Pattern Recognition and Machine Learning
Indian Institute of Technology, Jodhpur

LAB 5

Report

Harshil Kaneria

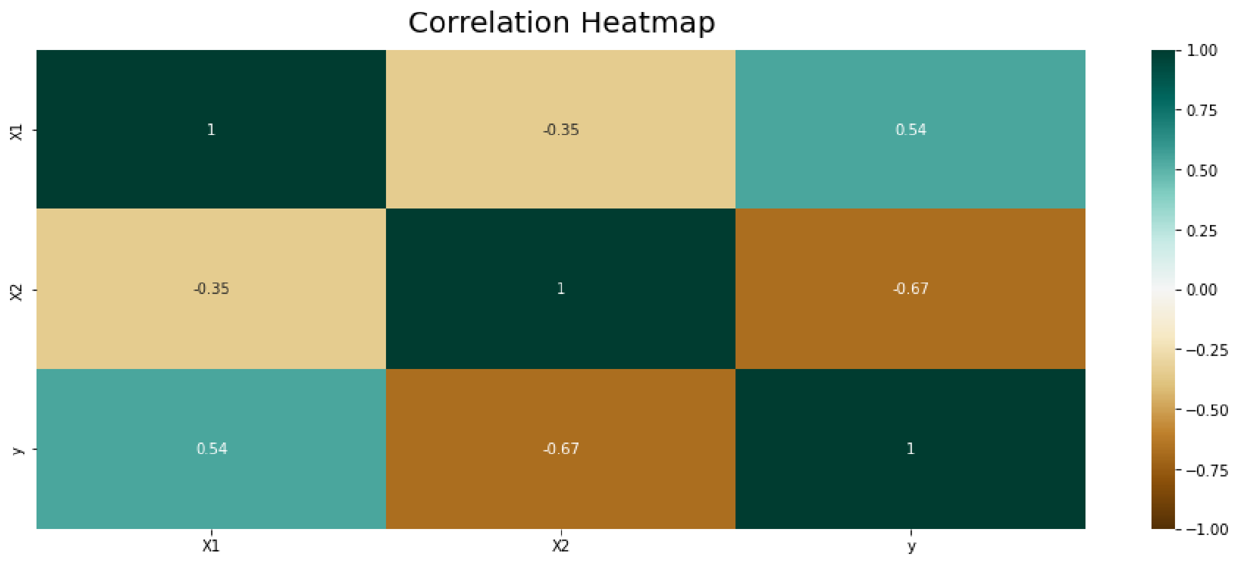
Department of Computer Science, IIT Jodhpur

Feb 27, 2023

Q1)

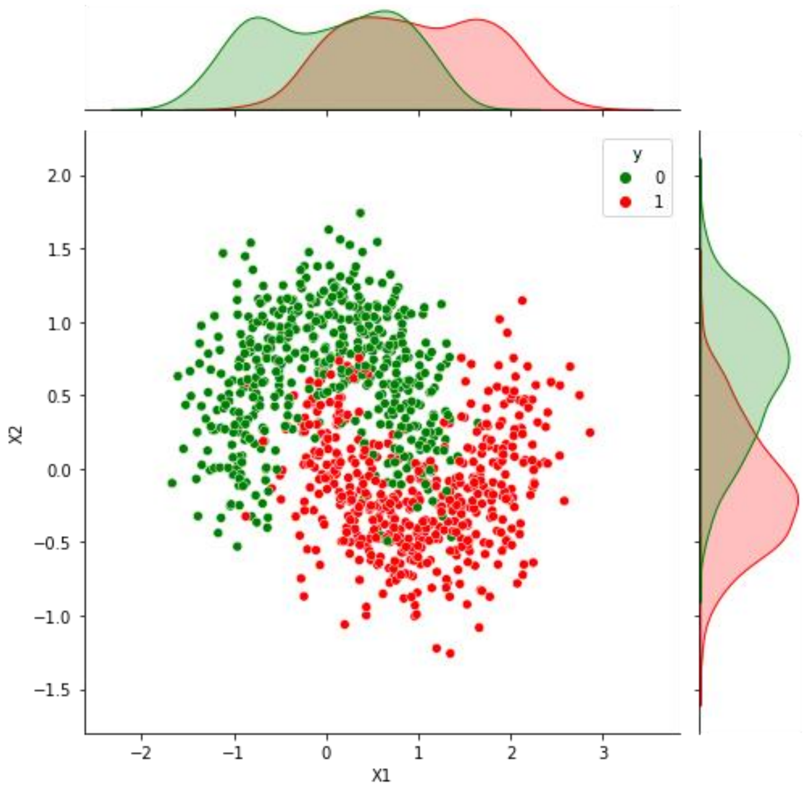
1) Pre-processing and Exploratory Analysis:

Plotted a **Correlation Heatmap**.

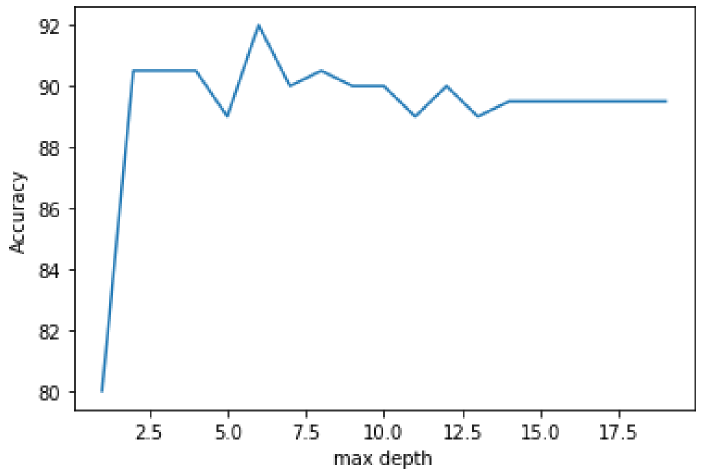


As we can see from correlation heatmap, there is poor correlation between columns “X1” and “X2”, which feature vectors of our dataset. Removing the datapoint lying far away from the distribution is the pre-processing step done. The **Outlier** datapoints are those which lie beyond **mean+3*(standard deviation)** and **mean-3*(standard deviation).**

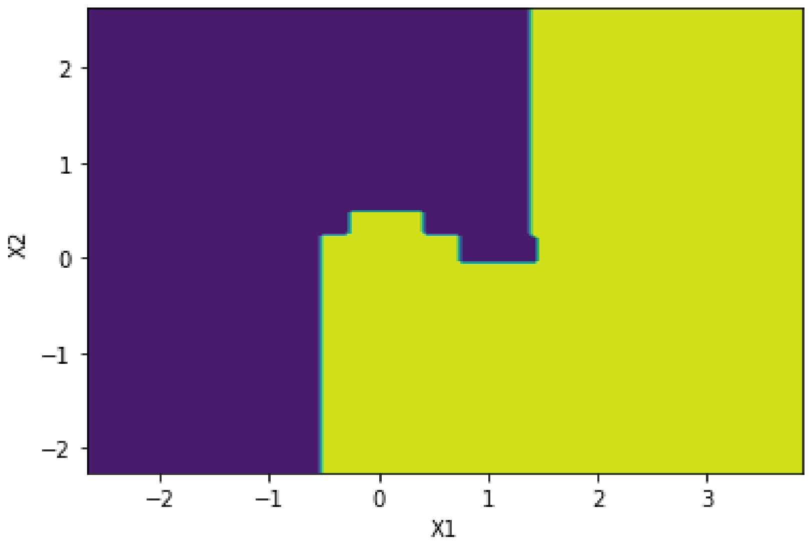
Generated Dataset (Scatter Plot with distribution):



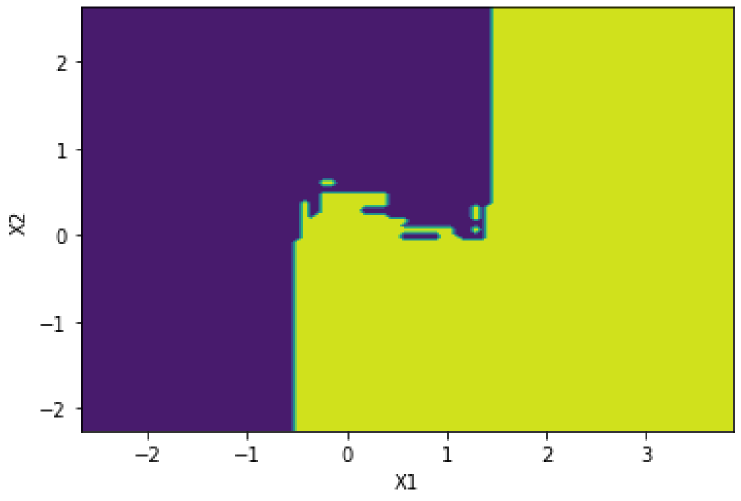
Trained a Decision Tree Classifier on the dataset generated and performed hyperparameter tuning for finding optimal max_depth. It comes out that **optimal max depth is 6**.



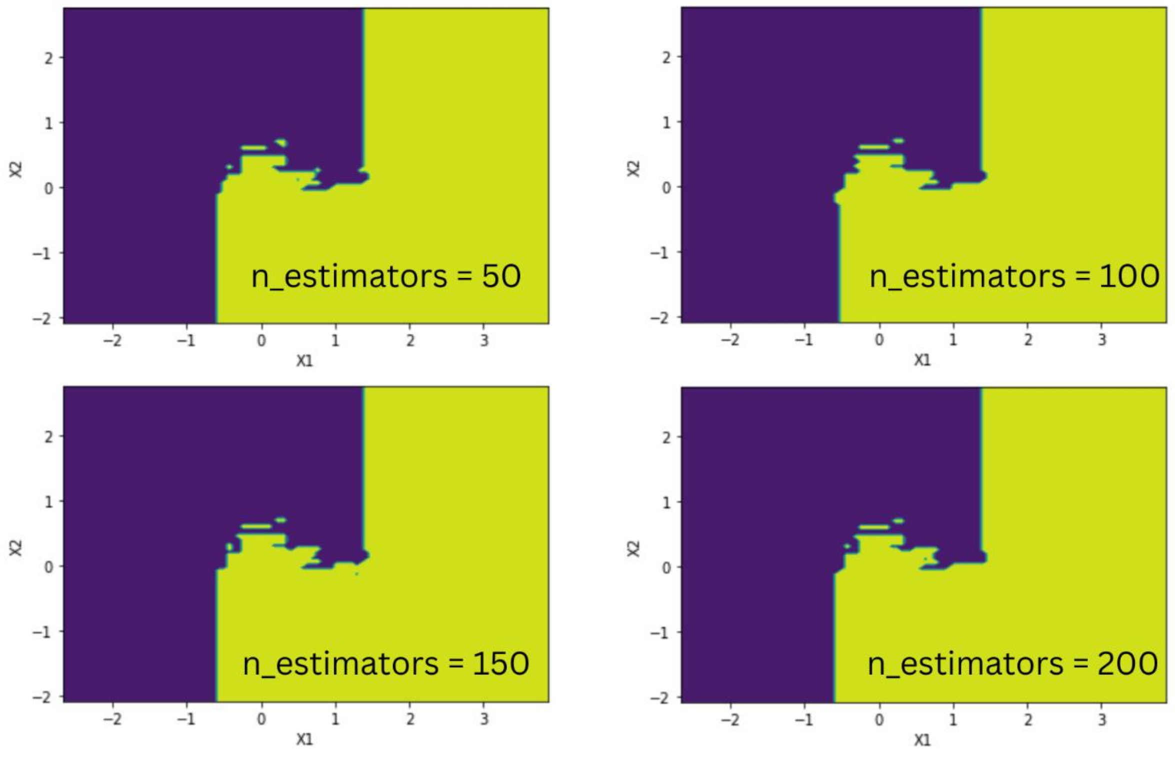
Trained a Decision Tree classifier at max depth = 6 and plotted the decision boundary for the same as follows:



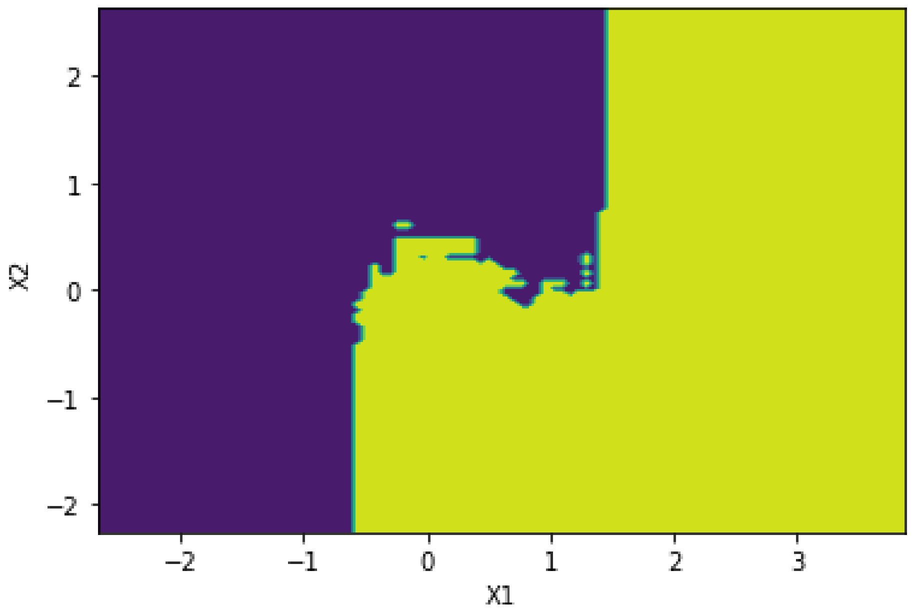
Trained a Bagging Classifier from sklearn and plotted the decision boundary as follows:



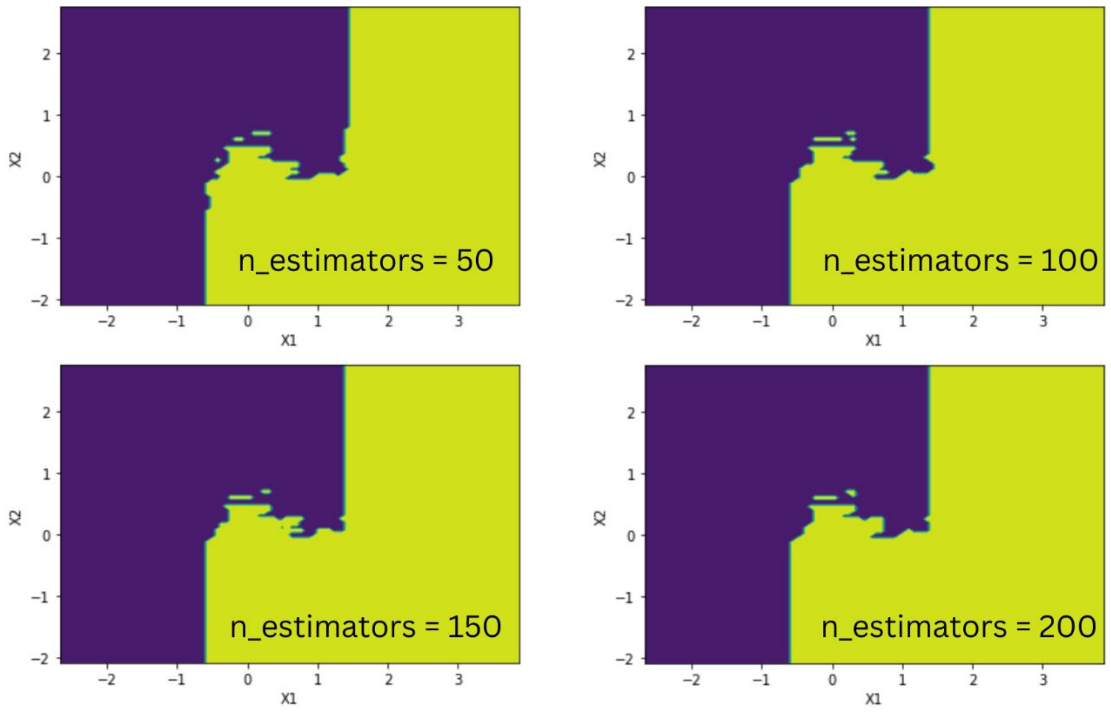
- Trained Bagging Classifier for different number of estimators starting from n_estimators = 50 to n_estimators = 300 with a gap of 50.
- Found that Maximum Accuracy of Bagging Classifier occurs at n_estimators = 100 and accuracy is 92%.
- The decision boundary plots of some cases are as below:



Trained a Random Forest Classifier from sklearn and plotted the decision boundary as follows:



- Trained Random Forest Classifier for different number of estimators starting from n_estimators = 50 to n_estimators = 300 with a gap of 50.
- Found that Maximum Accuracy of Random Forest Classifier occurs at n_estimators = 50 and accuracy is 92%.
- The decision boundary plots of some cases are as below:



The Random Forest Classifier is better choice for classification. This is due to the random feature selection in Random Forest Classifier, the trees are more independent of one another compared to regular bagging, which often results in better predictive performance (due to better variance-bias trade-offs), and we can also say that it is faster than bagging because each tree only learns from a subset of features.

1) **Implementing Bagging Classifier from Scratch:**

The Class for Bagging Classifier is named "Bagging_Classifier" and beholds the following:

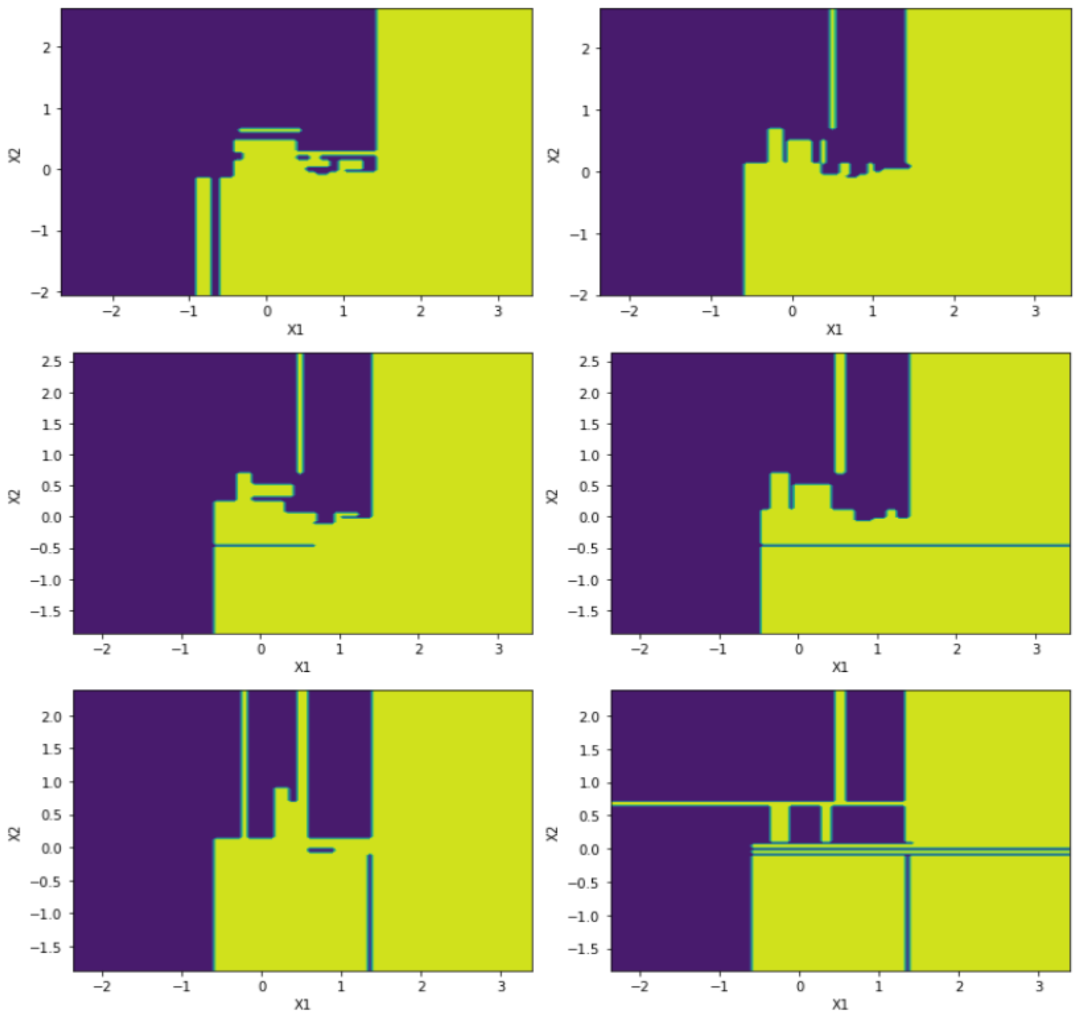
1. **train** – Trains Decision tree classifier model.
2. **predict** – Predicts the class by voting method.
3. **score** – return the accuracy score of classifiers.
4. **EachTreeAc**c – Returns the accuracy of each tree trained.
5. **EachTreeDB** – plots decision boundary of each tree.

Overall performance of Bagging Classifier for generated dataset: **90.5%**

Performance of Tree model trained:

[89.0 , 87.5 , 88.5 , 87.5 , 85.5 , 90 .0 , 89.5 , 85.5 , 84.0 , 84.5]

Decision Boundaries of some of the trees are shown below:



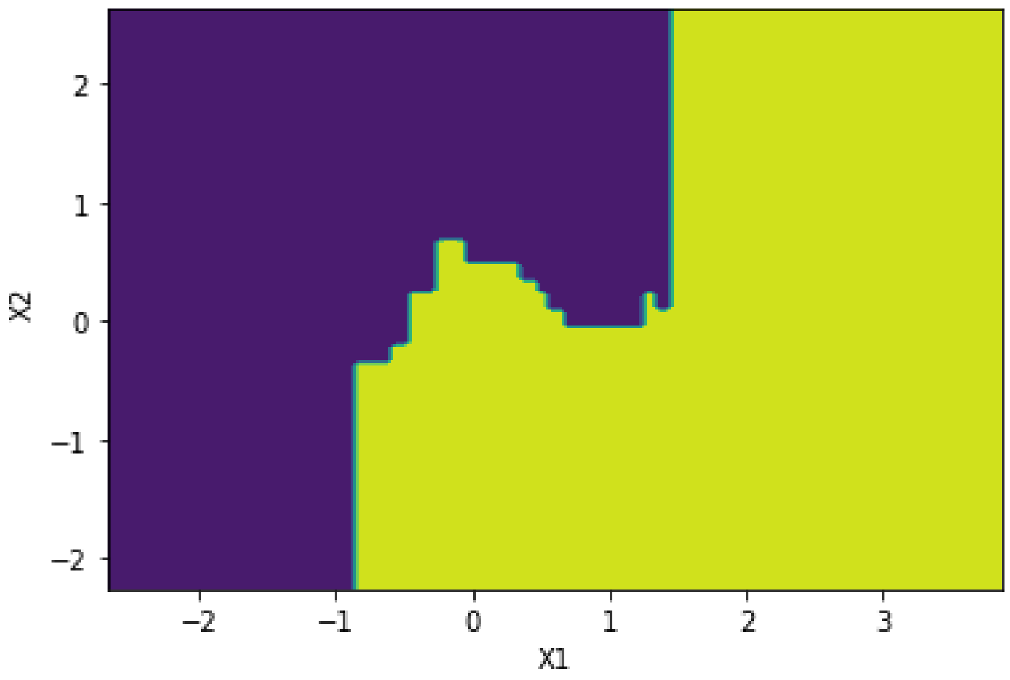
Hence, each tree has discrete decision limits and varying accuracies of two to three percent. We know that a Bagging classifier votes on several models to classify data. When we train too many decision trees with various chunks of data from our train set, each tree has a unique classification approach, preventing overfitting and underfitting. Bagging, unlike typical decision trees, examines many trees, preventing overfitted and underfitted trees.

Q2)

1) **Trained AdaBoost Classifier:**

   Accuracy on Train data: 94%   Accuracy on Test data: 89%

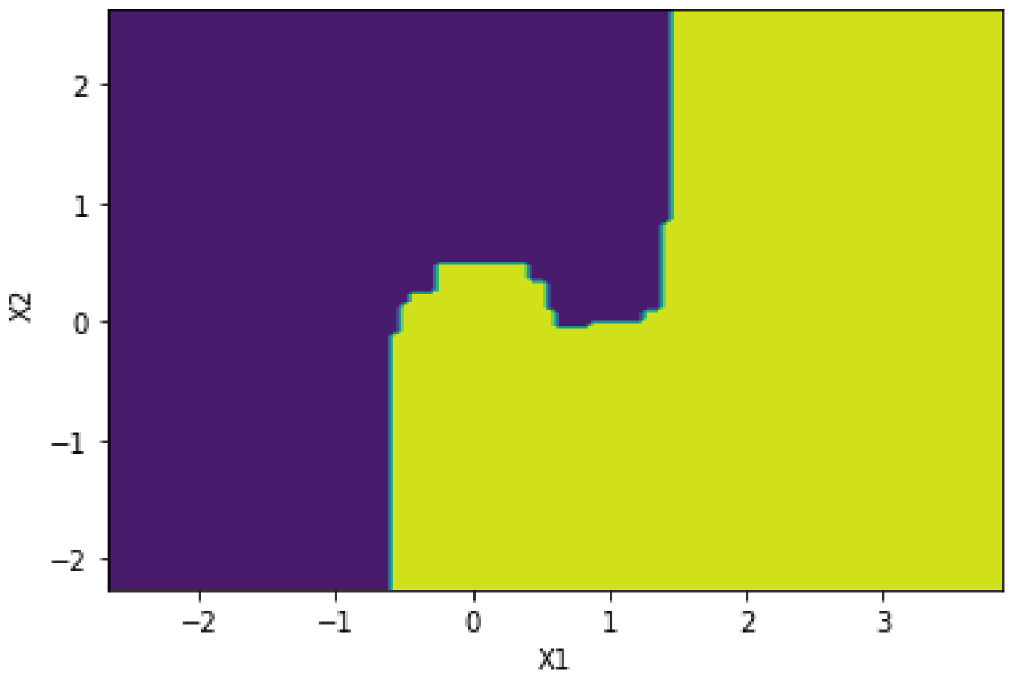
   Decision Boundary for the same:



2) **Trained XgBoost Classifier:**

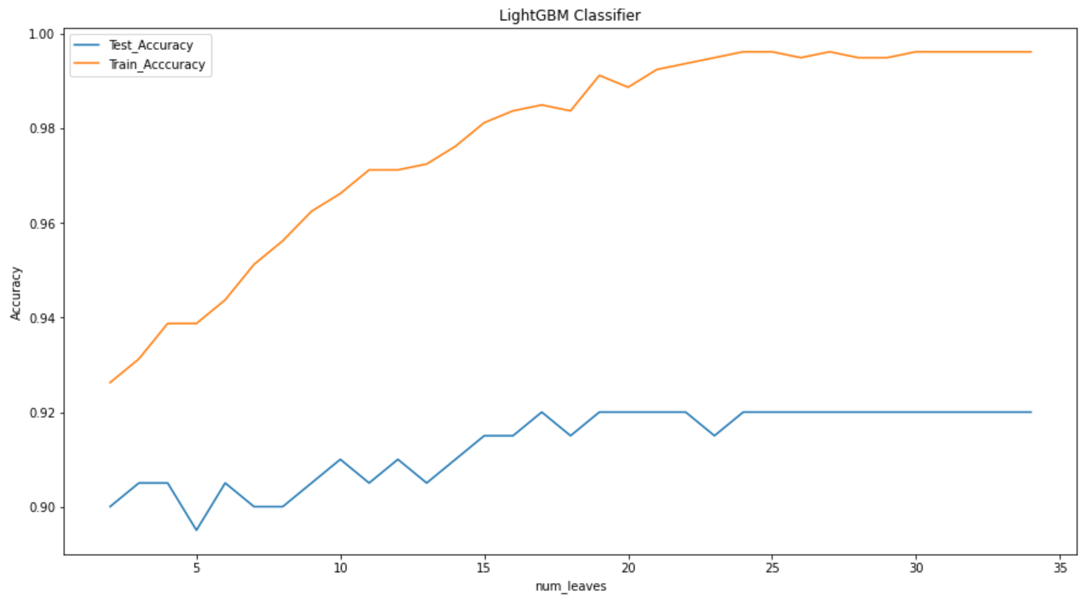
   Accuracy on Train data: 94%   Accuracy on Test data: 91%

   Decision Boundary for the same:



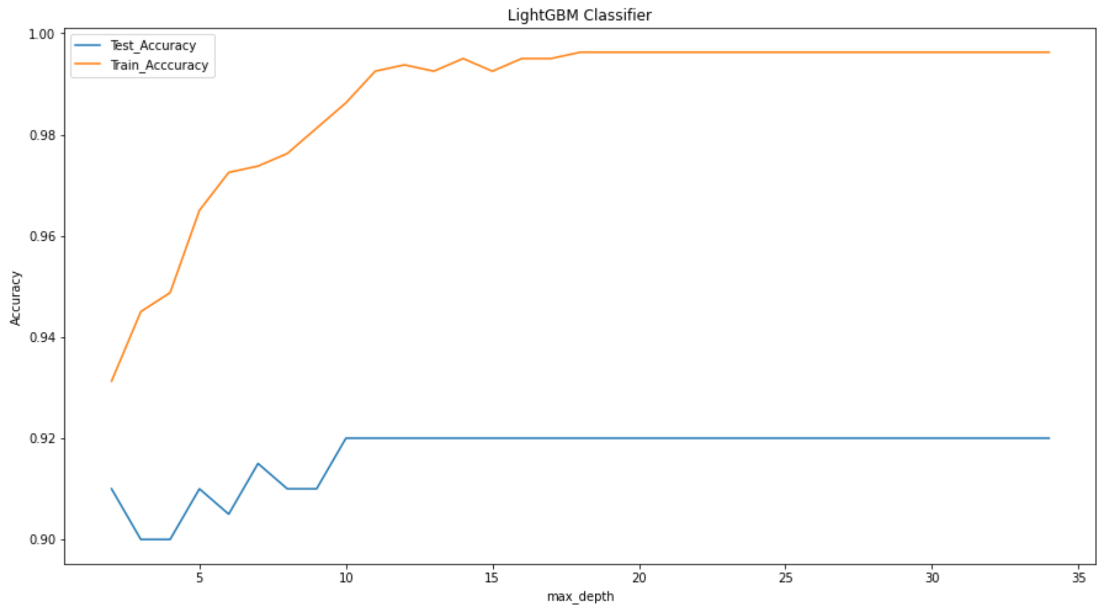
3) Trained LightGBM Classifier (varying num_leaves):

Following is the plot for accuracies on Train and Test data:



4) Trained LightGBM Classifier (varying max_depth):

Following is the plot for accuracies on Train and Test data:

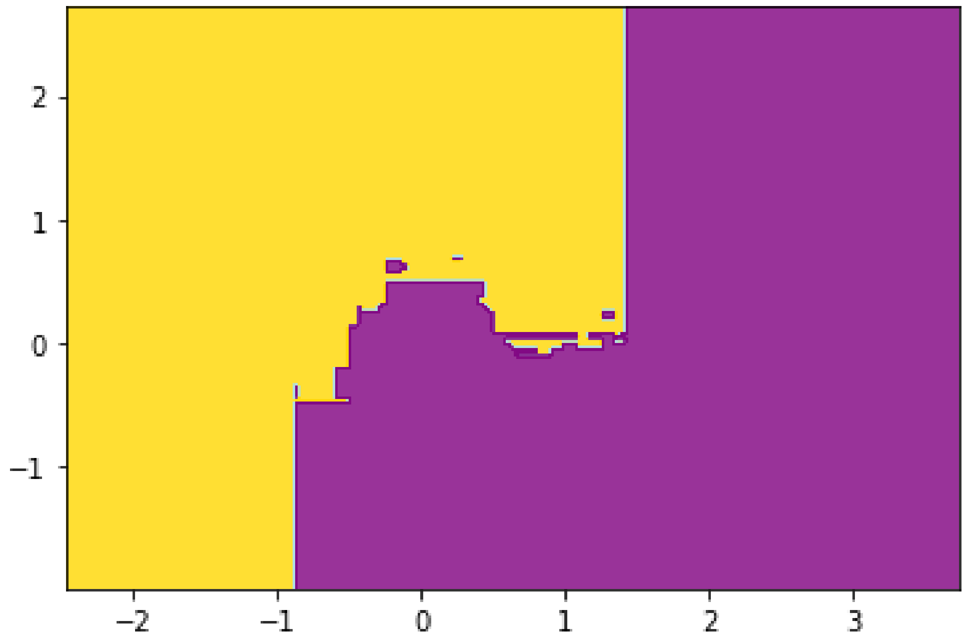


We can see from the above two graphs that after certain number of num_leaves and max_depth the accuracy doesn't change for the classifier. This indicates that after that point the model starts overfitting on training dataset. For the num_leaves the overfitting starts around num_leaves = 25. For the max_depth the overfitting starts around max_depth = 12.

In general, increasing max_depth allows the tree to capture more complex relationships in the data, which can lead to better accuracy on the training set.

In contrast, increasing num_leaves allows the model to split the data into more partitions, which can also increase accuracy on the training set. We can see that the learning rate of Classifier when max_depth is varied is higher. So, the max depth can be better option for avoiding overfitting and detect the same at early stage.

5) **Trained LightGBM Classifier** (num_leaves = 7):
Accuracy on Train data: 95.125% Accuracy on Test data: 90%
Decision Boundary for the same:



The performance of LightGBM Classifier is better than others but there is not much difference in the accuracies.

Q3) Voting Classifier: Accuracy – 90%

Used the following four Classifiers for Voting Classifier:

a) Gaussian Naïve Bayes Classifier – 85%
b) LightGBM Classifier – 90%
c) AdaBoost Classifier – 89%
d) XGBoost Classifier – 91%

We can see that Accuracy of XGBoost Classifier is maximum and Gaussian Naïve Bayes Classifier is least.