023265 (587) total: 2h 2m 5s remaining: 1h 25m 33s
588: learn: 0.0017321 test: 0.0023260 best: 0.0023260 (588) total: 2h 2m 18s remaining: 1h 25m 20s
589: learn: 0.0017282 test: 0.0023223 best: 0.0023223 (589) total: 2h 2m 31s remaining: 1h 25m 8s
590: learn: 0.0017123 test: 0.0023058 best: 0.0023058 (590) total: 2h 2m 44s remaining: 1h 24m 56s
591: learn: 0.0017083 test: 0.0023021 best: 0.0023021 (591) total: 2h 2m 57s remaining: 1h 24m 44s
592: learn: 0.0017012 test: 0.0022955 best: 0.0022955 (592) total: 2h 3m 9s remaining: 1h 24m 31s
593: learn: 0.0016908 test: 0.0022840 best: 0.0022840 (593) total: 2h 3m 23s remaining: 1h 24m 20s
594: learn: 0.0016838 test: 0.0022762 best: 0.0022762 (594) total: 2h 3m 36s remaining: 1h 24m 7s
595: learn: 0.0016788 test: 0.0022712 best: 0.0022712 (595) total: 2h 3m 48s remaining: 1h 23m 55s
596: learn: 0.0016720 test: 0.0022636 best: 0.0022636 (596) total: 2h 4m remaining: 1h 23m 42s
597: learn: 0.0016680 test: 0.0022609 best: 0.0022609 (597) total: 2h 4m 12s remaining: 1h 23m 30s
598: learn: 0.0016625 test: 0.0022546 best: 0.0022546 (598) total: 2h 4m 24s remaining: 1h 23m 17s
599: learn: 0.0016566 test: 0.0022480 best: 0.0022480 (599) total: 2h 4m 37s remaining: 1h 23m 5s
600: learn: 0.0016536 test: 0.0022447 best: 0.0022447 (600) total: 2h 4m 49s remaining: 1h 22m 51s
601: learn: 0.0016521 test: 0.0022433 best: 0.0022433 (601) total: 2h 4m 59s remaining: 1h 22m 38s
602: learn: 0.0016481 test: 0.0022404 best: 0.0022404 (602) total: 2h 5m 11s remaining: 1h 22m 25s
603: learn: 0.0016421 test: 0.0022354 best: 0.0022354 (603) total: 2h 5m 25s remaining: 1h 22m 13s
604: learn: 0.0016391 test: 0.0022334 best: 0.0022334 (604) total: 2h 5m 38s remaining: 1h 22m 1s
605: learn: 0.0016373 test: 0.0022326 best: 0.0022326 (605) total: 2h 5m 50s remaining: 1h 21m 49s
606: learn: 0.0016343 test: 0.0022295 best: 0.0022295 (606) total: 2h 6m 3s remaining: 1h 21m 36s
607: learn: 0.0016323 test: 0.0022270 best: 0.0022270 (607) total: 2h 6m 14s remaining: 1h 21m 23s
608: learn: 0.0016283 test: 0.0022223 best: 0.0022223 (608) total: 2h 6m 26s remaining: 1h 21m 10s
609: learn: 0.0016228 test: 0.0022167 best: 0.0022167 (609) total: 2h 6m 39s remaining: 1h 20m 58s
610: learn: 0.0016187 test: 0.0022116 best: 0.0022116 (610) total: 2h 6m 50s remaining: 1h 20m 45s
611: learn: 0.0016144 test: 0.0022087 best: 0.0022087 (611) total: 2h 7m 2s remaining: 1h 20m 32s
612: learn: 0.0016113 test: 0.0022077 best: 0.0022077 (612) total: 2h 7m 14s remaining: 1h 20m 20s
613: learn: 0.0016085 test: 0.0022071 best: 0.0022071 (613) total: 2h 7m 27s remaining: 1h 20m 7s
614: learn: 0.0016014 test: 0.0022012 best: 0.0022012 (614) total: 2h 7m 41s remaining: 1h 19m 56s
615: learn: 0.0015999 test: 0.0022007 best: 0.0022007 (615) total: 2h 7m 53s remaining: 1h 19m 43s
616: learn: 0.0015971 test: 0.0021989 best: 0.0021989 (616) total: 2h 8m 4s remaining: 1h 19m 30s

```
617: learn: 0.0015941 test: 0.0021965 best: 0.0021965 (617) total: 2h 8m 16s remaining: 1h 19m 17s
618: learn: 0.0015810 test: 0.0021824 best: 0.0021824 (618) total: 2h 8m 29s remaining: 1h 19m 5s
619: learn: 0.0015772 test: 0.0021794 best: 0.0021794 (619) total: 2h 8m 42s remaining: 1h 18m 52s
620: learn: 0.0015691 test: 0.0021705 best: 0.0021705 (620) total: 2h 8m 53s remaining: 1h 18m 40s
621: learn: 0.0015651 test: 0.0021670 best: 0.0021670 (621) total: 2h 9m 6s remaining: 1h 18m 27s
622: learn: 0.0015613 test: 0.0021660 best: 0.0021660 (622) total: 2h 9m 20s remaining: 1h 18m 15s
623: learn: 0.0015596 test: 0.0021644 best: 0.0021644 (623) total: 2h 9m 31s remaining: 1h 18m 2s
624: learn: 0.0015579 test: 0.0021635 best: 0.0021635 (624) total: 2h 9m 43s remaining: 1h 17m 50s
625: learn: 0.0015543 test: 0.0021600 best: 0.0021600 (625) total: 2h 9m 54s remaining: 1h 17m 36s
626: learn: 0.0015478 test: 0.0021519 best: 0.0021519 (626) total: 2h 10m 7s remaining: 1h 17m 24s
627: learn: 0.0015416 test: 0.0021473 best: 0.0021473 (627) total: 2h 10m 20s remaining: 1h 17m 12
s
628: learn: 0.0015375 test: 0.0021430 best: 0.0021430 (628) total: 2h 10m 32s remaining: 1h 17m
629: learn: 0.0015348 test: 0.0021409 best: 0.0021409 (629) total: 2h 10m 44s remaining: 1h 16m 46
s
630: learn: 0.0015328 test: 0.0021385 best: 0.0021385 (630) total: 2h 10m 55s remaining: 1h 16m 33
s
631: learn: 0.0015309 test: 0.0021367 best: 0.0021367 (631) total: 2h 11m 7s remaining: 1h 16m 21s
632: learn: 0.0015258 test: 0.0021313 best: 0.0021313 (632) total: 2h 11m 21s remaining: 1h 16m 9s
633: learn: 0.0015247 test: 0.0021307 best: 0.0021307 (633) total: 2h 11m 31s remaining: 1h 15m 55
s
634: learn: 0.0015190 test: 0.0021242 best: 0.0021242 (634) total: 2h 11m 44s remaining: 1h 15m 43
s
635: learn: 0.0015150 test: 0.0021217 best: 0.0021217 (635) total: 2h 11m 57s remaining: 1h 15m 31
s
636: learn: 0.0015102 test: 0.0021162 best: 0.0021162 (636) total: 2h 12m 10s remaining: 1h 15m 19
s
637: learn: 0.0015029 test: 0.0021091 best: 0.0021091 (637) total: 2h 12m 24s remaining: 1h 15m 7s
638: learn: 0.0015009 test: 0.0021076 best: 0.0021076 (638) total: 2h 12m 36s remaining: 1h 14m 54
s
639: learn: 0.0014911 test: 0.0020977 best: 0.0020977 (639) total: 2h 12m 49s remaining: 1h 14m 43
s
640: learn: 0.0014859 test: 0.0020918 best: 0.0020918 (640) total: 2h 13m remaining: 1h 14m 29s
641: learn: 0.0014828 test: 0.0020883 best: 0.0020883 (641) total: 2h 13m 12s remaining: 1h 14m 16
s
642: learn: 0.0014810 test: 0.0020869 best: 0.0020869 (642) total: 2h 13m 23s remaining: 1h 14m 3s
643: learn: 0.0014782 test: 0.0020856 best: 0.0020856 (643) total: 2h 13m 36s remaining: 1h 13m 51
s
644: learn: 0.0014761 test: 0.0020850 best: 0.0020850 (644) total: 2h 13m 48s remaining: 1h 13m 39
s
645: learn: 0.0014724 test: 0.0020806 best: 0.0020806 (645) total: 2h 14m 1s remaining: 1h 13m 26s
646: learn: 0.0014693 test: 0.0020768 best: 0.0020768 (646) total: 2h 14m 13s remaining: 1h 13m 14
s
647: learn: 0.0014665 test: 0.0020740 best: 0.0020740 (647) total: 2h 14m 26s remaining: 1h 13m 1s
648: learn: 0.0014652 test: 0.0020731 best: 0.0020731 (648) total: 2h 14m 36s remaining: 1h 12m 48
s
649: learn: 0.0014628 test: 0.0020712 best: 0.0020712 (649) total: 2h 14m 49s remaining: 1h 12m 36
s
650: learn: 0.0014611 test: 0.0020702 best: 0.0020702 (650) total: 2h 15m 1s remaining: 1h 12m 23s
651: learn: 0.0014596 test: 0.0020695 best: 0.0020695 (651) total: 2h 15m 12s remaining: 1h 12m 10
s
652: learn: 0.0014580 test: 0.0020677 best: 0.0020677 (652) total: 2h 15m 24s remaining: 1h 11m 57
s
653: learn: 0.0014554 test: 0.0020658 best: 0.0020658 (653) total: 2h 15m 37s remaining: 1h 11m 45
s
654: learn: 0.0014527 test: 0.0020634 best: 0.0020634 (654) total: 2h 15m 49s remaining: 1h 11m 32
s
655: learn: 0.0014500 test: 0.0020618 best: 0.0020618 (655) total: 2h 16m 3s remaining: 1h 11m 20s
656: learn: 0.0014449 test: 0.0020586 best: 0.0020586 (656) total: 2h 16m 17s remaining: 1h 11m 9s
657: learn: 0.0014406 test: 0.0020547 best: 0.0020547 (657) total: 2h 16m 30s remaining: 1h 10m 57
s
658: learn: 0.0014393 test: 0.0020538 best: 0.0020538 (658) total: 2h 16m 41s remaining: 1h 10m 44
s
659: learn: 0.0014326 test: 0.0020475 best: 0.0020475 (659) total: 2h 16m 56s remaining: 1h 10m 32
s
660: learn: 0.0014277 test: 0.0020418 best: 0.0020418 (660) total: 2h 17m 8s remaining: 1h 10m 19s
661: learn: 0.0014248 test: 0.0020409 best: 0.0020409 (661) total: 2h 17m 21s remaining: 1h 10m 7s
662: learn: 0.0014197 test: 0.0020358 best: 0.0020358 (662) total: 2h 17m 34s remaining: 1h 9m 55s
663: learn: 0.0014139 test: 0.0020303 best: 0.0020303 (663) total: 2h 17m 48s remaining: 1h 9m 43s
664: learn: 0.0014129 test: 0.0020297 best: 0.0020297 (664) total: 2h 18m remaining: 1h 9m 31s
665: learn: 0.0014103 test: 0.0020284 best: 0.0020284 (665) total: 2h 18m 13s remaining: 1h 9m 19s
666: learn: 0.0014072 test: 0.0020263 best: 0.0020263 (666) total: 2h 18m 26s remaining: 1h 9m 6s
667: learn: 0.0014030 test: 0.0020231 best: 0.0020231 (667) total: 2h 18m 40s remaining: 1h 8m 55s
668: learn: 0.0013992 test: 0.0020209 best: 0.0020209 (668) total: 2h 18m 53s remaining: 1h 8m 43s
669: learn: 0.0013972 test: 0.0020192 best: 0.0020192 (669) total: 2h 19m 5s remaining: 1h 8m 30s
670: learn: 0.0013940 test: 0.0020163 best: 0.0020163 (670) total: 2h 19m 19s remaining: 1h 8m 18s
671: learn: 0.0013912 test: 0.0020136 best: 0.0020136 (671) total: 2h 19m 32s remaining: 1h 8m 6s
```

```
672: learn: 0.0013890 test: 0.0020109 best: 0.0020109 (672) total: 2h 19m 44s remaining: 1h 7m 54s
673: learn: 0.0013842 test: 0.0020066 best: 0.0020066 (673) total: 2h 19m 58s remaining: 1h 7m 42s
674: learn: 0.0013746 test: 0.0019967 best: 0.0019967 (674) total: 2h 20m 11s remaining: 1h 7m 29s
675: learn: 0.0013725 test: 0.0019944 best: 0.0019944 (675) total: 2h 20m 23s remaining: 1h 7m 17s
676: learn: 0.0013703 test: 0.0019934 best: 0.0019934 (676) total: 2h 20m 36s remaining: 1h 7m 4s
677: learn: 0.0013645 test: 0.0019880 best: 0.0019880 (677) total: 2h 20m 48s remaining: 1h 6m 52s
678: learn: 0.0013617 test: 0.0019856 best: 0.0019856 (678) total: 2h 20m 59s remaining: 1h 6m 39s
679: learn: 0.0013573 test: 0.0019827 best: 0.0019827 (679) total: 2h 21m 14s remaining: 1h 6m 27s
680: learn: 0.0013550 test: 0.0019814 best: 0.0019814 (680) total: 2h 21m 27s remaining: 1h 6m 15s
681: learn: 0.0013513 test: 0.0019769 best: 0.0019769 (681) total: 2h 21m 40s remaining: 1h 6m 3s
682: learn: 0.0013503 test: 0.0019767 best: 0.0019767 (682) total: 2h 21m 51s remaining: 1h 5m 50s
683: learn: 0.0013473 test: 0.0019749 best: 0.0019749 (683) total: 2h 22m 3s remaining: 1h 5m 37s
684: learn: 0.0013360 test: 0.0019618 best: 0.0019618 (684) total: 2h 22m 17s remaining: 1h 5m 26s
685: learn: 0.0013318 test: 0.0019575 best: 0.0019575 (685) total: 2h 22m 31s remaining: 1h 5m 14s
686: learn: 0.0013263 test: 0.0019515 best: 0.0019515 (686) total: 2h 22m 43s remaining: 1h 5m 1s
687: learn: 0.0013224 test: 0.0019473 best: 0.0019473 (687) total: 2h 22m 55s remaining: 1h 4m 49s
688: learn: 0.0013187 test: 0.0019422 best: 0.0019422 (688) total: 2h 23m 8s remaining: 1h 4m 36s
689: learn: 0.0013160 test: 0.0019392 best: 0.0019392 (689) total: 2h 23m 20s remaining: 1h 4m 23s
690: learn: 0.0013143 test: 0.0019382 best: 0.0019382 (690) total: 2h 23m 32s remaining: 1h 4m 11s
691: learn: 0.0013113 test: 0.0019364 best: 0.0019364 (691) total: 2h 23m 45s remaining: 1h 3m 59s
692: learn: 0.0013091 test: 0.0019349 best: 0.0019349 (692) total: 2h 23m 58s remaining: 1h 3m 47s
693: learn: 0.0013067 test: 0.0019322 best: 0.0019322 (693) total: 2h 24m 10s remaining: 1h 3m 34s
694: learn: 0.0012977 test: 0.0019232 best: 0.0019232 (694) total: 2h 24m 24s remaining: 1h 3m 22s
695: learn: 0.0012922 test: 0.0019165 best: 0.0019165 (695) total: 2h 24m 36s remaining: 1h 3m 9s
696: learn: 0.0012895 test: 0.0019145 best: 0.0019145 (696) total: 2h 24m 49s remaining: 1h 2m 57s
697: learn: 0.0012838 test: 0.0019082 best: 0.0019082 (697) total: 2h 25m 1s remaining: 1h 2m 44s
698: learn: 0.0012826 test: 0.0019076 best: 0.0019076 (698) total: 2h 25m 13s remaining: 1h 2m 32s
699: learn: 0.0012795 test: 0.0019045 best: 0.0019045 (699) total: 2h 25m 26s remaining: 1h 2m 20s
700: learn: 0.0012730 test: 0.0018981 best: 0.0018981 (700) total: 2h 25m 39s remaining: 1h 2m 7s
701: learn: 0.0012633 test: 0.0018870 best: 0.0018870 (701) total: 2h 25m 52s remaining: 1h 1m 55s
702: learn: 0.0012621 test: 0.0018859 best: 0.0018859 (702) total: 2h 26m 2s remaining: 1h 1m 41s
703: learn: 0.0012603 test: 0.0018841 best: 0.0018841 (703) total: 2h 26m 14s remaining: 1h 1m 29s
704: learn: 0.0012587 test: 0.0018828 best: 0.0018828 (704) total: 2h 26m 26s remaining: 1h 1m 16s
705: learn: 0.0012578 test: 0.0018817 best: 0.0018817 (705) total: 2h 26m 36s remaining: 1h 1m 3s
706: learn: 0.0012553 test: 0.0018793 best: 0.0018793 (706) total: 2h 26m 48s remaining: 1h 50s
707: learn: 0.0012520 test: 0.0018774 best: 0.0018774 (707) total: 2h 27m 1s remaining: 1h 38s
708: learn: 0.0012510 test: 0.0018764 best: 0.0018764 (708) total: 2h 27m 12s remaining: 1h 25s
709: learn: 0.0012484 test: 0.0018735 best: 0.0018735 (709) total: 2h 27m 23s remaining: 1h 12s
710: learn: 0.0012469 test: 0.0018716 best: 0.0018716 (710) total: 2h 27m 34s remaining: 59m 59s
711: learn: 0.0012454 test: 0.0018700 best: 0.0018700 (711) total: 2h 27m 45s remaining: 59m 45s
712: learn: 0.0012352 test: 0.0018594 best: 0.0018594 (712) total: 2h 27m 57s remaining: 59m 33s
713: learn: 0.0012332 test: 0.0018581 best: 0.0018581 (713) total: 2h 28m 9s remaining: 59m 20s
714: learn: 0.0012318 test: 0.0018565 best: 0.0018565 (714) total: 2h 28m 21s remaining: 59m 8s
715: learn: 0.0012266 test: 0.0018509 best: 0.0018509 (715) total: 2h 28m 35s remaining: 58m 56s
716: learn: 0.0012228 test: 0.0018471 best: 0.0018471 (716) total: 2h 28m 47s remaining: 58m 43s
717: learn: 0.0012204 test: 0.0018454 best: 0.0018454 (717) total: 2h 29m 1s remaining: 58m 31s
718: learn: 0.0012141 test: 0.0018389 best: 0.0018389 (718) total: 2h 29m 14s remaining: 58m 19s
719: learn: 0.0012116 test: 0.0018374 best: 0.0018374 (719) total: 2h 29m 27s remaining: 58m 7s
720: learn: 0.0012077 test: 0.0018340 best: 0.0018340 (720) total: 2h 29m 40s remaining: 57m 55s
721: learn: 0.0012038 test: 0.0018298 best: 0.0018298 (721) total: 2h 29m 53s remaining: 57m 42s
722: learn: 0.0011988 test: 0.0018252 best: 0.0018252 (722) total: 2h 30m 5s remaining: 57m 30s
723: learn: 0.0011969 test: 0.0018232 best: 0.0018232 (723) total: 2h 30m 17s remaining: 57m 17s
724: learn: 0.0011940 test: 0.0018205 best: 0.0018205 (724) total: 2h 30m 30s remaining: 57m 5s
725: learn: 0.0011914 test: 0.0018174 best: 0.0018174 (725) total: 2h 30m 41s remaining: 56m 52s
726: learn: 0.0011899 test: 0.0018166 best: 0.0018166 (726) total: 2h 30m 54s remaining: 56m 40s
727: learn: 0.0011875 test: 0.0018148 best: 0.0018148 (727) total: 2h 31m 7s remaining: 56m 27s
728: learn: 0.0011820 test: 0.0018083 best: 0.0018083 (728) total: 2h 31m 21s remaining: 56m 16s
729: learn: 0.0011790 test: 0.0018052 best: 0.0018052 (729) total: 2h 31m 33s remaining: 56m 3s
730: learn: 0.0011760 test: 0.0018015 best: 0.0018015 (730) total: 2h 31m 47s remaining: 55m 51s
731: learn: 0.0011722 test: 0.0017974 best: 0.0017974 (731) total: 2h 32m remaining: 55m 39s
732: learn: 0.0011707 test: 0.0017964 best: 0.0017964 (732) total: 2h 32m 12s remaining: 55m 26s
733: learn: 0.0011692 test: 0.0017953 best: 0.0017953 (733) total: 2h 32m 25s remaining: 55m 14s
734: learn: 0.0011651 test: 0.0017911 best: 0.0017911 (734) total: 2h 32m 38s remaining: 55m 1s
735: learn: 0.0011611 test: 0.0017878 best: 0.0017878 (735) total: 2h 32m 51s remaining: 54m 49s
736: learn: 0.0011573 test: 0.0017828 best: 0.0017828 (736) total: 2h 33m 3s remaining: 54m 37s
737: learn: 0.0011543 test: 0.0017800 best: 0.0017800 (737) total: 2h 33m 16s remaining: 54m 24s
738: learn: 0.0011510 test: 0.0017771 best: 0.0017771 (738) total: 2h 33m 29s remaining: 54m 12s
739: learn: 0.0011451 test: 0.0017707 best: 0.0017707 (739) total: 2h 33m 41s remaining: 53m 59s
740: learn: 0.0011424 test: 0.0017685 best: 0.0017685 (740) total: 2h 33m 54s remaining: 53m 47s
741: learn: 0.0011389 test: 0.0017653 best: 0.0017653 (741) total: 2h 34m 7s remaining: 53m 35s
742: learn: 0.0011350 test: 0.0017626 best: 0.0017626 (742) total: 2h 34m 21s remaining: 53m 23s
743: learn: 0.0011285 test: 0.0017574 best: 0.0017574 (743) total: 2h 34m 36s remaining: 53m 11s
744: learn: 0.0011262 test: 0.0017555 best: 0.0017555 (744) total: 2h 34m 48s remaining: 52m 59s
745: learn: 0.0011241 test: 0.0017548 best: 0.0017548 (745) total: 2h 35m 1s remaining: 52m 47s
746: learn: 0.0011190 test: 0.0017503 best: 0.0017503 (746) total: 2h 35m 14s remaining: 52m 34s
747: learn: 0.0011162 test: 0.0017479 best: 0.0017479 (747) total: 2h 35m 27s remaining: 52m 22s
748: learn: 0.0011105 test: 0.0017432 best: 0.0017432 (748) total: 2h 35m 41s remaining: 52m 10s
```

```
749: learn: 0.0011088 test: 0.0017413 best: 0.0017413 (749) total: 2h 35m 53s remaining: 51m 57s
750: learn: 0.0011050 test: 0.0017382 best: 0.0017382 (750) total: 2h 36m 6s remaining: 51m 45s
751: learn: 0.0011028 test: 0.0017356 best: 0.0017356 (751) total: 2h 36m 19s remaining: 51m 33s
752: learn: 0.0010998 test: 0.0017320 best: 0.0017320 (752) total: 2h 36m 32s remaining: 51m 21s
753: learn: 0.0010981 test: 0.0017311 best: 0.0017311 (753) total: 2h 36m 45s remaining: 51m 8s
754: learn: 0.0010960 test: 0.0017290 best: 0.0017290 (754) total: 2h 36m 57s remaining: 50m 55s
755: learn: 0.0010903 test: 0.0017238 best: 0.0017238 (755) total: 2h 37m 10s remaining: 50m 43s
756: learn: 0.0010886 test: 0.0017221 best: 0.0017221 (756) total: 2h 37m 22s remaining: 50m 31s
757: learn: 0.0010857 test: 0.0017198 best: 0.0017198 (757) total: 2h 37m 35s remaining: 50m 18s
758: learn: 0.0010841 test: 0.0017189 best: 0.0017189 (758) total: 2h 37m 46s remaining: 50m 5s
759: learn: 0.0010826 test: 0.0017176 best: 0.0017176 (759) total: 2h 37m 58s remaining: 49m 53s
760: learn: 0.0010816 test: 0.0017165 best: 0.0017165 (760) total: 2h 38m 8s remaining: 49m 40s
761: learn: 0.0010807 test: 0.0017160 best: 0.0017160 (761) total: 2h 38m 20s remaining: 49m 27s
762: learn: 0.0010798 test: 0.0017150 best: 0.0017150 (762) total: 2h 38m 31s remaining: 49m 14s
763: learn: 0.0010781 test: 0.0017139 best: 0.0017139 (763) total: 2h 38m 44s remaining: 49m 2s
764: learn: 0.0010753 test: 0.0017112 best: 0.0017112 (764) total: 2h 38m 57s remaining: 48m 49s
765: learn: 0.0010735 test: 0.0017092 best: 0.0017092 (765) total: 2h 39m 9s remaining: 48m 37s
766: learn: 0.0010723 test: 0.0017084 best: 0.0017084 (766) total: 2h 39m 22s remaining: 48m 24s
767: learn: 0.0010715 test: 0.0017077 best: 0.0017077 (767) total: 2h 39m 34s remaining: 48m 12s
768: learn: 0.0010693 test: 0.0017063 best: 0.0017063 (768) total: 2h 39m 46s remaining: 47m 59s
769: learn: 0.0010685 test: 0.0017051 best: 0.0017051 (769) total: 2h 39m 57s remaining: 47m 46s
770: learn: 0.0010674 test: 0.0017046 best: 0.0017046 (770) total: 2h 40m 8s remaining: 47m 33s
771: learn: 0.0010629 test: 0.0016993 best: 0.0016993 (771) total: 2h 40m 21s remaining: 47m 21s
772: learn: 0.0010599 test: 0.0016971 best: 0.0016971 (772) total: 2h 40m 35s remaining: 47m 9s
773: learn: 0.0010578 test: 0.0016964 best: 0.0016964 (773) total: 2h 40m 48s remaining: 46m 57s
774: learn: 0.0010565 test: 0.0016957 best: 0.0016957 (774) total: 2h 41m remaining: 46m 44s
775: learn: 0.0010544 test: 0.0016939 best: 0.0016939 (775) total: 2h 41m 14s remaining: 46m 32s
776: learn: 0.0010517 test: 0.0016908 best: 0.0016908 (776) total: 2h 41m 27s remaining: 46m 20s
777: learn: 0.0010483 test: 0.0016873 best: 0.0016873 (777) total: 2h 41m 40s remaining: 46m 8s
778: learn: 0.0010473 test: 0.0016872 best: 0.0016872 (778) total: 2h 41m 51s remaining: 45m 55s
779: learn: 0.0010450 test: 0.0016849 best: 0.0016849 (779) total: 2h 42m 3s remaining: 45m 42s
780: learn: 0.0010424 test: 0.0016824 best: 0.0016824 (780) total: 2h 42m 17s remaining: 45m 30s
781: learn: 0.0010410 test: 0.0016820 best: 0.0016820 (781) total: 2h 42m 29s remaining: 45m 17s
782: learn: 0.0010390 test: 0.0016814 best: 0.0016814 (782) total: 2h 42m 41s remaining: 45m 5s
783: learn: 0.0010377 test: 0.0016801 best: 0.0016801 (783) total: 2h 42m 54s remaining: 44m 52s
784: learn: 0.0010353 test: 0.0016788 best: 0.0016788 (784) total: 2h 43m 6s remaining: 44m 40s
785: learn: 0.0010312 test: 0.0016741 best: 0.0016741 (785) total: 2h 43m 18s remaining: 44m 27s
786: learn: 0.0010299 test: 0.0016734 best: 0.0016734 (786) total: 2h 43m 29s remaining: 44m 14s
787: learn: 0.0010230 test: 0.0016649 best: 0.0016649 (787) total: 2h 43m 42s remaining: 44m 2s
788: learn: 0.0010207 test: 0.0016625 best: 0.0016625 (788) total: 2h 43m 54s remaining: 43m 49s
789: learn: 0.0010164 test: 0.0016588 best: 0.0016588 (789) total: 2h 44m 7s remaining: 43m 37s
790: learn: 0.0010141 test: 0.0016573 best: 0.0016573 (790) total: 2h 44m 19s remaining: 43m 25s
791: learn: 0.0010118 test: 0.0016553 best: 0.0016553 (791) total: 2h 44m 31s remaining: 43m 12s
792: learn: 0.0010111 test: 0.0016543 best: 0.0016543 (792) total: 2h 44m 42s remaining: 42m 59s
793: learn: 0.0010097 test: 0.0016526 best: 0.0016526 (793) total: 2h 44m 53s remaining: 42m 46s
794: learn: 0.0010033 test: 0.0016457 best: 0.0016457 (794) total: 2h 45m 7s remaining: 42m 34s
795: learn: 0.0010002 test: 0.0016440 best: 0.0016440 (795) total: 2h 45m 20s remaining: 42m 22s
796: learn: 0.0009981 test: 0.0016428 best: 0.0016428 (796) total: 2h 45m 32s remaining: 42m 9s
797: learn: 0.0009968 test: 0.0016412 best: 0.0016412 (797) total: 2h 45m 43s remaining: 41m 56s
798: learn: 0.0009955 test: 0.0016398 best: 0.0016398 (798) total: 2h 45m 54s remaining: 41m 44s
799: learn: 0.0009944 test: 0.0016381 best: 0.0016381 (799) total: 2h 46m 6s remaining: 41m 31s
800: learn: 0.0009900 test: 0.0016349 best: 0.0016349 (800) total: 2h 46m 20s remaining: 41m 19s
801: learn: 0.0009873 test: 0.0016333 best: 0.0016333 (801) total: 2h 46m 34s remaining: 41m 7s
802: learn: 0.0009862 test: 0.0016318 best: 0.0016318 (802) total: 2h 46m 45s remaining: 40m 54s
803: learn: 0.0009838 test: 0.0016300 best: 0.0016300 (803) total: 2h 46m 59s remaining: 40m 42s
804: learn: 0.0009813 test: 0.0016272 best: 0.0016272 (804) total: 2h 47m 13s remaining: 40m 30s
805: learn: 0.0009785 test: 0.0016240 best: 0.0016240 (805) total: 2h 47m 25s remaining: 40m 17s
806: learn: 0.0009760 test: 0.0016213 best: 0.0016213 (806) total: 2h 47m 37s remaining: 40m 5s
807: learn: 0.0009752 test: 0.0016204 best: 0.0016204 (807) total: 2h 47m 48s remaining: 39m 52s
808: learn: 0.0009739 test: 0.0016185 best: 0.0016185 (808) total: 2h 48m 1s remaining: 39m 40s
809: learn: 0.0009727 test: 0.0016178 best: 0.0016178 (809) total: 2h 48m 13s remaining: 39m 27s
810: learn: 0.0009709 test: 0.0016167 best: 0.0016167 (810) total: 2h 48m 27s remaining: 39m 15s
811: learn: 0.0009657 test: 0.0016112 best: 0.0016112 (811) total: 2h 48m 40s remaining: 39m 3s
812: learn: 0.0009629 test: 0.0016103 best: 0.0016103 (812) total: 2h 48m 54s remaining: 38m 51s
813: learn: 0.0009593 test: 0.0016073 best: 0.0016073 (813) total: 2h 49m 7s remaining: 38m 38s
814: learn: 0.0009584 test: 0.0016067 best: 0.0016067 (814) total: 2h 49m 19s remaining: 38m 26s
815: learn: 0.0009579 test: 0.0016062 best: 0.0016062 (815) total: 2h 49m 29s remaining: 38m 13s
816: learn: 0.0009556 test: 0.0016034 best: 0.0016034 (816) total: 2h 49m 41s remaining: 38m
817: learn: 0.0009528 test: 0.0016016 best: 0.0016016 (817) total: 2h 49m 55s remaining: 37m 48s
818: learn: 0.0009507 test: 0.0015999 best: 0.0015999 (818) total: 2h 50m 6s remaining: 37m 35s
819: learn: 0.0009491 test: 0.0015991 best: 0.0015991 (819) total: 2h 50m 19s remaining: 37m 23s
820: learn: 0.0009467 test: 0.0015979 best: 0.0015979 (820) total: 2h 50m 33s remaining: 37m 11s
821: learn: 0.0009445 test: 0.0015964 best: 0.0015964 (821) total: 2h 50m 45s remaining: 36m 58s
822: learn: 0.0009434 test: 0.0015956 best: 0.0015956 (822) total: 2h 50m 57s remaining: 36m 46s
823: learn: 0.0009403 test: 0.0015926 best: 0.0015926 (823) total: 2h 51m 11s remaining: 36m 33s
824: learn: 0.0009392 test: 0.0015918 best: 0.0015918 (824) total: 2h 51m 23s remaining: 36m 21s
825: learn: 0.0009371 test: 0.0015888 best: 0.0015888 (825) total: 2h 51m 36s remaining: 36m 8s
```

```
826: learn: 0.0009326 test: 0.0015844 best: 0.0015844 (826) total: 2h 51m 50s remaining: 35m 56s
827: learn: 0.0009316 test: 0.0015834 best: 0.0015834 (827) total: 2h 52m 2s remaining: 35m 44s
828: learn: 0.0009303 test: 0.0015831 best: 0.0015831 (828) total: 2h 52m 14s remaining: 35m 31s
829: learn: 0.0009288 test: 0.0015831 best: 0.0015831 (829) total: 2h 52m 27s remaining: 35m 19s
830: learn: 0.0009241 test: 0.0015789 best: 0.0015789 (830) total: 2h 52m 41s remaining: 35m 7s
831: learn: 0.0009233 test: 0.0015785 best: 0.0015785 (831) total: 2h 52m 52s remaining: 34m 54s
832: learn: 0.0009217 test: 0.0015767 best: 0.0015767 (832) total: 2h 53m 4s remaining: 34m 41s
833: learn: 0.0009203 test: 0.0015756 best: 0.0015756 (833) total: 2h 53m 16s remaining: 34m 29s
834: learn: 0.0009196 test: 0.0015750 best: 0.0015750 (834) total: 2h 53m 27s remaining: 34m 16s
835: learn: 0.0009182 test: 0.0015739 best: 0.0015739 (835) total: 2h 53m 40s remaining: 34m 4s
836: learn: 0.0009163 test: 0.0015728 best: 0.0015728 (836) total: 2h 53m 54s remaining: 33m 52s
837: learn: 0.0009136 test: 0.0015714 best: 0.0015714 (837) total: 2h 54m 8s remaining: 33m 39s
838: learn: 0.0009133 test: 0.0015713 best: 0.0015713 (838) total: 2h 54m 18s remaining: 33m 26s
839: learn: 0.0009128 test: 0.0015708 best: 0.0015708 (839) total: 2h 54m 28s remaining: 33m 14s
840: learn: 0.0009083 test: 0.0015663 best: 0.0015663 (840) total: 2h 54m 43s remaining: 33m 2s
841: learn: 0.0009064 test: 0.0015650 best: 0.0015650 (841) total: 2h 54m 57s remaining: 32m 49s
842: learn: 0.0009040 test: 0.0015632 best: 0.0015632 (842) total: 2h 55m 8s remaining: 32m 37s
843: learn: 0.0009022 test: 0.0015610 best: 0.0015610 (843) total: 2h 55m 21s remaining: 32m 24s
844: learn: 0.0009004 test: 0.0015591 best: 0.0015591 (844) total: 2h 55m 35s remaining: 32m 12s
845: learn: 0.0008990 test: 0.0015576 best: 0.0015576 (845) total: 2h 55m 47s remaining: 31m 59s
846: learn: 0.0008973 test: 0.0015551 best: 0.0015551 (846) total: 2h 56m remaining: 31m 47s
847: learn: 0.0008968 test: 0.0015547 best: 0.0015547 (847) total: 2h 56m 10s remaining: 31m 34s
848: learn: 0.0008946 test: 0.0015528 best: 0.0015528 (848) total: 2h 56m 24s remaining: 31m 22s
849: learn: 0.0008919 test: 0.0015503 best: 0.0015503 (849) total: 2h 56m 37s remaining: 31m 10s
850: learn: 0.0008863 test: 0.0015437 best: 0.0015437 (850) total: 2h 56m 51s remaining: 30m 57s
851: learn: 0.0008851 test: 0.0015432 best: 0.0015432 (851) total: 2h 57m 4s remaining: 30m 45s
852: learn: 0.0008828 test: 0.0015412 best: 0.0015412 (852) total: 2h 57m 18s remaining: 30m 33s
853: learn: 0.0008810 test: 0.0015400 best: 0.0015400 (853) total: 2h 57m 31s remaining: 30m 20s
854: learn: 0.0008794 test: 0.0015390 best: 0.0015390 (854) total: 2h 57m 43s remaining: 30m 8s
855: learn: 0.0008777 test: 0.0015376 best: 0.0015376 (855) total: 2h 57m 56s remaining: 29m 56s
856: learn: 0.0008755 test: 0.0015355 best: 0.0015355 (856) total: 2h 58m 8s remaining: 29m 43s
857: learn: 0.0008739 test: 0.0015335 best: 0.0015335 (857) total: 2h 58m 20s remaining: 29m 30s
858: learn: 0.0008720 test: 0.0015315 best: 0.0015315 (858) total: 2h 58m 33s remaining: 29m 18s
859: learn: 0.0008705 test: 0.0015300 best: 0.0015300 (859) total: 2h 58m 46s remaining: 29m 6s
860: learn: 0.0008695 test: 0.0015289 best: 0.0015289 (860) total: 2h 58m 58s remaining: 28m 53s
861: learn: 0.0008685 test: 0.0015277 best: 0.0015277 (861) total: 2h 59m 10s remaining: 28m 41s
862: learn: 0.0008675 test: 0.0015269 best: 0.0015269 (862) total: 2h 59m 22s remaining: 28m 28s
863: learn: 0.0008649 test: 0.0015247 best: 0.0015247 (863) total: 2h 59m 36s remaining: 28m 16s
864: learn: 0.0008641 test: 0.0015246 best: 0.0015246 (864) total: 2h 59m 48s remaining: 28m 3s
865: learn: 0.0008633 test: 0.0015241 best: 0.0015241 (865) total: 3h remaining: 27m 51s
866: learn: 0.0008625 test: 0.0015242 best: 0.0015241 (865) total: 3h 13s remaining: 27m 38s
867: learn: 0.0008612 test: 0.0015216 best: 0.0015216 (867) total: 3h 26s remaining: 27m 26s
868: learn: 0.0008562 test: 0.0015160 best: 0.0015160 (868) total: 3h 39s remaining: 27m 14s
869: learn: 0.0008549 test: 0.0015153 best: 0.0015153 (869) total: 3h 52s remaining: 27m 1s
870: learn: 0.0008527 test: 0.0015123 best: 0.0015123 (870) total: 3h 1m 4s remaining: 26m 49s
871: learn: 0.0008511 test: 0.0015111 best: 0.0015111 (871) total: 3h 1m 17s remaining: 26m 36s
872: learn: 0.0008491 test: 0.0015107 best: 0.0015107 (872) total: 3h 1m 30s remaining: 26m 24s
873: learn: 0.0008481 test: 0.0015099 best: 0.0015099 (873) total: 3h 1m 42s remaining: 26m 11s
874: learn: 0.0008464 test: 0.0015087 best: 0.0015087 (874) total: 3h 1m 56s remaining: 25m 59s
875: learn: 0.0008445 test: 0.0015058 best: 0.0015058 (875) total: 3h 2m 9s remaining: 25m 47s
876: learn: 0.0008420 test: 0.0015029 best: 0.0015029 (876) total: 3h 2m 23s remaining: 25m 34s
877: learn: 0.0008338 test: 0.0014942 best: 0.0014942 (877) total: 3h 2m 36s remaining: 25m 22s
878: learn: 0.0008305 test: 0.0014908 best: 0.0014908 (878) total: 3h 2m 48s remaining: 25m 9s
879: learn: 0.0008300 test: 0.0014904 best: 0.0014904 (879) total: 3h 2m 59s remaining: 24m 57s
880: learn: 0.0008291 test: 0.0014900 best: 0.0014900 (880) total: 3h 3m 11s remaining: 24m 44s
881: learn: 0.0008276 test: 0.0014890 best: 0.0014890 (881) total: 3h 3m 24s remaining: 24m 32s
882: learn: 0.0008261 test: 0.0014878 best: 0.0014878 (882) total: 3h 3m 37s remaining: 24m 19s
883: learn: 0.0008253 test: 0.0014879 best: 0.0014878 (882) total: 3h 3m 49s remaining: 24m 7s
884: learn: 0.0008246 test: 0.0014869 best: 0.0014869 (884) total: 3h 3m 59s remaining: 23m 54s
885: learn: 0.0008236 test: 0.0014855 best: 0.0014855 (885) total: 3h 4m 11s remaining: 23m 42s
886: learn: 0.0008219 test: 0.0014838 best: 0.0014838 (886) total: 3h 4m 24s remaining: 23m 29s
887: learn: 0.0008203 test: 0.0014829 best: 0.0014829 (887) total: 3h 4m 35s remaining: 23m 16s
888: learn: 0.0008192 test: 0.0014815 best: 0.0014815 (888) total: 3h 4m 47s remaining: 23m 4s
889: learn: 0.0008163 test: 0.0014792 best: 0.0014792 (889) total: 3h 4m 59s remaining: 22m 51s
890: learn: 0.0008146 test: 0.0014776 best: 0.0014776 (890) total: 3h 5m 12s remaining: 22m 39s
891: learn: 0.0008134 test: 0.0014763 best: 0.0014763 (891) total: 3h 5m 25s remaining: 22m 27s
892: learn: 0.0008120 test: 0.0014739 best: 0.0014739 (892) total: 3h 5m 39s remaining: 22m 14s
893: learn: 0.0008109 test: 0.0014726 best: 0.0014726 (893) total: 3h 5m 52s remaining: 22m 2s
894: learn: 0.0008087 test: 0.0014710 best: 0.0014710 (894) total: 3h 6m 5s remaining: 21m 49s
895: learn: 0.0008075 test: 0.0014707 best: 0.0014707 (895) total: 3h 6m 19s remaining: 21m 37s
896: learn: 0.0008052 test: 0.0014701 best: 0.0014701 (896) total: 3h 6m 33s remaining: 21m 25s
897: learn: 0.0008045 test: 0.0014698 best: 0.0014698 (897) total: 3h 6m 46s remaining: 21m 12s
898: learn: 0.0008036 test: 0.0014697 best: 0.0014697 (898) total: 3h 6m 57s remaining: 21m
899: learn: 0.0008019 test: 0.0014683 best: 0.0014683 (899) total: 3h 7m 10s remaining: 20m 47s
900: learn: 0.0008013 test: 0.0014678 best: 0.0014678 (900) total: 3h 7m 22s remaining: 20m 35s
901: learn: 0.0007995 test: 0.0014673 best: 0.0014673 (901) total: 3h 7m 36s remaining: 20m 22s
902: learn: 0.0007991 test: 0.0014674 best: 0.0014673 (901) total: 3h 7m 46s remaining: 20m 10s
```

```
903: learn: 0.0007975 test: 0.0014659 best: 0.0014659 (903) total: 3h 7m 58s remaining: 19m 57s
904: learn: 0.0007963 test: 0.0014642 best: 0.0014642 (904) total: 3h 8m 10s remaining: 19m 45s
905: learn: 0.0007948 test: 0.0014635 best: 0.0014635 (905) total: 3h 8m 23s remaining: 19m 32s
906: learn: 0.0007940 test: 0.0014627 best: 0.0014627 (906) total: 3h 8m 34s remaining: 19m 20s
907: learn: 0.0007928 test: 0.0014622 best: 0.0014622 (907) total: 3h 8m 46s remaining: 19m 7s
908: learn: 0.0007922 test: 0.0014620 best: 0.0014620 (908) total: 3h 8m 57s remaining: 18m 55s
909: learn: 0.0007909 test: 0.0014612 best: 0.0014612 (909) total: 3h 9m 10s remaining: 18m 42s
910: learn: 0.0007899 test: 0.0014608 best: 0.0014608 (910) total: 3h 9m 24s remaining: 18m 30s
911: learn: 0.0007882 test: 0.0014598 best: 0.0014598 (911) total: 3h 9m 37s remaining: 18m 17s
912: learn: 0.0007858 test: 0.0014579 best: 0.0014579 (912) total: 3h 9m 51s remaining: 18m 5s
913: learn: 0.0007841 test: 0.0014556 best: 0.0014556 (913) total: 3h 10m 4s remaining: 17m 53s
914: learn: 0.0007824 test: 0.0014540 best: 0.0014540 (914) total: 3h 10m 15s remaining: 17m 40s
915: learn: 0.0007809 test: 0.0014524 best: 0.0014524 (915) total: 3h 10m 28s remaining: 17m 28s
916: learn: 0.0007784 test: 0.0014496 best: 0.0014496 (916) total: 3h 10m 41s remaining: 17m 15s
917: learn: 0.0007760 test: 0.0014481 best: 0.0014481 (917) total: 3h 10m 55s remaining: 17m 3s
918: learn: 0.0007736 test: 0.0014451 best: 0.0014451 (918) total: 3h 11m 6s remaining: 16m 50s
919: learn: 0.0007729 test: 0.0014444 best: 0.0014444 (919) total: 3h 11m 18s remaining: 16m 38s
920: learn: 0.0007713 test: 0.0014428 best: 0.0014428 (920) total: 3h 11m 30s remaining: 16m 25s
921: learn: 0.0007702 test: 0.0014421 best: 0.0014421 (921) total: 3h 11m 42s remaining: 16m 13s
922: learn: 0.0007679 test: 0.0014401 best: 0.0014401 (922) total: 3h 11m 57s remaining: 16m
923: learn: 0.0007674 test: 0.0014401 best: 0.0014401 (923) total: 3h 12m 9s remaining: 15m 48s
924: learn: 0.0007656 test: 0.0014389 best: 0.0014389 (924) total: 3h 12m 23s remaining: 15m 35s
925: learn: 0.0007643 test: 0.0014378 best: 0.0014378 (925) total: 3h 12m 35s remaining: 15m 23s
926: learn: 0.0007616 test: 0.0014362 best: 0.0014362 (926) total: 3h 12m 51s remaining: 15m 11s
927: learn: 0.0007582 test: 0.0014314 best: 0.0014314 (927) total: 3h 13m 4s remaining: 14m 58s
928: learn: 0.0007564 test: 0.0014304 best: 0.0014304 (928) total: 3h 13m 19s remaining: 14m 46s
929: learn: 0.0007556 test: 0.0014299 best: 0.0014299 (929) total: 3h 13m 31s remaining: 14m 33s
930: learn: 0.0007542 test: 0.0014285 best: 0.0014285 (930) total: 3h 13m 43s remaining: 14m 21s
931: learn: 0.0007531 test: 0.0014274 best: 0.0014274 (931) total: 3h 13m 54s remaining: 14m 8s
932: learn: 0.0007505 test: 0.0014256 best: 0.0014256 (932) total: 3h 14m 7s remaining: 13m 56s
933: learn: 0.0007486 test: 0.0014233 best: 0.0014233 (933) total: 3h 14m 20s remaining: 13m 43s
934: learn: 0.0007477 test: 0.0014230 best: 0.0014230 (934) total: 3h 14m 32s remaining: 13m 31s
935: learn: 0.0007465 test: 0.0014226 best: 0.0014226 (935) total: 3h 14m 46s remaining: 13m 19s
936: learn: 0.0007461 test: 0.0014223 best: 0.0014223 (936) total: 3h 14m 56s remaining: 13m 6s
937: learn: 0.0007451 test: 0.0014215 best: 0.0014215 (937) total: 3h 15m 8s remaining: 12m 53s
938: learn: 0.0007434 test: 0.0014205 best: 0.0014205 (938) total: 3h 15m 20s remaining: 12m 41s
939: learn: 0.0007429 test: 0.0014205 best: 0.0014205 (939) total: 3h 15m 31s remaining: 12m 28s
940: learn: 0.0007412 test: 0.0014188 best: 0.0014188 (940) total: 3h 15m 45s remaining: 12m 16s
941: learn: 0.0007386 test: 0.0014162 best: 0.0014162 (941) total: 3h 15m 59s remaining: 12m 4s
942: learn: 0.0007371 test: 0.0014149 best: 0.0014149 (942) total: 3h 16m 13s remaining: 11m 51s
943: learn: 0.0007365 test: 0.0014143 best: 0.0014143 (943) total: 3h 16m 25s remaining: 11m 39s
944: learn: 0.0007358 test: 0.0014135 best: 0.0014135 (944) total: 3h 16m 36s remaining: 11m 26s
945: learn: 0.0007321 test: 0.0014096 best: 0.0014096 (945) total: 3h 16m 50s remaining: 11m 14s
946: learn: 0.0007312 test: 0.0014093 best: 0.0014093 (946) total: 3h 17m 2s remaining: 11m 1s
947: learn: 0.0007302 test: 0.0014087 best: 0.0014087 (947) total: 3h 17m 15s remaining: 10m 49s
948: learn: 0.0007278 test: 0.0014063 best: 0.0014063 (948) total: 3h 17m 28s remaining: 10m 36s
949: learn: 0.0007250 test: 0.0014029 best: 0.0014029 (949) total: 3h 17m 40s remaining: 10m 24s
950: learn: 0.0007245 test: 0.0014023 best: 0.0014023 (950) total: 3h 17m 51s remaining: 10m 11s
951: learn: 0.0007225 test: 0.0013997 best: 0.0013997 (951) total: 3h 18m 5s remaining: 9m 59s
952: learn: 0.0007215 test: 0.0013995 best: 0.0013995 (952) total: 3h 18m 18s remaining: 9m 46s
953: learn: 0.0007204 test: 0.0013984 best: 0.0013984 (953) total: 3h 18m 32s remaining: 9m 34s
954: learn: 0.0007192 test: 0.0013980 best: 0.0013980 (954) total: 3h 18m 44s remaining: 9m 21s
955: learn: 0.0007167 test: 0.0013962 best: 0.0013962 (955) total: 3h 18m 58s remaining: 9m 9s
956: learn: 0.0007145 test: 0.0013935 best: 0.0013935 (956) total: 3h 19m 12s remaining: 8m 57s
957: learn: 0.0007129 test: 0.0013914 best: 0.0013914 (957) total: 3h 19m 26s remaining: 8m 44s
958: learn: 0.0007064 test: 0.0013822 best: 0.0013822 (958) total: 3h 19m 39s remaining: 8m 32s
959: learn: 0.0007059 test: 0.0013819 best: 0.0013819 (959) total: 3h 19m 50s remaining: 8m 19s
960: learn: 0.0007046 test: 0.0013802 best: 0.0013802 (960) total: 3h 20m 3s remaining: 8m 7s
961: learn: 0.0007015 test: 0.0013756 best: 0.0013756 (961) total: 3h 20m 16s remaining: 7m 54s
962: learn: 0.0007006 test: 0.0013757 best: 0.0013756 (961) total: 3h 20m 29s remaining: 7m 42s
963: learn: 0.0006998 test: 0.0013749 best: 0.0013749 (963) total: 3h 20m 41s remaining: 7m 29s
964: learn: 0.0006984 test: 0.0013731 best: 0.0013731 (964) total: 3h 20m 54s remaining: 7m 17s
965: learn: 0.0006956 test: 0.0013699 best: 0.0013699 (965) total: 3h 21m 8s remaining: 7m 4s
966: learn: 0.0006941 test: 0.0013699 best: 0.0013699 (966) total: 3h 21m 20s remaining: 6m 52s
967: learn: 0.0006927 test: 0.0013693 best: 0.0013693 (967) total: 3h 21m 33s remaining: 6m 39s
968: learn: 0.0006912 test: 0.0013679 best: 0.0013679 (968) total: 3h 21m 46s remaining: 6m 27s
969: learn: 0.0006901 test: 0.0013669 best: 0.0013669 (969) total: 3h 21m 58s remaining: 6m 14s
970: learn: 0.0006878 test: 0.0013648 best: 0.0013648 (970) total: 3h 22m 12s remaining: 6m 2s
971: learn: 0.0006871 test: 0.0013643 best: 0.0013643 (971) total: 3h 22m 24s remaining: 5m 49s
972: learn: 0.0006847 test: 0.0013617 best: 0.0013617 (972) total: 3h 22m 38s remaining: 5m 37s
973: learn: 0.0006822 test: 0.0013595 best: 0.0013595 (973) total: 3h 22m 52s remaining: 5m 24s
974: learn: 0.0006818 test: 0.0013596 best: 0.0013595 (973) total: 3h 23m 2s remaining: 5m 12s
975: learn: 0.0006795 test: 0.0013573 best: 0.0013573 (975) total: 3h 23m 16s remaining: 4m 59s
976: learn: 0.0006786 test: 0.0013570 best: 0.0013570 (976) total: 3h 23m 29s remaining: 4m 47s
977: learn: 0.0006767 test: 0.0013549 best: 0.0013549 (977) total: 3h 23m 42s remaining: 4m 34s
978: learn: 0.0006761 test: 0.0013546 best: 0.0013546 (978) total: 3h 23m 54s remaining: 4m 22s
979: learn: 0.0006749 test: 0.0013538 best: 0.0013538 (979) total: 3h 24m 7s remaining: 4m 9s
```

```
980: learn: 0.0006742 test: 0.0013539 best: 0.0013538 (979) total: 3h 24m 19s remaining: 3m 57s
981: learn: 0.0006737 test: 0.0013538 best: 0.0013538 (981) total: 3h 24m 31s remaining: 3m 44s
982: learn: 0.0006726 test: 0.0013533 best: 0.0013533 (982) total: 3h 24m 44s remaining: 3m 32s
983: learn: 0.0006724 test: 0.0013528 best: 0.0013528 (983) total: 3h 24m 54s remaining: 3m 19s
984: learn: 0.0006720 test: 0.0013522 best: 0.0013522 (984) total: 3h 25m 4s remaining: 3m 7s
985: learn: 0.0006707 test: 0.0013516 best: 0.0013516 (985) total: 3h 25m 19s remaining: 2m 54s
986: learn: 0.0006689 test: 0.0013502 best: 0.0013502 (986) total: 3h 25m 32s remaining: 2m 42s
987: learn: 0.0006659 test: 0.0013464 best: 0.0013464 (987) total: 3h 25m 46s remaining: 2m 29s
988: learn: 0.0006648 test: 0.0013457 best: 0.0013457 (988) total: 3h 26m remaining: 2m 17s
989: learn: 0.0006645 test: 0.0013451 best: 0.0013451 (989) total: 3h 26m 11s remaining: 2m 4s
990: learn: 0.0006632 test: 0.0013446 best: 0.0013446 (990) total: 3h 26m 23s remaining: 1m 52s
991: learn: 0.0006627 test: 0.0013443 best: 0.0013443 (991) total: 3h 26m 36s remaining: 1m 39s
992: learn: 0.0006619 test: 0.0013433 best: 0.0013433 (992) total: 3h 26m 48s remaining: 1m 27s
993: learn: 0.0006606 test: 0.0013416 best: 0.0013416 (993) total: 3h 27m 1s remaining: 1m 14s
994: learn: 0.0006593 test: 0.0013414 best: 0.0013414 (994) total: 3h 27m 15s remaining: 1m 2s
995: learn: 0.0006584 test: 0.0013415 best: 0.0013414 (994) total: 3h 27m 28s remaining: 50s
996: learn: 0.0006576 test: 0.0013409 best: 0.0013409 (996) total: 3h 27m 40s remaining: 37.5s
997: learn: 0.0006540 test: 0.0013375 best: 0.0013375 (997) total: 3h 27m 53s remaining: 25s
998: learn: 0.0006521 test: 0.0013373 best: 0.0013373 (998) total: 3h 28m 7s remaining: 12.5s
999: learn: 0.0006509 test: 0.0013369 best: 0.0013369 (999) total: 3h 28m 22s remaining: 0us

bestTest = 0.00133690021
bestIteration = 999

Count of trees in model = 1000
```

We can see from CV loss, the performance of catboost is poor than the XGBoost. So we'll move to next model.

In [ ]:

```
```

## SVM (SGDClassifier)

In [6]:

```
df=new_train.sample(80000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df=df.drop(featr,axis=1)
df.shape
```

Out[6]:

```
(80000, 120)
```

In [7]:

```
train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=
32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[7]:

```
((64000, 120), (16000, 120), 64000, 16000)
```

In [23]:

```
# https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html#

alpha = [10 ** x for x in range(-3, 3)]
for i in alpha:
    clf = make_pipeline(StandardScaler(),SGDClassifier(loss='hinge',penalty='l2',class_weight='bala
nced',eta0=i,n_jobs=-1))
    clf.fit(train_df, y_train)
    sig_clf = CalibratedClassifierCV(clf)
    sig_clf.fit(train_df, y_train)
    predict_y = sig_clf.predict_proba(train_df)
```

```
    print('TRAIN LOSS is ',i,log_loss(y_train, predict_y))
    predict_y=sig_clf.predict_proba(val_df)
    print('CV LOSS is ',i,log_loss(y_val, predict_y))
```

```
TRAIN LOSS is  0.001 0.8720556432501266
CV LOSS is  0.001 0.8747265886714645
TRAIN LOSS is  0.01 0.8664445588266754
CV LOSS is  0.01 0.8696406818949567
TRAIN LOSS is  0.1 0.8678965864187506
CV LOSS is  0.1 0.8706483655081254
TRAIN LOSS is  1 0.8658558732377809
CV LOSS is  1 0.8681625883776843
TRAIN LOSS is  10 0.8670826925857118
CV LOSS is  10 0.8692886328944602
TRAIN LOSS is  100 0.8654509588439843
CV LOSS is  100 0.8682710922282438
```

In [24]:

```
# full data
y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[24]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [25]:

```
clf =
make_pipeline(StandardScaler(),SGDClassifier(loss='hinge',penalty='l2',class_weight='balanced',eta
0=1,n_jobs=-1))
clf.fit(train_df, y_train)
sig_clf = CalibratedClassifierCV(clf)
sig_clf.fit(train_df, y_train)
predict_y = sig_clf.predict_proba(train_df)
print('TRAIN LOSS is ',log_loss(y_train, predict_y))
predict_y=sig_clf.predict_proba(val_df)
print('CV LOSS is ',log_loss(y_val, predict_y))
```

```
TRAIN LOSS is  0.8698698116734259
CV LOSS is  0.8698199539967327
```

We can see from CV loss, the performance of SVM is poor than the XGBoost ( XGBoost- best till now). So we'll move to next model.

## Logistic Regression (SGDClassifier)

In [22]:

```
df=new_train.sample(80000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df=df.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=
32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[22]:

```
((64000, 120), (16000, 120), 64000, 16000)
```

In [11]:

```
from sklearn.linear_model import SGDClassifier
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import make_pipeline

alpha = [10 ** x for x in range(-3, 3)]
for i in alpha:
    clf = make_pipeline(StandardScaler(),SGDClassifier(loss='log',penalty='l2',class_weight='balanc
ed',eta0=i,n_jobs=-1))
    clf.fit(train_df, y_train)
    sig_clf = CalibratedClassifierCV(clf)
    sig_clf.fit(train_df, y_train)
    predict_y = sig_clf.predict_proba(train_df)
    print('TRAIN LOSS is ',log_loss(y_train, predict_y))
    predict_y=sig_clf.predict_proba(val_df)
    print('CV LOSS is ',log_loss(y_val, predict_y))
```

```
TRAIN LOSS is  0.8531547428670425
CV LOSS is  0.8573718257954563
TRAIN LOSS is  0.8520681057508018
CV LOSS is  0.8555931013879499
TRAIN LOSS is  0.8543615100825678
CV LOSS is  0.8601929524497007
TRAIN LOSS is  0.8534581367639129
CV LOSS is  0.8585225472996906
TRAIN LOSS is  0.85384547207583
CV LOSS is  0.8590822718273724
TRAIN LOSS is  0.8538086442815045
CV LOSS is  0.8586933339967835
```

In [15]:

```
# full data
y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[15]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [16]:

```
clf = make_pipeline(StandardScaler(),SGDClassifier(loss='log',penalty='l2',class_weight='balanced'
,eta0=0.01,n_jobs=-1))
clf.fit(train_df, y_train)
sig_clf = CalibratedClassifierCV(clf)
sig_clf.fit(train_df, y_train)
predict_y = sig_clf.predict_proba(train_df)
print('TRAIN LOSS is ',log_loss(y_train, predict_y))
predict_y=sig_clf.predict_proba(val_df)
print('CV LOSS is ',log_loss(y_val, predict_y))
```

```
TRAIN LOSS is  0.8526831596361619
CV LOSS is  0.8526577092862271
```

We can see from CV loss, the performance of logistic regression is poor than the XGBoost ( XGBoost- best till now). So we'll move to next model.

In [ ]:

```
```

## XgBoost 2

We have 2 forms of XGBoost:

1. Direct xgboost library (xgb)
2. sklearn wrapper for XGBoost (XGBClassifier)

In [8]:

```
# https://blog.cambridgespark.com/hyperparameter-tuning-in-xgboost-4ff9100a3b2f
# https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/

import xgboost as xgb
from xgboost.sklearn import XGBClassifier

df=new_train.sample(90000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df=df.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[8]:

```
((72000, 120), (18000, 120), 72000, 18000)
```

### Tuning Parameters with xgb

In [15]:

```
dtrain = xgb.DMatrix(train_df, label=y_train)
dcv = xgb.DMatrix(val_df, label=y_val)

params = {'max_depth':6,
    'min_child_weight': 1,
    'gamma':0,
    'eta':.3,
    'subsample': 1,
    'colsample_bytree': 1,
    'num_class':4,
    'scale_pos_weight':1,
    'objective':'multi:softprob', 'eval_metric':'mlogloss'}

model = xgb.train(params,dtrain,num_boost_round=999,evals=[(dcv, "Test")],early_stopping_rounds=10)

print(model.best_score,model.best_iteration+1)

# training with default params for later reference. next will do the hyperparameter tuning
```

```
[0] Test-mlogloss:1.12452
Will train until Test-mlogloss hasn't improved in 10 rounds.
[1] Test-mlogloss:0.95366
[2] Test-mlogloss:0.84074
[3] Test-mlogloss:0.75555
[4] Test-mlogloss:0.68949
[5] Test-mlogloss:0.62257
[6] Test-mlogloss:0.57108
[7] Test-mlogloss:0.51839
[8] Test-mlogloss:0.47667
[9] Test-mlogloss:0.44308
[10] Test-mlogloss:0.40764
[11] Test-mlogloss:0.38641
[12] Test-mlogloss:0.36883
[13] Test-mlogloss:0.34923
[14] Test-mlogloss:0.32720
[15] Test-mlogloss:0.30753
[16] Test-mlogloss:0.29181
[17] Test-mlogloss:0.28132
[18] Test-mlogloss:0.26581
[19] Test-mlogloss:0.25529
[20] Test-mlogloss:0.24168
[21] Test-mlogloss:0.23234
```

```
[22] Test-mlogloss:0.22613
[23] Test-mlogloss:0.21920
[24] Test-mlogloss:0.21218
[25] Test-mlogloss:0.20705
[26] Test-mlogloss:0.20300
[27] Test-mlogloss:0.19861
[28] Test-mlogloss:0.19311
[29] Test-mlogloss:0.18810
[30] Test-mlogloss:0.18424
[31] Test-mlogloss:0.17473
[32] Test-mlogloss:0.17167
[33] Test-mlogloss:0.16564
[34] Test-mlogloss:0.15912
[35] Test-mlogloss:0.15581
[36] Test-mlogloss:0.15296
[37] Test-mlogloss:0.15010
[38] Test-mlogloss:0.14688
[39] Test-mlogloss:0.14458
[40] Test-mlogloss:0.14238
[41] Test-mlogloss:0.14111
[42] Test-mlogloss:0.13703
[43] Test-mlogloss:0.13376
[44] Test-mlogloss:0.13156
[45] Test-mlogloss:0.13004
[46] Test-mlogloss:0.12823
[47] Test-mlogloss:0.12595
[48] Test-mlogloss:0.12282
[49] Test-mlogloss:0.12141
[50] Test-mlogloss:0.11953
[51] Test-mlogloss:0.11828
[52] Test-mlogloss:0.11577
[53] Test-mlogloss:0.11319
[54] Test-mlogloss:0.11162
[55] Test-mlogloss:0.11067
[56] Test-mlogloss:0.10945
[57] Test-mlogloss:0.10773
[58] Test-mlogloss:0.10667
[59] Test-mlogloss:0.10567
[60] Test-mlogloss:0.10364
[61] Test-mlogloss:0.10253
[62] Test-mlogloss:0.10106
[63] Test-mlogloss:0.09966
[64] Test-mlogloss:0.09853
[65] Test-mlogloss:0.09701
[66] Test-mlogloss:0.09577
[67] Test-mlogloss:0.09429
[68] Test-mlogloss:0.09336
[69] Test-mlogloss:0.09254
[70] Test-mlogloss:0.09128
[71] Test-mlogloss:0.09021
[72] Test-mlogloss:0.08926
[73] Test-mlogloss:0.08866
[74] Test-mlogloss:0.08737
[75] Test-mlogloss:0.08650
[76] Test-mlogloss:0.08593
[77] Test-mlogloss:0.08506
[78] Test-mlogloss:0.08426
[79] Test-mlogloss:0.08366
[80] Test-mlogloss:0.08277
[81] Test-mlogloss:0.08166
[82] Test-mlogloss:0.08121
[83] Test-mlogloss:0.08084
[84] Test-mlogloss:0.08032
[85] Test-mlogloss:0.07943
[86] Test-mlogloss:0.07873
[87] Test-mlogloss:0.07751
[88] Test-mlogloss:0.07674
[89] Test-mlogloss:0.07576
[90] Test-mlogloss:0.07535
[91] Test-mlogloss:0.07508
[92] Test-mlogloss:0.07426
[93] Test-mlogloss:0.07374
[94] Test-mlogloss:0.07318
[95] Test-mlogloss:0.07284
[96] Test-mlogloss:0.07226
[97] Test-mlogloss:0.07157
[98] Test-mlogloss:0.07135
```

```
[99] Test-mlogloss:0.07092
[100] Test-mlogloss:0.07040
[101] Test-mlogloss:0.06947
[102] Test-mlogloss:0.06864
[103] Test-mlogloss:0.06840
[104] Test-mlogloss:0.06802
[105] Test-mlogloss:0.06769
[106] Test-mlogloss:0.06726
[107] Test-mlogloss:0.06668
[108] Test-mlogloss:0.06628
[109] Test-mlogloss:0.06586
[110] Test-mlogloss:0.06551
[111] Test-mlogloss:0.06486
[112] Test-mlogloss:0.06445
[113] Test-mlogloss:0.06384
[114] Test-mlogloss:0.06313
[115] Test-mlogloss:0.06255
[116] Test-mlogloss:0.06219
[117] Test-mlogloss:0.06180
[118] Test-mlogloss:0.06118
[119] Test-mlogloss:0.06084
[120] Test-mlogloss:0.06052
[121] Test-mlogloss:0.06021
[122] Test-mlogloss:0.06008
[123] Test-mlogloss:0.05968
[124] Test-mlogloss:0.05949
[125] Test-mlogloss:0.05933
[126] Test-mlogloss:0.05903
[127] Test-mlogloss:0.05863
[128] Test-mlogloss:0.05855
[129] Test-mlogloss:0.05816
[130] Test-mlogloss:0.05786
[131] Test-mlogloss:0.05763
[132] Test-mlogloss:0.05718
[133] Test-mlogloss:0.05694
[134] Test-mlogloss:0.05674
[135] Test-mlogloss:0.05659
[136] Test-mlogloss:0.05609
[137] Test-mlogloss:0.05595
[138] Test-mlogloss:0.05582
[139] Test-mlogloss:0.05552
[140] Test-mlogloss:0.05498
[141] Test-mlogloss:0.05482
[142] Test-mlogloss:0.05465
[143] Test-mlogloss:0.05441
[144] Test-mlogloss:0.05428
[145] Test-mlogloss:0.05397
[146] Test-mlogloss:0.05380
[147] Test-mlogloss:0.05379
[148] Test-mlogloss:0.05365
[149] Test-mlogloss:0.05353
[150] Test-mlogloss:0.05346
[151] Test-mlogloss:0.05327
[152] Test-mlogloss:0.05301
[153] Test-mlogloss:0.05263
[154] Test-mlogloss:0.05253
[155] Test-mlogloss:0.05248
[156] Test-mlogloss:0.05247
[157] Test-mlogloss:0.05239
[158] Test-mlogloss:0.05207
[159] Test-mlogloss:0.05193
[160] Test-mlogloss:0.05177
[161] Test-mlogloss:0.05163
[162] Test-mlogloss:0.05137
[163] Test-mlogloss:0.05120
[164] Test-mlogloss:0.05112
[165] Test-mlogloss:0.05096
[166] Test-mlogloss:0.05079
[167] Test-mlogloss:0.05068
[168] Test-mlogloss:0.05067
[169] Test-mlogloss:0.05056
[170] Test-mlogloss:0.05058
[171] Test-mlogloss:0.05052
[172] Test-mlogloss:0.05037
[173] Test-mlogloss:0.04999
[174] Test-mlogloss:0.04995
[175] Test-mlogloss:0.04987
```

```
[176] Test-mlogloss:0.04967
[177] Test-mlogloss:0.04964
[178] Test-mlogloss:0.04954
[179] Test-mlogloss:0.04950
[180] Test-mlogloss:0.04949
[181] Test-mlogloss:0.04945
[182] Test-mlogloss:0.04941
[183] Test-mlogloss:0.04930
[184] Test-mlogloss:0.04923
[185] Test-mlogloss:0.04920
[186] Test-mlogloss:0.04918
[187] Test-mlogloss:0.04906
[188] Test-mlogloss:0.04888
[189] Test-mlogloss:0.04891
[190] Test-mlogloss:0.04884
[191] Test-mlogloss:0.04867
[192] Test-mlogloss:0.04858
[193] Test-mlogloss:0.04853
[194] Test-mlogloss:0.04851
[195] Test-mlogloss:0.04849
[196] Test-mlogloss:0.04841
[197] Test-mlogloss:0.04840
[198] Test-mlogloss:0.04830
[199] Test-mlogloss:0.04822
[200] Test-mlogloss:0.04815
[201] Test-mlogloss:0.04814
[202] Test-mlogloss:0.04805
[203] Test-mlogloss:0.04808
[204] Test-mlogloss:0.04803
[205] Test-mlogloss:0.04791
[206] Test-mlogloss:0.04788
[207] Test-mlogloss:0.04779
[208] Test-mlogloss:0.04775
[209] Test-mlogloss:0.04765
[210] Test-mlogloss:0.04761
[211] Test-mlogloss:0.04768
[212] Test-mlogloss:0.04756
[213] Test-mlogloss:0.04743
[214] Test-mlogloss:0.04742
[215] Test-mlogloss:0.04725
[216] Test-mlogloss:0.04715
[217] Test-mlogloss:0.04722
[218] Test-mlogloss:0.04713
[219] Test-mlogloss:0.04712
[220] Test-mlogloss:0.04710
[221] Test-mlogloss:0.04702
[222] Test-mlogloss:0.04692
[223] Test-mlogloss:0.04697
[224] Test-mlogloss:0.04692
[225] Test-mlogloss:0.04688
[226] Test-mlogloss:0.04686
[227] Test-mlogloss:0.04685
[228] Test-mlogloss:0.04668
[229] Test-mlogloss:0.04664
[230] Test-mlogloss:0.04649
[231] Test-mlogloss:0.04648
[232] Test-mlogloss:0.04648
[233] Test-mlogloss:0.04651
[234] Test-mlogloss:0.04649
[235] Test-mlogloss:0.04649
[236] Test-mlogloss:0.04652
[237] Test-mlogloss:0.04653
[238] Test-mlogloss:0.04653
[239] Test-mlogloss:0.04648
[240] Test-mlogloss:0.04636
[241] Test-mlogloss:0.04628
[242] Test-mlogloss:0.04628
[243] Test-mlogloss:0.04626
[244] Test-mlogloss:0.04624
[245] Test-mlogloss:0.04629
[246] Test-mlogloss:0.04626
[247] Test-mlogloss:0.04623
[248] Test-mlogloss:0.04619
[249] Test-mlogloss:0.04610
[250] Test-mlogloss:0.04602
[251] Test-mlogloss:0.04598
[252] Test-mlogloss:0.04592
```

```
[253] Test-mlogloss:0.04585
[254] Test-mlogloss:0.04584
[255] Test-mlogloss:0.04585
[256] Test-mlogloss:0.04577
[257] Test-mlogloss:0.04573
[258] Test-mlogloss:0.04570
[259] Test-mlogloss:0.04560
[260] Test-mlogloss:0.04550
[261] Test-mlogloss:0.04556
[262] Test-mlogloss:0.04555
[263] Test-mlogloss:0.04554
[264] Test-mlogloss:0.04545
[265] Test-mlogloss:0.04542
[266] Test-mlogloss:0.04543
[267] Test-mlogloss:0.04550
[268] Test-mlogloss:0.04544
[269] Test-mlogloss:0.04540
[270] Test-mlogloss:0.04542
[271] Test-mlogloss:0.04539
[272] Test-mlogloss:0.04542
[273] Test-mlogloss:0.04542
[274] Test-mlogloss:0.04553
[275] Test-mlogloss:0.04551
[276] Test-mlogloss:0.04553
[277] Test-mlogloss:0.04552
[278] Test-mlogloss:0.04544
[279] Test-mlogloss:0.04544
[280] Test-mlogloss:0.04542
[281] Test-mlogloss:0.04542
Stopping. Best iteration:
[271] Test-mlogloss:0.04539

0.045387 272
```

In [24]:

```
def xgboost_cal(f1,f2,gridsearch_params):
    ''' takes features and do param tuning with xgb.csv '''

    min_loss = float("Inf")
    best_params = None


    for a, b in gridsearch_params:
        params[f1] = a
        params[f2] = b

        cv_results = xgb.cv(
            params,
            dtrain,
            num_boost_round=999,
            seed=42,
            nfold=3,
            metrics={'mlogloss'},
            early_stopping_rounds=10
        )

        mean_loss = cv_results['test-mlogloss-mean'].min()
        boost_rounds = cv_results['test-mlogloss-mean'].argmin()
        print(mean_loss, boost_rounds)
        if mean_loss < min_loss:
            min_loss = mean_loss
            best_params = (a,b)
    print("Best params: {}, {}, Loss: {}".format(best_params[0], best_params[1], min_loss))
    return best_params[0], best_params[1]
```

**Step-1 : tuning 'max_depth' and 'min_child_weight'**

In [25]:

```
# tuning 'max_depth' and 'min_child_weight'

gridsearch_params = [(depth, child_weight) for depth in range(9,12) for child_weight in range(5,8)]
depth,c weight=xgboost cal('max depth','min child weight',gridsearch params)
```

```
params['max_depth'] = depth
params['min_child_weight'] = c_weight
```

```
0.06649100000000001 129
0.06830066666666666 142
0.06823166666666666 160
0.06722333333333333 111
0.06872 123
0.06779433333333333 134
0.06713766666666666 109
0.06776833333333333 113
0.069507 133
Best params: 9, 5, Loss: 0.06649100000000001
```

**Step-2 : tuning 'subsample' and 'colsample_bytree'**

In [27]:

```
# tuning 'subsample' and 'colsample_bytree'

gridsearch_params = [(s, c) for s in [i/10. for i in range(7,11)] for c in [i/10. for i in range(7,1
1)]]
subsample,colsample_bytree=xgboost_cal('subsample','colsample_bytree',gridsearch_params)
params['subsample'] = subsample
params['colsample_bytree'] = colsample_bytree
```

```
0.07875566666666667 164
0.07774066666666667 133
0.07695466666666666 132
0.07596333333333333 124
0.07623266666666667 149
0.073443 128
0.07268566666666666 135
0.07318000000000001 127
0.072394 138
0.071999 139
0.07063533333333334 138
0.07056733333333333 116
0.07034566666666668 145
0.069494 151
0.06819533333333333 131
0.06649100000000001 129
Best params: 1.0, 1.0, Loss: 0.06649100000000001
```

**Step-3 : tuning Learning rate**

In [8]:

```
# tuning 'learning rate' or 'eta'

for i in [0.01,0.1,0.2,0.3,0.38]:
    params['eta'] = i
    cv_results = xgb.cv(params,dtrain,num_boost_round=999,seed=42,nfold=3,metrics=['mlogloss'],earl
y_stopping_rounds=10)
    mean_loss = cv_results['test-mlogloss-mean'].min()
    boost_rounds = cv_results['test-mlogloss-mean'].argmin()
    print(mean_loss, boost_rounds)
    if mean_loss < min_loss:
        min_loss = mean_loss
        best_params = i
print("Best params: {}, Loss: {}".format(best_params, min_loss))
params['eta']=best_params
```

```
0.10636433333333332 998
0.06819233333333333 383
0.068537 225
0.06932033333333333 125
0.06988266666666666 112
Best params: 0.1, Loss: 0.06819233333333333
```

In [10]:

```
# tuning 'learning rate' or 'eta' ---> narrowing down the range

min_loss = float("Inf")
best_params = None

for i in [0.08,0.12,0.15]:
    params['eta'] = i
    cv_results = xgb.cv(params,dtrain,num_boost_round=999,seed=42,nfold=3,metrics=['mlogloss'],earl
y_stopping_rounds=10)
    mean_loss = cv_results['test-mlogloss-mean'].min()
    boost_rounds = cv_results['test-mlogloss-mean'].argmin()
    print(mean_loss, boost_rounds)
    if mean_loss < min_loss:
        min_loss = mean_loss
        best_params = i
print("Best params: {}, Loss: {}".format(best_params, min_loss))
params['eta']=best_params
```

```
0.067603 519
0.06806566666666668 331
0.06918733333333334 262
Best params: 0.08, Loss: 0.067603
```

params = {'max_depth':9, 'min_child_weight': 5, 'eta':0.08, 'subsample': 1, 'colsample_bytree': 1, 'num_class':4, 'scale_pos_weight':1, 'objective':'multi:softprob', 'eval_metric':'mlogloss'}

**Step-4 : finding optimal estimators**

In [48]:

```
# finding 'num_boost_round'

dtrain = xgb.DMatrix(train_df, label=y_train)
dcv = xgb.DMatrix(val_df, label=y_val)

cvresult = xgb.cv(params, dtrain, num_boost_round=999, nfold=3,seed=42,
            metrics=['mlogloss'], early_stopping_rounds=10)
n_estimators=cvresult.shape[0]

n_estimators
```

Out[48]:

```
425
```

**Final training**

In [13]:

```
# full data

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[13]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [53]:

```
dtrain = xgb.DMatrix(train_df, label=y_train)
dcv = xgb.DMatrix(val_df, label=y_val)
```

```
model = xgb.train(params,dtrain,num_boost_round=n_estimators,evals=[(dcv, "Test")])
```

```
[12:14:43] WARNING: /workspace/src/learner.cc:480:
Parameters: { scale_pos_weight } might not be used.

  This may not be accurate due to some parameters are only used in language bindings but
  passed down to XGBoost core.  Or some parameters are not used but slip through this
  verification. Please open an issue if you find above cases.


[0] Test-mlogloss:1.30171
[1] Test-mlogloss:1.21594
[2] Test-mlogloss:1.14316
[3] Test-mlogloss:1.07392
[4] Test-mlogloss:1.01561
[5] Test-mlogloss:0.95997
[6] Test-mlogloss:0.90928
[7] Test-mlogloss:0.86623
[8] Test-mlogloss:0.82268
[9] Test-mlogloss:0.78995
[10] Test-mlogloss:0.75958
[11] Test-mlogloss:0.72835
[12] Test-mlogloss:0.69708
[13] Test-mlogloss:0.67191
[14] Test-mlogloss:0.64933
[15] Test-mlogloss:0.62838
[16] Test-mlogloss:0.60762
[17] Test-mlogloss:0.58873
[18] Test-mlogloss:0.57142
[19] Test-mlogloss:0.54702
[20] Test-mlogloss:0.53069
[21] Test-mlogloss:0.51020
[22] Test-mlogloss:0.48978
[23] Test-mlogloss:0.47505
[24] Test-mlogloss:0.46006
[25] Test-mlogloss:0.44311
[26] Test-mlogloss:0.42391
[27] Test-mlogloss:0.40935
[28] Test-mlogloss:0.39571
[29] Test-mlogloss:0.38281
[30] Test-mlogloss:0.36720
[31] Test-mlogloss:0.35462
[32] Test-mlogloss:0.34167
[33] Test-mlogloss:0.32931
[34] Test-mlogloss:0.32141
[35] Test-mlogloss:0.31097
[36] Test-mlogloss:0.30126
[37] Test-mlogloss:0.29085
[38] Test-mlogloss:0.28351
[39] Test-mlogloss:0.27461
[40] Test-mlogloss:0.26797
[41] Test-mlogloss:0.26173
[42] Test-mlogloss:0.25511
[43] Test-mlogloss:0.24980
[44] Test-mlogloss:0.24405
[45] Test-mlogloss:0.23850
[46] Test-mlogloss:0.23317
[47] Test-mlogloss:0.22734
[48] Test-mlogloss:0.22230
[49] Test-mlogloss:0.21717
[50] Test-mlogloss:0.21248
[51] Test-mlogloss:0.20846
[52] Test-mlogloss:0.20506
[53] Test-mlogloss:0.20083
[54] Test-mlogloss:0.19688
[55] Test-mlogloss:0.19279
[56] Test-mlogloss:0.18865
[57] Test-mlogloss:0.18452
[58] Test-mlogloss:0.17995
[59] Test-mlogloss:0.17654
[60] Test-mlogloss:0.17398
[61] Test-mlogloss:0.17051
[62] Test-mlogloss:0.16711
[63] Test-mlogloss:0.16403
[64] Test-mlogloss:0.16123
[65] Test-mlogloss:0.15885
[66] Test-mlogloss:0.15652
```

```
[66] Test-mlogloss:0.15652
[67] Test-mlogloss:0.15369
[68] Test-mlogloss:0.15102
[69] Test-mlogloss:0.14860
[70] Test-mlogloss:0.14648
[71] Test-mlogloss:0.14475
[72] Test-mlogloss:0.14258
[73] Test-mlogloss:0.14043
[74] Test-mlogloss:0.13870
[75] Test-mlogloss:0.13679
[76] Test-mlogloss:0.13483
[77] Test-mlogloss:0.13346
[78] Test-mlogloss:0.13151
[79] Test-mlogloss:0.12991
[80] Test-mlogloss:0.12790
[81] Test-mlogloss:0.12648
[82] Test-mlogloss:0.12497
[83] Test-mlogloss:0.12345
[84] Test-mlogloss:0.12228
[85] Test-mlogloss:0.12114
[86] Test-mlogloss:0.11961
[87] Test-mlogloss:0.11848
[88] Test-mlogloss:0.11737
[89] Test-mlogloss:0.11605
[90] Test-mlogloss:0.11488
[91] Test-mlogloss:0.11366
[92] Test-mlogloss:0.11255
[93] Test-mlogloss:0.11121
[94] Test-mlogloss:0.11010
[95] Test-mlogloss:0.10937
[96] Test-mlogloss:0.10838
[97] Test-mlogloss:0.10733
[98] Test-mlogloss:0.10645
[99] Test-mlogloss:0.10530
[100] Test-mlogloss:0.10416
[101] Test-mlogloss:0.10324
[102] Test-mlogloss:0.10233
[103] Test-mlogloss:0.10147
[104] Test-mlogloss:0.10064
[105] Test-mlogloss:0.09968
[106] Test-mlogloss:0.09914
[107] Test-mlogloss:0.09835
[108] Test-mlogloss:0.09770
[109] Test-mlogloss:0.09688
[110] Test-mlogloss:0.09627
[111] Test-mlogloss:0.09561
[112] Test-mlogloss:0.09482
[113] Test-mlogloss:0.09424
[114] Test-mlogloss:0.09345
[115] Test-mlogloss:0.09296
[116] Test-mlogloss:0.09222
[117] Test-mlogloss:0.09159
[118] Test-mlogloss:0.09107
[119] Test-mlogloss:0.09052
[120] Test-mlogloss:0.08998
[121] Test-mlogloss:0.08938
[122] Test-mlogloss:0.08902
[123] Test-mlogloss:0.08869
[124] Test-mlogloss:0.08798
[125] Test-mlogloss:0.08764
[126] Test-mlogloss:0.08729
[127] Test-mlogloss:0.08656
[128] Test-mlogloss:0.08604
[129] Test-mlogloss:0.08558
[130] Test-mlogloss:0.08495
[131] Test-mlogloss:0.08447
[132] Test-mlogloss:0.08409
[133] Test-mlogloss:0.08355
[134] Test-mlogloss:0.08327
[135] Test-mlogloss:0.08279
[136] Test-mlogloss:0.08246
[137] Test-mlogloss:0.08189
[138] Test-mlogloss:0.08157
[139] Test-mlogloss:0.08122
[140] Test-mlogloss:0.08086
[141] Test-mlogloss:0.08054
[142] Test-mlogloss:0.08026
```

```
[143] Test-mlogloss:0.07999
[144] Test-mlogloss:0.07965
[145] Test-mlogloss:0.07930
[146] Test-mlogloss:0.07904
[147] Test-mlogloss:0.07861
[148] Test-mlogloss:0.07823
[149] Test-mlogloss:0.07798
[150] Test-mlogloss:0.07783
[151] Test-mlogloss:0.07755
[152] Test-mlogloss:0.07710
[153] Test-mlogloss:0.07682
[154] Test-mlogloss:0.07655
[155] Test-mlogloss:0.07615
[156] Test-mlogloss:0.07573
[157] Test-mlogloss:0.07547
[158] Test-mlogloss:0.07517
[159] Test-mlogloss:0.07485
[160] Test-mlogloss:0.07452
[161] Test-mlogloss:0.07430
[162] Test-mlogloss:0.07401
[163] Test-mlogloss:0.07373
[164] Test-mlogloss:0.07342
[165] Test-mlogloss:0.07306
[166] Test-mlogloss:0.07272
[167] Test-mlogloss:0.07232
[168] Test-mlogloss:0.07209
[169] Test-mlogloss:0.07183
[170] Test-mlogloss:0.07153
[171] Test-mlogloss:0.07116
[172] Test-mlogloss:0.07087
[173] Test-mlogloss:0.07058
[174] Test-mlogloss:0.07031
[175] Test-mlogloss:0.07002
[176] Test-mlogloss:0.06981
[177] Test-mlogloss:0.06950
[178] Test-mlogloss:0.06931
[179] Test-mlogloss:0.06913
[180] Test-mlogloss:0.06897
[181] Test-mlogloss:0.06876
[182] Test-mlogloss:0.06858
[183] Test-mlogloss:0.06833
[184] Test-mlogloss:0.06800
[185] Test-mlogloss:0.06779
[186] Test-mlogloss:0.06764
[187] Test-mlogloss:0.06737
[188] Test-mlogloss:0.06716
[189] Test-mlogloss:0.06692
[190] Test-mlogloss:0.06679
[191] Test-mlogloss:0.06642
[192] Test-mlogloss:0.06622
[193] Test-mlogloss:0.06590
[194] Test-mlogloss:0.06576
[195] Test-mlogloss:0.06560
[196] Test-mlogloss:0.06544
[197] Test-mlogloss:0.06527
[198] Test-mlogloss:0.06509
[199] Test-mlogloss:0.06494
[200] Test-mlogloss:0.06470
[201] Test-mlogloss:0.06445
[202] Test-mlogloss:0.06429
[203] Test-mlogloss:0.06414
[204] Test-mlogloss:0.06392
[205] Test-mlogloss:0.06377
[206] Test-mlogloss:0.06351
[207] Test-mlogloss:0.06335
[208] Test-mlogloss:0.06319
[209] Test-mlogloss:0.06305
[210] Test-mlogloss:0.06288
[211] Test-mlogloss:0.06275
[212] Test-mlogloss:0.06261
[213] Test-mlogloss:0.06246
[214] Test-mlogloss:0.06214
[215] Test-mlogloss:0.06202
[216] Test-mlogloss:0.06179
[217] Test-mlogloss:0.06168
[218] Test-mlogloss:0.06155
[219] Test-mlogloss:0.06139
```

```
[220] Test-mlogloss:0.06129
[221] Test-mlogloss:0.06111
[222] Test-mlogloss:0.06101
[223] Test-mlogloss:0.06087
[224] Test-mlogloss:0.06073
[225] Test-mlogloss:0.06051
[226] Test-mlogloss:0.06043
[227] Test-mlogloss:0.06022
[228] Test-mlogloss:0.06009
[229] Test-mlogloss:0.06000
[230] Test-mlogloss:0.05970
[231] Test-mlogloss:0.05965
[232] Test-mlogloss:0.05953
[233] Test-mlogloss:0.05930
[234] Test-mlogloss:0.05914
[235] Test-mlogloss:0.05906
[236] Test-mlogloss:0.05894
[237] Test-mlogloss:0.05881
[238] Test-mlogloss:0.05873
[239] Test-mlogloss:0.05857
[240] Test-mlogloss:0.05847
[241] Test-mlogloss:0.05836
[242] Test-mlogloss:0.05826
[243] Test-mlogloss:0.05805
[244] Test-mlogloss:0.05794
[245] Test-mlogloss:0.05776
[246] Test-mlogloss:0.05758
[247] Test-mlogloss:0.05744
[248] Test-mlogloss:0.05736
[249] Test-mlogloss:0.05729
[250] Test-mlogloss:0.05686
[251] Test-mlogloss:0.05671
[252] Test-mlogloss:0.05666
[253] Test-mlogloss:0.05649
[254] Test-mlogloss:0.05626
[255] Test-mlogloss:0.05600
[256] Test-mlogloss:0.05583
[257] Test-mlogloss:0.05580
[258] Test-mlogloss:0.05568
[259] Test-mlogloss:0.05549
[260] Test-mlogloss:0.05545
[261] Test-mlogloss:0.05538
[262] Test-mlogloss:0.05534
[263] Test-mlogloss:0.05527
[264] Test-mlogloss:0.05521
[265] Test-mlogloss:0.05511
[266] Test-mlogloss:0.05503
[267] Test-mlogloss:0.05496
[268] Test-mlogloss:0.05489
[269] Test-mlogloss:0.05485
[270] Test-mlogloss:0.05454
[271] Test-mlogloss:0.05434
[272] Test-mlogloss:0.05426
[273] Test-mlogloss:0.05421
[274] Test-mlogloss:0.05412
[275] Test-mlogloss:0.05394
[276] Test-mlogloss:0.05387
[277] Test-mlogloss:0.05387
[278] Test-mlogloss:0.05372
[279] Test-mlogloss:0.05367
[280] Test-mlogloss:0.05362
[281] Test-mlogloss:0.05357
[282] Test-mlogloss:0.05341
[283] Test-mlogloss:0.05333
[284] Test-mlogloss:0.05325
[285] Test-mlogloss:0.05320
[286] Test-mlogloss:0.05314
[287] Test-mlogloss:0.05308
[288] Test-mlogloss:0.05305
[289] Test-mlogloss:0.05303
[290] Test-mlogloss:0.05293
[291] Test-mlogloss:0.05291
[292] Test-mlogloss:0.05284
[293] Test-mlogloss:0.05282
[294] Test-mlogloss:0.05277
[295] Test-mlogloss:0.05273
[296] Test-mlogloss:0.05270
```

```
[297] Test-mlogloss:0.05267
[298] Test-mlogloss:0.05262
[299] Test-mlogloss:0.05259
[300] Test-mlogloss:0.05245
[301] Test-mlogloss:0.05232
[302] Test-mlogloss:0.05228
[303] Test-mlogloss:0.05220
[304] Test-mlogloss:0.05217
[305] Test-mlogloss:0.05199
[306] Test-mlogloss:0.05196
[307] Test-mlogloss:0.05187
[308] Test-mlogloss:0.05182
[309] Test-mlogloss:0.05173
[310] Test-mlogloss:0.05169
[311] Test-mlogloss:0.05164
[312] Test-mlogloss:0.05154
[313] Test-mlogloss:0.05150
[314] Test-mlogloss:0.05146
[315] Test-mlogloss:0.05142
[316] Test-mlogloss:0.05136
[317] Test-mlogloss:0.05131
[318] Test-mlogloss:0.05129
[319] Test-mlogloss:0.05121
[320] Test-mlogloss:0.05116
[321] Test-mlogloss:0.05109
[322] Test-mlogloss:0.05107
[323] Test-mlogloss:0.05099
[324] Test-mlogloss:0.05093
[325] Test-mlogloss:0.05088
[326] Test-mlogloss:0.05086
[327] Test-mlogloss:0.05079
[328] Test-mlogloss:0.05073
[329] Test-mlogloss:0.05072
[330] Test-mlogloss:0.05069
[331] Test-mlogloss:0.05064
[332] Test-mlogloss:0.05057
[333] Test-mlogloss:0.05053
[334] Test-mlogloss:0.05054
[335] Test-mlogloss:0.05045
[336] Test-mlogloss:0.05042
[337] Test-mlogloss:0.05039
[338] Test-mlogloss:0.05037
[339] Test-mlogloss:0.05033
[340] Test-mlogloss:0.05025
[341] Test-mlogloss:0.05022
[342] Test-mlogloss:0.05013
[343] Test-mlogloss:0.05009
[344] Test-mlogloss:0.05005
[345] Test-mlogloss:0.05004
[346] Test-mlogloss:0.04997
[347] Test-mlogloss:0.04996
[348] Test-mlogloss:0.04988
[349] Test-mlogloss:0.04983
[350] Test-mlogloss:0.04979
[351] Test-mlogloss:0.04975
[352] Test-mlogloss:0.04965
[353] Test-mlogloss:0.04961
[354] Test-mlogloss:0.04952
[355] Test-mlogloss:0.04949
[356] Test-mlogloss:0.04944
[357] Test-mlogloss:0.04940
[358] Test-mlogloss:0.04937
[359] Test-mlogloss:0.04935
[360] Test-mlogloss:0.04931
[361] Test-mlogloss:0.04930
[362] Test-mlogloss:0.04928
[363] Test-mlogloss:0.04926
[364] Test-mlogloss:0.04922
[365] Test-mlogloss:0.04920
[366] Test-mlogloss:0.04921
[367] Test-mlogloss:0.04919
[368] Test-mlogloss:0.04912
[369] Test-mlogloss:0.04909
[370] Test-mlogloss:0.04904
[371] Test-mlogloss:0.04900
[372] Test-mlogloss:0.04895
[373] Test-mlogloss:0.04891
```

```
[374] Test-mlogloss:0.04889
[375] Test-mlogloss:0.04888
[376] Test-mlogloss:0.04887
[377] Test-mlogloss:0.04884
[378] Test-mlogloss:0.04879
[379] Test-mlogloss:0.04878
[380] Test-mlogloss:0.04875
[381] Test-mlogloss:0.04872
[382] Test-mlogloss:0.04870
[383] Test-mlogloss:0.04865
[384] Test-mlogloss:0.04863
[385] Test-mlogloss:0.04859
[386] Test-mlogloss:0.04855
[387] Test-mlogloss:0.04850
[388] Test-mlogloss:0.04843
[389] Test-mlogloss:0.04839
[390] Test-mlogloss:0.04836
[391] Test-mlogloss:0.04831
[392] Test-mlogloss:0.04828
[393] Test-mlogloss:0.04826
[394] Test-mlogloss:0.04824
[395] Test-mlogloss:0.04818
[396] Test-mlogloss:0.04814
[397] Test-mlogloss:0.04812
[398] Test-mlogloss:0.04810
[399] Test-mlogloss:0.04810
[400] Test-mlogloss:0.04805
[401] Test-mlogloss:0.04804
[402] Test-mlogloss:0.04804
[403] Test-mlogloss:0.04799
[404] Test-mlogloss:0.04797
[405] Test-mlogloss:0.04795
[406] Test-mlogloss:0.04794
[407] Test-mlogloss:0.04787
[408] Test-mlogloss:0.04788
[409] Test-mlogloss:0.04786
[410] Test-mlogloss:0.04781
[411] Test-mlogloss:0.04780
[412] Test-mlogloss:0.04777
[413] Test-mlogloss:0.04775
[414] Test-mlogloss:0.04770
[415] Test-mlogloss:0.04768
[416] Test-mlogloss:0.04767
[417] Test-mlogloss:0.04763
[418] Test-mlogloss:0.04760
[419] Test-mlogloss:0.04758
[420] Test-mlogloss:0.04755
[421] Test-mlogloss:0.04749
[422] Test-mlogloss:0.04743
[423] Test-mlogloss:0.04742
[424] Test-mlogloss:0.04742
```

In [66]:

```
pr=model.predict(dtrain)
log_loss(y_train, pr)
```

Out[66]:

```
0.007250603937307788
```

In [67]:

```
pr=model.predict(dcv)
log_loss(y_val,pr)
```

Out[67]:

```
0.047419433995244564
```

Param tuning with xgb.cv and learning rate 0.08, model performance is not that good. May be if we increase either learning rate or n_estimators we get better performance.

## Tuning Parameters with XGBClassifier

#### Step-1 : set learning rate to 0.3 and find number of estimators (n_estimators)

In [68]:

```
# https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/

x_cfl=XGBClassifier(
            objective='multi:softprob',eval_metric='mlogloss', max_delta_step=5, tree_method='hist'
,
            n_estimators=1000, max_depth=9, min_child_weight=5, learning_rate=0.3, n_jobs=-1)

evaluation = [(train_df, y_train), ( val_df, y_val)]

x_cfl.fit(train_df,y_train,eval_set=evaluation, eval_metric="mlogloss", early_stopping_rounds=10)
print(x_cfl.evals_result())
```

```
[0] validation_0-mlogloss:1.05886 validation_1-mlogloss:1.06784
Multiple eval metrics have been passed: 'validation_1-mlogloss' will be used for early stopping.

Will train until validation_1-mlogloss hasn't improved in 10 rounds.
[1] validation_0-mlogloss:0.85813 validation_1-mlogloss:0.86952
[2] validation_0-mlogloss:0.69955 validation_1-mlogloss:0.71330
[3] validation_0-mlogloss:0.58964 validation_1-mlogloss:0.60531
[4] validation_0-mlogloss:0.49906 validation_1-mlogloss:0.51513
[5] validation_0-mlogloss:0.43085 validation_1-mlogloss:0.44819
[6] validation_0-mlogloss:0.37322 validation_1-mlogloss:0.39190
[7] validation_0-mlogloss:0.32637 validation_1-mlogloss:0.34625
[8] validation_0-mlogloss:0.28656 validation_1-mlogloss:0.30794
[9] validation_0-mlogloss:0.25230 validation_1-mlogloss:0.27498
[10] validation_0-mlogloss:0.22398 validation_1-mlogloss:0.24826
[11] validation_0-mlogloss:0.20047 validation_1-mlogloss:0.22623
[12] validation_0-mlogloss:0.18327 validation_1-mlogloss:0.20961
[13] validation_0-mlogloss:0.16855 validation_1-mlogloss:0.19554
[14] validation_0-mlogloss:0.15484 validation_1-mlogloss:0.18281
[15] validation_0-mlogloss:0.14257 validation_1-mlogloss:0.17072
[16] validation_0-mlogloss:0.13163 validation_1-mlogloss:0.16032
[17] validation_0-mlogloss:0.12306 validation_1-mlogloss:0.15185
[18] validation_0-mlogloss:0.11408 validation_1-mlogloss:0.14317
[19] validation_0-mlogloss:0.10690 validation_1-mlogloss:0.13696
[20] validation_0-mlogloss:0.10045 validation_1-mlogloss:0.13080
[21] validation_0-mlogloss:0.09418 validation_1-mlogloss:0.12519
[22] validation_0-mlogloss:0.08869 validation_1-mlogloss:0.12046
[23] validation_0-mlogloss:0.08310 validation_1-mlogloss:0.11572
[24] validation_0-mlogloss:0.07897 validation_1-mlogloss:0.11164
[25] validation_0-mlogloss:0.07537 validation_1-mlogloss:0.10858
[26] validation_0-mlogloss:0.07184 validation_1-mlogloss:0.10538
[27] validation_0-mlogloss:0.06691 validation_1-mlogloss:0.10095
[28] validation_0-mlogloss:0.06370 validation_1-mlogloss:0.09819
[29] validation_0-mlogloss:0.06064 validation_1-mlogloss:0.09544
[30] validation_0-mlogloss:0.05791 validation_1-mlogloss:0.09290
[31] validation_0-mlogloss:0.05541 validation_1-mlogloss:0.09085
[32] validation_0-mlogloss:0.05156 validation_1-mlogloss:0.08756
[33] validation_0-mlogloss:0.04834 validation_1-mlogloss:0.08425
[34] validation_0-mlogloss:0.04659 validation_1-mlogloss:0.08268
[35] validation_0-mlogloss:0.04547 validation_1-mlogloss:0.08149
[36] validation_0-mlogloss:0.04351 validation_1-mlogloss:0.08012
[37] validation_0-mlogloss:0.04215 validation_1-mlogloss:0.07884
[38] validation_0-mlogloss:0.04080 validation_1-mlogloss:0.07742
[39] validation_0-mlogloss:0.03947 validation_1-mlogloss:0.07647
[40] validation_0-mlogloss:0.03803 validation_1-mlogloss:0.07543
[41] validation_0-mlogloss:0.03732 validation_1-mlogloss:0.07486
[42] validation_0-mlogloss:0.03618 validation_1-mlogloss:0.07390
[43] validation_0-mlogloss:0.03436 validation_1-mlogloss:0.07206
[44] validation_0-mlogloss:0.03264 validation_1-mlogloss:0.07058
[45] validation_0-mlogloss:0.03145 validation_1-mlogloss:0.06950
[46] validation_0-mlogloss:0.03031 validation_1-mlogloss:0.06881
[47] validation_0-mlogloss:0.02915 validation_1-mlogloss:0.06795
[48] validation_0-mlogloss:0.02822 validation_1-mlogloss:0.06742
[49] validation_0-mlogloss:0.02692 validation_1-mlogloss:0.06622
[50] validation_0-mlogloss:0.02609 validation_1-mlogloss:0.06525
[51] validation_0-mlogloss:0.02521 validation_1-mlogloss:0.06488
```

```
[51] validation_0-mlogloss:0.02521 validation_1-mlogloss:0.06466
[52] validation_0-mlogloss:0.02461 validation_1-mlogloss:0.06424
[53] validation_0-mlogloss:0.02386 validation_1-mlogloss:0.06379
[54] validation_0-mlogloss:0.02273 validation_1-mlogloss:0.06285
[55] validation_0-mlogloss:0.02198 validation_1-mlogloss:0.06221
[56] validation_0-mlogloss:0.02125 validation_1-mlogloss:0.06142
[57] validation_0-mlogloss:0.02065 validation_1-mlogloss:0.06084
[58] validation_0-mlogloss:0.02018 validation_1-mlogloss:0.06047
[59] validation_0-mlogloss:0.01956 validation_1-mlogloss:0.05989
[60] validation_0-mlogloss:0.01890 validation_1-mlogloss:0.05911
[61] validation_0-mlogloss:0.01827 validation_1-mlogloss:0.05850
[62] validation_0-mlogloss:0.01779 validation_1-mlogloss:0.05824
[63] validation_0-mlogloss:0.01722 validation_1-mlogloss:0.05776
[64] validation_0-mlogloss:0.01676 validation_1-mlogloss:0.05753
[65] validation_0-mlogloss:0.01647 validation_1-mlogloss:0.05728
[66] validation_0-mlogloss:0.01579 validation_1-mlogloss:0.05637
[67] validation_0-mlogloss:0.01544 validation_1-mlogloss:0.05606
[68] validation_0-mlogloss:0.01510 validation_1-mlogloss:0.05567
[69] validation_0-mlogloss:0.01475 validation_1-mlogloss:0.05543
[70] validation_0-mlogloss:0.01446 validation_1-mlogloss:0.05522
[71] validation_0-mlogloss:0.01410 validation_1-mlogloss:0.05486
[72] validation_0-mlogloss:0.01371 validation_1-mlogloss:0.05454
[73] validation_0-mlogloss:0.01332 validation_1-mlogloss:0.05421
[74] validation_0-mlogloss:0.01295 validation_1-mlogloss:0.05389
[75] validation_0-mlogloss:0.01262 validation_1-mlogloss:0.05372
[76] validation_0-mlogloss:0.01223 validation_1-mlogloss:0.05326
[77] validation_0-mlogloss:0.01194 validation_1-mlogloss:0.05298
[78] validation_0-mlogloss:0.01168 validation_1-mlogloss:0.05272
[79] validation_0-mlogloss:0.01150 validation_1-mlogloss:0.05270
[80] validation_0-mlogloss:0.01121 validation_1-mlogloss:0.05239
[81] validation_0-mlogloss:0.01104 validation_1-mlogloss:0.05230
[82] validation_0-mlogloss:0.01085 validation_1-mlogloss:0.05222
[83] validation_0-mlogloss:0.01068 validation_1-mlogloss:0.05206
[84] validation_0-mlogloss:0.01042 validation_1-mlogloss:0.05177
[85] validation_0-mlogloss:0.01019 validation_1-mlogloss:0.05153
[86] validation_0-mlogloss:0.00999 validation_1-mlogloss:0.05145
[87] validation_0-mlogloss:0.00983 validation_1-mlogloss:0.05137
[88] validation_0-mlogloss:0.00962 validation_1-mlogloss:0.05116
[89] validation_0-mlogloss:0.00949 validation_1-mlogloss:0.05112
[90] validation_0-mlogloss:0.00935 validation_1-mlogloss:0.05101
[91] validation_0-mlogloss:0.00918 validation_1-mlogloss:0.05087
[92] validation_0-mlogloss:0.00903 validation_1-mlogloss:0.05075
[93] validation_0-mlogloss:0.00890 validation_1-mlogloss:0.05067
[94] validation_0-mlogloss:0.00868 validation_1-mlogloss:0.05058
[95] validation_0-mlogloss:0.00847 validation_1-mlogloss:0.05038
[96] validation_0-mlogloss:0.00834 validation_1-mlogloss:0.05028
[97] validation_0-mlogloss:0.00818 validation_1-mlogloss:0.05031
[98] validation_0-mlogloss:0.00808 validation_1-mlogloss:0.05031
[99] validation_0-mlogloss:0.00794 validation_1-mlogloss:0.05021
[100] validation_0-mlogloss:0.00779 validation_1-mlogloss:0.04994
[101] validation_0-mlogloss:0.00768 validation_1-mlogloss:0.04984
[102] validation_0-mlogloss:0.00760 validation_1-mlogloss:0.04975
[103] validation_0-mlogloss:0.00749 validation_1-mlogloss:0.04964
[104] validation_0-mlogloss:0.00739 validation_1-mlogloss:0.04955
[105] validation_0-mlogloss:0.00720 validation_1-mlogloss:0.04935
[106] validation_0-mlogloss:0.00705 validation_1-mlogloss:0.04926
[107] validation_0-mlogloss:0.00695 validation_1-mlogloss:0.04920
[108] validation_0-mlogloss:0.00684 validation_1-mlogloss:0.04921
[109] validation_0-mlogloss:0.00670 validation_1-mlogloss:0.04912
[110] validation_0-mlogloss:0.00656 validation_1-mlogloss:0.04887
[111] validation_0-mlogloss:0.00645 validation_1-mlogloss:0.04879
[112] validation_0-mlogloss:0.00634 validation_1-mlogloss:0.04861
[113] validation_0-mlogloss:0.00626 validation_1-mlogloss:0.04856
[114] validation_0-mlogloss:0.00619 validation_1-mlogloss:0.04849
[115] validation_0-mlogloss:0.00607 validation_1-mlogloss:0.04840
[116] validation_0-mlogloss:0.00600 validation_1-mlogloss:0.04835
[117] validation_0-mlogloss:0.00592 validation_1-mlogloss:0.04834
[118] validation_0-mlogloss:0.00583 validation_1-mlogloss:0.04822
[119] validation_0-mlogloss:0.00573 validation_1-mlogloss:0.04809
[120] validation_0-mlogloss:0.00565 validation_1-mlogloss:0.04799
[121] validation_0-mlogloss:0.00557 validation_1-mlogloss:0.04789
[122] validation_0-mlogloss:0.00548 validation_1-mlogloss:0.04791
[123] validation_0-mlogloss:0.00543 validation_1-mlogloss:0.04793
[124] validation_0-mlogloss:0.00535 validation_1-mlogloss:0.04789
[125] validation_0-mlogloss:0.00526 validation_1-mlogloss:0.04774
[126] validation_0-mlogloss:0.00518 validation_1-mlogloss:0.04756
[127] validation_0-mlogloss:0.00512 validation_1-mlogloss:0.04753
[128] validation_0-mlogloss:0.00508 validation_1-mlogloss:0.04748
```

[128] validation_0-mlogloss:0.00306 validation_1-mlogloss:0.04748
[129] validation_0-mlogloss:0.00502 validation_1-mlogloss:0.04741
[130] validation_0-mlogloss:0.00494 validation_1-mlogloss:0.04728
[131] validation_0-mlogloss:0.00489 validation_1-mlogloss:0.04718
[132] validation_0-mlogloss:0.00485 validation_1-mlogloss:0.04716
[133] validation_0-mlogloss:0.00478 validation_1-mlogloss:0.04723
[134] validation_0-mlogloss:0.00473 validation_1-mlogloss:0.04717
[135] validation_0-mlogloss:0.00469 validation_1-mlogloss:0.04715
[136] validation_0-mlogloss:0.00463 validation_1-mlogloss:0.04715
[137] validation_0-mlogloss:0.00459 validation_1-mlogloss:0.04714
[138] validation_0-mlogloss:0.00453 validation_1-mlogloss:0.04715
[139] validation_0-mlogloss:0.00448 validation_1-mlogloss:0.04715
[140] validation_0-mlogloss:0.00445 validation_1-mlogloss:0.04716
[141] validation_0-mlogloss:0.00442 validation_1-mlogloss:0.04713
[142] validation_0-mlogloss:0.00438 validation_1-mlogloss:0.04707
[143] validation_0-mlogloss:0.00432 validation_1-mlogloss:0.04698
[144] validation_0-mlogloss:0.00427 validation_1-mlogloss:0.04703
[145] validation_0-mlogloss:0.00422 validation_1-mlogloss:0.04696
[146] validation_0-mlogloss:0.00418 validation_1-mlogloss:0.04695
[147] validation_0-mlogloss:0.00414 validation_1-mlogloss:0.04690
[148] validation_0-mlogloss:0.00409 validation_1-mlogloss:0.04689
[149] validation_0-mlogloss:0.00404 validation_1-mlogloss:0.04679
[150] validation_0-mlogloss:0.00400 validation_1-mlogloss:0.04684
[151] validation_0-mlogloss:0.00395 validation_1-mlogloss:0.04681
[152] validation_0-mlogloss:0.00392 validation_1-mlogloss:0.04682
[153] validation_0-mlogloss:0.00388 validation_1-mlogloss:0.04684
[154] validation_0-mlogloss:0.00381 validation_1-mlogloss:0.04673
[155] validation_0-mlogloss:0.00378 validation_1-mlogloss:0.04663
[156] validation_0-mlogloss:0.00373 validation_1-mlogloss:0.04661
[157] validation_0-mlogloss:0.00371 validation_1-mlogloss:0.04661
[158] validation_0-mlogloss:0.00367 validation_1-mlogloss:0.04661
[159] validation_0-mlogloss:0.00363 validation_1-mlogloss:0.04666
[160] validation_0-mlogloss:0.00360 validation_1-mlogloss:0.04665
[161] validation_0-mlogloss:0.00357 validation_1-mlogloss:0.04660
[162] validation_0-mlogloss:0.00352 validation_1-mlogloss:0.04654
[163] validation_0-mlogloss:0.00350 validation_1-mlogloss:0.04652
[164] validation_0-mlogloss:0.00346 validation_1-mlogloss:0.04650
[165] validation_0-mlogloss:0.00342 validation_1-mlogloss:0.04647
[166] validation_0-mlogloss:0.00338 validation_1-mlogloss:0.04639
[167] validation_0-mlogloss:0.00336 validation_1-mlogloss:0.04640
[168] validation_0-mlogloss:0.00333 validation_1-mlogloss:0.04635
[169] validation_0-mlogloss:0.00330 validation_1-mlogloss:0.04637
[170] validation_0-mlogloss:0.00328 validation_1-mlogloss:0.04641
[171] validation_0-mlogloss:0.00325 validation_1-mlogloss:0.04635
[172] validation_0-mlogloss:0.00322 validation_1-mlogloss:0.04634
[173] validation_0-mlogloss:0.00320 validation_1-mlogloss:0.04640
[174] validation_0-mlogloss:0.00316 validation_1-mlogloss:0.04631
[175] validation_0-mlogloss:0.00314 validation_1-mlogloss:0.04630
[176] validation_0-mlogloss:0.00311 validation_1-mlogloss:0.04628
[177] validation_0-mlogloss:0.00308 validation_1-mlogloss:0.04627
[178] validation_0-mlogloss:0.00306 validation_1-mlogloss:0.04630
[179] validation_0-mlogloss:0.00303 validation_1-mlogloss:0.04637
[180] validation_0-mlogloss:0.00300 validation_1-mlogloss:0.04641
[181] validation_0-mlogloss:0.00299 validation_1-mlogloss:0.04636
[182] validation_0-mlogloss:0.00298 validation_1-mlogloss:0.04634
[183] validation_0-mlogloss:0.00296 validation_1-mlogloss:0.04637
[184] validation_0-mlogloss:0.00293 validation_1-mlogloss:0.04635
[185] validation_0-mlogloss:0.00291 validation_1-mlogloss:0.04641
[186] validation_0-mlogloss:0.00289 validation_1-mlogloss:0.04647
[187] validation_0-mlogloss:0.00287 validation_1-mlogloss:0.04646
Stopping. Best iteration:
[177] validation_0-mlogloss:0.00308 validation_1-mlogloss:0.04627

{'validation_0': {'mlogloss': [1.058861, 0.858127, 0.699552, 0.589643, 0.499059, 0.430855,
0.373224, 0.326373, 0.286564, 0.252297, 0.223977, 0.200468, 0.183266, 0.168545, 0.154839,
0.142574, 0.131634, 0.123064, 0.114076, 0.106903, 0.100449, 0.094178, 0.088693, 0.083103,
0.078967, 0.075366, 0.071836, 0.06691, 0.063704, 0.060639, 0.057914, 0.05541, 0.051561, 0.048339,
0.04659, 0.04547, 0.04351, 0.042151, 0.040797, 0.039468, 0.038028, 0.037315, 0.036184, 0.034364, 0
.032643, 0.031454, 0.030308, 0.029151, 0.028217, 0.026921, 0.026091, 0.02521, 0.024611, 0.023856,
0.022735, 0.021979, 0.021246, 0.020652, 0.020179, 0.019564, 0.018898, 0.018274, 0.017789,
0.017219, 0.016765, 0.016473, 0.015791, 0.015439, 0.015098, 0.014747, 0.014461, 0.014097,
0.013713, 0.01332, 0.012952, 0.012624, 0.012226, 0.011941, 0.011685, 0.011502, 0.011208, 0.011036,
0.010845, 0.01068, 0.010422, 0.010193, 0.009986, 0.009826, 0.009615, 0.009494, 0.00935, 0.009182,
0.009031, 0.008903, 0.008681, 0.008473, 0.008343, 0.008181, 0.008077, 0.007943, 0.007795, 0.00768,
0.007598, 0.007492, 0.007394, 0.007204, 0.00705, 0.006946, 0.006841, 0.006699, 0.00656, 0.006449,
0.006344, 0.006263, 0.006195, 0.006073, 0.006005, 0.005917, 0.005825, 0.005732, 0.005651,
0.005568, 0.005482, 0.005435, 0.005354, 0.005257, 0.005183, 0.005118, 0.005079, 0.005019, 0.00494,

```
0.004889, 0.004849, 0.004782, 0.004731, 0.004691, 0.004628, 0.004587, 0.00453, 0.004484, 0.004452,
0.004423, 0.004382, 0.004319, 0.00427, 0.004223, 0.004184, 0.004137, 0.004094, 0.004038, 0.003998,
0.00395, 0.003921, 0.003876, 0.003812, 0.003776, 0.00373, 0.003707, 0.003666, 0.003631, 0.003597,
0.003573, 0.003524, 0.003498, 0.003464, 0.003421, 0.003382, 0.003359, 0.003333, 0.003301, 0.00328,
0.003253, 0.003221, 0.003202, 0.003164, 0.003138, 0.003113, 0.003079, 0.003059, 0.003035,
0.003004, 0.002992, 0.002978, 0.002959, 0.002932, 0.002909, 0.00289]}, 'validation_1':
{'mlogloss': [1.067835, 0.869525, 0.713301, 0.605311, 0.515132, 0.44819, 0.391904, 0.346255, 0.307
942, 0.274977, 0.248261, 0.226225, 0.209614, 0.195542, 0.182813, 0.170724, 0.160321, 0.151845, 0.1
43166, 0.136958, 0.130803, 0.125189, 0.120462, 0.115717, 0.111645, 0.108576, 0.105381, 0.100949, 0
.098189, 0.095439, 0.092903, 0.090854, 0.087559, 0.084254, 0.082679, 0.081494, 0.080123, 0.078838,
0.077422, 0.076469, 0.075432, 0.074856, 0.073897, 0.072056, 0.070582, 0.069503, 0.068811,
0.067949, 0.067425, 0.066222, 0.065247, 0.064879, 0.06424, 0.063792, 0.062852, 0.062211, 0.061417,
0.060839, 0.060467, 0.059894, 0.059113, 0.0585, 0.058235, 0.057755, 0.05753, 0.057282, 0.056369, 0
.056061, 0.05567, 0.055433, 0.055221, 0.054859, 0.054538, 0.054208, 0.053891, 0.053724, 0.053263,
0.052977, 0.052724, 0.052697, 0.052387, 0.052298, 0.052218, 0.052057, 0.051771, 0.051535,
0.051452, 0.051375, 0.051159, 0.051115, 0.051012, 0.050875, 0.050751, 0.050668, 0.05058, 0.050378,
0.050282, 0.050307, 0.050313, 0.050209, 0.049938, 0.049836, 0.049754, 0.049642, 0.049551,
0.049354, 0.049258, 0.049196, 0.049207, 0.049124, 0.048874, 0.048793, 0.048606, 0.048563,
0.048489, 0.048397, 0.048347, 0.04834, 0.048216, 0.048089, 0.047987, 0.047886, 0.047906, 0.047932,
0.047888, 0.047736, 0.047558, 0.047532, 0.047483, 0.047412, 0.047281, 0.047184, 0.04716, 0.047229,
0.047173, 0.047153, 0.047147, 0.047137, 0.047151, 0.047147, 0.047156, 0.047134, 0.047067,
0.046985, 0.047034, 0.046958, 0.046946, 0.046902, 0.046892, 0.046793, 0.046839, 0.046806,
0.046816, 0.046835, 0.046729, 0.046626, 0.046611, 0.04661, 0.046607, 0.046657, 0.046649, 0.046599,
0.046539, 0.046516, 0.046501, 0.046469, 0.046394, 0.0464, 0.046347, 0.046375, 0.046412, 0.046349,
0.046344, 0.046401, 0.046306, 0.046297, 0.046279, 0.046265, 0.046301, 0.046366, 0.046405,
0.046359, 0.04634, 0.046373, 0.046353, 0.046409, 0.046474]}}
```

In [ ]:

```
```

**Step-2 : finding max_depth and min_child_weight**

In [11]:

```
from sklearn.model_selection import GridSearchCV

param_test1 = {
 'max_depth':range(7,14,2),
 'min_child_weight':range(4,7)}

random_cfl=RandomizedSearchCV(XGBClassifier(
                                        learning_rate =0.3, n_estimators=177,
                                        objective='multi:softprob',eval_metric='mlogloss', nthread=
,
                                        seed=27),param_distributions=param_test1,cv=3,verbose=10,n_
obs=-1)
random_cfl.fit(train_df, y_train)
random_cfl.best_params_,random_cfl.best_score_
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed: 25.6min
[Parallel(n_jobs=-1)]: Done   9 tasks      | elapsed: 56.4min
[Parallel(n_jobs=-1)]: Done  19 out of  30 | elapsed: 88.5min remaining: 51.2min
[Parallel(n_jobs=-1)]: Done  23 out of  30 | elapsed: 101.4min remaining: 30.9min
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed: 115.7min remaining: 12.9min
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed: 117.6min finished
```

Out[11]:

```
({'max_depth': 13, 'min_child_weight': 4}, 0.977763888888889)
```

In [13]:

```
x_cfl=XGBClassifier(
                objective='multi:softprob',eval_metric='mlogloss',
                max_delta_step=14, tree_method='hist',
                n_estimators=177, learning_rate =0.3,
                max_depth = 13, seed=27,nthread=4,
                min_child_weight = 4, n_jobs=-1)
```

```
x_cfl.fit(train_df,y_train,verbose=True)

predict_y = x_cfl.predict_proba(train_df)
print("The train log loss is:",log_loss(y_train, predict_y))
predict_y = x_cfl.predict_proba(val_df)
print("The cross validation log loss is:",log_loss(y_val, predict_y))
```

```
The train log loss is: 0.0014369631745504334
The cross validation log loss is: 0.0378758928825291
```

**Step-3 : finding gamma**

In [12]:

```
param_test1 = {'gamma':[i/10.0 for i in range(0,3)]}

random_cfl=RandomizedSearchCV(XGBClassifier(
                                        learning_rate =0.3, n_estimators=177,max_depth=13,
min_child_weight=4,
                                        objective='multi:softprob',eval_metric='mlogloss', nthread=
,
                                        seed=27),param_distributions=param_test1,cv=3,verbose=10,n_
obs=-1)
random_cfl.fit(train_df, y_train)
random_cfl.best_params_,random_cfl.best_score_
```

```
Fitting 3 folds for each of 3 candidates, totalling 9 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 out of   9 | elapsed: 36.9min remaining: 129.3min
[Parallel(n_jobs=-1)]: Done   3 out of   9 | elapsed: 37.1min remaining: 74.2min
[Parallel(n_jobs=-1)]: Done   4 out of   9 | elapsed: 38.1min remaining: 47.6min
[Parallel(n_jobs=-1)]: Done   5 out of   9 | elapsed: 38.2min remaining: 30.6min
[Parallel(n_jobs=-1)]: Done   6 out of   9 | elapsed: 38.3min remaining: 19.1min
[Parallel(n_jobs=-1)]: Done   7 out of   9 | elapsed: 38.8min remaining: 11.1min
[Parallel(n_jobs=-1)]: Done   9 out of   9 | elapsed: 43.3min remaining:    0.0s
[Parallel(n_jobs=-1)]: Done   9 out of   9 | elapsed: 43.3min finished
```

Out[12]:

```
({'gamma': 0.0}, 0.977763888888889)
```

In [19]:

```
x_cfl=XGBClassifier(
                objective='multi:softprob',eval_metric='mlogloss',
                max_delta_step=14, tree_method='hist',
                n_estimators=177, learning_rate =0.3,
                max_depth = 13, seed=27,nthread=4, gamma=0,
                min_child_weight = 4, n_jobs=-1)

x_cfl.fit(train_df,y_train,verbose=True)

predict_y = x_cfl.predict_proba(train_df)
print("The train log loss is:",log_loss(y_train, predict_y))
predict_y = x_cfl.predict_proba(val_df)
print("The cross validation log loss is:",log_loss(y_val, predict_y))
```

```
The train log loss is: 0.0014369631745504334
The cross validation log loss is: 0.0378758928825291
```

**Step-4 : finding subsample and colsample_bytree**

In [21]:

```
param_test1 = {
 'subsample':[i/100.0 for i in range(80,101,5)],
```

```
 'colsample_bytree':[i/100.0 for i in range(80,101,5)]}

random_cfl=RandomizedSearchCV(XGBClassifier(
                                        objective='multi:softprob',eval_metric='mlogloss', nthread=
,
                                        max_delta_step=14, tree_method='hist',seed=27, learning_rat
=0.3,
                                        n_estimators=177, max_depth=13, min_child_weight=4, gamma=0
                                        ),param_distributions=param_test1,cv=3,verbose=10,n_jobs=-1
random_cfl.fit(train_df, y_train)
random_cfl.best_params_,random_cfl.best_score_
```

```
Fitting 3 folds for each of 10 candidates, totalling 30 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed: 11.4min
[Parallel(n_jobs=-1)]: Done   9 tasks      | elapsed: 22.8min
[Parallel(n_jobs=-1)]: Done  19 out of  30 | elapsed: 34.3min remaining: 19.9min
[Parallel(n_jobs=-1)]: Done  23 out of  30 | elapsed: 35.0min remaining: 10.6min
[Parallel(n_jobs=-1)]: Done  27 out of  30 | elapsed: 41.9min remaining:  4.7min
[Parallel(n_jobs=-1)]: Done  30 out of  30 | elapsed: 42.0min finished
```

Out[21]:

```
({'colsample_bytree': 0.95, 'subsample': 1.0}, 0.9763472222222221)
```

In [22]:

```
x_cfl=XGBClassifier(
                objective='multi:softprob',eval_metric='mlogloss',
                max_delta_step=14, tree_method='hist',
                n_estimators=177, learning_rate =0.3,
                max_depth = 13, seed=27, nthread=4, gamma=0,
                min_child_weight = 4, n_jobs=-1, subsample=1, colsample_bytree=0.95)

x_cfl.fit(train_df,y_train,verbose=True)

predict_y = x_cfl.predict_proba(train_df)
print("The train log loss is:",log_loss(y_train, predict_y))
predict_y = x_cfl.predict_proba(val_df)
print("The cross validation log loss is:",log_loss(y_val, predict_y))
```

```
The train log loss is: 0.0014308520202687305
The cross validation log loss is: 0.037640506114396434
```

#### Step-5 : Tuning Regularization Parameters

In [23]:

```
for i in [1e-3, 1e-2, 0.1, 1, 10]:
    random_cfl=XGBClassifier(learning_rate=0.3, n_estimators=177,
                             max_delta_step=14, tree_method='hist', min_child_weight = 4,
                             objective='multi:softprob',eval_metric='mlogloss', nthread=4, max_dept
= 13,
                             seed=27,reg_alpha=i,n_jobs=-1,subsample=1, colsample_bytree=0.95)
    random_cfl.fit(train_df, y_train)
    predict_y = random_cfl.predict_proba(train_df)
    print("The train log loss is:",log_loss(y_train, predict_y))
    predict_y = random_cfl.predict_proba(val_df)
    print("The cross validation log loss is:",log_loss(y_val, predict_y))
```

```
The train log loss is: 0.0014489360005278623
The cross validation log loss is: 0.03860493085944829
The train log loss is: 0.001450179148062999
The cross validation log loss is: 0.0394539491604009
The train log loss is: 0.0014821685200999358
The cross validation log loss is: 0.03786734197049854
The train log loss is: 0.002982256365096046
The cross validation log loss is: 0.038207542646398224
The train log loss is: 0.021006336835214925
The cross validation log loss is: 0.05890475003377247
```

**Final training**

In [6]:

```
# full data
from xgboost.sklearn import XGBClassifier

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[6]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [27]:

```
random_cfl=XGBClassifier(learning_rate=0.3, n_estimators=177,
                            max_delta_step=14, tree_method='hist', min_child_weight = 4,
                            objective='multi:softprob',eval_metric='mlogloss', nthread=4, max_dept
= 13,
                            seed=27,n_jobs=-1,subsample=1, colsample_bytree=0.95,reg_alpha=0.1)
random_cfl.fit(train_df, y_train)

predict_y = random_cfl.predict_proba(train_df)
print("The train log loss is:",log_loss(y_train, predict_y))
predict_y = random_cfl.predict_proba(val_df)
print("The cross validation log loss is:",log_loss(y_val, predict_y))
```

```
The train log loss is: 0.000154881986323408
The cross validation log loss is: 0.000754514872950122
```

In [18]:

```
# saving the model as it's the best model yet

pickle.dump(random_cfl, open("xgb_model.pickle.dat", "wb"))
```

In [14]:

```
# test data predictions

list_tup=[]
for tup in [(x,y) for x in np.unique(test['crew']) for y in np.unique(test['seat'])]:
    list_tup.append(tup)

test_id_f=[]
pred=[]
for i in range(0,18):
    test_df,test_id=test_data_FE(i,list_tup)
    test_id_f.append(test_id)
    test_df=test_df.drop(featr,axis=1)
    y_test_pred=random_cfl.predict_proba(test_df)
    pred.append(y_test_pred)
    print('---------------------------------- loop : ',i+1,' ----------------------------------')
```

```
---------------------------------- loop :  1  ----------------------------------
---------------------------------- loop :  2  ----------------------------------
---------------------------------- loop :  3  ----------------------------------
---------------------------------- loop :  4  ----------------------------------
missing gsr
---------------------------------- loop :  5  ----------------------------------
---------------------------------- loop :  6  ----------------------------------
---------------------------------- loop :  7  ----------------------------------
---------------------------------- loop :  8  ----------------------------------
---------------------------------- loop :  9  ----------------------------------
failed "gsr_amp" extraction
```

```
failed  gsr_amp  extraction
---------------------------------- loop :  10  ----------------------------------
---------------------------------- loop :  11  ----------------------------------
---------------------------------- loop :  12  ----------------------------------
missing gsr
---------------------------------- loop :  13  ----------------------------------
---------------------------------- loop :  14  ----------------------------------
---------------------------------- loop :  15  ----------------------------------
---------------------------------- loop :  16  ----------------------------------
---------------------------------- loop :  17  ----------------------------------
---------------------------------- loop :  18  ----------------------------------
```

In [15]:

```
test_pr=[]
tst_id=[]
for i in pred:
    for j in i:
        test_pr.append(j)

for i in test_id_f:
    for j in i:
        tst_id.append(j)

sub=pd.DataFrame(test_pr,columns=['A','B','C','D'])
sub['id']=tst_id
sub=sub[['id','A','B','C','D']]
sub=sub.sort_values(by=['id'])
sub.reset_index(inplace = True)
sub=sub.drop('index',axis=1)
sub.head()
```

Out[15]:

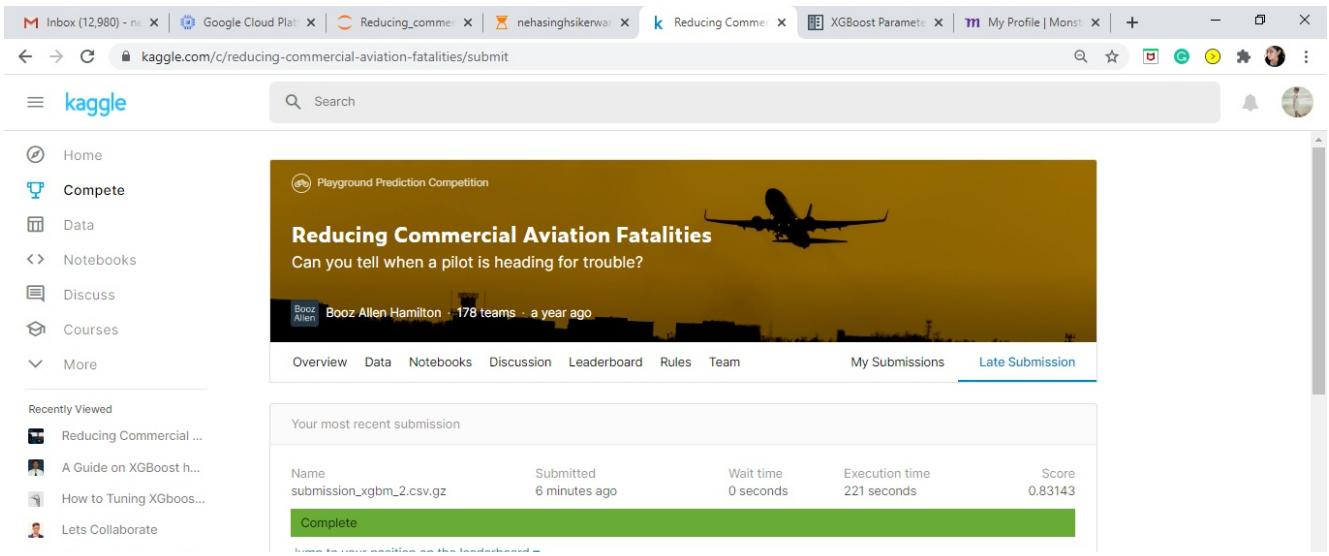
| | id | A | B | C | D |
|---|---|---|---|---|---|
| **0** | 0 | 0.994353 | 0.001942 | 0.003600 | 0.000106 |
| **1** | 1 | 0.955771 | 0.003552 | 0.037387 | 0.003290 |
| **2** | 2 | 0.991723 | 0.002599 | 0.005394 | 0.000284 |
| **3** | 3 | 0.879917 | 0.003502 | 0.114386 | 0.002195 |
| **4** | 4 | 0.990729 | 0.002589 | 0.006597 | 0.000084 |

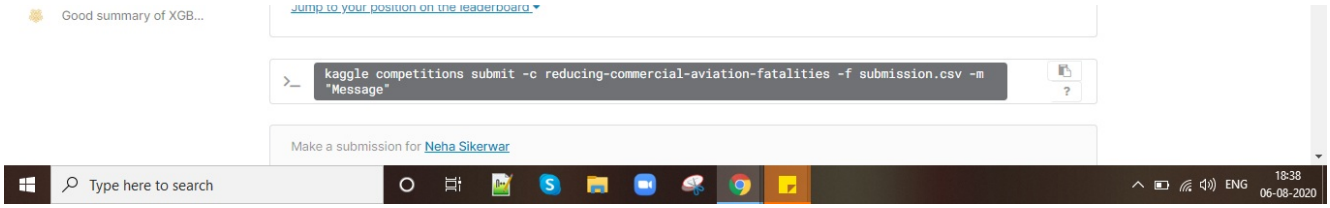
In [16]:

```
sub.to_csv('submission_xgbm_2.csv.gz',index=False, compression='gzip')
```

In [ ]:

```

```






In [ ]:

```
```

## xgboost part-3

In [18]:

```
df=new_train.sample(90000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df=df.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=
32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[18]:

```
((72000, 120), (18000, 120), 72000, 18000)
```

In [26]:

```
# https://www.kaggle.com/prashant111/a-guide-on-xgboost-hyperparameters-tuning

import xgboost as xgb
from hyperopt import STATUS_OK, Trials, fmin, hp, tpe

param={'max_depth': hp.quniform("max_depth", 3, 15, 1),
        'gamma': hp.uniform ('gamma', 1,6),
        'reg_alpha' : hp.quniform('reg_alpha', 0,50,1),
        'reg_lambda' : hp.uniform('reg_lambda', 0,1),
        'colsample_bytree' : hp.uniform('colsample_bytree', 0.5,1),
        'min_child_weight' : hp.quniform('min_child_weight', 0, 10, 1),
        'n_estimators': 1000,
        'seed': 0}


def tuning_para(param):
    clf=xgb.XGBClassifier(objective='multi:softprob',eval_metric='mlogloss',
                    n_estimators =param['n_estimators'], max_depth = int(param['max_depth']), gamma
= param['gamma'],
                    reg_alpha =
int(param['reg_alpha']),min_child_weight=int(param['min_child_weight']),
                    colsample_bytree=int(param['colsample_bytree']))

    evaluation = [(train_df, y_train), ( val_df, y_val)]

    clf.fit(train_df, y_train,
            eval_set=evaluation, eval_metric="mlogloss",
            early_stopping_rounds=10,verbose=False)

    predict = clf.predict_proba(val_df)
    #print(predict)
    loss=log_loss(y_val, predict)
    print ("val loss :", loss)
    return {'loss': loss, 'status': STATUS_OK }

hyperparams=fmin(fn=tuning_para, space=param, algo=tpe.suggest, max_evals=100, trials=Trials())
hyperparams
```

```
val loss :
```

0.37175550568432986
val loss :
0.30791516320944035
val loss :
0.3825919120676376
val loss :
0.49702040890159294
val loss :
0.3632421203510668
val loss :
0.3213784543173226
val loss :
0.4224146760702133
val loss :
0.501205605092329
val loss :
0.3171295280196209
val loss :
0.4077187454376577
val loss :
0.415252703225266
val loss :
0.40122764769597496
val loss :
0.3832681457074941
val loss :
0.3335823371960885
val loss :
0.3426075901129322
val loss :
0.398250886882319
val loss :
0.40712648276648383
val loss :
0.4165083803867973
val loss :
0.4286442336990084
val loss :
0.27115371571874775
val loss :
0.3106071055586249
val loss :
0.21790613248975366
val loss :
0.35931452698325866
val loss :
0.30977353122543394
val loss :
0.33850860265908217
val loss :
0.3101174802851294
val loss :
0.22082946512387652
val loss :
0.33855413178810784
val loss :
0.3144703877953612
val loss :
0.27898777697977334
val loss :
0.2592376386191916
val loss :
0.23542264506753255
val loss :
0.3419288645774068
val loss :
0.36093111531330374
val loss :
0.3491611592250734
val loss :
0.2777975230588151
val loss :
0.24668387324987595
val loss :
0.40826440302696493
val loss :
0.29026540681627616

```
val loss :
0.40423700241214183
val loss :
0.31159494564697443
val loss :
0.23892922992438254
val loss :
0.41679965063520813
val loss :
0.3620833871380568
val loss :
0.38279858633336455
val loss :
0.35089167088329043
val loss :
0.3768223258283817
val loss :
0.3571702275973124
val loss :
0.37606494280496716
val loss :
0.41341780017983143
val loss :
0.3769570470276004
val loss :
0.27111410774503664
val loss :
0.3706445112723353
val loss :
0.3322202618860594
val loss :
0.3887925154130854
val loss :
0.3394639660115483
val loss :
0.2932772375187342
val loss :
0.25581382636455563
val loss :
0.4945881235815533
val loss :
0.2950829122760915
val loss :
0.33099697104833387
val loss :
0.3820564157723372
val loss :
0.2870805396158426
val loss :
0.3540579606781248
val loss :
0.2531598417300742
val loss :
0.2701822494710731
val loss :
0.2637253615433551
val loss :
0.23502363863013387
val loss :
0.24155782495237427
val loss :
0.21696003308022047
val loss :
0.2856209209544419
val loss :
0.25192868255819
val loss :
0.3839983224082971
val loss :
0.29835510574911131
val loss :
0.38535993611950997
val loss :
0.221714086505928
val loss :
0.31998040507168 56
val loss :
```

```
0.2942184285411026
val loss :
0.2572288882183671
val loss :
0.31891627369599396
val loss :
0.3108019512934067
val loss :
0.36112607787513923
val loss :
0.3876387785968319
val loss :
0.34190051828835405
val loss :
0.35229511888979725
val loss :
0.330095410661856
val loss :
0.30411633881720707
val loss :
0.44133407402866415
val loss :
0.2410530544090798
val loss :
0.314860438202925
val loss :
0.28158459340506914
val loss :
0.3125914413801446
val loss :
0.379737354032954
val loss :
0.33109613314801634
val loss :
0.3910939463494255
val loss :
0.3746538607865546
val loss :
0.2885571114850043
val loss :
0.2999604372845562
val loss :
0.36226966794005905
100%|██████████| 100/100 [3:10:18<00:00, 114.19s/trial, best loss: 0.21696003308022047]
```

Out[26]:

```
{'colsample_bytree': 0.7824039477407645,
 'gamma': 1.1590357546482402,
 'max_depth': 15.0,
 'min_child_weight': 0.0,
 'reg_alpha': 0.0,
 'reg_lambda': 0.7093293854468173}
```

In [27]:

```
# full data

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[27]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [ ]:

```
x_cfl=xgb.XGBClassifier(
                objective='multi:softprob',eval_metric='mlogloss',
                max_delta_step=14, tree_method='hist',
                n_estimators=1000, colsample_bytree = int(0.7824039477407645)
```

```
                n_estimators=1000, colsample_bytree = int(0.7624059477407645),
                gamma= 1.1590357546482402, max_depth = int(15),
                min_child_weight =0,reg_alpha =0,
                reg_lambda= 0.7093293854468173, n_jobs=-1)

evaluation = [(train_df, y_train), ( val_df, y_val)]

x_cfl.fit(train_df, y_train,
            eval_set=evaluation, eval_metric="mlogloss",
            early_stopping_rounds=10)
```

```
[0] validation_0-mlogloss:1.23674 validation_1-mlogloss:1.23503
Multiple eval metrics have been passed: 'validation_1-mlogloss' will be used for early stopping.

Will train until validation_1-mlogloss hasn't improved in 10 rounds.
[1] validation_0-mlogloss:1.13586 validation_1-mlogloss:1.13450
[2] validation_0-mlogloss:1.05103 validation_1-mlogloss:1.05014
[3] validation_0-mlogloss:0.99795 validation_1-mlogloss:1.00222
[4] validation_0-mlogloss:0.96609 validation_1-mlogloss:0.96729
[5] validation_0-mlogloss:0.94279 validation_1-mlogloss:0.94109
[6] validation_0-mlogloss:0.92294 validation_1-mlogloss:0.92061
[7] validation_0-mlogloss:0.89452 validation_1-mlogloss:0.89377
[8] validation_0-mlogloss:0.88253 validation_1-mlogloss:0.88243
[9] validation_0-mlogloss:0.87441 validation_1-mlogloss:0.87480
[10] validation_0-mlogloss:0.86597 validation_1-mlogloss:0.86697
[11] validation_0-mlogloss:0.86014 validation_1-mlogloss:0.86133
[12] validation_0-mlogloss:0.83902 validation_1-mlogloss:0.83981
[13] validation_0-mlogloss:0.83354 validation_1-mlogloss:0.83409
[14] validation_0-mlogloss:0.83005 validation_1-mlogloss:0.83063
[15] validation_0-mlogloss:0.81978 validation_1-mlogloss:0.82021
[16] validation_0-mlogloss:0.81586 validation_1-mlogloss:0.81632
[17] validation_0-mlogloss:0.81327 validation_1-mlogloss:0.81377
[18] validation_0-mlogloss:0.81134 validation_1-mlogloss:0.81187
[19] validation_0-mlogloss:0.80905 validation_1-mlogloss:0.80963
[20] validation_0-mlogloss:0.80539 validation_1-mlogloss:0.80602
[21] validation_0-mlogloss:0.80177 validation_1-mlogloss:0.80239
[22] validation_0-mlogloss:0.79911 validation_1-mlogloss:0.79977
[23] validation_0-mlogloss:0.79722 validation_1-mlogloss:0.79788
[24] validation_0-mlogloss:0.79107 validation_1-mlogloss:0.79166
[25] validation_0-mlogloss:0.78716 validation_1-mlogloss:0.78783
[26] validation_0-mlogloss:0.78441 validation_1-mlogloss:0.78511
[27] validation_0-mlogloss:0.78228 validation_1-mlogloss:0.78312
[28] validation_0-mlogloss:0.75850 validation_1-mlogloss:0.75945
[29] validation_0-mlogloss:0.75603 validation_1-mlogloss:0.75703
[30] validation_0-mlogloss:0.75365 validation_1-mlogloss:0.75470
[31] validation_0-mlogloss:0.75115 validation_1-mlogloss:0.75230
[32] validation_0-mlogloss:0.74826 validation_1-mlogloss:0.74952
[33] validation_0-mlogloss:0.74374 validation_1-mlogloss:0.74509
[34] validation_0-mlogloss:0.74235 validation_1-mlogloss:0.74372
[35] validation_0-mlogloss:0.73641 validation_1-mlogloss:0.73790
[36] validation_0-mlogloss:0.73378 validation_1-mlogloss:0.73531
[37] validation_0-mlogloss:0.73128 validation_1-mlogloss:0.73288
[38] validation_0-mlogloss:0.72955 validation_1-mlogloss:0.73117
[39] validation_0-mlogloss:0.72814 validation_1-mlogloss:0.72981
[40] validation_0-mlogloss:0.72623 validation_1-mlogloss:0.72789
[41] validation_0-mlogloss:0.72446 validation_1-mlogloss:0.72616
[42] validation_0-mlogloss:0.72161 validation_1-mlogloss:0.72332
[43] validation_0-mlogloss:0.71951 validation_1-mlogloss:0.72128
[44] validation_0-mlogloss:0.71758 validation_1-mlogloss:0.71940
[45] validation_0-mlogloss:0.71531 validation_1-mlogloss:0.71714
[46] validation_0-mlogloss:0.71329 validation_1-mlogloss:0.71514
[47] validation_0-mlogloss:0.71014 validation_1-mlogloss:0.71201
[48] validation_0-mlogloss:0.70754 validation_1-mlogloss:0.70941
[49] validation_0-mlogloss:0.70611 validation_1-mlogloss:0.70803
[50] validation_0-mlogloss:0.70412 validation_1-mlogloss:0.70604
[51] validation_0-mlogloss:0.70077 validation_1-mlogloss:0.70269
[52] validation_0-mlogloss:0.69924 validation_1-mlogloss:0.70117
[53] validation_0-mlogloss:0.69738 validation_1-mlogloss:0.69933
[54] validation_0-mlogloss:0.69454 validation_1-mlogloss:0.69651
[55] validation_0-mlogloss:0.69302 validation_1-mlogloss:0.69501
[56] validation_0-mlogloss:0.69072 validation_1-mlogloss:0.69275
[57] validation_0-mlogloss:0.68967 validation_1-mlogloss:0.69172
[58] validation_0-mlogloss:0.66878 validation_1-mlogloss:0.67089
[59] validation_0-mlogloss:0.66677 validation_1-mlogloss:0.66885
[60] validation_0-mlogloss:0.66563 validation_1-mlogloss:0.66768
[61] validation_0-mlogloss:0.66425 validation_1-mlogloss:0.66635
[62] validation_0-mlogloss:0.66291 validation_1-mlogloss:0.66505
```

```
[63] validation_0-mlogloss:0.66131 validation_1-mlogloss:0.66348
[64] validation_0-mlogloss:0.66020 validation_1-mlogloss:0.66242
[65] validation_0-mlogloss:0.65875 validation_1-mlogloss:0.66099
[66] validation_0-mlogloss:0.65703 validation_1-mlogloss:0.65927
[67] validation_0-mlogloss:0.65184 validation_1-mlogloss:0.65407
[68] validation_0-mlogloss:0.64616 validation_1-mlogloss:0.64841
[69] validation_0-mlogloss:0.64404 validation_1-mlogloss:0.64629
[70] validation_0-mlogloss:0.64269 validation_1-mlogloss:0.64497
[71] validation_0-mlogloss:0.63987 validation_1-mlogloss:0.64216
[72] validation_0-mlogloss:0.63800 validation_1-mlogloss:0.64032
[73] validation_0-mlogloss:0.63611 validation_1-mlogloss:0.63849
[74] validation_0-mlogloss:0.62971 validation_1-mlogloss:0.63208
[75] validation_0-mlogloss:0.62846 validation_1-mlogloss:0.63089
[76] validation_0-mlogloss:0.62629 validation_1-mlogloss:0.62872
[77] validation_0-mlogloss:0.62420 validation_1-mlogloss:0.62667
[78] validation_0-mlogloss:0.62178 validation_1-mlogloss:0.62422
[79] validation_0-mlogloss:0.61973 validation_1-mlogloss:0.62221
[80] validation_0-mlogloss:0.61822 validation_1-mlogloss:0.62071
[81] validation_0-mlogloss:0.61658 validation_1-mlogloss:0.61907
[82] validation_0-mlogloss:0.61471 validation_1-mlogloss:0.61720
[83] validation_0-mlogloss:0.61007 validation_1-mlogloss:0.61262
[84] validation_0-mlogloss:0.60713 validation_1-mlogloss:0.60966
[85] validation_0-mlogloss:0.60555 validation_1-mlogloss:0.60812
[86] validation_0-mlogloss:0.60462 validation_1-mlogloss:0.60720
[87] validation_0-mlogloss:0.60416 validation_1-mlogloss:0.60675
[88] validation_0-mlogloss:0.60307 validation_1-mlogloss:0.60567
[89] validation_0-mlogloss:0.60230 validation_1-mlogloss:0.60494
[90] validation_0-mlogloss:0.60139 validation_1-mlogloss:0.60406
[91] validation_0-mlogloss:0.60049 validation_1-mlogloss:0.60314
[92] validation_0-mlogloss:0.59918 validation_1-mlogloss:0.60185
[93] validation_0-mlogloss:0.59671 validation_1-mlogloss:0.59936
[94] validation_0-mlogloss:0.59488 validation_1-mlogloss:0.59755
[95] validation_0-mlogloss:0.59222 validation_1-mlogloss:0.59489
[96] validation_0-mlogloss:0.59132 validation_1-mlogloss:0.59400
[97] validation_0-mlogloss:0.58995 validation_1-mlogloss:0.59265
[98] validation_0-mlogloss:0.58811 validation_1-mlogloss:0.59083
[99] validation_0-mlogloss:0.58634 validation_1-mlogloss:0.58908
[100] validation_0-mlogloss:0.58496 validation_1-mlogloss:0.58772
[101] validation_0-mlogloss:0.58378 validation_1-mlogloss:0.58656
[102] validation_0-mlogloss:0.58287 validation_1-mlogloss:0.58566
[103] validation_0-mlogloss:0.58155 validation_1-mlogloss:0.58435
[104] validation_0-mlogloss:0.57996 validation_1-mlogloss:0.58276
[105] validation_0-mlogloss:0.57913 validation_1-mlogloss:0.58196
[106] validation_0-mlogloss:0.57770 validation_1-mlogloss:0.58057
[107] validation_0-mlogloss:0.57305 validation_1-mlogloss:0.57593
[108] validation_0-mlogloss:0.57227 validation_1-mlogloss:0.57517
[109] validation_0-mlogloss:0.57110 validation_1-mlogloss:0.57403
[110] validation_0-mlogloss:0.56998 validation_1-mlogloss:0.57289
[111] validation_0-mlogloss:0.55210 validation_1-mlogloss:0.55500
[112] validation_0-mlogloss:0.55077 validation_1-mlogloss:0.55368
[113] validation_0-mlogloss:0.54573 validation_1-mlogloss:0.54857
[114] validation_0-mlogloss:0.54445 validation_1-mlogloss:0.54733
[115] validation_0-mlogloss:0.54406 validation_1-mlogloss:0.54696
[116] validation_0-mlogloss:0.54359 validation_1-mlogloss:0.54652
[117] validation_0-mlogloss:0.54190 validation_1-mlogloss:0.54487
[118] validation_0-mlogloss:0.54045 validation_1-mlogloss:0.54337
[119] validation_0-mlogloss:0.53950 validation_1-mlogloss:0.54245
[120] validation_0-mlogloss:0.53839 validation_1-mlogloss:0.54135
[121] validation_0-mlogloss:0.53800 validation_1-mlogloss:0.54096
[122] validation_0-mlogloss:0.52466 validation_1-mlogloss:0.52762
[123] validation_0-mlogloss:0.52383 validation_1-mlogloss:0.52678
[124] validation_0-mlogloss:0.52313 validation_1-mlogloss:0.52610
[125] validation_0-mlogloss:0.52245 validation_1-mlogloss:0.52543
[126] validation_0-mlogloss:0.52160 validation_1-mlogloss:0.52458
[127] validation_0-mlogloss:0.52080 validation_1-mlogloss:0.52380
[128] validation_0-mlogloss:0.51979 validation_1-mlogloss:0.52284
[129] validation_0-mlogloss:0.51894 validation_1-mlogloss:0.52197
[130] validation_0-mlogloss:0.51775 validation_1-mlogloss:0.52078
[131] validation_0-mlogloss:0.51592 validation_1-mlogloss:0.51894
[132] validation_0-mlogloss:0.51463 validation_1-mlogloss:0.51768
[133] validation_0-mlogloss:0.51343 validation_1-mlogloss:0.51646
[134] validation_0-mlogloss:0.51286 validation_1-mlogloss:0.51594
[135] validation_0-mlogloss:0.51188 validation_1-mlogloss:0.51498
[136] validation_0-mlogloss:0.51050 validation_1-mlogloss:0.51360
[137] validation_0-mlogloss:0.50954 validation_1-mlogloss:0.51264
[138] validation_0-mlogloss:0.50847 validation_1-mlogloss:0.51155
[139] validation_0-mlogloss:0.50666 validation_1-mlogloss:0.50975
```

```
[140] validation_0-mlogloss:0.50537 validation_1-mlogloss:0.50849
[141] validation_0-mlogloss:0.50415 validation_1-mlogloss:0.50726
[142] validation_0-mlogloss:0.50300 validation_1-mlogloss:0.50613
[143] validation_0-mlogloss:0.50198 validation_1-mlogloss:0.50513
[144] validation_0-mlogloss:0.50089 validation_1-mlogloss:0.50405
[145] validation_0-mlogloss:0.49985 validation_1-mlogloss:0.50304
[146] validation_0-mlogloss:0.49854 validation_1-mlogloss:0.50174
[147] validation_0-mlogloss:0.49669 validation_1-mlogloss:0.49993
[148] validation_0-mlogloss:0.49572 validation_1-mlogloss:0.49895
[149] validation_0-mlogloss:0.49494 validation_1-mlogloss:0.49818
[150] validation_0-mlogloss:0.49405 validation_1-mlogloss:0.49732
[151] validation_0-mlogloss:0.49248 validation_1-mlogloss:0.49575
[152] validation_0-mlogloss:0.49150 validation_1-mlogloss:0.49475
[153] validation_0-mlogloss:0.48867 validation_1-mlogloss:0.49195
[154] validation_0-mlogloss:0.48750 validation_1-mlogloss:0.49077
[155] validation_0-mlogloss:0.48659 validation_1-mlogloss:0.48989
[156] validation_0-mlogloss:0.48592 validation_1-mlogloss:0.48924
[157] validation_0-mlogloss:0.46537 validation_1-mlogloss:0.46868
[158] validation_0-mlogloss:0.46421 validation_1-mlogloss:0.46754
[159] validation_0-mlogloss:0.46349 validation_1-mlogloss:0.46682
[160] validation_0-mlogloss:0.46224 validation_1-mlogloss:0.46557
[161] validation_0-mlogloss:0.46179 validation_1-mlogloss:0.46515
[162] validation_0-mlogloss:0.46042 validation_1-mlogloss:0.46377
[163] validation_0-mlogloss:0.45967 validation_1-mlogloss:0.46304
[164] validation_0-mlogloss:0.45891 validation_1-mlogloss:0.46230
[165] validation_0-mlogloss:0.45797 validation_1-mlogloss:0.46136
[166] validation_0-mlogloss:0.45741 validation_1-mlogloss:0.46078
[167] validation_0-mlogloss:0.45674 validation_1-mlogloss:0.46014
[168] validation_0-mlogloss:0.45581 validation_1-mlogloss:0.45922
[169] validation_0-mlogloss:0.45517 validation_1-mlogloss:0.45858
[170] validation_0-mlogloss:0.45423 validation_1-mlogloss:0.45766
[171] validation_0-mlogloss:0.45382 validation_1-mlogloss:0.45727
[172] validation_0-mlogloss:0.45328 validation_1-mlogloss:0.45671
[173] validation_0-mlogloss:0.45269 validation_1-mlogloss:0.45613
[174] validation_0-mlogloss:0.45101 validation_1-mlogloss:0.45443
[175] validation_0-mlogloss:0.45028 validation_1-mlogloss:0.45373
[176] validation_0-mlogloss:0.44911 validation_1-mlogloss:0.45260
[177] validation_0-mlogloss:0.44666 validation_1-mlogloss:0.45016
[178] validation_0-mlogloss:0.44406 validation_1-mlogloss:0.44754
[179] validation_0-mlogloss:0.44316 validation_1-mlogloss:0.44662
[180] validation_0-mlogloss:0.44269 validation_1-mlogloss:0.44617
[181] validation_0-mlogloss:0.44207 validation_1-mlogloss:0.44557
[182] validation_0-mlogloss:0.44138 validation_1-mlogloss:0.44487
[183] validation_0-mlogloss:0.44019 validation_1-mlogloss:0.44368
[184] validation_0-mlogloss:0.43949 validation_1-mlogloss:0.44300
[185] validation_0-mlogloss:0.43875 validation_1-mlogloss:0.44225
[186] validation_0-mlogloss:0.43675 validation_1-mlogloss:0.44023
[187] validation_0-mlogloss:0.43561 validation_1-mlogloss:0.43909
[188] validation_0-mlogloss:0.43496 validation_1-mlogloss:0.43845
[189] validation_0-mlogloss:0.43426 validation_1-mlogloss:0.43776
[190] validation_0-mlogloss:0.43290 validation_1-mlogloss:0.43640
[191] validation_0-mlogloss:0.43129 validation_1-mlogloss:0.43478
[192] validation_0-mlogloss:0.43086 validation_1-mlogloss:0.43436
[193] validation_0-mlogloss:0.42986 validation_1-mlogloss:0.43339
[194] validation_0-mlogloss:0.42905 validation_1-mlogloss:0.43257
[195] validation_0-mlogloss:0.42789 validation_1-mlogloss:0.43140
[196] validation_0-mlogloss:0.42229 validation_1-mlogloss:0.42582
[197] validation_0-mlogloss:0.42177 validation_1-mlogloss:0.42534
[198] validation_0-mlogloss:0.42137 validation_1-mlogloss:0.42494
[199] validation_0-mlogloss:0.42059 validation_1-mlogloss:0.42418
[200] validation_0-mlogloss:0.41993 validation_1-mlogloss:0.42353
[201] validation_0-mlogloss:0.41932 validation_1-mlogloss:0.42290
[202] validation_0-mlogloss:0.41873 validation_1-mlogloss:0.42232
[203] validation_0-mlogloss:0.41818 validation_1-mlogloss:0.42177
[204] validation_0-mlogloss:0.41756 validation_1-mlogloss:0.42117
[205] validation_0-mlogloss:0.41713 validation_1-mlogloss:0.42075
[206] validation_0-mlogloss:0.41639 validation_1-mlogloss:0.42002
[207] validation_0-mlogloss:0.41608 validation_1-mlogloss:0.41974
[208] validation_0-mlogloss:0.41475 validation_1-mlogloss:0.41842
[209] validation_0-mlogloss:0.40856 validation_1-mlogloss:0.41223
[210] validation_0-mlogloss:0.40793 validation_1-mlogloss:0.41160
[211] validation_0-mlogloss:0.40753 validation_1-mlogloss:0.41121
[212] validation_0-mlogloss:0.40651 validation_1-mlogloss:0.41018
[213] validation_0-mlogloss:0.40595 validation_1-mlogloss:0.40963
[214] validation_0-mlogloss:0.40546 validation_1-mlogloss:0.40917
[215] validation_0-mlogloss:0.40491 validation_1-mlogloss:0.40863
[216] validation_0-mlogloss:0.40426 validation_1-mlogloss:0.40796
```

[217] validation_0-mlogloss:0.40377 validation_1-mlogloss:0.40748
[218] validation_0-mlogloss:0.40359 validation_1-mlogloss:0.40732
[219] validation_0-mlogloss:0.40315 validation_1-mlogloss:0.40686
[220] validation_0-mlogloss:0.40250 validation_1-mlogloss:0.40622
[221] validation_0-mlogloss:0.39755 validation_1-mlogloss:0.40129
[222] validation_0-mlogloss:0.39720 validation_1-mlogloss:0.40096
[223] validation_0-mlogloss:0.39691 validation_1-mlogloss:0.40066
[224] validation_0-mlogloss:0.39652 validation_1-mlogloss:0.40029
[225] validation_0-mlogloss:0.39567 validation_1-mlogloss:0.39942
[226] validation_0-mlogloss:0.39484 validation_1-mlogloss:0.39859
[227] validation_0-mlogloss:0.39435 validation_1-mlogloss:0.39811
[228] validation_0-mlogloss:0.39385 validation_1-mlogloss:0.39761
[229] validation_0-mlogloss:0.39256 validation_1-mlogloss:0.39632
[230] validation_0-mlogloss:0.39187 validation_1-mlogloss:0.39562
[231] validation_0-mlogloss:0.39142 validation_1-mlogloss:0.39516
[232] validation_0-mlogloss:0.39040 validation_1-mlogloss:0.39418
[233] validation_0-mlogloss:0.38651 validation_1-mlogloss:0.39032
[234] validation_0-mlogloss:0.38467 validation_1-mlogloss:0.38850
[235] validation_0-mlogloss:0.38418 validation_1-mlogloss:0.38802
[236] validation_0-mlogloss:0.38382 validation_1-mlogloss:0.38767
[237] validation_0-mlogloss:0.38336 validation_1-mlogloss:0.38722
[238] validation_0-mlogloss:0.38277 validation_1-mlogloss:0.38663
[239] validation_0-mlogloss:0.38251 validation_1-mlogloss:0.38637
[240] validation_0-mlogloss:0.38179 validation_1-mlogloss:0.38564
[241] validation_0-mlogloss:0.37723 validation_1-mlogloss:0.38103
[242] validation_0-mlogloss:0.37660 validation_1-mlogloss:0.38040
[243] validation_0-mlogloss:0.37616 validation_1-mlogloss:0.37998
[244] validation_0-mlogloss:0.37299 validation_1-mlogloss:0.37686
[245] validation_0-mlogloss:0.37241 validation_1-mlogloss:0.37627
[246] validation_0-mlogloss:0.37184 validation_1-mlogloss:0.37573
[247] validation_0-mlogloss:0.37038 validation_1-mlogloss:0.37428
[248] validation_0-mlogloss:0.36983 validation_1-mlogloss:0.37372
[249] validation_0-mlogloss:0.36926 validation_1-mlogloss:0.37316
[250] validation_0-mlogloss:0.36887 validation_1-mlogloss:0.37275
[251] validation_0-mlogloss:0.36854 validation_1-mlogloss:0.37244
[252] validation_0-mlogloss:0.36824 validation_1-mlogloss:0.37217
[253] validation_0-mlogloss:0.36748 validation_1-mlogloss:0.37141
[254] validation_0-mlogloss:0.36580 validation_1-mlogloss:0.36973
[255] validation_0-mlogloss:0.36531 validation_1-mlogloss:0.36923
[256] validation_0-mlogloss:0.35247 validation_1-mlogloss:0.35635
[257] validation_0-mlogloss:0.35208 validation_1-mlogloss:0.35595
[258] validation_0-mlogloss:0.35125 validation_1-mlogloss:0.35513
[259] validation_0-mlogloss:0.35066 validation_1-mlogloss:0.35452
[260] validation_0-mlogloss:0.34065 validation_1-mlogloss:0.34447
[261] validation_0-mlogloss:0.34014 validation_1-mlogloss:0.34397
[262] validation_0-mlogloss:0.33973 validation_1-mlogloss:0.34353
[263] validation_0-mlogloss:0.33948 validation_1-mlogloss:0.34329
[264] validation_0-mlogloss:0.33882 validation_1-mlogloss:0.34261
[265] validation_0-mlogloss:0.33853 validation_1-mlogloss:0.34232
[266] validation_0-mlogloss:0.33780 validation_1-mlogloss:0.34159
[267] validation_0-mlogloss:0.33720 validation_1-mlogloss:0.34101
[268] validation_0-mlogloss:0.33679 validation_1-mlogloss:0.34060
[269] validation_0-mlogloss:0.33611 validation_1-mlogloss:0.33992
[270] validation_0-mlogloss:0.33585 validation_1-mlogloss:0.33966
[271] validation_0-mlogloss:0.33553 validation_1-mlogloss:0.33934
[272] validation_0-mlogloss:0.33392 validation_1-mlogloss:0.33771
[273] validation_0-mlogloss:0.33353 validation_1-mlogloss:0.33734
[274] validation_0-mlogloss:0.33231 validation_1-mlogloss:0.33610
[275] validation_0-mlogloss:0.33209 validation_1-mlogloss:0.33589
[276] validation_0-mlogloss:0.33072 validation_1-mlogloss:0.33455
[277] validation_0-mlogloss:0.33038 validation_1-mlogloss:0.33422
[278] validation_0-mlogloss:0.32998 validation_1-mlogloss:0.33380
[279] validation_0-mlogloss:0.32942 validation_1-mlogloss:0.33327
[280] validation_0-mlogloss:0.32896 validation_1-mlogloss:0.33281
[281] validation_0-mlogloss:0.32478 validation_1-mlogloss:0.32858
[282] validation_0-mlogloss:0.32328 validation_1-mlogloss:0.32706
[283] validation_0-mlogloss:0.32287 validation_1-mlogloss:0.32664
[284] validation_0-mlogloss:0.32250 validation_1-mlogloss:0.32627
[285] validation_0-mlogloss:0.32192 validation_1-mlogloss:0.32569
[286] validation_0-mlogloss:0.32170 validation_1-mlogloss:0.32548
[287] validation_0-mlogloss:0.32154 validation_1-mlogloss:0.32532
[288] validation_0-mlogloss:0.32103 validation_1-mlogloss:0.32483
[289] validation_0-mlogloss:0.32071 validation_1-mlogloss:0.32451
[290] validation_0-mlogloss:0.32049 validation_1-mlogloss:0.32430
[291] validation_0-mlogloss:0.32008 validation_1-mlogloss:0.32390
[292] validation_0-mlogloss:0.31971 validation_1-mlogloss:0.32354
[293] validation_0-mlogloss:0.31944 validation_1-mlogloss:0.32328

[294] validation_0-mlogloss:0.31907 validation_1-mlogloss:0.32293
[295] validation_0-mlogloss:0.31886 validation_1-mlogloss:0.32272
[296] validation_0-mlogloss:0.31858 validation_1-mlogloss:0.32245
[297] validation_0-mlogloss:0.31807 validation_1-mlogloss:0.32195
[298] validation_0-mlogloss:0.31666 validation_1-mlogloss:0.32053
[299] validation_0-mlogloss:0.31586 validation_1-mlogloss:0.31970
[300] validation_0-mlogloss:0.31546 validation_1-mlogloss:0.31932
[301] validation_0-mlogloss:0.31527 validation_1-mlogloss:0.31912
[302] validation_0-mlogloss:0.31485 validation_1-mlogloss:0.31869
[303] validation_0-mlogloss:0.31472 validation_1-mlogloss:0.31856
[304] validation_0-mlogloss:0.31435 validation_1-mlogloss:0.31821
[305] validation_0-mlogloss:0.31392 validation_1-mlogloss:0.31777
[306] validation_0-mlogloss:0.31314 validation_1-mlogloss:0.31697
[307] validation_0-mlogloss:0.31284 validation_1-mlogloss:0.31669
[308] validation_0-mlogloss:0.31242 validation_1-mlogloss:0.31627
[309] validation_0-mlogloss:0.31176 validation_1-mlogloss:0.31561
[310] validation_0-mlogloss:0.31132 validation_1-mlogloss:0.31518
[311] validation_0-mlogloss:0.31090 validation_1-mlogloss:0.31475
[312] validation_0-mlogloss:0.31050 validation_1-mlogloss:0.31435
[313] validation_0-mlogloss:0.31002 validation_1-mlogloss:0.31388
[314] validation_0-mlogloss:0.30979 validation_1-mlogloss:0.31366
[315] validation_0-mlogloss:0.30946 validation_1-mlogloss:0.31335
[316] validation_0-mlogloss:0.30896 validation_1-mlogloss:0.31283
[317] validation_0-mlogloss:0.30880 validation_1-mlogloss:0.31268
[318] validation_0-mlogloss:0.30845 validation_1-mlogloss:0.31234
[319] validation_0-mlogloss:0.30817 validation_1-mlogloss:0.31207
[320] validation_0-mlogloss:0.30802 validation_1-mlogloss:0.31193
[321] validation_0-mlogloss:0.30796 validation_1-mlogloss:0.31186
[322] validation_0-mlogloss:0.30758 validation_1-mlogloss:0.31151
[323] validation_0-mlogloss:0.30716 validation_1-mlogloss:0.31111
[324] validation_0-mlogloss:0.30670 validation_1-mlogloss:0.31066
[325] validation_0-mlogloss:0.30352 validation_1-mlogloss:0.30745
[326] validation_0-mlogloss:0.30325 validation_1-mlogloss:0.30718
[327] validation_0-mlogloss:0.30302 validation_1-mlogloss:0.30695
[328] validation_0-mlogloss:0.30217 validation_1-mlogloss:0.30607
[329] validation_0-mlogloss:0.30183 validation_1-mlogloss:0.30571
[330] validation_0-mlogloss:0.30164 validation_1-mlogloss:0.30552
[331] validation_0-mlogloss:0.30110 validation_1-mlogloss:0.30500
[332] validation_0-mlogloss:0.29866 validation_1-mlogloss:0.30254
[333] validation_0-mlogloss:0.29823 validation_1-mlogloss:0.30212
[334] validation_0-mlogloss:0.29785 validation_1-mlogloss:0.30171
[335] validation_0-mlogloss:0.29763 validation_1-mlogloss:0.30149
[336] validation_0-mlogloss:0.29720 validation_1-mlogloss:0.30107
[337] validation_0-mlogloss:0.29688 validation_1-mlogloss:0.30076
[338] validation_0-mlogloss:0.29576 validation_1-mlogloss:0.29962
[339] validation_0-mlogloss:0.29213 validation_1-mlogloss:0.29599
[340] validation_0-mlogloss:0.29174 validation_1-mlogloss:0.29562
[341] validation_0-mlogloss:0.29114 validation_1-mlogloss:0.29501
[342] validation_0-mlogloss:0.29091 validation_1-mlogloss:0.29478
[343] validation_0-mlogloss:0.29047 validation_1-mlogloss:0.29434
[344] validation_0-mlogloss:0.29017 validation_1-mlogloss:0.29404
[345] validation_0-mlogloss:0.28990 validation_1-mlogloss:0.29377
[346] validation_0-mlogloss:0.28959 validation_1-mlogloss:0.29347
[347] validation_0-mlogloss:0.28911 validation_1-mlogloss:0.29298
[348] validation_0-mlogloss:0.28889 validation_1-mlogloss:0.29278
[349] validation_0-mlogloss:0.28840 validation_1-mlogloss:0.29230
[350] validation_0-mlogloss:0.28818 validation_1-mlogloss:0.29209
[351] validation_0-mlogloss:0.28703 validation_1-mlogloss:0.29096
[352] validation_0-mlogloss:0.28620 validation_1-mlogloss:0.29011
[353] validation_0-mlogloss:0.28597 validation_1-mlogloss:0.28989
[354] validation_0-mlogloss:0.28576 validation_1-mlogloss:0.28970
[355] validation_0-mlogloss:0.28545 validation_1-mlogloss:0.28939
[356] validation_0-mlogloss:0.28527 validation_1-mlogloss:0.28921
[357] validation_0-mlogloss:0.28499 validation_1-mlogloss:0.28893
[358] validation_0-mlogloss:0.28446 validation_1-mlogloss:0.28840
[359] validation_0-mlogloss:0.28415 validation_1-mlogloss:0.28809
[360] validation_0-mlogloss:0.28401 validation_1-mlogloss:0.28796
[361] validation_0-mlogloss:0.28379 validation_1-mlogloss:0.28774
[362] validation_0-mlogloss:0.28352 validation_1-mlogloss:0.28747
[363] validation_0-mlogloss:0.28083 validation_1-mlogloss:0.28480
[364] validation_0-mlogloss:0.27890 validation_1-mlogloss:0.28285
[365] validation_0-mlogloss:0.27730 validation_1-mlogloss:0.28122
[366] validation_0-mlogloss:0.27696 validation_1-mlogloss:0.28087
[367] validation_0-mlogloss:0.27668 validation_1-mlogloss:0.28061
[368] validation_0-mlogloss:0.27640 validation_1-mlogloss:0.28033
[369] validation_0-mlogloss:0.27621 validation_1-mlogloss:0.28014

[370] validation_0-mlogloss:0.27162 validation_1-mlogloss:0.27555
[371] validation_0-mlogloss:0.27139 validation_1-mlogloss:0.27531
[372] validation_0-mlogloss:0.27120 validation_1-mlogloss:0.27512
[373] validation_0-mlogloss:0.27102 validation_1-mlogloss:0.27494
[374] validation_0-mlogloss:0.27085 validation_1-mlogloss:0.27476
[375] validation_0-mlogloss:0.27057 validation_1-mlogloss:0.27450
[376] validation_0-mlogloss:0.27013 validation_1-mlogloss:0.27407
[377] validation_0-mlogloss:0.26913 validation_1-mlogloss:0.27305
[378] validation_0-mlogloss:0.26829 validation_1-mlogloss:0.27220
[379] validation_0-mlogloss:0.26816 validation_1-mlogloss:0.27208
[380] validation_0-mlogloss:0.26780 validation_1-mlogloss:0.27174
[381] validation_0-mlogloss:0.26742 validation_1-mlogloss:0.27134
[382] validation_0-mlogloss:0.26360 validation_1-mlogloss:0.26753
[383] validation_0-mlogloss:0.26288 validation_1-mlogloss:0.26682
[384] validation_0-mlogloss:0.26267 validation_1-mlogloss:0.26662
[385] validation_0-mlogloss:0.26232 validation_1-mlogloss:0.26628
[386] validation_0-mlogloss:0.26225 validation_1-mlogloss:0.26621
[387] validation_0-mlogloss:0.26186 validation_1-mlogloss:0.26582
[388] validation_0-mlogloss:0.26074 validation_1-mlogloss:0.26468
[389] validation_0-mlogloss:0.25986 validation_1-mlogloss:0.26379
[390] validation_0-mlogloss:0.25967 validation_1-mlogloss:0.26360
[391] validation_0-mlogloss:0.25952 validation_1-mlogloss:0.26344
[392] validation_0-mlogloss:0.25920 validation_1-mlogloss:0.26311
[393] validation_0-mlogloss:0.25915 validation_1-mlogloss:0.26305
[394] validation_0-mlogloss:0.25895 validation_1-mlogloss:0.26287
[395] validation_0-mlogloss:0.25875 validation_1-mlogloss:0.26266
[396] validation_0-mlogloss:0.25811 validation_1-mlogloss:0.26203
[397] validation_0-mlogloss:0.25785 validation_1-mlogloss:0.26177
[398] validation_0-mlogloss:0.25746 validation_1-mlogloss:0.26139
[399] validation_0-mlogloss:0.25716 validation_1-mlogloss:0.26110
[400] validation_0-mlogloss:0.25698 validation_1-mlogloss:0.26092
[401] validation_0-mlogloss:0.25641 validation_1-mlogloss:0.26034
[402] validation_0-mlogloss:0.25618 validation_1-mlogloss:0.26011
[403] validation_0-mlogloss:0.25585 validation_1-mlogloss:0.25977
[404] validation_0-mlogloss:0.25560 validation_1-mlogloss:0.25952
[405] validation_0-mlogloss:0.25530 validation_1-mlogloss:0.25924
[406] validation_0-mlogloss:0.25484 validation_1-mlogloss:0.25878
[407] validation_0-mlogloss:0.25455 validation_1-mlogloss:0.25848
[408] validation_0-mlogloss:0.25432 validation_1-mlogloss:0.25825
[409] validation_0-mlogloss:0.25408 validation_1-mlogloss:0.25803
[410] validation_0-mlogloss:0.25387 validation_1-mlogloss:0.25783
[411] validation_0-mlogloss:0.25366 validation_1-mlogloss:0.25762
[412] validation_0-mlogloss:0.25339 validation_1-mlogloss:0.25736
[413] validation_0-mlogloss:0.25317 validation_1-mlogloss:0.25714
[414] validation_0-mlogloss:0.25296 validation_1-mlogloss:0.25694
[415] validation_0-mlogloss:0.25278 validation_1-mlogloss:0.25676
[416] validation_0-mlogloss:0.25250 validation_1-mlogloss:0.25649
[417] validation_0-mlogloss:0.25228 validation_1-mlogloss:0.25628
[418] validation_0-mlogloss:0.25195 validation_1-mlogloss:0.25595
[419] validation_0-mlogloss:0.25180 validation_1-mlogloss:0.25581
[420] validation_0-mlogloss:0.25170 validation_1-mlogloss:0.25572
[421] validation_0-mlogloss:0.25146 validation_1-mlogloss:0.25548
[422] validation_0-mlogloss:0.25128 validation_1-mlogloss:0.25532
[423] validation_0-mlogloss:0.25116 validation_1-mlogloss:0.25520
[424] validation_0-mlogloss:0.25100 validation_1-mlogloss:0.25504
[425] validation_0-mlogloss:0.25080 validation_1-mlogloss:0.25484
[426] validation_0-mlogloss:0.25054 validation_1-mlogloss:0.25460
[427] validation_0-mlogloss:0.25029 validation_1-mlogloss:0.25435
[428] validation_0-mlogloss:0.24995 validation_1-mlogloss:0.25402
[429] validation_0-mlogloss:0.24988 validation_1-mlogloss:0.25395
[430] validation_0-mlogloss:0.24964 validation_1-mlogloss:0.25371
[431] validation_0-mlogloss:0.24943 validation_1-mlogloss:0.25350
[432] validation_0-mlogloss:0.24910 validation_1-mlogloss:0.25319
[433] validation_0-mlogloss:0.24887 validation_1-mlogloss:0.25295
[434] validation_0-mlogloss:0.24861 validation_1-mlogloss:0.25269
[435] validation_0-mlogloss:0.24642 validation_1-mlogloss:0.25050
[436] validation_0-mlogloss:0.24630 validation_1-mlogloss:0.25039
[437] validation_0-mlogloss:0.24608 validation_1-mlogloss:0.25017
[438] validation_0-mlogloss:0.24584 validation_1-mlogloss:0.24995
[439] validation_0-mlogloss:0.24563 validation_1-mlogloss:0.24974
[440] validation_0-mlogloss:0.24543 validation_1-mlogloss:0.24953
[441] validation_0-mlogloss:0.24505 validation_1-mlogloss:0.24915
[442] validation_0-mlogloss:0.24486 validation_1-mlogloss:0.24896
[443] validation_0-mlogloss:0.24458 validation_1-mlogloss:0.24869
[444] validation_0-mlogloss:0.24427 validation_1-mlogloss:0.24838
[445] validation_0-mlogloss:0.24242 validation_1-mlogloss:0.24654
[446] validation_0-mlogloss:0.24231 validation_1-mlogloss:0.24644
[447] validation_0-mlogloss:0.24216 validation_1-mlogloss:0.24628

[447] validation_0-mlogloss:0.24216 validation_1-mlogloss:0.24628
[448] validation_0-mlogloss:0.24199 validation_1-mlogloss:0.24614
[449] validation_0-mlogloss:0.24182 validation_1-mlogloss:0.24597
[450] validation_0-mlogloss:0.24164 validation_1-mlogloss:0.24580
[451] validation_0-mlogloss:0.24126 validation_1-mlogloss:0.24541
[452] validation_0-mlogloss:0.24110 validation_1-mlogloss:0.24526
[453] validation_0-mlogloss:0.24093 validation_1-mlogloss:0.24510
[454] validation_0-mlogloss:0.24073 validation_1-mlogloss:0.24490
[455] validation_0-mlogloss:0.24065 validation_1-mlogloss:0.24483
[456] validation_0-mlogloss:0.24040 validation_1-mlogloss:0.24457
[457] validation_0-mlogloss:0.24024 validation_1-mlogloss:0.24443
[458] validation_0-mlogloss:0.23876 validation_1-mlogloss:0.24293
[459] validation_0-mlogloss:0.23864 validation_1-mlogloss:0.24280
[460] validation_0-mlogloss:0.23840 validation_1-mlogloss:0.24258
[461] validation_0-mlogloss:0.23816 validation_1-mlogloss:0.24234
[462] validation_0-mlogloss:0.23803 validation_1-mlogloss:0.24220
[463] validation_0-mlogloss:0.23722 validation_1-mlogloss:0.24139
[464] validation_0-mlogloss:0.23705 validation_1-mlogloss:0.24122
[465] validation_0-mlogloss:0.23683 validation_1-mlogloss:0.24102
[466] validation_0-mlogloss:0.23663 validation_1-mlogloss:0.24083
[467] validation_0-mlogloss:0.23652 validation_1-mlogloss:0.24072
[468] validation_0-mlogloss:0.23642 validation_1-mlogloss:0.24063
[469] validation_0-mlogloss:0.23616 validation_1-mlogloss:0.24037
[470] validation_0-mlogloss:0.23594 validation_1-mlogloss:0.24016
[471] validation_0-mlogloss:0.23577 validation_1-mlogloss:0.23998
[472] validation_0-mlogloss:0.23548 validation_1-mlogloss:0.23970
[473] validation_0-mlogloss:0.23525 validation_1-mlogloss:0.23947
[474] validation_0-mlogloss:0.23511 validation_1-mlogloss:0.23934
[475] validation_0-mlogloss:0.23489 validation_1-mlogloss:0.23911
[476] validation_0-mlogloss:0.23478 validation_1-mlogloss:0.23900
[477] validation_0-mlogloss:0.23471 validation_1-mlogloss:0.23894
[478] validation_0-mlogloss:0.23470 validation_1-mlogloss:0.23893
[479] validation_0-mlogloss:0.23462 validation_1-mlogloss:0.23885
[480] validation_0-mlogloss:0.23436 validation_1-mlogloss:0.23860
[481] validation_0-mlogloss:0.23421 validation_1-mlogloss:0.23845
[482] validation_0-mlogloss:0.23409 validation_1-mlogloss:0.23832
[483] validation_0-mlogloss:0.23393 validation_1-mlogloss:0.23819
[484] validation_0-mlogloss:0.23375 validation_1-mlogloss:0.23800
[485] validation_0-mlogloss:0.23347 validation_1-mlogloss:0.23772
[486] validation_0-mlogloss:0.23329 validation_1-mlogloss:0.23755
[487] validation_0-mlogloss:0.23266 validation_1-mlogloss:0.23692
[488] validation_0-mlogloss:0.23242 validation_1-mlogloss:0.23668
[489] validation_0-mlogloss:0.23227 validation_1-mlogloss:0.23653
[490] validation_0-mlogloss:0.23200 validation_1-mlogloss:0.23627
[491] validation_0-mlogloss:0.23189 validation_1-mlogloss:0.23617
[492] validation_0-mlogloss:0.23160 validation_1-mlogloss:0.23586
[493] validation_0-mlogloss:0.23128 validation_1-mlogloss:0.23554
[494] validation_0-mlogloss:0.23114 validation_1-mlogloss:0.23538
[495] validation_0-mlogloss:0.23074 validation_1-mlogloss:0.23500
[496] validation_0-mlogloss:0.23054 validation_1-mlogloss:0.23480
[497] validation_0-mlogloss:0.23030 validation_1-mlogloss:0.23456
[498] validation_0-mlogloss:0.23012 validation_1-mlogloss:0.23438
[499] validation_0-mlogloss:0.22996 validation_1-mlogloss:0.23422
[500] validation_0-mlogloss:0.22979 validation_1-mlogloss:0.23405
[501] validation_0-mlogloss:0.22956 validation_1-mlogloss:0.23383
[502] validation_0-mlogloss:0.22937 validation_1-mlogloss:0.23364
[503] validation_0-mlogloss:0.22919 validation_1-mlogloss:0.23345
[504] validation_0-mlogloss:0.22897 validation_1-mlogloss:0.23323
[505] validation_0-mlogloss:0.22790 validation_1-mlogloss:0.23216
[506] validation_0-mlogloss:0.22708 validation_1-mlogloss:0.23131
[507] validation_0-mlogloss:0.22693 validation_1-mlogloss:0.23116
[508] validation_0-mlogloss:0.22615 validation_1-mlogloss:0.23036
[509] validation_0-mlogloss:0.22598 validation_1-mlogloss:0.23020
[510] validation_0-mlogloss:0.22279 validation_1-mlogloss:0.22702
[511] validation_0-mlogloss:0.22274 validation_1-mlogloss:0.22695
[512] validation_0-mlogloss:0.22257 validation_1-mlogloss:0.22680
[513] validation_0-mlogloss:0.22242 validation_1-mlogloss:0.22664
[514] validation_0-mlogloss:0.22220 validation_1-mlogloss:0.22642
[515] validation_0-mlogloss:0.22214 validation_1-mlogloss:0.22636
[516] validation_0-mlogloss:0.22202 validation_1-mlogloss:0.22624
[517] validation_0-mlogloss:0.22179 validation_1-mlogloss:0.22600
[518] validation_0-mlogloss:0.22165 validation_1-mlogloss:0.22588
[519] validation_0-mlogloss:0.22142 validation_1-mlogloss:0.22566
[520] validation_0-mlogloss:0.22021 validation_1-mlogloss:0.22446
[521] validation_0-mlogloss:0.22000 validation_1-mlogloss:0.22424
[522] validation_0-mlogloss:0.21977 validation_1-mlogloss:0.22402
[523] validation_0-mlogloss:0.21968 validation_1-mlogloss:0.22393
[524] validation_0-mlogloss:0.21933 validation_1-mlogloss:0.22357

[524] validation_0-mlogloss:0.21933 validation_1-mlogloss:0.22357
[525] validation_0-mlogloss:0.21919 validation_1-mlogloss:0.22344
[526] validation_0-mlogloss:0.21893 validation_1-mlogloss:0.22317
[527] validation_0-mlogloss:0.21873 validation_1-mlogloss:0.22298
[528] validation_0-mlogloss:0.21862 validation_1-mlogloss:0.22287
[529] validation_0-mlogloss:0.21848 validation_1-mlogloss:0.22273
[530] validation_0-mlogloss:0.21834 validation_1-mlogloss:0.22258
[531] validation_0-mlogloss:0.21818 validation_1-mlogloss:0.22243
[532] validation_0-mlogloss:0.21803 validation_1-mlogloss:0.22229
[533] validation_0-mlogloss:0.21794 validation_1-mlogloss:0.22221
[534] validation_0-mlogloss:0.21779 validation_1-mlogloss:0.22205
[535] validation_0-mlogloss:0.21772 validation_1-mlogloss:0.22198
[536] validation_0-mlogloss:0.21693 validation_1-mlogloss:0.22119
[537] validation_0-mlogloss:0.21682 validation_1-mlogloss:0.22109
[538] validation_0-mlogloss:0.21662 validation_1-mlogloss:0.22089
[539] validation_0-mlogloss:0.21650 validation_1-mlogloss:0.22077
[540] validation_0-mlogloss:0.21639 validation_1-mlogloss:0.22065
[541] validation_0-mlogloss:0.21625 validation_1-mlogloss:0.22052
[542] validation_0-mlogloss:0.21615 validation_1-mlogloss:0.22043
[543] validation_0-mlogloss:0.21602 validation_1-mlogloss:0.22032
[544] validation_0-mlogloss:0.21591 validation_1-mlogloss:0.22020
[545] validation_0-mlogloss:0.21577 validation_1-mlogloss:0.22006
[546] validation_0-mlogloss:0.21539 validation_1-mlogloss:0.21968
[547] validation_0-mlogloss:0.21512 validation_1-mlogloss:0.21941
[548] validation_0-mlogloss:0.21432 validation_1-mlogloss:0.21859
[549] validation_0-mlogloss:0.21418 validation_1-mlogloss:0.21845
[550] validation_0-mlogloss:0.21211 validation_1-mlogloss:0.21640
[551] validation_0-mlogloss:0.21201 validation_1-mlogloss:0.21631
[552] validation_0-mlogloss:0.21188 validation_1-mlogloss:0.21618
[553] validation_0-mlogloss:0.21174 validation_1-mlogloss:0.21605
[554] validation_0-mlogloss:0.21160 validation_1-mlogloss:0.21590
[555] validation_0-mlogloss:0.21036 validation_1-mlogloss:0.21469
[556] validation_0-mlogloss:0.21027 validation_1-mlogloss:0.21460
[557] validation_0-mlogloss:0.21009 validation_1-mlogloss:0.21442
[558] validation_0-mlogloss:0.20992 validation_1-mlogloss:0.21425
[559] validation_0-mlogloss:0.20978 validation_1-mlogloss:0.21411
[560] validation_0-mlogloss:0.20962 validation_1-mlogloss:0.21395
[561] validation_0-mlogloss:0.20951 validation_1-mlogloss:0.21383
[562] validation_0-mlogloss:0.20943 validation_1-mlogloss:0.21376
[563] validation_0-mlogloss:0.20925 validation_1-mlogloss:0.21358
[564] validation_0-mlogloss:0.20915 validation_1-mlogloss:0.21348
[565] validation_0-mlogloss:0.20884 validation_1-mlogloss:0.21317
[566] validation_0-mlogloss:0.20876 validation_1-mlogloss:0.21309
[567] validation_0-mlogloss:0.20870 validation_1-mlogloss:0.21303
[568] validation_0-mlogloss:0.20851 validation_1-mlogloss:0.21284
[569] validation_0-mlogloss:0.20841 validation_1-mlogloss:0.21275
[570] validation_0-mlogloss:0.20817 validation_1-mlogloss:0.21250
[571] validation_0-mlogloss:0.20798 validation_1-mlogloss:0.21231
[572] validation_0-mlogloss:0.20780 validation_1-mlogloss:0.21215
[573] validation_0-mlogloss:0.19849 validation_1-mlogloss:0.20275
[574] validation_0-mlogloss:0.19837 validation_1-mlogloss:0.20265
[575] validation_0-mlogloss:0.19819 validation_1-mlogloss:0.20247
[576] validation_0-mlogloss:0.19810 validation_1-mlogloss:0.20238
[577] validation_0-mlogloss:0.19268 validation_1-mlogloss:0.19693
[578] validation_0-mlogloss:0.19257 validation_1-mlogloss:0.19682
[579] validation_0-mlogloss:0.18638 validation_1-mlogloss:0.19054
[580] validation_0-mlogloss:0.18623 validation_1-mlogloss:0.19040
[581] validation_0-mlogloss:0.18612 validation_1-mlogloss:0.19030
[582] validation_0-mlogloss:0.18578 validation_1-mlogloss:0.18995
[583] validation_0-mlogloss:0.18553 validation_1-mlogloss:0.18969
[584] validation_0-mlogloss:0.18547 validation_1-mlogloss:0.18963
[585] validation_0-mlogloss:0.18527 validation_1-mlogloss:0.18942
[586] validation_0-mlogloss:0.18515 validation_1-mlogloss:0.18929
[587] validation_0-mlogloss:0.18508 validation_1-mlogloss:0.18923
[588] validation_0-mlogloss:0.18489 validation_1-mlogloss:0.18905
[589] validation_0-mlogloss:0.18474 validation_1-mlogloss:0.18889
[590] validation_0-mlogloss:0.18461 validation_1-mlogloss:0.18877
[591] validation_0-mlogloss:0.18435 validation_1-mlogloss:0.18851
[592] validation_0-mlogloss:0.18333 validation_1-mlogloss:0.18749
[593] validation_0-mlogloss:0.18320 validation_1-mlogloss:0.18738
[594] validation_0-mlogloss:0.18314 validation_1-mlogloss:0.18731
[595] validation_0-mlogloss:0.18305 validation_1-mlogloss:0.18723
[596] validation_0-mlogloss:0.18291 validation_1-mlogloss:0.18709
[597] validation_0-mlogloss:0.18276 validation_1-mlogloss:0.18694
[598] validation_0-mlogloss:0.18266 validation_1-mlogloss:0.18684
[599] validation_0-mlogloss:0.18235 validation_1-mlogloss:0.18653
[600] validation_0-mlogloss:0.18227 validation_1-mlogloss:0.18647
[601] validation_0-mlogloss:0.18220 validation_1-mlogloss:0.18640

[601] validation_0-mlogloss:0.18220 validation_1-mlogloss:0.18640
[602] validation_0-mlogloss:0.18201 validation_1-mlogloss:0.18622
[603] validation_0-mlogloss:0.18193 validation_1-mlogloss:0.18613
[604] validation_0-mlogloss:0.18182 validation_1-mlogloss:0.18603
[605] validation_0-mlogloss:0.18163 validation_1-mlogloss:0.18584
[606] validation_0-mlogloss:0.18152 validation_1-mlogloss:0.18574
[607] validation_0-mlogloss:0.18098 validation_1-mlogloss:0.18519
[608] validation_0-mlogloss:0.18083 validation_1-mlogloss:0.18504
[609] validation_0-mlogloss:0.18075 validation_1-mlogloss:0.18496
[610] validation_0-mlogloss:0.17987 validation_1-mlogloss:0.18407
[611] validation_0-mlogloss:0.17981 validation_1-mlogloss:0.18401
[612] validation_0-mlogloss:0.17905 validation_1-mlogloss:0.18325
[613] validation_0-mlogloss:0.17894 validation_1-mlogloss:0.18314
[614] validation_0-mlogloss:0.17879 validation_1-mlogloss:0.18300
[615] validation_0-mlogloss:0.17874 validation_1-mlogloss:0.18295
[616] validation_0-mlogloss:0.17862 validation_1-mlogloss:0.18284
[617] validation_0-mlogloss:0.17846 validation_1-mlogloss:0.18269
[618] validation_0-mlogloss:0.17833 validation_1-mlogloss:0.18256
[619] validation_0-mlogloss:0.17823 validation_1-mlogloss:0.18246
[620] validation_0-mlogloss:0.17818 validation_1-mlogloss:0.18242
[621] validation_0-mlogloss:0.17806 validation_1-mlogloss:0.18229
[622] validation_0-mlogloss:0.17797 validation_1-mlogloss:0.18221
[623] validation_0-mlogloss:0.17748 validation_1-mlogloss:0.18171
[624] validation_0-mlogloss:0.17733 validation_1-mlogloss:0.18156
[625] validation_0-mlogloss:0.17727 validation_1-mlogloss:0.18150
[626] validation_0-mlogloss:0.17705 validation_1-mlogloss:0.18127
[627] validation_0-mlogloss:0.17695 validation_1-mlogloss:0.18117
[628] validation_0-mlogloss:0.17689 validation_1-mlogloss:0.18111
[629] validation_0-mlogloss:0.17674 validation_1-mlogloss:0.18096
[630] validation_0-mlogloss:0.17649 validation_1-mlogloss:0.18072
[631] validation_0-mlogloss:0.17639 validation_1-mlogloss:0.18062
[632] validation_0-mlogloss:0.17632 validation_1-mlogloss:0.18055
[633] validation_0-mlogloss:0.17622 validation_1-mlogloss:0.18045
[634] validation_0-mlogloss:0.17605 validation_1-mlogloss:0.18028
[635] validation_0-mlogloss:0.17256 validation_1-mlogloss:0.17675
[636] validation_0-mlogloss:0.17248 validation_1-mlogloss:0.17668
[637] validation_0-mlogloss:0.17240 validation_1-mlogloss:0.17660
[638] validation_0-mlogloss:0.17156 validation_1-mlogloss:0.17577
[639] validation_0-mlogloss:0.17146 validation_1-mlogloss:0.17567
[640] validation_0-mlogloss:0.17103 validation_1-mlogloss:0.17524
[641] validation_0-mlogloss:0.17097 validation_1-mlogloss:0.17517
[642] validation_0-mlogloss:0.16665 validation_1-mlogloss:0.17080
[643] validation_0-mlogloss:0.16656 validation_1-mlogloss:0.17071
[644] validation_0-mlogloss:0.16646 validation_1-mlogloss:0.17060
[645] validation_0-mlogloss:0.16576 validation_1-mlogloss:0.16992
[646] validation_0-mlogloss:0.16566 validation_1-mlogloss:0.16982
[647] validation_0-mlogloss:0.16554 validation_1-mlogloss:0.16969
[648] validation_0-mlogloss:0.16532 validation_1-mlogloss:0.16948
[649] validation_0-mlogloss:0.16522 validation_1-mlogloss:0.16937
[650] validation_0-mlogloss:0.16514 validation_1-mlogloss:0.16930
[651] validation_0-mlogloss:0.16497 validation_1-mlogloss:0.16912
[652] validation_0-mlogloss:0.16488 validation_1-mlogloss:0.16904
[653] validation_0-mlogloss:0.16477 validation_1-mlogloss:0.16892
[654] validation_0-mlogloss:0.16469 validation_1-mlogloss:0.16885
[655] validation_0-mlogloss:0.16459 validation_1-mlogloss:0.16876
[656] validation_0-mlogloss:0.16452 validation_1-mlogloss:0.16869
[657] validation_0-mlogloss:0.16447 validation_1-mlogloss:0.16863
[658] validation_0-mlogloss:0.16373 validation_1-mlogloss:0.16789
[659] validation_0-mlogloss:0.16360 validation_1-mlogloss:0.16777
[660] validation_0-mlogloss:0.16355 validation_1-mlogloss:0.16770
[661] validation_0-mlogloss:0.16336 validation_1-mlogloss:0.16752
[662] validation_0-mlogloss:0.16321 validation_1-mlogloss:0.16736
[663] validation_0-mlogloss:0.16309 validation_1-mlogloss:0.16724
[664] validation_0-mlogloss:0.16300 validation_1-mlogloss:0.16716
[665] validation_0-mlogloss:0.16287 validation_1-mlogloss:0.16703
[666] validation_0-mlogloss:0.16281 validation_1-mlogloss:0.16696
[667] validation_0-mlogloss:0.15904 validation_1-mlogloss:0.16313
[668] validation_0-mlogloss:0.15840 validation_1-mlogloss:0.16248
[669] validation_0-mlogloss:0.15831 validation_1-mlogloss:0.16240
[670] validation_0-mlogloss:0.15822 validation_1-mlogloss:0.16231
[671] validation_0-mlogloss:0.15815 validation_1-mlogloss:0.16224
[672] validation_0-mlogloss:0.15806 validation_1-mlogloss:0.16215
[673] validation_0-mlogloss:0.15786 validation_1-mlogloss:0.16196
[674] validation_0-mlogloss:0.15734 validation_1-mlogloss:0.16142
[675] validation_0-mlogloss:0.15728 validation_1-mlogloss:0.16136
[676] validation_0-mlogloss:0.15712 validation_1-mlogloss:0.16120
[677] validation_0-mlogloss:0.15699 validation_1-mlogloss:0.16107

```
[678] validation_0-mlogloss:0.15692 validation_1-mlogloss:0.16100
[679] validation_0-mlogloss:0.15674 validation_1-mlogloss:0.16081
[680] validation_0-mlogloss:0.15602 validation_1-mlogloss:0.16010
[681] validation_0-mlogloss:0.15590 validation_1-mlogloss:0.15998
[682] validation_0-mlogloss:0.15582 validation_1-mlogloss:0.15991
[683] validation_0-mlogloss:0.15568 validation_1-mlogloss:0.15977
[684] validation_0-mlogloss:0.15565 validation_1-mlogloss:0.15974
[685] validation_0-mlogloss:0.15551 validation_1-mlogloss:0.15960
[686] validation_0-mlogloss:0.15541 validation_1-mlogloss:0.15951
[687] validation_0-mlogloss:0.15531 validation_1-mlogloss:0.15941
[688] validation_0-mlogloss:0.15519 validation_1-mlogloss:0.15929
[689] validation_0-mlogloss:0.15508 validation_1-mlogloss:0.15918
[690] validation_0-mlogloss:0.15485 validation_1-mlogloss:0.15895
[691] validation_0-mlogloss:0.15475 validation_1-mlogloss:0.15887
[692] validation_0-mlogloss:0.15420 validation_1-mlogloss:0.15831
[693] validation_0-mlogloss:0.15416 validation_1-mlogloss:0.15827
[694] validation_0-mlogloss:0.15383 validation_1-mlogloss:0.15794
[695] validation_0-mlogloss:0.15366 validation_1-mlogloss:0.15776
[696] validation_0-mlogloss:0.15357 validation_1-mlogloss:0.15768
[697] validation_0-mlogloss:0.15346 validation_1-mlogloss:0.15758
[698] validation_0-mlogloss:0.15338 validation_1-mlogloss:0.15749
[699] validation_0-mlogloss:0.15321 validation_1-mlogloss:0.15733
[700] validation_0-mlogloss:0.15310 validation_1-mlogloss:0.15721
[701] validation_0-mlogloss:0.15301 validation_1-mlogloss:0.15712
[702] validation_0-mlogloss:0.15297 validation_1-mlogloss:0.15709
[703] validation_0-mlogloss:0.15283 validation_1-mlogloss:0.15695
[704] validation_0-mlogloss:0.15278 validation_1-mlogloss:0.15690
[705] validation_0-mlogloss:0.15267 validation_1-mlogloss:0.15680
[706] validation_0-mlogloss:0.15258 validation_1-mlogloss:0.15671
[707] validation_0-mlogloss:0.15253 validation_1-mlogloss:0.15666
[708] validation_0-mlogloss:0.15251 validation_1-mlogloss:0.15664
[709] validation_0-mlogloss:0.15238 validation_1-mlogloss:0.15652
[710] validation_0-mlogloss:0.15233 validation_1-mlogloss:0.15646
[711] validation_0-mlogloss:0.15221 validation_1-mlogloss:0.15634
[712] validation_0-mlogloss:0.15194 validation_1-mlogloss:0.15607
[713] validation_0-mlogloss:0.15057 validation_1-mlogloss:0.15471
[714] validation_0-mlogloss:0.15052 validation_1-mlogloss:0.15466
[715] validation_0-mlogloss:0.15036 validation_1-mlogloss:0.15450
[716] validation_0-mlogloss:0.15030 validation_1-mlogloss:0.15444
[717] validation_0-mlogloss:0.15016 validation_1-mlogloss:0.15430
[718] validation_0-mlogloss:0.15007 validation_1-mlogloss:0.15421
[719] validation_0-mlogloss:0.14999 validation_1-mlogloss:0.15413
[720] validation_0-mlogloss:0.14989 validation_1-mlogloss:0.15403
[721] validation_0-mlogloss:0.14983 validation_1-mlogloss:0.15397
[722] validation_0-mlogloss:0.14928 validation_1-mlogloss:0.15342
[723] validation_0-mlogloss:0.14921 validation_1-mlogloss:0.15335
[724] validation_0-mlogloss:0.14916 validation_1-mlogloss:0.15331
[725] validation_0-mlogloss:0.14910 validation_1-mlogloss:0.15325
[726] validation_0-mlogloss:0.14887 validation_1-mlogloss:0.15301
[727] validation_0-mlogloss:0.14867 validation_1-mlogloss:0.15281
[728] validation_0-mlogloss:0.14856 validation_1-mlogloss:0.15270
[729] validation_0-mlogloss:0.14850 validation_1-mlogloss:0.15264
[730] validation_0-mlogloss:0.14838 validation_1-mlogloss:0.15253
[731] validation_0-mlogloss:0.14830 validation_1-mlogloss:0.15244
[732] validation_0-mlogloss:0.14825 validation_1-mlogloss:0.15240
[733] validation_0-mlogloss:0.14819 validation_1-mlogloss:0.15233
[734] validation_0-mlogloss:0.14815 validation_1-mlogloss:0.15229
[735] validation_0-mlogloss:0.14808 validation_1-mlogloss:0.15222
[736] validation_0-mlogloss:0.14800 validation_1-mlogloss:0.15214
[737] validation_0-mlogloss:0.14786 validation_1-mlogloss:0.15201
[738] validation_0-mlogloss:0.14774 validation_1-mlogloss:0.15189
[739] validation_0-mlogloss:0.14764 validation_1-mlogloss:0.15180
[740] validation_0-mlogloss:0.14746 validation_1-mlogloss:0.15161
[741] validation_0-mlogloss:0.14741 validation_1-mlogloss:0.15156
[742] validation_0-mlogloss:0.14539 validation_1-mlogloss:0.14954
[743] validation_0-mlogloss:0.14530 validation_1-mlogloss:0.14943
[744] validation_0-mlogloss:0.14523 validation_1-mlogloss:0.14937
[745] validation_0-mlogloss:0.14514 validation_1-mlogloss:0.14929
[746] validation_0-mlogloss:0.14505 validation_1-mlogloss:0.14919
[747] validation_0-mlogloss:0.14497 validation_1-mlogloss:0.14911
[748] validation_0-mlogloss:0.14492 validation_1-mlogloss:0.14907
[749] validation_0-mlogloss:0.14485 validation_1-mlogloss:0.14900
[750] validation_0-mlogloss:0.14481 validation_1-mlogloss:0.14896
[751] validation_0-mlogloss:0.14473 validation_1-mlogloss:0.14889
[752] validation_0-mlogloss:0.14435 validation_1-mlogloss:0.14851
[753] validation_0-mlogloss:0.14425 validation_1-mlogloss:0.14842
[754] validation_0-mlogloss:0.14419 validation_1-mlogloss:0.14836
```

```
[755] validation_0-mlogloss:0.14408 validation_1-mlogloss:0.14825
[756] validation_0-mlogloss:0.14398 validation_1-mlogloss:0.14815
[757] validation_0-mlogloss:0.14395 validation_1-mlogloss:0.14812
[758] validation_0-mlogloss:0.14385 validation_1-mlogloss:0.14801
[759] validation_0-mlogloss:0.14373 validation_1-mlogloss:0.14790
[760] validation_0-mlogloss:0.14360 validation_1-mlogloss:0.14776
[761] validation_0-mlogloss:0.14351 validation_1-mlogloss:0.14768
[762] validation_0-mlogloss:0.14342 validation_1-mlogloss:0.14759
[763] validation_0-mlogloss:0.14339 validation_1-mlogloss:0.14755
[764] validation_0-mlogloss:0.14325 validation_1-mlogloss:0.14742
[765] validation_0-mlogloss:0.14319 validation_1-mlogloss:0.14736
[766] validation_0-mlogloss:0.14313 validation_1-mlogloss:0.14730
[767] validation_0-mlogloss:0.14304 validation_1-mlogloss:0.14721
[768] validation_0-mlogloss:0.14296 validation_1-mlogloss:0.14714
[769] validation_0-mlogloss:0.14288 validation_1-mlogloss:0.14705
[770] validation_0-mlogloss:0.14282 validation_1-mlogloss:0.14700
[771] validation_0-mlogloss:0.14276 validation_1-mlogloss:0.14695
[772] validation_0-mlogloss:0.14272 validation_1-mlogloss:0.14690
[773] validation_0-mlogloss:0.14263 validation_1-mlogloss:0.14680
[774] validation_0-mlogloss:0.14258 validation_1-mlogloss:0.14676
[775] validation_0-mlogloss:0.14253 validation_1-mlogloss:0.14672
[776] validation_0-mlogloss:0.14246 validation_1-mlogloss:0.14664
[777] validation_0-mlogloss:0.14239 validation_1-mlogloss:0.14657
[778] validation_0-mlogloss:0.14204 validation_1-mlogloss:0.14621
[779] validation_0-mlogloss:0.14200 validation_1-mlogloss:0.14618
[780] validation_0-mlogloss:0.14194 validation_1-mlogloss:0.14612
[781] validation_0-mlogloss:0.14141 validation_1-mlogloss:0.14559
[782] validation_0-mlogloss:0.14128 validation_1-mlogloss:0.14546
[783] validation_0-mlogloss:0.14126 validation_1-mlogloss:0.14544
[784] validation_0-mlogloss:0.14115 validation_1-mlogloss:0.14533
[785] validation_0-mlogloss:0.13995 validation_1-mlogloss:0.14415
[786] validation_0-mlogloss:0.13987 validation_1-mlogloss:0.14407
[787] validation_0-mlogloss:0.13985 validation_1-mlogloss:0.14404
[788] validation_0-mlogloss:0.13979 validation_1-mlogloss:0.14399
[789] validation_0-mlogloss:0.13969 validation_1-mlogloss:0.14389
[790] validation_0-mlogloss:0.13952 validation_1-mlogloss:0.14372
[791] validation_0-mlogloss:0.13945 validation_1-mlogloss:0.14365
[792] validation_0-mlogloss:0.13935 validation_1-mlogloss:0.14354
[793] validation_0-mlogloss:0.13925 validation_1-mlogloss:0.14345
[794] validation_0-mlogloss:0.13870 validation_1-mlogloss:0.14290
[795] validation_0-mlogloss:0.13822 validation_1-mlogloss:0.14243
[796] validation_0-mlogloss:0.13816 validation_1-mlogloss:0.14237
[797] validation_0-mlogloss:0.13809 validation_1-mlogloss:0.14231
[798] validation_0-mlogloss:0.13801 validation_1-mlogloss:0.14222
[799] validation_0-mlogloss:0.13798 validation_1-mlogloss:0.14219
[800] validation_0-mlogloss:0.13785 validation_1-mlogloss:0.14207
[801] validation_0-mlogloss:0.13779 validation_1-mlogloss:0.14201
[802] validation_0-mlogloss:0.13768 validation_1-mlogloss:0.14190
[803] validation_0-mlogloss:0.13761 validation_1-mlogloss:0.14182
[804] validation_0-mlogloss:0.13758 validation_1-mlogloss:0.14180
[805] validation_0-mlogloss:0.13662 validation_1-mlogloss:0.14084
[806] validation_0-mlogloss:0.13653 validation_1-mlogloss:0.14076
[807] validation_0-mlogloss:0.13649 validation_1-mlogloss:0.14072
[808] validation_0-mlogloss:0.13647 validation_1-mlogloss:0.14070
[809] validation_0-mlogloss:0.13642 validation_1-mlogloss:0.14065
[810] validation_0-mlogloss:0.13635 validation_1-mlogloss:0.14059
[811] validation_0-mlogloss:0.13628 validation_1-mlogloss:0.14052
[812] validation_0-mlogloss:0.13591 validation_1-mlogloss:0.14012
[813] validation_0-mlogloss:0.13583 validation_1-mlogloss:0.14005
[814] validation_0-mlogloss:0.13549 validation_1-mlogloss:0.13971
[815] validation_0-mlogloss:0.13540 validation_1-mlogloss:0.13962
[816] validation_0-mlogloss:0.13533 validation_1-mlogloss:0.13956
[817] validation_0-mlogloss:0.13522 validation_1-mlogloss:0.13945
[818] validation_0-mlogloss:0.13514 validation_1-mlogloss:0.13936
[819] validation_0-mlogloss:0.13510 validation_1-mlogloss:0.13932
[820] validation_0-mlogloss:0.13484 validation_1-mlogloss:0.13905
[821] validation_0-mlogloss:0.13472 validation_1-mlogloss:0.13892
[822] validation_0-mlogloss:0.13467 validation_1-mlogloss:0.13886
[823] validation_0-mlogloss:0.13457 validation_1-mlogloss:0.13877
[824] validation_0-mlogloss:0.13452 validation_1-mlogloss:0.13872
[825] validation_0-mlogloss:0.13448 validation_1-mlogloss:0.13869
[826] validation_0-mlogloss:0.13442 validation_1-mlogloss:0.13863
[827] validation_0-mlogloss:0.13431 validation_1-mlogloss:0.13853
[828] validation_0-mlogloss:0.13423 validation_1-mlogloss:0.13844
[829] validation_0-mlogloss:0.13419 validation_1-mlogloss:0.13840
[830] validation_0-mlogloss:0.13129 validation_1-mlogloss:0.13547
[831] validation_0-mlogloss:0.13121 validation_1-mlogloss:0.13539
```

```
[832] validation_0-mlogloss:0.13113 validation_1-mlogloss:0.13531
[833] validation_0-mlogloss:0.13090 validation_1-mlogloss:0.13508
[834] validation_0-mlogloss:0.13086 validation_1-mlogloss:0.13504
[835] validation_0-mlogloss:0.13069 validation_1-mlogloss:0.13486
[836] validation_0-mlogloss:0.12839 validation_1-mlogloss:0.13254
[837] validation_0-mlogloss:0.12832 validation_1-mlogloss:0.13247
[838] validation_0-mlogloss:0.12826 validation_1-mlogloss:0.13242
[839] validation_0-mlogloss:0.12818 validation_1-mlogloss:0.13233
[840] validation_0-mlogloss:0.12812 validation_1-mlogloss:0.13227
[841] validation_0-mlogloss:0.12806 validation_1-mlogloss:0.13221
[842] validation_0-mlogloss:0.12797 validation_1-mlogloss:0.13211
[843] validation_0-mlogloss:0.12788 validation_1-mlogloss:0.13204
[844] validation_0-mlogloss:0.12775 validation_1-mlogloss:0.13190
[845] validation_0-mlogloss:0.12686 validation_1-mlogloss:0.13101
[846] validation_0-mlogloss:0.12682 validation_1-mlogloss:0.13097
[847] validation_0-mlogloss:0.12676 validation_1-mlogloss:0.13092
[848] validation_0-mlogloss:0.12610 validation_1-mlogloss:0.13026
[849] validation_0-mlogloss:0.12607 validation_1-mlogloss:0.13022
[850] validation_0-mlogloss:0.12602 validation_1-mlogloss:0.13018
[851] validation_0-mlogloss:0.12591 validation_1-mlogloss:0.13007
[852] validation_0-mlogloss:0.12582 validation_1-mlogloss:0.12999
[853] validation_0-mlogloss:0.12577 validation_1-mlogloss:0.12993
[854] validation_0-mlogloss:0.12572 validation_1-mlogloss:0.12988
[855] validation_0-mlogloss:0.12561 validation_1-mlogloss:0.12977
[856] validation_0-mlogloss:0.12547 validation_1-mlogloss:0.12963
[857] validation_0-mlogloss:0.12542 validation_1-mlogloss:0.12958
[858] validation_0-mlogloss:0.12535 validation_1-mlogloss:0.12951
[859] validation_0-mlogloss:0.12533 validation_1-mlogloss:0.12949
[860] validation_0-mlogloss:0.12525 validation_1-mlogloss:0.12941
[861] validation_0-mlogloss:0.12518 validation_1-mlogloss:0.12934
[862] validation_0-mlogloss:0.12514 validation_1-mlogloss:0.12930
[863] validation_0-mlogloss:0.12508 validation_1-mlogloss:0.12924
[864] validation_0-mlogloss:0.12501 validation_1-mlogloss:0.12917
[865] validation_0-mlogloss:0.12494 validation_1-mlogloss:0.12911
[866] validation_0-mlogloss:0.12488 validation_1-mlogloss:0.12904
[867] validation_0-mlogloss:0.12483 validation_1-mlogloss:0.12899
[868] validation_0-mlogloss:0.12473 validation_1-mlogloss:0.12890
[869] validation_0-mlogloss:0.12463 validation_1-mlogloss:0.12880
[870] validation_0-mlogloss:0.12454 validation_1-mlogloss:0.12871
[871] validation_0-mlogloss:0.12449 validation_1-mlogloss:0.12866
[872] validation_0-mlogloss:0.12441 validation_1-mlogloss:0.12859
[873] validation_0-mlogloss:0.12431 validation_1-mlogloss:0.12848
[874] validation_0-mlogloss:0.12412 validation_1-mlogloss:0.12830
[875] validation_0-mlogloss:0.12400 validation_1-mlogloss:0.12817
[876] validation_0-mlogloss:0.12392 validation_1-mlogloss:0.12809
[877] validation_0-mlogloss:0.12384 validation_1-mlogloss:0.12801
[878] validation_0-mlogloss:0.12380 validation_1-mlogloss:0.12797
[879] validation_0-mlogloss:0.12376 validation_1-mlogloss:0.12793
[880] validation_0-mlogloss:0.12368 validation_1-mlogloss:0.12785
[881] validation_0-mlogloss:0.12363 validation_1-mlogloss:0.12780
[882] validation_0-mlogloss:0.12355 validation_1-mlogloss:0.12773
[883] validation_0-mlogloss:0.12351 validation_1-mlogloss:0.12769
[884] validation_0-mlogloss:0.12347 validation_1-mlogloss:0.12765
[885] validation_0-mlogloss:0.12304 validation_1-mlogloss:0.12722
[886] validation_0-mlogloss:0.12299 validation_1-mlogloss:0.12716
[887] validation_0-mlogloss:0.12294 validation_1-mlogloss:0.12712
[888] validation_0-mlogloss:0.12290 validation_1-mlogloss:0.12707
[889] validation_0-mlogloss:0.12286 validation_1-mlogloss:0.12703
[890] validation_0-mlogloss:0.12281 validation_1-mlogloss:0.12699
[891] validation_0-mlogloss:0.12273 validation_1-mlogloss:0.12691
[892] validation_0-mlogloss:0.12266 validation_1-mlogloss:0.12684
[893] validation_0-mlogloss:0.12257 validation_1-mlogloss:0.12675
[894] validation_0-mlogloss:0.12250 validation_1-mlogloss:0.12668
[895] validation_0-mlogloss:0.12242 validation_1-mlogloss:0.12661
[896] validation_0-mlogloss:0.12177 validation_1-mlogloss:0.12596
[897] validation_0-mlogloss:0.12170 validation_1-mlogloss:0.12589
[898] validation_0-mlogloss:0.12165 validation_1-mlogloss:0.12585
[899] validation_0-mlogloss:0.12160 validation_1-mlogloss:0.12579
[900] validation_0-mlogloss:0.12154 validation_1-mlogloss:0.12573
[901] validation_0-mlogloss:0.12146 validation_1-mlogloss:0.12565
[902] validation_0-mlogloss:0.12111 validation_1-mlogloss:0.12530
[903] validation_0-mlogloss:0.12102 validation_1-mlogloss:0.12521
[904] validation_0-mlogloss:0.12098 validation_1-mlogloss:0.12517
[905] validation_0-mlogloss:0.12093 validation_1-mlogloss:0.12512
[906] validation_0-mlogloss:0.12087 validation_1-mlogloss:0.12505
[907] validation_0-mlogloss:0.12080 validation_1-mlogloss:0.12499
[908] validation_0-mlogloss:0.12073 validation_1-mlogloss:0.12492
```

```
[909] validation_0-mlogloss:0.12068 validation_1-mlogloss:0.12487
[910] validation_0-mlogloss:0.12064 validation_1-mlogloss:0.12484
[911] validation_0-mlogloss:0.12056 validation_1-mlogloss:0.12475
[912] validation_0-mlogloss:0.12048 validation_1-mlogloss:0.12468
[913] validation_0-mlogloss:0.12047 validation_1-mlogloss:0.12467
[914] validation_0-mlogloss:0.12043 validation_1-mlogloss:0.12462
[915] validation_0-mlogloss:0.12036 validation_1-mlogloss:0.12455
[916] validation_0-mlogloss:0.12030 validation_1-mlogloss:0.12450
[917] validation_0-mlogloss:0.12025 validation_1-mlogloss:0.12445
[918] validation_0-mlogloss:0.12011 validation_1-mlogloss:0.12431
[919] validation_0-mlogloss:0.12008 validation_1-mlogloss:0.12428
[920] validation_0-mlogloss:0.12001 validation_1-mlogloss:0.12422
[921] validation_0-mlogloss:0.11783 validation_1-mlogloss:0.12201
[922] validation_0-mlogloss:0.11777 validation_1-mlogloss:0.12195
[923] validation_0-mlogloss:0.11768 validation_1-mlogloss:0.12187
[924] validation_0-mlogloss:0.11765 validation_1-mlogloss:0.12184
[925] validation_0-mlogloss:0.11758 validation_1-mlogloss:0.12177
[926] validation_0-mlogloss:0.11748 validation_1-mlogloss:0.12167
[927] validation_0-mlogloss:0.11695 validation_1-mlogloss:0.12113
[928] validation_0-mlogloss:0.11688 validation_1-mlogloss:0.12107
[929] validation_0-mlogloss:0.11678 validation_1-mlogloss:0.12097
[930] validation_0-mlogloss:0.11673 validation_1-mlogloss:0.12092
[931] validation_0-mlogloss:0.11664 validation_1-mlogloss:0.12082
[932] validation_0-mlogloss:0.11660 validation_1-mlogloss:0.12079
[933] validation_0-mlogloss:0.11651 validation_1-mlogloss:0.12072
[934] validation_0-mlogloss:0.11645 validation_1-mlogloss:0.12065
[935] validation_0-mlogloss:0.11640 validation_1-mlogloss:0.12059
[936] validation_0-mlogloss:0.11631 validation_1-mlogloss:0.12051
[937] validation_0-mlogloss:0.11507 validation_1-mlogloss:0.11925
[938] validation_0-mlogloss:0.11502 validation_1-mlogloss:0.11921
[939] validation_0-mlogloss:0.11498 validation_1-mlogloss:0.11916
[940] validation_0-mlogloss:0.11493 validation_1-mlogloss:0.11912
[941] validation_0-mlogloss:0.11488 validation_1-mlogloss:0.11907
[942] validation_0-mlogloss:0.11483 validation_1-mlogloss:0.11902
[943] validation_0-mlogloss:0.11475 validation_1-mlogloss:0.11893
[944] validation_0-mlogloss:0.11472 validation_1-mlogloss:0.11891
[945] validation_0-mlogloss:0.11466 validation_1-mlogloss:0.11885
[946] validation_0-mlogloss:0.11461 validation_1-mlogloss:0.11880
[947] validation_0-mlogloss:0.11457 validation_1-mlogloss:0.11876
[948] validation_0-mlogloss:0.11453 validation_1-mlogloss:0.11872
[949] validation_0-mlogloss:0.11447 validation_1-mlogloss:0.11867
[950] validation_0-mlogloss:0.11446 validation_1-mlogloss:0.11866
[951] validation_0-mlogloss:0.11439 validation_1-mlogloss:0.11859
[952] validation_0-mlogloss:0.11434 validation_1-mlogloss:0.11854
[953] validation_0-mlogloss:0.11430 validation_1-mlogloss:0.11850
[954] validation_0-mlogloss:0.11427 validation_1-mlogloss:0.11847
[955] validation_0-mlogloss:0.11416 validation_1-mlogloss:0.11836
[956] validation_0-mlogloss:0.11410 validation_1-mlogloss:0.11830
[957] validation_0-mlogloss:0.11406 validation_1-mlogloss:0.11826
[958] validation_0-mlogloss:0.11398 validation_1-mlogloss:0.11819
[959] validation_0-mlogloss:0.11365 validation_1-mlogloss:0.11786
[960] validation_0-mlogloss:0.11362 validation_1-mlogloss:0.11782
[961] validation_0-mlogloss:0.11357 validation_1-mlogloss:0.11778
[962] validation_0-mlogloss:0.11354 validation_1-mlogloss:0.11775
[963] validation_0-mlogloss:0.11351 validation_1-mlogloss:0.11772
[964] validation_0-mlogloss:0.11344 validation_1-mlogloss:0.11765
[965] validation_0-mlogloss:0.11342 validation_1-mlogloss:0.11763
[966] validation_0-mlogloss:0.11339 validation_1-mlogloss:0.11760
[967] validation_0-mlogloss:0.11163 validation_1-mlogloss:0.11581
[968] validation_0-mlogloss:0.11159 validation_1-mlogloss:0.11578
[969] validation_0-mlogloss:0.11156 validation_1-mlogloss:0.11575
[970] validation_0-mlogloss:0.11152 validation_1-mlogloss:0.11571
[971] validation_0-mlogloss:0.11147 validation_1-mlogloss:0.11566
[972] validation_0-mlogloss:0.11142 validation_1-mlogloss:0.11561
[973] validation_0-mlogloss:0.11137 validation_1-mlogloss:0.11557
[974] validation_0-mlogloss:0.11132 validation_1-mlogloss:0.11552
[975] validation_0-mlogloss:0.11125 validation_1-mlogloss:0.11545
[976] validation_0-mlogloss:0.11117 validation_1-mlogloss:0.11538
[977] validation_0-mlogloss:0.11113 validation_1-mlogloss:0.11533
[978] validation_0-mlogloss:0.11105 validation_1-mlogloss:0.11525
[979] validation_0-mlogloss:0.11096 validation_1-mlogloss:0.11516
[980] validation_0-mlogloss:0.11088 validation_1-mlogloss:0.11508
[981] validation_0-mlogloss:0.11083 validation_1-mlogloss:0.11503
[982] validation_0-mlogloss:0.11071 validation_1-mlogloss:0.11490
[983] validation_0-mlogloss:0.11067 validation_1-mlogloss:0.11487
[984] validation_0-mlogloss:0.10978 validation_1-mlogloss:0.11397
[985] validation_0-mlogloss:0.10974 validation_1-mlogloss:0.11394
```

```
[986] validation_0-mlogloss:0.10965 validation_1-mlogloss:0.11384
[987] validation_0-mlogloss:0.10961 validation_1-mlogloss:0.11379
[988] validation_0-mlogloss:0.10951 validation_1-mlogloss:0.11370
[989] validation_0-mlogloss:0.10949 validation_1-mlogloss:0.11368
[990] validation_0-mlogloss:0.10946 validation_1-mlogloss:0.11365
[991] validation_0-mlogloss:0.10937 validation_1-mlogloss:0.11356
[992] validation_0-mlogloss:0.10932 validation_1-mlogloss:0.11351
[993] validation_0-mlogloss:0.10921 validation_1-mlogloss:0.11339
[994] validation_0-mlogloss:0.10917 validation_1-mlogloss:0.11335
[995] validation_0-mlogloss:0.10911 validation_1-mlogloss:0.11330
[996] validation_0-mlogloss:0.10906 validation_1-mlogloss:0.11325
[997] validation_0-mlogloss:0.10897 validation_1-mlogloss:0.11316
[998] validation_0-mlogloss:0.10891 validation_1-mlogloss:0.11310
[999] validation_0-mlogloss:0.10890 validation_1-mlogloss:0.11309
```

param tuning with hyperopt not giving better model performance. I have memory and computational constraints, otherwise may be incresing either learning rate or n_esimators improve the performance. For 1000 estimators it took nearly 10 hours.

## Deep Neural Networks - 1d-CNN

In [5]:

```
# https://github.com/meitetsu3/1DCNN/blob/master/1DCNN_demo.ipynb

import keras
from keras.models import Sequential
from keras.layers import Conv1D, MaxPooling1D, Flatten, Dense, Dropout
from keras.callbacks import ModelCheckpoint
```

In [12]:

```
y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)

Y_train_D_hot = keras.utils.to_categorical(y-1, 4)

train_df, val_df, y_train, y_val = train_test_split(new_train, Y_train_D_hot, stratify=y, test_size
=0.2, random_state=32)

train_df=np.asarray(train_df).astype('float32').reshape(train_df.shape[0], train_df.shape[1], 1)
val_df=np.asarray(val_df).astype('float32').reshape(val_df.shape[0], val_df.shape[1],1)

train_df.shape, val_df.shape, y_train.shape, y_val.shape
```

Out[12]:

```
((3346256, 120, 1), (836564, 120, 1), (3346256, 4), (836564, 4))
```

In [6]:

```
# model architecture

model_n = Sequential()

# Layer 1
model_n.add(Conv1D(filters=16, kernel_size=64,strides = 16, padding='same', activation='relu', inpu
t_shape=(120,1)))
model_n.add(MaxPooling1D(pool_size=2,padding='same'))

# Layer 2
model_n.add(Conv1D(filters=16, kernel_size=3, strides = 1, padding='same', activation='relu'))
model_n.add(MaxPooling1D(pool_size=2,padding='same'))

# Layer 3
model_n.add(Conv1D(filters=32, kernel_size=3, strides = 1, padding='same', activation='relu'))
model_n.add(MaxPooling1D(pool_size=2,padding='same'))
model_n.add(Dropout(0.2))

# Layer 4
model_n.add(Conv1D(filters=32, kernel_size=3, strides = 1, padding='same', activation='relu'))
```

```
model_n.add(MaxPooling1D(pool_size=2,padding='same'))
model_n.add(Dropout(0.2))

# Layer 5
model_n.add(Conv1D(filters=32, kernel_size=3, strides = 1, padding='same', activation='relu'))
model_n.add(keras.layers.GlobalMaxPooling1D())

# Dense Layer
model_n.add(Dense(50, activation='relu'))
model_n.add(Dropout(0.2))

# Output Layer
model_n.add(Dense(4, activation='softmax'))

model_n.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv1d (Conv1D)              (None, 8, 16)             1040
_________________________________________________________________
max_pooling1d (MaxPooling1D) (None, 4, 16)             0
_________________________________________________________________
conv1d_1 (Conv1D)            (None, 4, 16)             784
_________________________________________________________________
max_pooling1d_1 (MaxPooling1 (None, 2, 16)             0
_________________________________________________________________
conv1d_2 (Conv1D)            (None, 2, 32)             1568
_________________________________________________________________
max_pooling1d_2 (MaxPooling1 (None, 1, 32)             0
_________________________________________________________________
dropout (Dropout)            (None, 1, 32)             0
_________________________________________________________________
conv1d_3 (Conv1D)            (None, 1, 32)             3104
_________________________________________________________________
max_pooling1d_3 (MaxPooling1 (None, 1, 32)             0
_________________________________________________________________
dropout_1 (Dropout)          (None, 1, 32)             0
_________________________________________________________________
conv1d_4 (Conv1D)            (None, 1, 32)             3104
_________________________________________________________________
global_max_pooling1d (Global (None, 32)                0
_________________________________________________________________
dense (Dense)                (None, 50)                1650
_________________________________________________________________
dropout_2 (Dropout)          (None, 50)                0
_________________________________________________________________
dense_1 (Dense)              (None, 4)                 204
=================================================================
Total params: 11,454
Trainable params: 11,454
Non-trainable params: 0
_________________________________________________________________
```

In [15]:

```
from keras.callbacks import TensorBoard
import datetime

# Compile
model_n.compile(loss='categorical_crossentropy', optimizer=keras.optimizers.Adam(),
metrics=['accuracy'])


#Callbacks
log_dir="logs" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = TensorBoard(log_dir=log_dir, histogram_freq=1, write_graph=True,write_grads=
True)
checkpoint = ModelCheckpoint(filepath='CNNC2.weights.best.hdf5', verbose=1, save_best_only=True)

# training
model_n.fit(train_df, y_train, batch_size=32, epochs=10,
          validation_data=(val_df, y_val), callbacks=[checkpoint,tensorboard_callback],
          verbose=1, shuffle=True)
```

```
# load the weights that yielded the best validation accuracy
model_n.load_weights('CNNC2.weights.best.hdf5')

# evaluate and print test accuracy

score = model_n.evaluate(train_df, y_train, verbose=0)
print('\n', 'CNN train accuracy:', score[1])

score = model_n.evaluate(val_df, y_val, verbose=0)
print('\n', 'CNN validation accuracy:', score[1])
```

```
WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` Callback.
Epoch 1/10
     2/104571 [..............................] - ETA: 4:42:39 - accuracy: 0.8281 - loss:
0.4937WARNING:tensorflow:Method (on_train_batch_end) is slow compared to the batch update
(0.161768). Check your callbacks.
104569/104571 [============================>.] - ETA: 0s - accuracy: 0.7585 - loss: 0.6112
Epoch 00001: val_loss improved from inf to 0.56838, saving model to CNNC2.weights.best.hdf5
104571/104571 [==============================] - 270s 3ms/step - accuracy: 0.7584 - loss: 0.6112 -
val_accuracy: 0.7746 - val_loss: 0.5684
Epoch 2/10
 98885/104571 [===========================>..] - ETA: 13s - accuracy: 0.7732 - loss: 0.5771
Epoch 00002: val_loss improved from 0.56838 to 0.56626, saving model to CNNC2.weights.best.hdf5
104571/104571 [==============================] - 270s 3ms/step - accuracy: 0.7737 - loss: 0.5765 -
val_accuracy: 0.7761 - val_loss: 0.5663
Epoch 3/10
104565/104571 [============================>.] - ETA: 0s - accuracy: 0.7856 - loss: 0.5509
Epoch 00003: val_loss improved from 0.56626 to 0.51858, saving model to CNNC2.weights.best.hdf5
104571/104571 [==============================] - 265s 3ms/step - accuracy: 0.7856 - loss: 0.5509 -
val_accuracy: 0.8086 - val_loss: 0.5186
Epoch 4/10
104554/104571 [============================>.] - ETA: 0s - accuracy: 0.7939 - loss: 0.5477
Epoch 00004: val_loss did not improve from 0.51858
104571/104571 [==============================] - 265s 3ms/step - accuracy: 0.7939 - loss: 0.5477 -
val_accuracy: 0.8036 - val_loss: 0.5296
Epoch 5/10
104558/104571 [============================>.] - ETA: 0s - accuracy: 0.7961 - loss: 0.5371
Epoch 00005: val_loss did not improve from 0.51858
104571/104571 [==============================] - 266s 3ms/step - accuracy: 0.7961 - loss: 0.5371 -
val_accuracy: 0.7916 - val_loss: 0.6516
Epoch 6/10
104555/104571 [============================>.] - ETA: 0s - accuracy: 0.7986 - loss: 0.5379
Epoch 00006: val_loss improved from 0.51858 to 0.49670, saving model to CNNC2.weights.best.hdf5
104571/104571 [==============================] - 266s 3ms/step - accuracy: 0.7986 - loss: 0.5379 -
val_accuracy: 0.8136 - val_loss: 0.4967
Epoch 7/10
104552/104571 [============================>.] - ETA: 0s - accuracy: 0.8134 - loss: 0.5151
Epoch 00007: val_loss did not improve from 0.49670
104571/104571 [==============================] - 264s 3ms/step - accuracy: 0.8133 - loss: 0.5151 -
val_accuracy: 0.7890 - val_loss: 0.5189
Epoch 8/10
104567/104571 [============================>.] - ETA: 0s - accuracy: 0.8191 - loss: 0.4966
Epoch 00008: val_loss did not improve from 0.49670
104571/104571 [==============================] - 262s 3ms/step - accuracy: 0.8191 - loss: 0.4966 -
val_accuracy: 0.7804 - val_loss: 0.5265
Epoch 9/10
104569/104571 [============================>.] - ETA: 0s - accuracy: 0.7966 - loss: 0.5386
Epoch 00009: val_loss improved from 0.49670 to 0.45057, saving model to CNNC2.weights.best.hdf5
104571/104571 [==============================] - 260s 2ms/step - accuracy: 0.7966 - loss: 0.5386 -
val_accuracy: 0.8296 - val_loss: 0.4506
Epoch 10/10
104555/104571 [============================>.] - ETA: 0s - accuracy: 0.8085 - loss: 0.5179
Epoch 00010: val_loss did not improve from 0.45057
104571/104571 [==============================] - 256s 2ms/step - accuracy: 0.8085 - loss: 0.5179 -
val_accuracy: 0.8154 - val_loss: 0.5064

 CNN train accuracy: 0.8289996385574341

 CNN validation accuracy: 0.8295803070068359
```

In [17]:

```
print('CNN train loss is', score[0])

print('CNN validation loss is', score[0])
```

```
CNN train loss is 0.4505651593208313
CNN validation loss is 0.4505651593208313
```

In [22]:

```
# test data predictions

list_tup=[]
for tup in [(x,y) for x in np.unique(test['crew']) for y in np.unique(test['seat'])]:
    list_tup.append(tup)

test_id_f=[]
pred=[]
for i in range(0,18):
    test_df,test_id=test_data_FE(i,list_tup)
    test_id_f.append(test_id)
    test_df=test_df.drop(featr,axis=1)
    test_df=test_df.replace(np.nan,0)
    test_df=np.asarray(test_df).astype('float32').reshape(test_df.shape[0], test_df.shape[1], 1)
    y_test_pred=model_n.predict(test_df)
    pred.append(y_test_pred)
    print('---------------------------------- loop : ',i+1,' ----------------------------------')
```

```
---------------------------------- loop :  1  ----------------------------------
---------------------------------- loop :  2  ----------------------------------
---------------------------------- loop :  3  ----------------------------------
---------------------------------- loop :  4  ----------------------------------
missing gsr
---------------------------------- loop :  5  ----------------------------------
---------------------------------- loop :  6  ----------------------------------
---------------------------------- loop :  7  ----------------------------------
---------------------------------- loop :  8  ----------------------------------
---------------------------------- loop :  9  ----------------------------------
failed "gsr_amp" extraction
---------------------------------- loop :  10  ----------------------------------
---------------------------------- loop :  11  ----------------------------------
---------------------------------- loop :  12  ----------------------------------
missing gsr
---------------------------------- loop :  13  ----------------------------------
---------------------------------- loop :  14  ----------------------------------
---------------------------------- loop :  15  ----------------------------------
---------------------------------- loop :  16  ----------------------------------
---------------------------------- loop :  17  ----------------------------------
---------------------------------- loop :  18  ----------------------------------
```

In [23]:

```
test_pr=[]
tst_id=[]
for i in pred:
    for j in i:
        test_pr.append(j)

for i in test_id_f:
    for j in i:
        tst_id.append(j)

sub=pd.DataFrame(test_pr,columns=['A','B','C','D'])
sub['id']=tst_id
sub=sub[['id','A','B','C','D']]
sub=sub.sort_values(by=['id'])
sub.reset_index(inplace = True)
sub=sub.drop('index',axis=1)
sub.head()
```

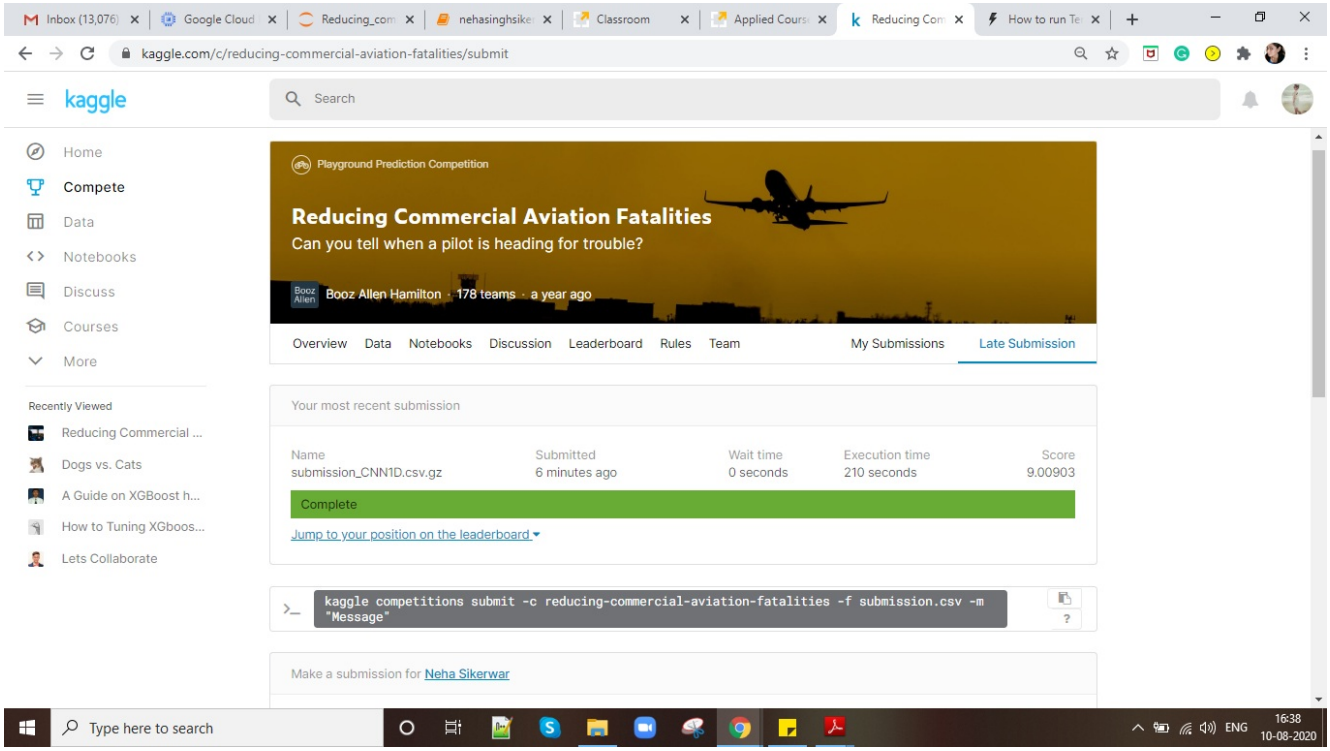
Out[23]:

| | id | A | B | C | D |
|---|---|---|---|---|---|
| **0** | 0 | 0.000000e+00 | 0.0 | 2.971665e-31 | 1.0 |
| **1** | 1 | 2.733429e-10 | 0.0 | 1.245971e-11 | 1.0 |

| | | | | | |
|---|---|---|---|---|---|
| 2 | 2 | 0.000000e+00 | 0.0 | 2.377329e-30 | 1.0 |
| 3 | 3 | 1.708287e-09 | 0.0 | 7.901020e-11 | 1.0 |
| 4 | 4 | 0.000000e+00 | 0.0 | 2.562986e-31 | 1.0 |

In [25]:

```
sub.to_csv('submission_CNN1D.csv.gz',index=False, compression='gzip')
```




In [ ]:

```
```

## Dense Neural Network

In [45]:

```
import keras
from keras.models import Sequential
from keras.layers import Flatten, Dense, Dropout
from keras.callbacks import ModelCheckpoint

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)

Y_train_D_hot = keras.utils.to_categorical(y-1, 4)

train_df, val_df, y_train, y_val = train_test_split(new_train, Y_train_D_hot, stratify=y, test_size
=0.2, random_state=32)
train_df.shape, val_df.shape, y_train.shape, y_val.shape
```

Out[45]:

```
((3346256, 120), (836564, 120), (3346256, 4), (836564, 4))
```

In [11]:

```
# model architecture

model_n = Sequential()
```

```
model_n.add(keras.Input(shape=(120,)))

# Dense layer 1
model_n.add(Dense(50,activation='relu',kernel_initializer=keras.initializers.he_uniform(seed=30)))

# Dense layer 2
model_n.add(Dense(50,activation='relu',kernel_initializer=keras.initializers.he_uniform(seed=30)))

# Dense layer 3
model_n.add(Dense(50,activation='relu',kernel_initializer=keras.initializers.he_uniform(seed=30)))

# Dense layer 4
model_n.add(Dense(50,activation='relu',kernel_initializer=keras.initializers.he_uniform(seed=30)))

# Dense layer 5
model_n.add(Dense(50,activation='relu',kernel_initializer=keras.initializers.he_uniform(seed=30)))

#output layer
model_n.add(Dense(4,activation='softmax',kernel_initializer=keras.initializers.he_uniform(seed=30)
))

model_n.summary()
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense (Dense)                (None, 50)                6050
_________________________________________________________________
dense_1 (Dense)              (None, 50)                2550
_________________________________________________________________
dense_2 (Dense)              (None, 50)                2550
_________________________________________________________________
dense_3 (Dense)              (None, 50)                2550
_________________________________________________________________
dense_4 (Dense)              (None, 50)                2550
_________________________________________________________________
dense_5 (Dense)              (None, 4)                 204
=================================================================
Total params: 16,454
Trainable params: 16,454
Non-trainable params: 0
_________________________________________________________________
```

In [9]:

```
from keras.callbacks import TensorBoard
import datetime

# Compile
model_n.compile(loss='categorical_crossentropy', optimizer=keras.optimizers.Adam(),
metrics=['accuracy'])


#Callbacks
log_dir="logs" + datetime.datetime.now().strftime("%Y%m%d-%H%M%S")
tensorboard_callback = TensorBoard(log_dir=log_dir, histogram_freq=1, write_graph=True,write_grads=
True)
checkpoint = ModelCheckpoint(filepath='DenseNet.weights.best.hdf5', verbose=1, save_best_only=True)

# training
model_n.fit(train_df, y_train, batch_size=32, epochs=12,
          validation_data=(val_df, y_val), callbacks=[checkpoint,tensorboard_callback],
          verbose=1, shuffle=True)

# load the weights that yielded the best validation accuracy
model_n.load_weights('DenseNet.weights.best.hdf5')

# evaluate and print test accuracy

score = model_n.evaluate(train_df, y_train, verbose=0)
print('\n', 'train loss:', score[0])

score = model_n.evaluate(val_df, y_val, verbose=0)
print('\n', 'validation loss:', score[0])
```

```
WARNING:tensorflow:`write_grads` will be ignored in TensorFlow 2.0 for the `TensorBoard` Callback.
Epoch 1/12
104571/104571 [==============================] - ETA: 0s - loss: 0.8853 - accuracy: 0.7391
Epoch 00001: val_loss improved from inf to 0.54442, saving model to DenseNet.weights.best.hdf5
104571/104571 [==============================] - 281s 3ms/step - loss: 0.8853 - accuracy: 0.7391 -
val_accuracy: 0.7799 - val_loss: 0.5444
Epoch 2/12
104555/104571 [==============================>.] - ETA: 0s - loss: 0.5412 - accuracy: 0.7887
Epoch 00002: val_loss improved from 0.54442 to 0.53116, saving model to DenseNet.weights.best.hdf5
104571/104571 [==============================] - 281s 3ms/step - loss: 0.5412 - accuracy: 0.7887 -
val_accuracy: 0.7919 - val_loss: 0.5312
Epoch 3/12
104563/104571 [==============================>.] - ETA: 0s - loss: 0.5250 - accuracy: 0.7948
Epoch 00003: val_loss improved from 0.53116 to 0.48993, saving model to DenseNet.weights.best.hdf5
104571/104571 [==============================] - 280s 3ms/step - loss: 0.5250 - accuracy: 0.7948 -
val_accuracy: 0.8079 - val_loss: 0.4899
Epoch 4/12
104556/104571 [==============================>.] - ETA: 0s - loss: 0.4978 - accuracy: 0.8056
Epoch 00004: val_loss improved from 0.48993 to 0.39763, saving model to DenseNet.weights.best.hdf5
104571/104571 [==============================] - 275s 3ms/step - loss: 0.4978 - accuracy: 0.8056 -
val_accuracy: 0.8631 - val_loss: 0.3976
Epoch 5/12
104570/104571 [==============================>.] - ETA: 0s - loss: 0.5038 - accuracy: 0.8040
Epoch 00005: val_loss did not improve from 0.39763
104571/104571 [==============================] - 278s 3ms/step - loss: 0.5038 - accuracy: 0.8040 -
val_accuracy: 0.8200 - val_loss: 0.4566
Epoch 6/12
104561/104571 [==============================>.] - ETA: 0s - loss: 0.4619 - accuracy: 0.8252
Epoch 00006: val_loss did not improve from 0.39763
104571/104571 [==============================] - 273s 3ms/step - loss: 0.4619 - accuracy: 0.8252 -
val_accuracy: 0.8094 - val_loss: 0.4804
Epoch 7/12
104566/104571 [==============================>.] - ETA: 0s - loss: 0.4319 - accuracy: 0.8464
Epoch 00007: val_loss improved from 0.39763 to 0.39090, saving model to DenseNet.weights.best.hdf5
104571/104571 [==============================] - 274s 3ms/step - loss: 0.4319 - accuracy: 0.8464 -
val_accuracy: 0.8665 - val_loss: 0.3909
Epoch 8/12
104561/104571 [==============================>.] - ETA: 0s - loss: 0.3884 - accuracy: 0.8681
Epoch 00008: val_loss improved from 0.39090 to 0.34880, saving model to DenseNet.weights.best.hdf5
104571/104571 [==============================] - 276s 3ms/step - loss: 0.3884 - accuracy: 0.8681 -
val_accuracy: 0.8814 - val_loss: 0.3488
Epoch 9/12
104567/104571 [==============================>.] - ETA: 0s - loss: 0.4184 - accuracy: 0.8598
Epoch 00009: val_loss did not improve from 0.34880
104571/104571 [==============================] - 279s 3ms/step - loss: 0.4184 - accuracy: 0.8598 -
val_accuracy: 0.8844 - val_loss: 0.3635
Epoch 10/12
104553/104571 [==============================>.] - ETA: 0s - loss: 0.4368 - accuracy: 0.8604
Epoch 00010: val_loss did not improve from 0.34880
104571/104571 [==============================] - 276s 3ms/step - loss: 0.4368 - accuracy: 0.8604 -
val_accuracy: 0.8671 - val_loss: 0.4116
Epoch 11/12
104562/104571 [==============================>.] - ETA: 0s - loss: 0.4054 - accuracy: 0.8663
Epoch 00011: val_loss did not improve from 0.34880
104571/104571 [==============================] - 270s 3ms/step - loss: 0.4054 - accuracy: 0.8663 -
val_accuracy: 0.8722 - val_loss: 0.3854
Epoch 12/12
104551/104571 [==============================>.] - ETA: 0s - loss: 0.4939 - accuracy: 0.8261
Epoch 00012: val_loss did not improve from 0.34880
104571/104571 [==============================] - 267s 3ms/step - loss: 0.4939 - accuracy: 0.8260 -
val_accuracy: 0.6895 - val_loss: 0.7826

 train loss: 0.3478294014930725

 validation loss: 0.3487970232963562
```

In [38]:

```
# test data predictions

list_tup=[]
for tup in [(x,y) for x in np.unique(test['crew']) for y in np.unique(test['seat'])]:
    list_tup.append(tup)

test_id_f=[]
```

```
test_id_f=[]
pred=[]
for i in range(0,18):
    test_df,test_id=test_data_FE(i,list_tup)
    test_id_f.append(test_id)
    test_df=test_df.drop(featr,axis=1)
    test_df=test_df.replace(np.nan,0)
    y_test_pred=model_n.predict(test_df)
    pred.append(y_test_pred)
    print('---------------------------------- loop : ',i+1,' ----------------------------------')
```

```
---------------------------------- loop :  1  ----------------------------------
---------------------------------- loop :  2  ----------------------------------
---------------------------------- loop :  3  ----------------------------------
---------------------------------- loop :  4  ----------------------------------
missing gsr
---------------------------------- loop :  5  ----------------------------------
---------------------------------- loop :  6  ----------------------------------
---------------------------------- loop :  7  ----------------------------------
---------------------------------- loop :  8  ----------------------------------
---------------------------------- loop :  9  ----------------------------------
failed "gsr_amp" extraction
---------------------------------- loop :  10  ----------------------------------
---------------------------------- loop :  11  ----------------------------------
---------------------------------- loop :  12  ----------------------------------
missing gsr
---------------------------------- loop :  13  ----------------------------------
---------------------------------- loop :  14  ----------------------------------
---------------------------------- loop :  15  ----------------------------------
---------------------------------- loop :  16  ----------------------------------
---------------------------------- loop :  17  ----------------------------------
---------------------------------- loop :  18  ----------------------------------
```

In [39]:

```
test_pr=[]
tst_id=[]
for i in pred:
    for j in i:
        test_pr.append(j)

for i in test_id_f:
    for j in i:
        tst_id.append(j)

sub=pd.DataFrame(test_pr,columns=['A','B','C','D'])
sub['id']=tst_id
sub=sub[['id','A','B','C','D']]
sub=sub.sort_values(by=['id'])
sub.reset_index(inplace = True)
sub=sub.drop('index',axis=1)
sub.head()
```

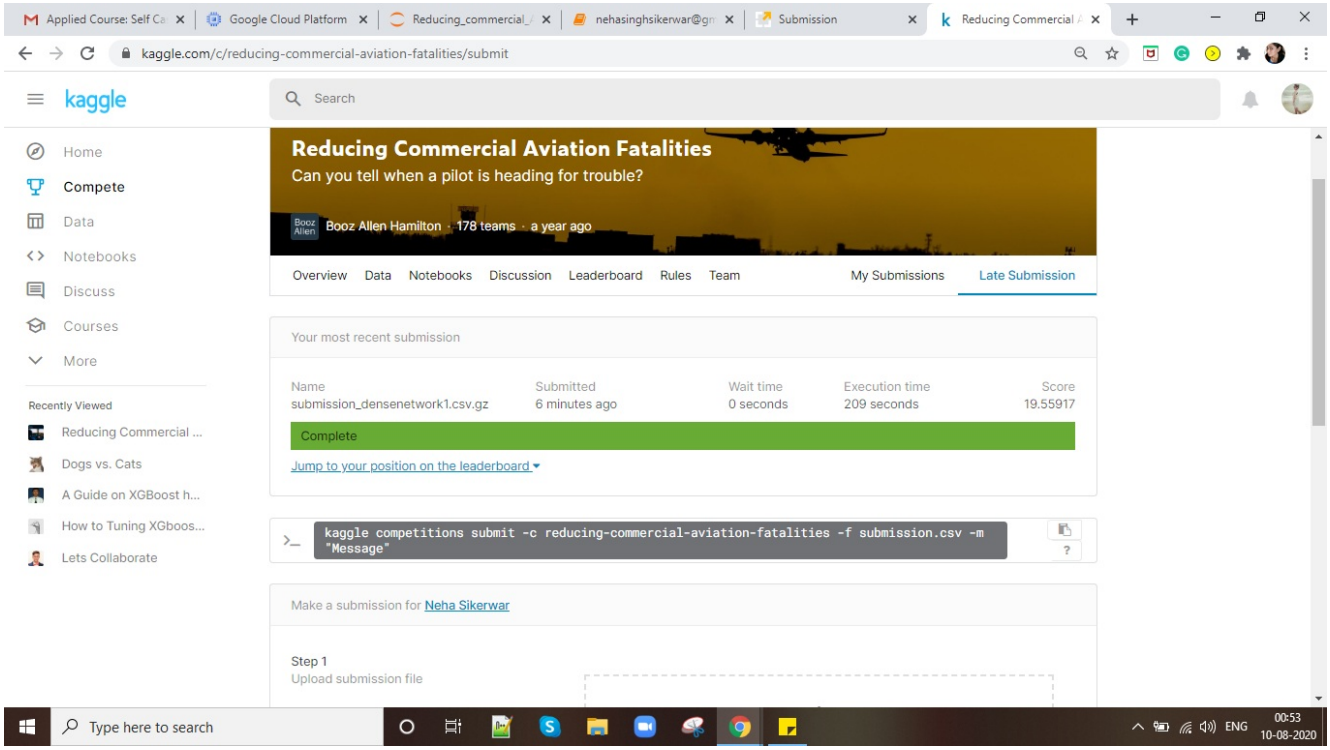
Out[39]:

| | id | A | B | C | D |
|---|---|---|---|---|---|
| **0** | 0 | 0.0 | 0.0 | 0.0 | 1.0 |
| **1** | 1 | 0.0 | 0.0 | 0.0 | 1.0 |
| **2** | 2 | 0.0 | 0.0 | 0.0 | 1.0 |
| **3** | 3 | 0.0 | 0.0 | 0.0 | 1.0 |
| **4** | 4 | 0.0 | 0.0 | 0.0 | 1.0 |

In [41]:

```
sub.to_csv('submission_densenetwork1.csv.gz',index=False, compression='gzip')
```

In [ ]:




In [ ]:

```
```

## AdaBoostClassifier

In [6]:

```
df=new_train.sample(90000).copy()
df.reset_index(inplace = True)
y=df['event']
df=df.drop('event',axis=1)
df=df.drop('index',axis=1)
df=df.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(df, y, stratify=y, test_size=0.2, random_state=
32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[6]:

```
((72000, 120), (18000, 120), 72000, 18000)
```

In [7]:

```
from sklearn.ensemble import AdaBoostClassifier
from scipy.stats import uniform, randint

a_cfl=AdaBoostClassifier()
prams={'learning_rate':uniform(0.01, 0.4),
     'n_estimators':randint(50,200)}

random_cfl=RandomizedSearchCV(a_cfl,param_distributions=prams,verbose=10,n_jobs=-1)
random_cfl.fit(train_df, y_train)
print (random_cfl.best_params_)
```

```
Fitting 5 folds for each of 10 candidates, totalling 50 fits

[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done   2 tasks      | elapsed:  3.6min
[Parallel(n_jobs=-1)]: Done   9 tasks      | elapsed:  6.0min
[Parallel(n_jobs=-1)]: Done  16 tasks      | elapsed:  7.1min
```

```
[Parallel(n_jobs=-1)]: Done  25 tasks      | elapsed:  9.4min
[Parallel(n_jobs=-1)]: Done  34 tasks      | elapsed: 13.3min
[Parallel(n_jobs=-1)]: Done  41 out of  50 | elapsed: 15.4min remaining:  3.4min
[Parallel(n_jobs=-1)]: Done  47 out of  50 | elapsed: 16.7min remaining:  1.1min
[Parallel(n_jobs=-1)]: Done  50 out of  50 | elapsed: 17.0min finished
```

```
{'learning_rate': 0.3970066479726446, 'n_estimators': 104}
```

In [6]:

```
# full data

y=new_train['event']
new_train=new_train.drop('event',axis=1)
new_train=new_train.drop(featr,axis=1)
train_df, val_df, y_train, y_val = train_test_split(new_train, y, stratify=y, test_size=0.2, random
_state=32)
train_df.shape, val_df.shape, len(y_train), len(y_val)
```

Out[6]:

```
((3346256, 120), (836564, 120), 3346256, 836564)
```

In [7]:

```
from sklearn.ensemble import AdaBoostClassifier

cfl=AdaBoostClassifier(learning_rate=0.3970066479726446, n_estimators=104)
cfl.fit(train_df, y_train)

predict_y = cfl.predict_proba(train_df)
print("The train log loss is:",log_loss(y_train, predict_y))
predict_y = cfl.predict_proba(val_df)
print("The cross validation log loss is:",log_loss(y_val, predict_y))
```

```
The train log loss is: 1.30277666504951
The cross validation log loss is: 1.3028351930202728
```

In [ ]:

```

```

# Summary

In [29]:

```
df=pd.DataFrame(data=[['Random Model',1.6455166263817762],['LGBMClassifier',0.04811346513573305],
                      ['xgbClassifier 1',0.0008236892036873947],['Random Forest',0.11121686718047841
],
                      ['CatBoost',0.00133690021],['SVM',0.8698199539967327],['Logistic Regression',0
.8526577092862271],
                      ['xgb.cv',0.047419433995244564],['xgbClassifier 2',0.000754514872950122],
                      ['xgbClassifier 3 : hyperopt',0.11309],['Deep neural network: CNN-1D',0.450565
1593208313],
                      ['Dense Neural Network',0.3487970232963562],
['AdaBoostClassifier',1.3028351930202728]],
                      columns=['Model','Validation Loss'])
df
```

Out[29]:

| | Model | Validation Loss |
|---|---|---|
| 0 | Random Model | 1.645517 |
| 1 | LGBMClassifier | 0.048113 |
| 2 | xgbClassifier 1 | 0.000824 |
| 3 | Random Forest | 0.111217 |

| | Model | Validation Loss |
|---|---|---|
| **4** | CatBoost | 0.001337 |
| **5** | SVM | 0.869820 |
| **6** | Logistic Regression | 0.852658 |
| **7** | xgb.cv | 0.047419 |
| **8** | xgbClassifier 2 | 0.000755 |
| **9** | xgbClassifier 3 : hyperopt | 0.113090 |
| **10** | Deep neural network: CNN-1D | 0.450565 |
| **11** | Dense Neural Network | 0.348797 |
| **12** | AdaBoostClassifier | 1.302835 |

xgbClassifier gave best performance (loss: 0.000755) of all.

In [ ]: