Comparative Perspectives of Boosting Classifiers in Machine Learning

February 7, 2021

0.0.1 Title of the project- Comparative Perspectives of Boosting Classifiers in Machine Learning

- Project by: Sudip Pandit*

0.0.2 Description of the Project:

- The "Breast Cancer Dataset" is used in this project. It has df.shape=(569, 31) which means 569 rows and 32 columns.
- The link of the datset used in this project is -https://www.kaggle.com/uciml/breast-cancer-wisconsin-data
- I am importing the important python packages- skelarn, pandas, numpy, seaborn and matplotlib to complete the project.
- The machine learning models such as Logistic Regression, Decision Tree, Random Forest, XGBoost, AdaBoost and Gradient Boosting classifier have been used.
- The performance of the machine learnig models have been tested on the basis of accuracy score, confusion matrix, classification report, f1 score and roc auc score.
- I had tuned hyperparameters to improve the perforamnce for XGBoost model
- The good visualization is also important along with accuracy score in model building. The performance of the model have been visualized in this project.

0.0.3 Problem statement:

- The full form of XGBoost is eXtreme Gradient Boosting, also called winner for several kaggle competetion machine learning model. Most of the literatues of Machine Learning found in google has described this model as having best accuracy, efficient and feasibility.
- It is a decision-tree-based ensemble ML algorithm based on gradient boosting framework.
- It is considered that XGBoost provides a convenient way of cross-validation.
- Cross-validation technique is applied to test the model's overfitting during the training phase. If the model gives good accuracy in training dataset but the model works very poor in testing unseen dataset then it is called overfitting or a model of low bias and high variance.
- I have to calculate the model training and testing errors with different learning rates.As we know that the best technique to choose the learning rate value is between 0 and 1. I will be going to start the test by putting the learning rate as 0.01.
- It would easy to see the results through good visualization. I am also going to visualize the training and testing errors and accuracies by making a graph. Finally, I will tune the hyperparameters which helps us predict the testing datasets i.e. x_test.

0.0.4 Purpose of the project:

- The purpose of this project is to provide a quick overview of different classifiers used in machine learning.
- I compare different classifiers accuracies with XGBoost Classifier and predict why it is important in machine learning model building.
- This project is focused in Boosting classifier. It will not cover the bagging part (Here, my goal is to identify some of the classifiers accuracies and compare their accuracies with the XGBoost classfier.

0.0.5 Reference Documents:

- I would like to highly recommend to read the article link- https://medium.com/@saugata.paul1010/ensemble-learning-bagging-boosting-stacking-and-cascading-classifiers-in-machine-learning-9c66cb271674. The writer has explained very clearly about the ensemble method. The best part of this article is that he has described about the concept of cascading.
- As per Paul(2018), Cascading is one of the most powerful ensemble learning algorithm which is used by Machine Learning engineers and scientists when they want to be absolutely dead sure about the accuracy of a result.
- I read different articles to prepare this project which are down at end of the project as a reference.

0.0.6 Import the libraries

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

0.0.7 Load the dataset

[ ]:
```
df=pd.read_csv('cancer.csv')
```

[ ]:
```
df.head()
```

[ ]:
```
df.tail()
```

[5]:
```
df.drop('id', axis=1, inplace=True)
```

[6]:
```
df['diagnosis']=pd.get_dummies(df['diagnosis'])
```

0.0.8 Print the first five rows

[7]:
```
df.head()
```

[7]:

	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	\
0	0	17.99	10.38	122.80	1001.0	
1	0	20.57	17.77	132.90	1326.0	

```
2          0        19.69         21.25          130.00     1203.0
3          0        11.42         20.38           77.58      386.1
4          0        20.29         14.34          135.10     1297.0

   smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
0          0.11840           0.27760          0.3001              0.14710
1          0.08474           0.07864          0.0869              0.07017
2          0.10960           0.15990          0.1974              0.12790
3          0.14250           0.28390          0.2414              0.10520
4          0.10030           0.13280          0.1980              0.10430

   symmetry_mean  …  radius_worst  texture_worst  perimeter_worst  \
0         0.2419  …         25.38          17.33           184.60
1         0.1812  …         24.99          23.41           158.80
2         0.2069  …         23.57          25.53           152.50
3         0.2597  …         14.91          26.50            98.87
4         0.1809  …         22.54          16.67           152.20

   area_worst  smoothness_worst  compactness_worst  concavity_worst  \
0      2019.0            0.1622             0.6656           0.7119
1      1956.0            0.1238             0.1866           0.2416
2      1709.0            0.1444             0.4245           0.4504
3       567.7            0.2098             0.8663           0.6869
4      1575.0            0.1374             0.2050           0.4000

   concave points_worst  symmetry_worst  fractal_dimension_worst
0                0.2654          0.4601                  0.11890
1                0.1860          0.2750                  0.08902
2                0.2430          0.3613                  0.08758
3                0.2575          0.6638                  0.17300
4                0.1625          0.2364                  0.07678

[5 rows x 31 columns]
```

0.0.9 This gives the shape of the dataset

[76]:
```
df.shape
```

[76]:
```
(569, 31)
```

[8]:
```
df.dtypes
```

[8]:
```
diagnosis                 uint8
radius_mean             float64
texture_mean            float64
perimeter_mean          float64
area_mean               float64
```

```
smoothness_mean            float64
compactness_mean           float64
concavity_mean             float64
concave points_mean        float64
symmetry_mean              float64
fractal_dimension_mean     float64
radius_se                  float64
texture_se                 float64
perimeter_se               float64
area_se                    float64
smoothness_se              float64
compactness_se             float64
concavity_se               float64
concave points_se          float64
symmetry_se                float64
fractal_dimension_se       float64
radius_worst               float64
texture_worst              float64
perimeter_worst            float64
area_worst                 float64
smoothness_worst           float64
compactness_worst          float64
concavity_worst            float64
concave points_worst       float64
symmetry_worst             float64
fractal_dimension_worst    float64
dtype: object
```

[9]:
```
df.columns
```

[9]:
```
Index(['diagnosis', 'radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
       'symmetry_worst', 'fractal_dimension_worst'],
      dtype='object')
```

0.0.10 x and y variables (sampling)

[10]:
```
x=df.drop('diagnosis', axis=1).values
y=df['diagnosis']
```

0.0.11 Scaling the x features in the same scale

[11]:
```
from sklearn.preprocessing import StandardScaler
```

[12]:
```
ss=StandardScaler()
```

[13]:
```
x_scaled=ss.fit_transform(x)
```

0.0.12 Model Building

- Logistic Regression

[14]:
```
from sklearn.linear_model import LogisticRegression
```

[15]:
```
ll=LogisticRegression(solver='lbfgs')
```

0.0.13 Split the dataset into training and testing in order to train the model

[16]:
```
from sklearn.model_selection import train_test_split
```

[17]:
```
x_train, x_test, y_train, y_test=train_test_split(x_scaled, y, test_size=0.30,␣
↪random_state=62)
```

[18]:
```
ll.fit(x_train, y_train)
```

[18]:
```
LogisticRegression()
```

[19]:
```
y_pred=ll.predict(x_test)
```

[20]:
```
print(y_test)
```

```
465    1
274    0
505    1
212    0
87     0
      ..
440    1
349    1
26     0
231    1
168    0
Name: diagnosis, Length: 171, dtype: uint8
```

0.0.14 Import the following parameters to test each classification model

[21]:
```
from sklearn.metrics import confusion_matrix, classification_report, f1_score,␣
↪roc_curve, roc_auc_score, accuracy_score
```

[22]:
```
confusion_matrix(y_test, y_pred)
```

[22]:
```
array([[ 61,   8],
       [  0, 102]], dtype=int64)
```

[23]:
```
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      0.88      0.94        69
           1       0.93      1.00      0.96       102

    accuracy                           0.95       171
   macro avg       0.96      0.94      0.95       171
weighted avg       0.96      0.95      0.95       171
```

[24]:
```
f1_score(y_test, y_pred)
```

[24]:
```
0.9622641509433962
```

[25]:
```
print(roc_curve(y_test, y_pred))
```

```
(array([0.        , 0.11594203, 1.        ]), array([0., 1., 1.]), array([2, 1,
0]))
```

[26]:
```
roc_auc_score(y_test, y_pred)
```

[26]:
```
0.9420289855072465
```

0.0.15 Random Forest Classifier

[27]:
```
from sklearn.ensemble import RandomForestClassifier
```

[28]:
```
rr=RandomForestClassifier()
```

[29]:
```
rr.fit(x_train, y_train)
```

[29]:
```
RandomForestClassifier()
```

[30]:
```
y_pred_rr=rr.predict(x_test)
```

[31]:
```
y_pred_rr
```

[31]:
```
array([1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1,
       1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0], dtype=uint8)
```

[32]:
```
confusion_matrix(y_test, y_pred_rr)
```

[32]:
```
array([[62,  7],
       [ 4, 98]], dtype=int64)
```

[33]:
```
accuracy_score(y_test, y_pred_rr)
```

[33]:
```
0.935672514619883
```

[34]:
```
print(classification_report(y_test, y_pred_rr))
```

```
              precision    recall  f1-score   support

           0       0.94      0.90      0.92        69
           1       0.93      0.96      0.95       102

    accuracy                           0.94       171
   macro avg       0.94      0.93      0.93       171
weighted avg       0.94      0.94      0.94       171
```

[35]:
```
f1_score(y_test, y_pred_rr)
```

[35]:
```
0.9468599033816426
```

[36]:
```
roc_auc_score(y_test, y_pred_rr)
```

[36]:
```
0.9296675191815857
```

0.0.16 Decision Tree Classifier

[37]:
```
from sklearn.tree import DecisionTreeClassifier
```

[38]:
```
dt=DecisionTreeClassifier(max_depth=2)
```

[39]:
```
dt.fit(x_train, y_train)
```

[39]:
```
DecisionTreeClassifier(max_depth=2)
```

[40]:
```
y_pred_dt=dt.predict(x_test)
```

[41]:
```
%%time
y_pred_dt
```

```
Wall time: 0 ns
```

[41]:
```
array([1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 1,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0,
       1, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1,
       1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0], dtype=uint8)
```

[42]:
```
accuracy_score(y_test,y_pred_dt)
```

[42]:
```
0.9005847953216374
```

[43]:
```
f1_score(y_test, y_pred_dt)
```

[43]:
```
0.9170731707317074
```

[44]:
```
roc_auc_score(y_test, y_pred_dt)
```

[44]:
```
0.8955669224211422
```

[45]:
```
confusion_matrix(y_test, y_pred_dt)
```

[45]:
```
array([[60,  9],
       [ 8, 94]], dtype=int64)
```

[46]:
```
print(classification_report(y_test, y_pred_dt))
```

```
              precision    recall  f1-score   support

           0       0.88      0.87      0.88        69
           1       0.91      0.92      0.92       102

    accuracy                           0.90       171
   macro avg       0.90      0.90      0.90       171
weighted avg       0.90      0.90      0.90       171
```

0.0.17 XGBoost Classifier

[47]:
```
from xgboost import XGBClassifier
```

[48]:
```
%%time
model =XGBClassifier(max_depth=12,
```

```
                        subsample=0.33,
                        objective='binary:logistic',
                        n_estimators=300,
                        learning_rate = 0.01)
eval_set = [(x_train, y_train), (x_test, y_test)]
model.fit(x_train, y_train.values.ravel(), early_stopping_rounds=15,␣
 ↪eval_metric=["error", "logloss"], eval_set=eval_set, verbose=True)
```

```
[0]     validation_0-error:0.04020      validation_0-logloss:0.68482
validation_1-error:0.09942      validation_1-logloss:0.68604
Multiple eval metrics have been passed: 'validation_1-logloss' will be used for
early stopping.

Will train until validation_1-logloss hasn't improved in 15 rounds.
[1]     validation_0-error:0.04020      validation_0-logloss:0.67700
validation_1-error:0.09357      validation_1-logloss:0.67927
[2]     validation_0-error:0.03266      validation_0-logloss:0.66916
validation_1-error:0.08772      validation_1-logloss:0.67272
[3]     validation_0-error:0.01759      validation_0-logloss:0.66141
validation_1-error:0.06433      validation_1-logloss:0.66598
[4]     validation_0-error:0.02261      validation_0-logloss:0.65450
validation_1-error:0.07018      validation_1-logloss:0.65992
[5]     validation_0-error:0.02764      validation_0-logloss:0.64719
validation_1-error:0.06433      validation_1-logloss:0.65319
[6]     validation_0-error:0.02513      validation_0-logloss:0.64023
validation_1-error:0.06433      validation_1-logloss:0.64709
[7]     validation_0-error:0.02764      validation_0-logloss:0.63275
validation_1-error:0.06433      validation_1-logloss:0.64080
[8]     validation_0-error:0.01759      validation_0-logloss:0.62582
validation_1-error:0.07018      validation_1-logloss:0.63502
[9]     validation_0-error:0.02010      validation_0-logloss:0.61877
validation_1-error:0.05848      validation_1-logloss:0.62931
[10]    validation_0-error:0.02010      validation_0-logloss:0.61196
validation_1-error:0.05848      validation_1-logloss:0.62350
[11]    validation_0-error:0.02010      validation_0-logloss:0.60515
validation_1-error:0.07018      validation_1-logloss:0.61764
[12]    validation_0-error:0.02010      validation_0-logloss:0.59876
validation_1-error:0.07018      validation_1-logloss:0.61209
[13]    validation_0-error:0.02261      validation_0-logloss:0.59242
validation_1-error:0.07602      validation_1-logloss:0.60659
[14]    validation_0-error:0.02261      validation_0-logloss:0.58579
validation_1-error:0.08187      validation_1-logloss:0.60129
[15]    validation_0-error:0.02261      validation_0-logloss:0.57973
validation_1-error:0.08187      validation_1-logloss:0.59647
[16]    validation_0-error:0.02261      validation_0-logloss:0.57342
validation_1-error:0.07602      validation_1-logloss:0.59117
[17]    validation_0-error:0.02513      validation_0-logloss:0.56714
validation_1-error:0.08187      validation_1-logloss:0.58574
```

```
[18]    validation_0-error:0.02261      validation_0-logloss:0.56090
validation_1-error:0.08772      validation_1-logloss:0.58057
[19]    validation_0-error:0.02261      validation_0-logloss:0.55550
validation_1-error:0.08187      validation_1-logloss:0.57545
[20]    validation_0-error:0.02513      validation_0-logloss:0.54965
validation_1-error:0.08772      validation_1-logloss:0.57047
[21]    validation_0-error:0.02513      validation_0-logloss:0.54346
validation_1-error:0.09357      validation_1-logloss:0.56517
[22]    validation_0-error:0.02513      validation_0-logloss:0.53834
validation_1-error:0.09357      validation_1-logloss:0.56108
[23]    validation_0-error:0.02513      validation_0-logloss:0.53293
validation_1-error:0.09357      validation_1-logloss:0.55608
[24]    validation_0-error:0.02513      validation_0-logloss:0.52762
validation_1-error:0.08187      validation_1-logloss:0.55141
[25]    validation_0-error:0.02261      validation_0-logloss:0.52203
validation_1-error:0.08187      validation_1-logloss:0.54679
[26]    validation_0-error:0.02261      validation_0-logloss:0.51689
validation_1-error:0.08187      validation_1-logloss:0.54188
[27]    validation_0-error:0.02261      validation_0-logloss:0.51174
validation_1-error:0.07602      validation_1-logloss:0.53745
[28]    validation_0-error:0.02261      validation_0-logloss:0.50668
validation_1-error:0.07602      validation_1-logloss:0.53312
[29]    validation_0-error:0.02261      validation_0-logloss:0.50193
validation_1-error:0.07018      validation_1-logloss:0.52866
[30]    validation_0-error:0.02010      validation_0-logloss:0.49672
validation_1-error:0.07602      validation_1-logloss:0.52443
[31]    validation_0-error:0.02010      validation_0-logloss:0.49203
validation_1-error:0.07018      validation_1-logloss:0.52050
[32]    validation_0-error:0.02010      validation_0-logloss:0.48730
validation_1-error:0.07602      validation_1-logloss:0.51636
[33]    validation_0-error:0.02010      validation_0-logloss:0.48259
validation_1-error:0.07018      validation_1-logloss:0.51180
[34]    validation_0-error:0.02010      validation_0-logloss:0.47788
validation_1-error:0.07018      validation_1-logloss:0.50779
[35]    validation_0-error:0.02010      validation_0-logloss:0.47321
validation_1-error:0.07018      validation_1-logloss:0.50402
[36]    validation_0-error:0.02010      validation_0-logloss:0.46840
validation_1-error:0.07018      validation_1-logloss:0.50014
[37]    validation_0-error:0.02010      validation_0-logloss:0.46425
validation_1-error:0.07018      validation_1-logloss:0.49665
[38]    validation_0-error:0.02010      validation_0-logloss:0.45977
validation_1-error:0.07018      validation_1-logloss:0.49282
[39]    validation_0-error:0.02010      validation_0-logloss:0.45525
validation_1-error:0.07018      validation_1-logloss:0.48901
[40]    validation_0-error:0.02010      validation_0-logloss:0.45075
validation_1-error:0.07602      validation_1-logloss:0.48533
[41]    validation_0-error:0.02010      validation_0-logloss:0.44675
validation_1-error:0.07602      validation_1-logloss:0.48141
```

```
[42]    validation_0-error:0.02010      validation_0-logloss:0.44253
validation_1-error:0.07602      validation_1-logloss:0.47775
[43]    validation_0-error:0.02010      validation_0-logloss:0.43847
validation_1-error:0.07018      validation_1-logloss:0.47449
[44]    validation_0-error:0.02010      validation_0-logloss:0.43426
validation_1-error:0.07018      validation_1-logloss:0.47086
[45]    validation_0-error:0.02010      validation_0-logloss:0.43013
validation_1-error:0.07018      validation_1-logloss:0.46742
[46]    validation_0-error:0.02261      validation_0-logloss:0.42617
validation_1-error:0.07018      validation_1-logloss:0.46385
[47]    validation_0-error:0.02513      validation_0-logloss:0.42204
validation_1-error:0.07602      validation_1-logloss:0.46060
[48]    validation_0-error:0.02513      validation_0-logloss:0.41827
validation_1-error:0.07018      validation_1-logloss:0.45749
[49]    validation_0-error:0.02513      validation_0-logloss:0.41430
validation_1-error:0.07018      validation_1-logloss:0.45443
[50]    validation_0-error:0.02764      validation_0-logloss:0.41041
validation_1-error:0.07018      validation_1-logloss:0.45110
[51]    validation_0-error:0.02764      validation_0-logloss:0.40643
validation_1-error:0.07018      validation_1-logloss:0.44797
[52]    validation_0-error:0.02261      validation_0-logloss:0.40285
validation_1-error:0.07018      validation_1-logloss:0.44518
[53]    validation_0-error:0.02261      validation_0-logloss:0.39921
validation_1-error:0.07602      validation_1-logloss:0.44205
[54]    validation_0-error:0.02010      validation_0-logloss:0.39539
validation_1-error:0.07602      validation_1-logloss:0.43903
[55]    validation_0-error:0.02010      validation_0-logloss:0.39183
validation_1-error:0.07602      validation_1-logloss:0.43611
[56]    validation_0-error:0.02010      validation_0-logloss:0.38825
validation_1-error:0.07602      validation_1-logloss:0.43315
[57]    validation_0-error:0.02010      validation_0-logloss:0.38470
validation_1-error:0.07602      validation_1-logloss:0.43007
[58]    validation_0-error:0.02010      validation_0-logloss:0.38127
validation_1-error:0.07018      validation_1-logloss:0.42692
[59]    validation_0-error:0.02010      validation_0-logloss:0.37781
validation_1-error:0.07018      validation_1-logloss:0.42393
[60]    validation_0-error:0.02010      validation_0-logloss:0.37441
validation_1-error:0.07018      validation_1-logloss:0.42124
[61]    validation_0-error:0.02010      validation_0-logloss:0.37095
validation_1-error:0.07018      validation_1-logloss:0.41838
[62]    validation_0-error:0.02010      validation_0-logloss:0.36765
validation_1-error:0.07602      validation_1-logloss:0.41569
[63]    validation_0-error:0.02010      validation_0-logloss:0.36429
validation_1-error:0.07602      validation_1-logloss:0.41305
[64]    validation_0-error:0.02010      validation_0-logloss:0.36101
validation_1-error:0.07602      validation_1-logloss:0.41053
[65]    validation_0-error:0.02010      validation_0-logloss:0.35803
validation_1-error:0.07602      validation_1-logloss:0.40791
```

```
[66]    validation_0-error:0.02010      validation_0-logloss:0.35493
validation_1-error:0.07018      validation_1-logloss:0.40533
[67]    validation_0-error:0.02010      validation_0-logloss:0.35181
validation_1-error:0.07602      validation_1-logloss:0.40270
[68]    validation_0-error:0.02010      validation_0-logloss:0.34882
validation_1-error:0.07018      validation_1-logloss:0.39997
[69]    validation_0-error:0.02010      validation_0-logloss:0.34580
validation_1-error:0.07018      validation_1-logloss:0.39724
[70]    validation_0-error:0.02010      validation_0-logloss:0.34258
validation_1-error:0.07018      validation_1-logloss:0.39463
[71]    validation_0-error:0.02010      validation_0-logloss:0.33955
validation_1-error:0.07018      validation_1-logloss:0.39225
[72]    validation_0-error:0.02010      validation_0-logloss:0.33670
validation_1-error:0.07018      validation_1-logloss:0.38994
[73]    validation_0-error:0.02010      validation_0-logloss:0.33397
validation_1-error:0.07018      validation_1-logloss:0.38747
[74]    validation_0-error:0.02010      validation_0-logloss:0.33126
validation_1-error:0.07018      validation_1-logloss:0.38510
[75]    validation_0-error:0.02010      validation_0-logloss:0.32848
validation_1-error:0.07018      validation_1-logloss:0.38269
[76]    validation_0-error:0.02513      validation_0-logloss:0.32567
validation_1-error:0.07018      validation_1-logloss:0.38047
[77]    validation_0-error:0.02513      validation_0-logloss:0.32280
validation_1-error:0.07602      validation_1-logloss:0.37814
[78]    validation_0-error:0.02261      validation_0-logloss:0.32019
validation_1-error:0.07018      validation_1-logloss:0.37590
[79]    validation_0-error:0.02261      validation_0-logloss:0.31766
validation_1-error:0.07602      validation_1-logloss:0.37351
[80]    validation_0-error:0.02513      validation_0-logloss:0.31483
validation_1-error:0.07602      validation_1-logloss:0.37151
[81]    validation_0-error:0.02261      validation_0-logloss:0.31224
validation_1-error:0.07602      validation_1-logloss:0.36927
[82]    validation_0-error:0.02261      validation_0-logloss:0.30974
validation_1-error:0.07602      validation_1-logloss:0.36664
[83]    validation_0-error:0.02261      validation_0-logloss:0.30733
validation_1-error:0.07602      validation_1-logloss:0.36454
[84]    validation_0-error:0.02261      validation_0-logloss:0.30475
validation_1-error:0.07018      validation_1-logloss:0.36227
[85]    validation_0-error:0.02010      validation_0-logloss:0.30212
validation_1-error:0.07602      validation_1-logloss:0.36026
[86]    validation_0-error:0.02010      validation_0-logloss:0.29967
validation_1-error:0.07018      validation_1-logloss:0.35818
[87]    validation_0-error:0.02010      validation_0-logloss:0.29724
validation_1-error:0.07018      validation_1-logloss:0.35619
[88]    validation_0-error:0.02010      validation_0-logloss:0.29474
validation_1-error:0.07018      validation_1-logloss:0.35423
[89]    validation_0-error:0.02010      validation_0-logloss:0.29247
validation_1-error:0.07018      validation_1-logloss:0.35239
```

```
[90]    validation_0-error:0.02010      validation_0-logloss:0.29042
validation_1-error:0.07018      validation_1-logloss:0.35066
[91]    validation_0-error:0.02010      validation_0-logloss:0.28798
validation_1-error:0.07018      validation_1-logloss:0.34862
[92]    validation_0-error:0.02010      validation_0-logloss:0.28588
validation_1-error:0.07018      validation_1-logloss:0.34690
[93]    validation_0-error:0.02010      validation_0-logloss:0.28341
validation_1-error:0.07018      validation_1-logloss:0.34491
[94]    validation_0-error:0.02010      validation_0-logloss:0.28109
validation_1-error:0.07018      validation_1-logloss:0.34314
[95]    validation_0-error:0.02010      validation_0-logloss:0.27900
validation_1-error:0.07018      validation_1-logloss:0.34156
[96]    validation_0-error:0.02010      validation_0-logloss:0.27693
validation_1-error:0.07018      validation_1-logloss:0.33983
[97]    validation_0-error:0.02010      validation_0-logloss:0.27469
validation_1-error:0.07018      validation_1-logloss:0.33791
[98]    validation_0-error:0.02010      validation_0-logloss:0.27259
validation_1-error:0.07018      validation_1-logloss:0.33661
[99]    validation_0-error:0.02010      validation_0-logloss:0.27047
validation_1-error:0.07018      validation_1-logloss:0.33485
[100]   validation_0-error:0.02010      validation_0-logloss:0.26829
validation_1-error:0.07018      validation_1-logloss:0.33303
[101]   validation_0-error:0.02010      validation_0-logloss:0.26610
validation_1-error:0.07018      validation_1-logloss:0.33126
[102]   validation_0-error:0.02010      validation_0-logloss:0.26405
validation_1-error:0.07018      validation_1-logloss:0.32954
[103]   validation_0-error:0.02010      validation_0-logloss:0.26198
validation_1-error:0.07018      validation_1-logloss:0.32776
[104]   validation_0-error:0.02010      validation_0-logloss:0.26005
validation_1-error:0.07018      validation_1-logloss:0.32614
[105]   validation_0-error:0.02010      validation_0-logloss:0.25819
validation_1-error:0.07018      validation_1-logloss:0.32417
[106]   validation_0-error:0.02010      validation_0-logloss:0.25623
validation_1-error:0.07018      validation_1-logloss:0.32251
[107]   validation_0-error:0.02010      validation_0-logloss:0.25420
validation_1-error:0.07018      validation_1-logloss:0.32106
[108]   validation_0-error:0.02010      validation_0-logloss:0.25243
validation_1-error:0.07018      validation_1-logloss:0.31959
[109]   validation_0-error:0.02010      validation_0-logloss:0.25046
validation_1-error:0.07018      validation_1-logloss:0.31814
[110]   validation_0-error:0.02010      validation_0-logloss:0.24870
validation_1-error:0.07018      validation_1-logloss:0.31671
[111]   validation_0-error:0.02010      validation_0-logloss:0.24688
validation_1-error:0.07018      validation_1-logloss:0.31492
[112]   validation_0-error:0.02010      validation_0-logloss:0.24493
validation_1-error:0.07018      validation_1-logloss:0.31355
[113]   validation_0-error:0.02010      validation_0-logloss:0.24318
validation_1-error:0.07018      validation_1-logloss:0.31203
```

```
[114]   validation_0-error:0.02010      validation_0-logloss:0.24137
validation_1-error:0.07018      validation_1-logloss:0.31026
[115]   validation_0-error:0.02010      validation_0-logloss:0.23944
validation_1-error:0.07018      validation_1-logloss:0.30889
[116]   validation_0-error:0.02010      validation_0-logloss:0.23751
validation_1-error:0.07018      validation_1-logloss:0.30743
[117]   validation_0-error:0.02010      validation_0-logloss:0.23575
validation_1-error:0.07018      validation_1-logloss:0.30629
[118]   validation_0-error:0.02010      validation_0-logloss:0.23401
validation_1-error:0.07018      validation_1-logloss:0.30504
[119]   validation_0-error:0.02010      validation_0-logloss:0.23241
validation_1-error:0.07018      validation_1-logloss:0.30369
[120]   validation_0-error:0.02010      validation_0-logloss:0.23096
validation_1-error:0.07018      validation_1-logloss:0.30241
[121]   validation_0-error:0.02010      validation_0-logloss:0.22915
validation_1-error:0.07018      validation_1-logloss:0.30119
[122]   validation_0-error:0.02010      validation_0-logloss:0.22763
validation_1-error:0.07018      validation_1-logloss:0.29984
[123]   validation_0-error:0.02010      validation_0-logloss:0.22602
validation_1-error:0.07018      validation_1-logloss:0.29859
[124]   validation_0-error:0.02010      validation_0-logloss:0.22453
validation_1-error:0.07018      validation_1-logloss:0.29712
[125]   validation_0-error:0.02010      validation_0-logloss:0.22304
validation_1-error:0.07018      validation_1-logloss:0.29563
[126]   validation_0-error:0.02010      validation_0-logloss:0.22139
validation_1-error:0.07018      validation_1-logloss:0.29450
[127]   validation_0-error:0.02010      validation_0-logloss:0.21986
validation_1-error:0.07018      validation_1-logloss:0.29340
[128]   validation_0-error:0.02010      validation_0-logloss:0.21818
validation_1-error:0.07018      validation_1-logloss:0.29205
[129]   validation_0-error:0.02010      validation_0-logloss:0.21656
validation_1-error:0.07018      validation_1-logloss:0.29089
[130]   validation_0-error:0.02010      validation_0-logloss:0.21504
validation_1-error:0.07018      validation_1-logloss:0.28961
[131]   validation_0-error:0.02010      validation_0-logloss:0.21372
validation_1-error:0.07018      validation_1-logloss:0.28838
[132]   validation_0-error:0.02010      validation_0-logloss:0.21227
validation_1-error:0.07018      validation_1-logloss:0.28714
[133]   validation_0-error:0.02010      validation_0-logloss:0.21070
validation_1-error:0.07018      validation_1-logloss:0.28602
[134]   validation_0-error:0.02010      validation_0-logloss:0.20922
validation_1-error:0.07018      validation_1-logloss:0.28494
[135]   validation_0-error:0.02010      validation_0-logloss:0.20774
validation_1-error:0.07018      validation_1-logloss:0.28356
[136]   validation_0-error:0.02010      validation_0-logloss:0.20629
validation_1-error:0.07018      validation_1-logloss:0.28264
[137]   validation_0-error:0.02010      validation_0-logloss:0.20485
validation_1-error:0.07018      validation_1-logloss:0.28122
```

```
[138]   validation_0-error:0.02010      validation_0-logloss:0.20352
validation_1-error:0.07018      validation_1-logloss:0.27988
[139]   validation_0-error:0.02010      validation_0-logloss:0.20215
validation_1-error:0.06433      validation_1-logloss:0.27892
[140]   validation_0-error:0.02010      validation_0-logloss:0.20081
validation_1-error:0.07018      validation_1-logloss:0.27800
[141]   validation_0-error:0.02010      validation_0-logloss:0.19954
validation_1-error:0.07018      validation_1-logloss:0.27670
[142]   validation_0-error:0.02010      validation_0-logloss:0.19821
validation_1-error:0.07018      validation_1-logloss:0.27581
[143]   validation_0-error:0.02010      validation_0-logloss:0.19699
validation_1-error:0.07018      validation_1-logloss:0.27483
[144]   validation_0-error:0.02010      validation_0-logloss:0.19557
validation_1-error:0.06433      validation_1-logloss:0.27376
[145]   validation_0-error:0.02010      validation_0-logloss:0.19421
validation_1-error:0.06433      validation_1-logloss:0.27282
[146]   validation_0-error:0.02010      validation_0-logloss:0.19292
validation_1-error:0.06433      validation_1-logloss:0.27188
[147]   validation_0-error:0.02010      validation_0-logloss:0.19162
validation_1-error:0.06433      validation_1-logloss:0.27090
[148]   validation_0-error:0.02010      validation_0-logloss:0.19032
validation_1-error:0.06433      validation_1-logloss:0.27002
[149]   validation_0-error:0.02010      validation_0-logloss:0.18909
validation_1-error:0.06433      validation_1-logloss:0.26915
[150]   validation_0-error:0.02010      validation_0-logloss:0.18780
validation_1-error:0.06433      validation_1-logloss:0.26804
[151]   validation_0-error:0.02010      validation_0-logloss:0.18674
validation_1-error:0.06433      validation_1-logloss:0.26691
[152]   validation_0-error:0.02010      validation_0-logloss:0.18554
validation_1-error:0.06433      validation_1-logloss:0.26592
[153]   validation_0-error:0.02010      validation_0-logloss:0.18437
validation_1-error:0.06433      validation_1-logloss:0.26491
[154]   validation_0-error:0.02010      validation_0-logloss:0.18311
validation_1-error:0.07018      validation_1-logloss:0.26378
[155]   validation_0-error:0.02010      validation_0-logloss:0.18197
validation_1-error:0.07018      validation_1-logloss:0.26284
[156]   validation_0-error:0.02010      validation_0-logloss:0.18077
validation_1-error:0.07018      validation_1-logloss:0.26199
[157]   validation_0-error:0.02010      validation_0-logloss:0.17972
validation_1-error:0.07018      validation_1-logloss:0.26089
[158]   validation_0-error:0.02010      validation_0-logloss:0.17849
validation_1-error:0.07018      validation_1-logloss:0.26028
[159]   validation_0-error:0.02010      validation_0-logloss:0.17735
validation_1-error:0.07018      validation_1-logloss:0.25931
[160]   validation_0-error:0.02010      validation_0-logloss:0.17640
validation_1-error:0.07018      validation_1-logloss:0.25866
[161]   validation_0-error:0.02010      validation_0-logloss:0.17519
validation_1-error:0.07018      validation_1-logloss:0.25757
```

```
[162]   validation_0-error:0.02010      validation_0-logloss:0.17404
validation_1-error:0.07018      validation_1-logloss:0.25669
[163]   validation_0-error:0.02010      validation_0-logloss:0.17287
validation_1-error:0.07018      validation_1-logloss:0.25553
[164]   validation_0-error:0.02010      validation_0-logloss:0.17191
validation_1-error:0.07018      validation_1-logloss:0.25453
[165]   validation_0-error:0.02010      validation_0-logloss:0.17097
validation_1-error:0.07018      validation_1-logloss:0.25379
[166]   validation_0-error:0.02010      validation_0-logloss:0.17005
validation_1-error:0.07018      validation_1-logloss:0.25313
[167]   validation_0-error:0.02010      validation_0-logloss:0.16893
validation_1-error:0.07018      validation_1-logloss:0.25241
[168]   validation_0-error:0.02010      validation_0-logloss:0.16796
validation_1-error:0.07018      validation_1-logloss:0.25152
[169]   validation_0-error:0.02010      validation_0-logloss:0.16689
validation_1-error:0.07018      validation_1-logloss:0.25081
[170]   validation_0-error:0.02010      validation_0-logloss:0.16586
validation_1-error:0.07018      validation_1-logloss:0.25007
[171]   validation_0-error:0.02010      validation_0-logloss:0.16478
validation_1-error:0.07018      validation_1-logloss:0.24919
[172]   validation_0-error:0.02010      validation_0-logloss:0.16379
validation_1-error:0.07018      validation_1-logloss:0.24871
[173]   validation_0-error:0.02010      validation_0-logloss:0.16280
validation_1-error:0.07018      validation_1-logloss:0.24757
[174]   validation_0-error:0.02010      validation_0-logloss:0.16181
validation_1-error:0.06433      validation_1-logloss:0.24642
[175]   validation_0-error:0.02010      validation_0-logloss:0.16090
validation_1-error:0.06433      validation_1-logloss:0.24562
[176]   validation_0-error:0.02010      validation_0-logloss:0.15991
validation_1-error:0.06433      validation_1-logloss:0.24464
[177]   validation_0-error:0.02010      validation_0-logloss:0.15898
validation_1-error:0.06433      validation_1-logloss:0.24413
[178]   validation_0-error:0.02010      validation_0-logloss:0.15809
validation_1-error:0.06433      validation_1-logloss:0.24350
[179]   validation_0-error:0.02010      validation_0-logloss:0.15707
validation_1-error:0.06433      validation_1-logloss:0.24285
[180]   validation_0-error:0.02010      validation_0-logloss:0.15618
validation_1-error:0.06433      validation_1-logloss:0.24227
[181]   validation_0-error:0.02010      validation_0-logloss:0.15537
validation_1-error:0.07018      validation_1-logloss:0.24154
[182]   validation_0-error:0.02010      validation_0-logloss:0.15448
validation_1-error:0.07018      validation_1-logloss:0.24098
[183]   validation_0-error:0.02010      validation_0-logloss:0.15355
validation_1-error:0.06433      validation_1-logloss:0.24017
[184]   validation_0-error:0.02010      validation_0-logloss:0.15257
validation_1-error:0.06433      validation_1-logloss:0.23952
[185]   validation_0-error:0.02010      validation_0-logloss:0.15161
validation_1-error:0.06433      validation_1-logloss:0.23913
```

```
[186]   validation_0-error:0.02010      validation_0-logloss:0.15054
validation_1-error:0.06433      validation_1-logloss:0.23822
[187]   validation_0-error:0.02010      validation_0-logloss:0.14961
validation_1-error:0.06433      validation_1-logloss:0.23781
[188]   validation_0-error:0.02010      validation_0-logloss:0.14878
validation_1-error:0.06433      validation_1-logloss:0.23725
[189]   validation_0-error:0.02010      validation_0-logloss:0.14787
validation_1-error:0.06433      validation_1-logloss:0.23637
[190]   validation_0-error:0.02010      validation_0-logloss:0.14697
validation_1-error:0.06433      validation_1-logloss:0.23588
[191]   validation_0-error:0.02010      validation_0-logloss:0.14607
validation_1-error:0.06433      validation_1-logloss:0.23499
[192]   validation_0-error:0.02010      validation_0-logloss:0.14519
validation_1-error:0.06433      validation_1-logloss:0.23435
[193]   validation_0-error:0.02010      validation_0-logloss:0.14444
validation_1-error:0.06433      validation_1-logloss:0.23367
[194]   validation_0-error:0.02010      validation_0-logloss:0.14356
validation_1-error:0.06433      validation_1-logloss:0.23330
[195]   validation_0-error:0.02010      validation_0-logloss:0.14256
validation_1-error:0.06433      validation_1-logloss:0.23231
[196]   validation_0-error:0.02010      validation_0-logloss:0.14172
validation_1-error:0.06433      validation_1-logloss:0.23179
[197]   validation_0-error:0.02010      validation_0-logloss:0.14092
validation_1-error:0.06433      validation_1-logloss:0.23119
[198]   validation_0-error:0.02010      validation_0-logloss:0.14016
validation_1-error:0.06433      validation_1-logloss:0.23066
[199]   validation_0-error:0.02010      validation_0-logloss:0.13946
validation_1-error:0.06433      validation_1-logloss:0.23031
[200]   validation_0-error:0.02010      validation_0-logloss:0.13863
validation_1-error:0.06433      validation_1-logloss:0.22984
[201]   validation_0-error:0.02010      validation_0-logloss:0.13792
validation_1-error:0.06433      validation_1-logloss:0.22961
[202]   validation_0-error:0.02010      validation_0-logloss:0.13725
validation_1-error:0.06433      validation_1-logloss:0.22922
[203]   validation_0-error:0.02010      validation_0-logloss:0.13654
validation_1-error:0.06433      validation_1-logloss:0.22818
[204]   validation_0-error:0.02010      validation_0-logloss:0.13583
validation_1-error:0.06433      validation_1-logloss:0.22781
[205]   validation_0-error:0.02010      validation_0-logloss:0.13514
validation_1-error:0.06433      validation_1-logloss:0.22741
[206]   validation_0-error:0.02010      validation_0-logloss:0.13456
validation_1-error:0.06433      validation_1-logloss:0.22649
[207]   validation_0-error:0.02010      validation_0-logloss:0.13382
validation_1-error:0.06433      validation_1-logloss:0.22607
[208]   validation_0-error:0.02010      validation_0-logloss:0.13314
validation_1-error:0.06433      validation_1-logloss:0.22569
[209]   validation_0-error:0.02010      validation_0-logloss:0.13244
validation_1-error:0.06433      validation_1-logloss:0.22526
```

```
[210]   validation_0-error:0.02010      validation_0-logloss:0.13179
validation_1-error:0.06433      validation_1-logloss:0.22464
[211]   validation_0-error:0.02010      validation_0-logloss:0.13093
validation_1-error:0.06433      validation_1-logloss:0.22379
[212]   validation_0-error:0.02010      validation_0-logloss:0.13037
validation_1-error:0.06433      validation_1-logloss:0.22380
[213]   validation_0-error:0.02010      validation_0-logloss:0.12959
validation_1-error:0.06433      validation_1-logloss:0.22341
[214]   validation_0-error:0.01759      validation_0-logloss:0.12898
validation_1-error:0.06433      validation_1-logloss:0.22297
[215]   validation_0-error:0.01759      validation_0-logloss:0.12828
validation_1-error:0.06433      validation_1-logloss:0.22230
[216]   validation_0-error:0.01759      validation_0-logloss:0.12764
validation_1-error:0.06433      validation_1-logloss:0.22198
[217]   validation_0-error:0.01759      validation_0-logloss:0.12707
validation_1-error:0.06433      validation_1-logloss:0.22153
[218]   validation_0-error:0.01759      validation_0-logloss:0.12634
validation_1-error:0.06433      validation_1-logloss:0.22106
[219]   validation_0-error:0.01759      validation_0-logloss:0.12570
validation_1-error:0.06433      validation_1-logloss:0.22086
[220]   validation_0-error:0.01759      validation_0-logloss:0.12510
validation_1-error:0.06433      validation_1-logloss:0.22027
[221]   validation_0-error:0.01759      validation_0-logloss:0.12452
validation_1-error:0.06433      validation_1-logloss:0.21981
[222]   validation_0-error:0.01759      validation_0-logloss:0.12397
validation_1-error:0.06433      validation_1-logloss:0.21912
[223]   validation_0-error:0.01759      validation_0-logloss:0.12344
validation_1-error:0.06433      validation_1-logloss:0.21869
[224]   validation_0-error:0.01759      validation_0-logloss:0.12274
validation_1-error:0.06433      validation_1-logloss:0.21831
[225]   validation_0-error:0.01759      validation_0-logloss:0.12221
validation_1-error:0.06433      validation_1-logloss:0.21777
[226]   validation_0-error:0.01759      validation_0-logloss:0.12157
validation_1-error:0.06433      validation_1-logloss:0.21730
[227]   validation_0-error:0.01759      validation_0-logloss:0.12081
validation_1-error:0.06433      validation_1-logloss:0.21673
[228]   validation_0-error:0.01759      validation_0-logloss:0.12022
validation_1-error:0.06433      validation_1-logloss:0.21611
[229]   validation_0-error:0.01759      validation_0-logloss:0.11972
validation_1-error:0.06433      validation_1-logloss:0.21583
[230]   validation_0-error:0.01759      validation_0-logloss:0.11909
validation_1-error:0.06433      validation_1-logloss:0.21551
[231]   validation_0-error:0.01759      validation_0-logloss:0.11849
validation_1-error:0.06433      validation_1-logloss:0.21489
[232]   validation_0-error:0.01759      validation_0-logloss:0.11789
validation_1-error:0.06433      validation_1-logloss:0.21430
[233]   validation_0-error:0.01759      validation_0-logloss:0.11710
validation_1-error:0.06433      validation_1-logloss:0.21358
```

```
[234]   validation_0-error:0.01759      validation_0-logloss:0.11658
validation_1-error:0.06433      validation_1-logloss:0.21320
[235]   validation_0-error:0.01759      validation_0-logloss:0.11602
validation_1-error:0.06433      validation_1-logloss:0.21271
[236]   validation_0-error:0.01759      validation_0-logloss:0.11554
validation_1-error:0.06433      validation_1-logloss:0.21230
[237]   validation_0-error:0.01759      validation_0-logloss:0.11506
validation_1-error:0.06433      validation_1-logloss:0.21190
[238]   validation_0-error:0.01759      validation_0-logloss:0.11449
validation_1-error:0.06433      validation_1-logloss:0.21142
[239]   validation_0-error:0.01759      validation_0-logloss:0.11401
validation_1-error:0.06433      validation_1-logloss:0.21116
[240]   validation_0-error:0.01759      validation_0-logloss:0.11336
validation_1-error:0.06433      validation_1-logloss:0.21061
[241]   validation_0-error:0.01759      validation_0-logloss:0.11288
validation_1-error:0.06433      validation_1-logloss:0.21026
[242]   validation_0-error:0.01759      validation_0-logloss:0.11224
validation_1-error:0.06433      validation_1-logloss:0.21008
[243]   validation_0-error:0.01759      validation_0-logloss:0.11171
validation_1-error:0.06433      validation_1-logloss:0.20982
[244]   validation_0-error:0.01759      validation_0-logloss:0.11114
validation_1-error:0.06433      validation_1-logloss:0.20935
[245]   validation_0-error:0.01759      validation_0-logloss:0.11066
validation_1-error:0.06433      validation_1-logloss:0.20913
[246]   validation_0-error:0.01759      validation_0-logloss:0.11014
validation_1-error:0.06433      validation_1-logloss:0.20880
[247]   validation_0-error:0.01759      validation_0-logloss:0.10971
validation_1-error:0.05848      validation_1-logloss:0.20836
[248]   validation_0-error:0.01759      validation_0-logloss:0.10921
validation_1-error:0.05848      validation_1-logloss:0.20793
[249]   validation_0-error:0.01759      validation_0-logloss:0.10867
validation_1-error:0.05848      validation_1-logloss:0.20739
[250]   validation_0-error:0.01759      validation_0-logloss:0.10813
validation_1-error:0.05848      validation_1-logloss:0.20691
[251]   validation_0-error:0.01759      validation_0-logloss:0.10753
validation_1-error:0.05848      validation_1-logloss:0.20649
[252]   validation_0-error:0.01759      validation_0-logloss:0.10703
validation_1-error:0.05848      validation_1-logloss:0.20613
[253]   validation_0-error:0.01759      validation_0-logloss:0.10662
validation_1-error:0.05848      validation_1-logloss:0.20558
[254]   validation_0-error:0.01759      validation_0-logloss:0.10616
validation_1-error:0.05848      validation_1-logloss:0.20539
[255]   validation_0-error:0.01759      validation_0-logloss:0.10565
validation_1-error:0.05848      validation_1-logloss:0.20508
[256]   validation_0-error:0.01759      validation_0-logloss:0.10524
validation_1-error:0.06433      validation_1-logloss:0.20498
[257]   validation_0-error:0.01759      validation_0-logloss:0.10478
validation_1-error:0.06433      validation_1-logloss:0.20483
```

```
[258]   validation_0-error:0.01759      validation_0-logloss:0.10432
validation_1-error:0.05848      validation_1-logloss:0.20421
[259]   validation_0-error:0.01759      validation_0-logloss:0.10382
validation_1-error:0.05848      validation_1-logloss:0.20392
[260]   validation_0-error:0.01759      validation_0-logloss:0.10329
validation_1-error:0.05848      validation_1-logloss:0.20351
[261]   validation_0-error:0.01759      validation_0-logloss:0.10286
validation_1-error:0.05848      validation_1-logloss:0.20339
[262]   validation_0-error:0.01759      validation_0-logloss:0.10243
validation_1-error:0.05848      validation_1-logloss:0.20317
[263]   validation_0-error:0.01759      validation_0-logloss:0.10198
validation_1-error:0.05848      validation_1-logloss:0.20273
[264]   validation_0-error:0.01759      validation_0-logloss:0.10160
validation_1-error:0.05848      validation_1-logloss:0.20256
[265]   validation_0-error:0.01759      validation_0-logloss:0.10122
validation_1-error:0.05848      validation_1-logloss:0.20245
[266]   validation_0-error:0.01759      validation_0-logloss:0.10081
validation_1-error:0.05848      validation_1-logloss:0.20206
[267]   validation_0-error:0.01759      validation_0-logloss:0.10035
validation_1-error:0.05848      validation_1-logloss:0.20171
[268]   validation_0-error:0.01759      validation_0-logloss:0.09987
validation_1-error:0.05848      validation_1-logloss:0.20134
[269]   validation_0-error:0.01759      validation_0-logloss:0.09947
validation_1-error:0.05848      validation_1-logloss:0.20117
[270]   validation_0-error:0.01759      validation_0-logloss:0.09901
validation_1-error:0.05848      validation_1-logloss:0.20077
[271]   validation_0-error:0.01759      validation_0-logloss:0.09857
validation_1-error:0.05848      validation_1-logloss:0.20056
[272]   validation_0-error:0.01759      validation_0-logloss:0.09815
validation_1-error:0.05848      validation_1-logloss:0.20039
[273]   validation_0-error:0.01759      validation_0-logloss:0.09772
validation_1-error:0.05848      validation_1-logloss:0.20010
[274]   validation_0-error:0.01759      validation_0-logloss:0.09728
validation_1-error:0.05848      validation_1-logloss:0.19966
[275]   validation_0-error:0.01759      validation_0-logloss:0.09689
validation_1-error:0.05848      validation_1-logloss:0.19951
[276]   validation_0-error:0.01759      validation_0-logloss:0.09650
validation_1-error:0.05848      validation_1-logloss:0.19933
[277]   validation_0-error:0.01759      validation_0-logloss:0.09606
validation_1-error:0.05848      validation_1-logloss:0.19907
[278]   validation_0-error:0.01759      validation_0-logloss:0.09568
validation_1-error:0.05848      validation_1-logloss:0.19882
[279]   validation_0-error:0.01759      validation_0-logloss:0.09527
validation_1-error:0.05848      validation_1-logloss:0.19847
[280]   validation_0-error:0.01759      validation_0-logloss:0.09487
validation_1-error:0.05848      validation_1-logloss:0.19835
[281]   validation_0-error:0.01759      validation_0-logloss:0.09440
validation_1-error:0.05848      validation_1-logloss:0.19815
```

```
[282]   validation_0-error:0.01759      validation_0-logloss:0.09398
validation_1-error:0.05848      validation_1-logloss:0.19795
[283]   validation_0-error:0.01759      validation_0-logloss:0.09363
validation_1-error:0.05848      validation_1-logloss:0.19779
[284]   validation_0-error:0.01759      validation_0-logloss:0.09321
validation_1-error:0.05848      validation_1-logloss:0.19771
[285]   validation_0-error:0.01759      validation_0-logloss:0.09284
validation_1-error:0.05848      validation_1-logloss:0.19763
[286]   validation_0-error:0.01759      validation_0-logloss:0.09245
validation_1-error:0.05848      validation_1-logloss:0.19737
[287]   validation_0-error:0.01759      validation_0-logloss:0.09209
validation_1-error:0.05848      validation_1-logloss:0.19716
[288]   validation_0-error:0.01759      validation_0-logloss:0.09165
validation_1-error:0.05848      validation_1-logloss:0.19684
[289]   validation_0-error:0.01759      validation_0-logloss:0.09128
validation_1-error:0.05848      validation_1-logloss:0.19643
[290]   validation_0-error:0.01759      validation_0-logloss:0.09092
validation_1-error:0.05848      validation_1-logloss:0.19613
[291]   validation_0-error:0.01759      validation_0-logloss:0.09051
validation_1-error:0.05848      validation_1-logloss:0.19583
[292]   validation_0-error:0.01759      validation_0-logloss:0.09005
validation_1-error:0.05848      validation_1-logloss:0.19551
[293]   validation_0-error:0.01759      validation_0-logloss:0.08970
validation_1-error:0.05848      validation_1-logloss:0.19506
[294]   validation_0-error:0.01759      validation_0-logloss:0.08937
validation_1-error:0.05848      validation_1-logloss:0.19500
[295]   validation_0-error:0.01759      validation_0-logloss:0.08902
validation_1-error:0.05848      validation_1-logloss:0.19495
[296]   validation_0-error:0.01759      validation_0-logloss:0.08861
validation_1-error:0.05848      validation_1-logloss:0.19472
[297]   validation_0-error:0.01759      validation_0-logloss:0.08822
validation_1-error:0.05848      validation_1-logloss:0.19449
[298]   validation_0-error:0.01759      validation_0-logloss:0.08792
validation_1-error:0.05848      validation_1-logloss:0.19415
[299]   validation_0-error:0.01759      validation_0-logloss:0.08766
validation_1-error:0.05848      validation_1-logloss:0.19407
Wall time: 799 ms
```

```
[48]: XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
                    colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
                    importance_type='gain', interaction_constraints='',
                    learning_rate=0.01, max_delta_step=0, max_depth=12,
                    min_child_weight=1, missing=nan, monotone_constraints='()',
                    n_estimators=300, n_jobs=0, num_parallel_tree=1, random_state=0,
                    reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=0.33,
                    tree_method='exact', validate_parameters=1, verbosity=None)
```

[49]:
```
y_pred_xgb=model.predict(x_test)
```

[50]:
```
y_pred_xgb
```

[50]:
```
array([1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0,
       1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1,
       0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1,
       1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0], dtype=uint8)
```

[51]:
```
accuracy_score(y_test, y_pred_xgb)
```

[51]:
```
0.9415204678362573
```

[52]:
```
confusion_matrix(y_test, y_pred_xgb)
```

[52]:
```
array([[63,  6],
       [ 4, 98]], dtype=int64)
```

[53]:
```
print(classification_report(y_test, y_pred_xgb))
```

```
              precision    recall  f1-score   support

           0       0.94      0.91      0.93        69
           1       0.94      0.96      0.95       102

    accuracy                           0.94       171
   macro avg       0.94      0.94      0.94       171
weighted avg       0.94      0.94      0.94       171
```

[54]:
```
f1_score(y_test, y_pred_xgb)
```

[54]:
```
0.9514563106796117
```

[55]:
```
roc_auc_score(y_test, y_pred_xgb)
```

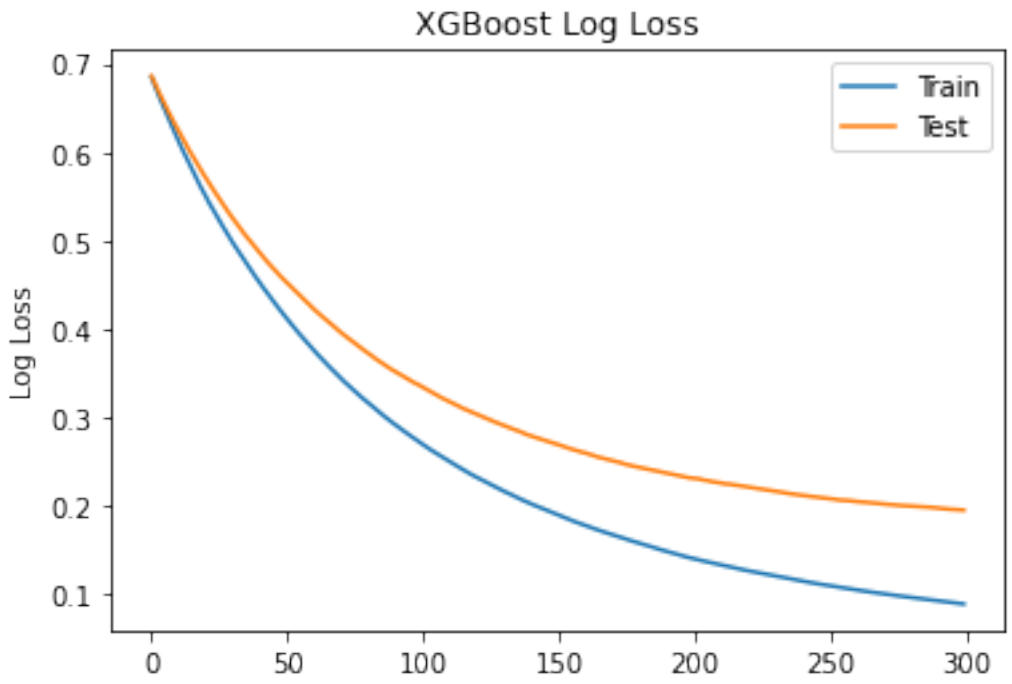
[55]:
```
0.93691389599318
```

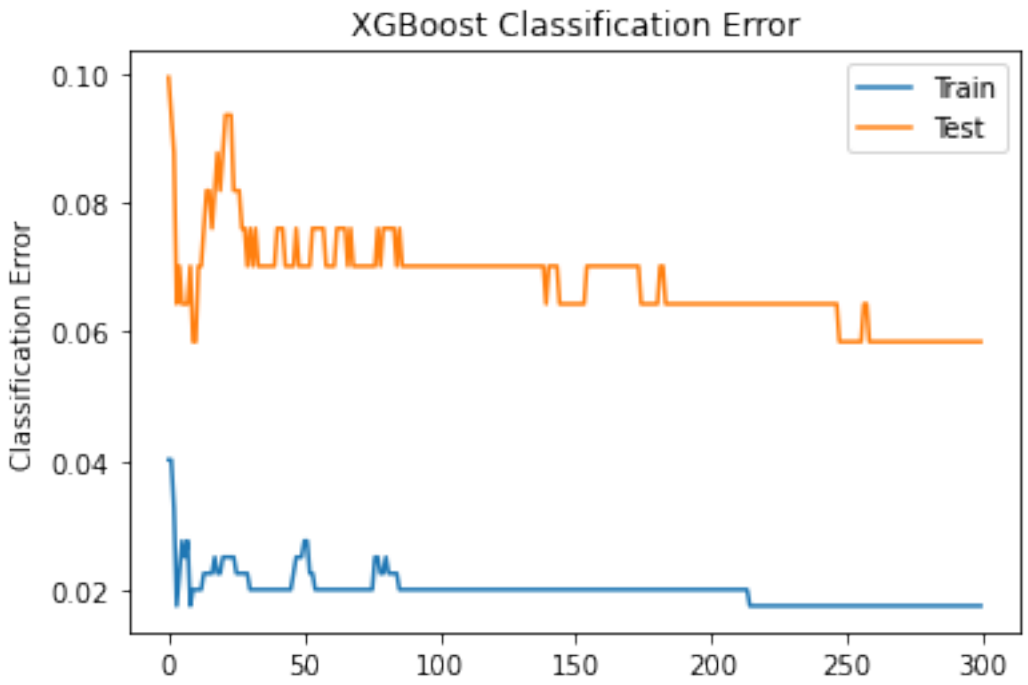
[56]:
```
accuracy = accuracy_score(y_test, y_pred_xgb)
print("Accuracy: %.2f%%" % (accuracy * 100.0))
```

```
Accuracy: 94.15%
```

0.0.18 Training and testing error in Validation

[57]:
```
# retrieve performance metrics
results = model.evals_result()
epochs = len(results['validation_0']['error'])
x_axis = range(0, epochs)
# plot log loss
fig, ax = plt.subplots()
ax.plot(x_axis, results['validation_0']['logloss'], label='Train')
ax.plot(x_axis, results['validation_1']['logloss'], label='Test')
ax.legend()
plt.ylabel('Log Loss')
plt.title('XGBoost Log Loss')
plt.show()
# plot classification error
fig, ax = plt.subplots()
ax.plot(x_axis, results['validation_0']['error'], label='Train')
ax.plot(x_axis, results['validation_1']['error'], label='Test')
ax.legend()
plt.ylabel('Classification Error')
plt.title('XGBoost Classification Error')
plt.show()
```





0.0.19 Hyperparameter Tuning

- let's try different learning rates like 0.001, 0.1, 0.2 ...and calculate the accuracy score

[59]:
```
#Hyper parameter tuning
from sklearn.model_selection import GridSearchCV

clf =XGBClassifier()
parameters = {
     "eta"    : [0.05, 0.10, 0.15, 0.20, 0.25, 0.30 ] ,
     "max_depth"        : [ 3, 4, 5, 6, 8, 10, 12, 15],
     "min_child_weight" : [ 1, 3, 5, 7 ],
     "gamma"            : [ 0.0, 0.1, 0.2 , 0.3, 0.4 ],
     "colsample_bytree" : [ 0.3, 0.4, 0.5 , 0.7 ]
     }

grid = GridSearchCV(clf,
                    parameters, n_jobs=4,
                    scoring="neg_log_loss",
                    cv=3)

grid.fit(x_train, y_train)
```

[59]:
```
GridSearchCV(cv=3,
             estimator=XGBClassifier(base_score=None, booster=None,
                                     colsample_bylevel=None,
                                     colsample_bynode=None,
                                     colsample_bytree=None, gamma=None,
                                     gpu_id=None, importance_type='gain',
                                     interaction_constraints=None,
                                     learning_rate=None, max_delta_step=None,
                                     max_depth=None, min_child_weight=None,
                                     missing=nan, monotone_constraints=None,
                                     n_estimators=100, n_jobs…
                                     num_parallel_tree=None, random_state=None,
                                     reg_alpha=None, reg_lambda=None,
                                     scale_pos_weight=None, subsample=None,
                                     tree_method=None, validate_parameters=None,
                                     verbosity=None),
             n_jobs=4,
             param_grid={'colsample_bytree': [0.3, 0.4, 0.5, 0.7],
                         'eta': [0.05, 0.1, 0.15, 0.2, 0.25, 0.3],
                         'gamma': [0.0, 0.1, 0.2, 0.3, 0.4],
                         'max_depth': [3, 4, 5, 6, 8, 10, 12, 15],
                         'min_child_weight': [1, 3, 5, 7]},
             scoring='neg_log_loss')
```

[60]:
```
y_pred_xgb_cv=grid.predict(x_test)
```

[61]:
```
accuracy_score(y_test, y_pred_xgb_cv)
```

[61]:
```
0.9473684210526315
```

[62]:
```
confusion_matrix(y_test, y_pred_xgb_cv)
```

[62]:
```
array([[ 62,   7],
       [  2, 100]], dtype=int64)
```

[63]:
```
print(classification_report(y_test, y_pred_xgb_cv))
```

```
              precision    recall  f1-score   support

           0       0.97      0.90      0.93        69
           1       0.93      0.98      0.96       102

    accuracy                           0.95       171
   macro avg       0.95      0.94      0.94       171
weighted avg       0.95      0.95      0.95       171
```

[64]:
```
print(f1_score(y_test, y_pred_xgb_cv))
```

0.9569377990430622

[65]:
```
roc_auc_score(y_test, y_pred_xgb_cv)
```

[65]: 0.9394714407502132

0.0.20 Different ensemble classifiers

- Here, I have calculated different ensemble classifiers and compare of each accuracies

[66]:
```
#ensemble method
from sklearn.ensemble import AdaBoostClassifier, GradientBoostingClassifier
from mlxtend.classifier import EnsembleVoteClassifier
from xgboost import XGBClassifier
```

[67]:
```
ada_boost = AdaBoostClassifier(n_estimators=5)
grad_boost = GradientBoostingClassifier(n_estimators=10)
xgb_boost = XGBClassifier(max_depth=5, learning_rate=0.001)
```

[68]:
```
from sklearn.model_selection import cross_val_score
ensemble_clf = EnsembleVoteClassifier(clfs=[ada_boost, grad_boost, xgb_boost],␣
 ↪voting='hard')
boosting_labels = ['Ada Boost', 'Gradient Boost', 'XG Boost', 'Ensemble']
```

[69]:
```
# classifiers performed on the cancer dataset
for clf, label in zip([ada_boost, grad_boost, xgb_boost, ensemble_clf],␣
 ↪boosting_labels):
    scores = cross_val_score(clf, x, y, cv=3, scoring='accuracy')
    print("Accuracy: {0:.3f}, Variance: (+/-) {1:.3f} [{2}]".format(scores.
 ↪mean(), scores.std(), label))
```

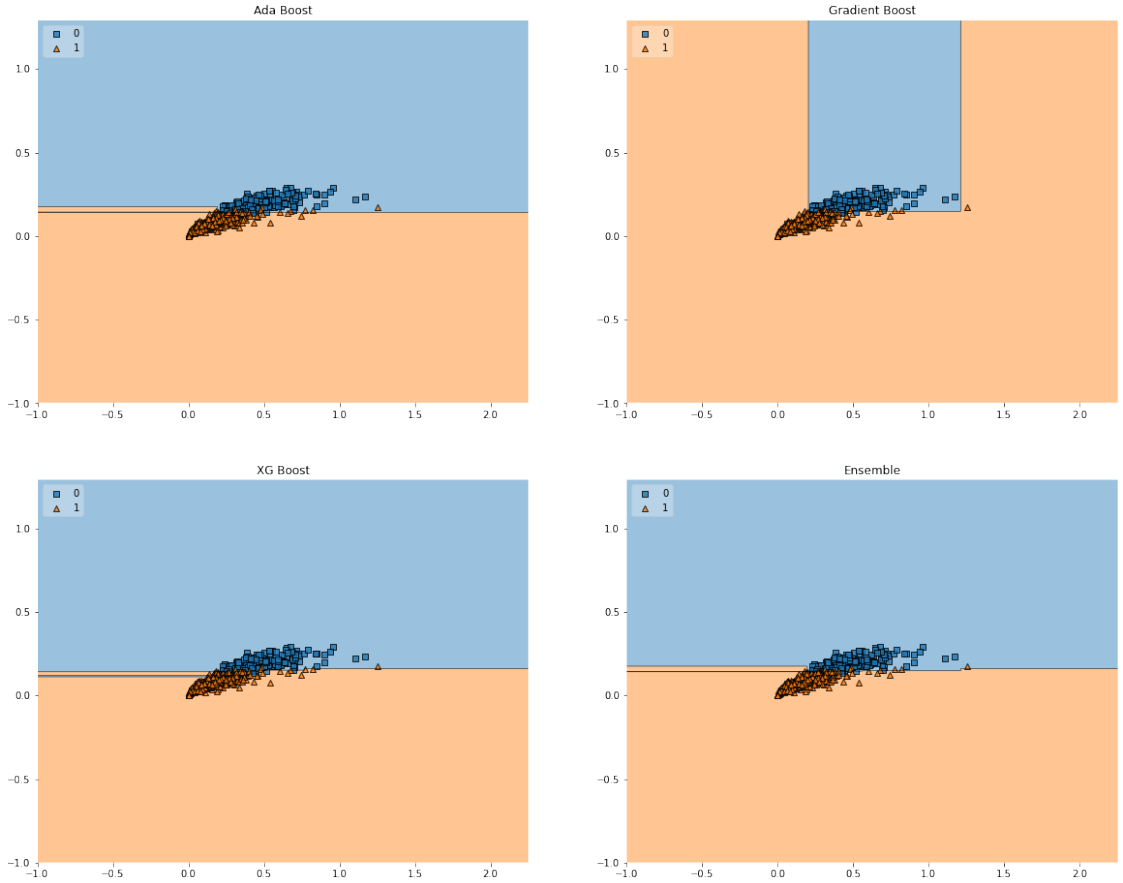
```
Accuracy: 0.933, Variance: (+/-) 0.011 [Ada Boost]
Accuracy: 0.931, Variance: (+/-) 0.015 [Gradient Boost]
Accuracy: 0.916, Variance: (+/-) 0.016 [XG Boost]
Accuracy: 0.928, Variance: (+/-) 0.022 [Ensemble]
```

0.0.21 Visualization through decision regions

- Decision regions for all the boosting algorithms
- let us consider only two independent variables at a time for how the classifiers are working

[70]:
```
x=['radius_mean', 'texture_mean', 'perimeter_mean',
       'area_mean', 'smoothness_mean', 'compactness_mean', 'concavity_mean',
       'concave points_mean', 'symmetry_mean', 'fractal_dimension_mean',
       'radius_se', 'texture_se', 'perimeter_se', 'area_se', 'smoothness_se',
       'compactness_se', 'concavity_se', 'concave points_se', 'symmetry_se',
       'fractal_dimension_se', 'radius_worst', 'texture_worst',
       'perimeter_worst', 'area_worst', 'smoothness_worst',
       'compactness_worst', 'concavity_worst', 'concave points_worst',
```

```
       'symmetry_worst', 'fractal_dimension_worst']
x=np.array(df[['concavity_worst', 'concave points_worst']])
y=np.array(y)
import matplotlib.pyplot as plt
from mlxtend.plotting import plot_decision_regions
import matplotlib.gridspec as gridspec
import itertools
gs = gridspec.GridSpec(2, 2)
fig = plt.figure(figsize=(20,16))
for clf, label, grd in zip([ada_boost, grad_boost, xgb_boost, ensemble_clf],␣
 ↪boosting_labels, itertools.product([0, 1], repeat=2)):
    clf.fit(x, y)
    ax = plt.subplot(gs[grd[0], grd[1]])
    fig = plot_decision_regions(x, y, clf=clf, legend=2)
    plt.title(label)
plt.show()
```



[71]:
```
# The above graph shows that ababoost is working good than other classifiers
ada_boost = AdaBoostClassifier(n_estimators=5)
```

```
ada_boost.fit(x_train, y_train)
y_pred_ada_boost=ada_boost.predict(x_test)
confusion_matrix(y_test, y_pred_ada_boost)
```

[71]:
```
array([[57, 12],
       [ 5, 97]], dtype=int64)
```

[72]:
```
print(classification_report(y_test, y_pred_ada_boost))
```

```
              precision    recall  f1-score   support

           0       0.92      0.83      0.87        69
           1       0.89      0.95      0.92       102

    accuracy                           0.90       171
   macro avg       0.90      0.89      0.89       171
weighted avg       0.90      0.90      0.90       171
```

[73]:
```
f1_score(y_test, y_pred_ada_boost)
```

[73]:
```
0.9194312796208531
```

[74]:
```
roc_auc_score(y_test, y_pred_ada_boost)
```

[74]:
```
0.8885336743393009
```

0.0.22 Results:

- While comparing accuracies of AdaBoost Classifier and XGBoost, XGBoost Classifier performs the best prediction, with an accuracy of 94.15%.

0.0.23 References:

- https://towardsdatascience.com/selecting-optimal-parameters-for-xgboost-model-training-c7cd9ed5e45e
- https://towardsdatascience.com/a-beginners-guide-to-xgboost-87f5d4c30ed7
- https://medium.com/@saugata.paul1010/ensemble-learning-bagging-boosting-stacking-and-cascading-classifiers-in-machine-4.learning-9c66cb271674
- https://rasbt.github.io/mlxtend/