Lab 5 Report

Nakkina Vinay (B21AI023)

Question 1

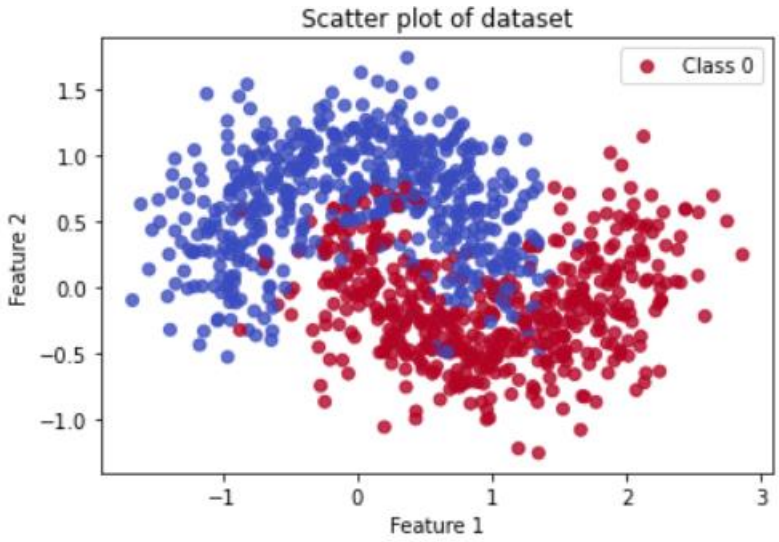
Subpart 1:

- Creating a dataset with 1000 samples using the make_moons() from sklearn
- Printing X and y

```
[[-0.17186341  0.59624885]
 [ 1.25328273 -0.26541353]
 [ 0.72322405  0.2319425 ]
 ...
 [ 1.77095705 -0.50943619]
 [-1.06177158  0.006786  ]
 [ 0.76117231  0.65196041]]
```

```
[1 1 1 1 0 1 1 1 1 0 1 0 1 1 1 1 0 1 1 0 1 0 0 1 0 0 0 1 1 1 0 0 1 0 0 0 1
 1 1 0 0 0 0 1 0 0 1 1 0 1 1 1 1 0 1 0 0 1 0 0 1 0 0 1 0 1 1 1 1 0 1 0 0 1 1
 0 0 1 0 1 0 1 0 0 0 0 1 1 1 1 0 0 0 1 0 1 0 1 0 0 1 1 0 1 0 1 1 1 1 1 0 1 1
 1 1 1 0 0 0 1 1 0 1 0 1 0 0 1 1 0 1 1 1 1 0 1 1 0 0 0 0 0 0 0 0 1 0 1 1 1 0
 1 0 1 0 1 0 1 0 0 0 1 0 1 1 1 1 1 1 1 1 0 1 0 0 0 0 0 1 0 0 0 0 1 1 0 1 0 1 1
 0 0 0 1 1 1 1 1 0 0 0 0 0 0 1 0 0 1 1 1 1 1 1 0 1 0 1 0 0 1 1 1 0 1 0 1 1 0 1
 1 0 1 0 1 0 1 1 1 0 1 0 1 0 0 0 1 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1 1 0 0 1 1 1
 0 1 0 0 0 0 1 1 0 1 0 0 0 1 0 1 0 0 1 0 1 1 1 0 0 0 1 0 0 0 0 1 1 1 1 1 0 0 0
 1 0 0 0 1 0 0 0 1 1 0 1 1 1 1 1 1 1 1 0 0 0 0 1 0 1 0 0 0 1 1 1 1 0 1 0 1 0 1
 1 0 0 1 1 1 1 1 0 0 0 0 0 0 1 1 0 1 0 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 1
 0 1 0 0 0 1 0 0 1 1 0 0 1 0 0 1 1 0 1 1 0 0 1 0 1 0 0 0 1 1 0 0 1 1 1 1 1 1
 0 0 1 1 1 1 0 1 1 1 1 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 1
 1 1 1 0 1 1 1 1 0 0 0 1 1 1 0 0 0 0 1 1 0 0 0 0 1 0 0 0 1 0 0 1 1 1 1 1 1 1
 0 0 0 1 0 0 0 0 0 1 1 1 0 0 0 0 0 1 1 1 0 0 1 1 1 1 0 1 1 0 1 0 0 0 0 1 0 0
 1 0 0 1 1 0 0 1 1 0 1 0 1 0 1 0 1 0 0 0 1 0 0 0 0 0 0 1 1 1 1 0 0 0 1 0 1 0 1
 1 0 0 0 0 0 1 1 1 0 0 1 0 0 1 0 0 1 0 0 0 1 0 0 0 1 1 1 0 1 1 0 0 0 1 0 1 1
 1 1 0 0 1 1 0 0 1 0 1 0 0 0 1 0 0 1 1 1 1 0 1 1 1 1 1 1 1 0 1 0 1 1 1 1 0 0
 1 0 0 0 1 1 1 1 0 0 0 0 1 0 1 1 1 0 1 0 0 1 0 0 1 1 1 1 1 0 1 0 0 0 1 1 1 1
 1 1 0 0 0 1 1 1 1 0 0 0 0 0 1 1 0 1 0 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 0 1 0
 1 1 1 1 0 1 1 1 1 1 1 1 1 0 1 1 0 1 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 1 0 1 1
 0 0 1 1 1 0 0 0 0 1 1 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 1 0 1 0 1 0 1 0 1 1 1
 1 1 1 1 0 1 0 1 1 1 0 1 1 0 0 1 0 1 1 0 0 1 1 1 1 1 1 1 1 1 0 1 1 1 0 1 0 0 1
 1 1 0 0 0 1 0 0 1 0 0 0 1 0 1 0 1 1 1 1 0 1 0 0 0 0 0 0 0 1 0 1 0 1 0 1 0 1 0
 1 0 1 1 1 0 1 0 1 0 0 1 1 1 0 0 0 1 1 0 1 0 1 1 0 1 0 0 1 1 1 0 0 0 1 1 0 1 0
 0 0 0 0 1 1 0 1 0 0 0 1 0 0 0 1 0 1 1 1 0 1 1 1 0 1 1 1 1 0 1 1 0 1 1 0 0 1 1
 1 1 0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 1 0 1 0 0 0 1 0 1
 0]
```

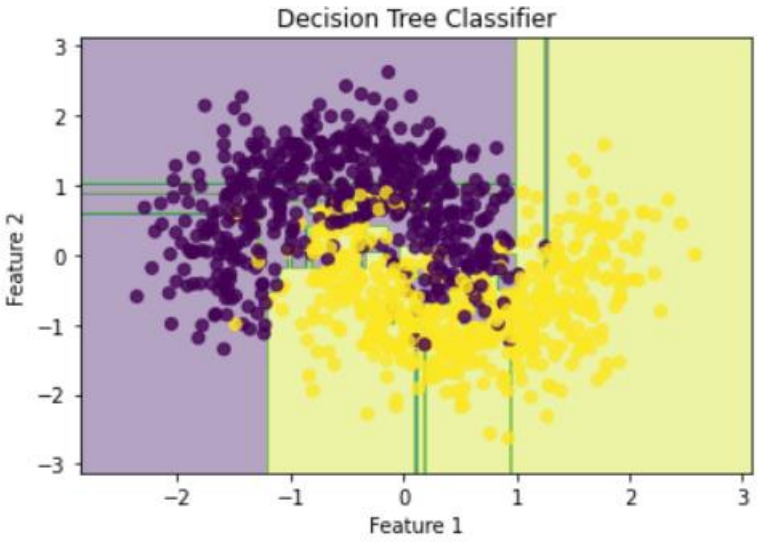
- The dataset was analyzed and it was a complete dataset
- Plotting the generated dataset using the scatter plot



- Performing preprocessing using StandardScaler() function and splitting the dataset into train and test sets

Decision Tree Classifier

- Training the model on out train sets using the DecisionTreeClassifier
- Plotting the decision boundaries on the dataset by making the grid points

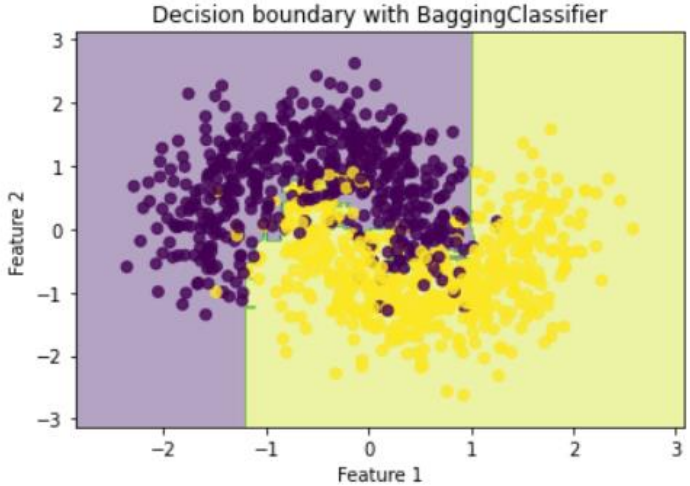


- Performing hyperparameter tuning for finding the best value of max_depth
- For this we are using the 5-fold Cross-Validation from GridSearchCV
- First defining the range of hyperparameters to be tuned
- Performing cross-validation using GridSearchCV and printing the best hyperparameters

```
Best Hyperparameters: {'max_depth': 2}
Best Value of max_depth is:  2
```

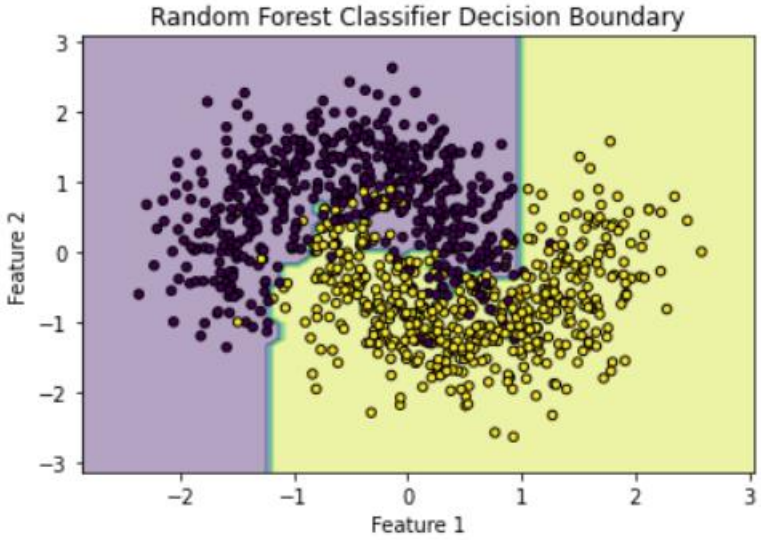
Bagging Classifier

- First defining the base classifier as DecisionTreeClassifier
- Training the BaggingClassifier with 100 estimators
- Plotting the decision boundary



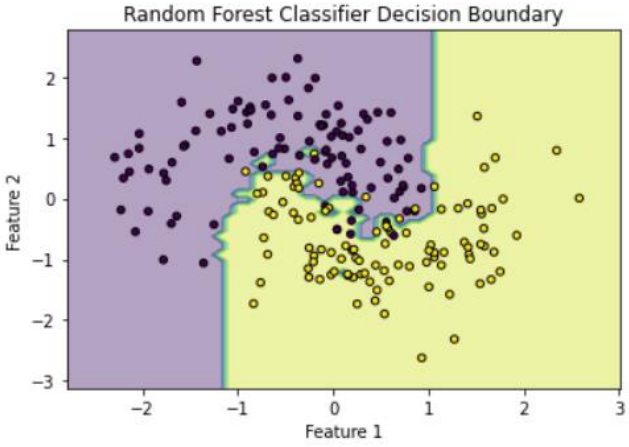
Random Forest Classifier

- Training a Random Forest Classifier with 100 trees
- Defining the boundaries of the meshgrid to create the decision boundary
- Predicting the labels of the meshgrid and plotting the decision boundary



Comparing the three Classifiers

- For comparing all the three classifiers Decision tree, Bagging and Random forest we will be calculating the evaluation metrics i.e Accuracy, Prediction, Recall, F1 Score and plotting the decision boundaries



- The best classifier will be the one with highest accuracy

```
Decision Tree Classifier:
 Accuracy: 0.905
 Precision: 0.9090909090909091
 Recall: 0.9
 F1-Score: 0.9045226130653266

Bagging Classifier:
 Accuracy: 0.895
 Precision: 0.898989898989899
 Recall: 0.89
 F1-Score: 0.8944723618090452

Random Forest Classifier:
 Accuracy: 0.905
 Precision: 0.9175257731958762
 Recall: 0.89
 F1-Score: 0.9035532994923858
```

Varying number of estimators

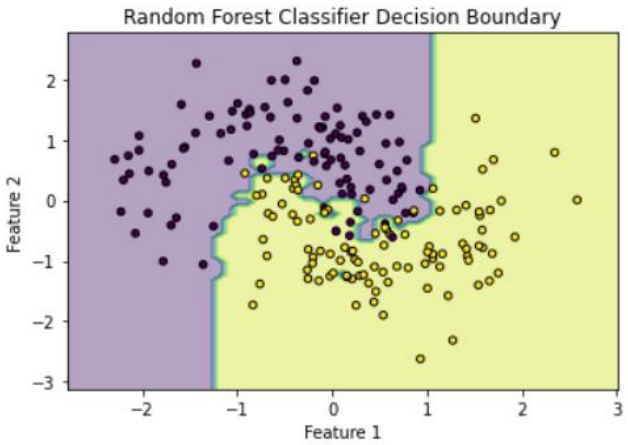
- Varying the number of estimators in Bagging classifier and Random forest classifier and calculating he accuracies of each one

```
Accuracy Score for Bagging Classifier when n_estimators are 100 is 0.895
Accuracy Score for Bagging Classifier when n_estimators are 110 is 0.895
Accuracy Score for Bagging Classifier when n_estimators are 120 is 0.9
Accuracy Score for Bagging Classifier when n_estimators are 130 is 0.9
Accuracy Score for Bagging Classifier when n_estimators are 140 is 0.9
Accuracy Score for Bagging Classifier when n_estimators are 150 is 0.9
Accuracy Score for Bagging Classifier when n_estimators are 160 is 0.905
Accuracy Score for Bagging Classifier when n_estimators are 170 is 0.905
Accuracy Score for Bagging Classifier when n_estimators are 180 is 0.905
Accuracy Score for Bagging Classifier when n_estimators are 190 is 0.9
Accuracy Score for Bagging Classifier when n_estimators are 200 is 0.9
```

```
Accuracy Score for Random Forest Classifier when n_estimators are 100 is 0.905
Accuracy Score for Random Forest Classifier when n_estimators are 110 is 0.905
Accuracy Score for Random Forest Classifier when n_estimators are 120 is 0.905
Accuracy Score for Random Forest Classifier when n_estimators are 130 is 0.915
Accuracy Score for Random Forest Classifier when n_estimators are 140 is 0.915
Accuracy Score for Random Forest Classifier when n_estimators are 150 is 0.915
Accuracy Score for Random Forest Classifier when n_estimators are 160 is 0.915
Accuracy Score for Random Forest Classifier when n_estimators are 170 is 0.915
Accuracy Score for Random Forest Classifier when n_estimators are 180 is 0.91
Accuracy Score for Random Forest Classifier when n_estimators are 190 is 0.91
Accuracy Score for Random Forest Classifier when n_estimators are 200 is 0.915
```

- Accuracies of Random forest classifier are higher than the accuracies of bagging classifier when varying the number of estimators
- Finding the maximum accuracy obtained when varying the number of estimators and training the model with Random forest classifier if the maximum accuracy is obtained under Random classifier or training the model

with Bagging classifier when the maximum accuracy is obtained under Bagging classifier and plotting the decision boundary of the same



Subpart 2:

- Implementing the Bagging Classifier from scratch
- Defining a BaggingClassifier class which trains multiple decision tree classifiers on bootstrap samples of the training data and combines their predictions by majority voting
- The class contains functions
    - __init__ for intialising the variables
    - Fit() : fits the bagging classifier to the training data
    - Predict() : predicts the class labels for the test data by majority voting over the predictionsof the individual classifiers
- Using this class intialising the bagging classifier with 10 estimators and max_depth=5
- Training the bagging classifier on the training data and making the predictions on the test data
- Calculating the accuracy

```
Bagging classifier accuracy: 0.9350
```

Question 2

Subpart 1: AdaBoost Model

- Intialising the AdaBoost classifier with DecisionTreeClassifier as the base estimator
- Training the AdaBoost classifier
- Evaluating the performance on the training set and testing set

```
Accuracy on the Training set: 1.0
Accuracy on the Testing set: 0.905
```

Subpart 2: XGBoost Model

- Defining the XGBoost model with subsample=0.7
- Training the XGBoost classifier
- Evaluating the performance on the training set and testing set

```
XGBoost model accuracy for Training set: 0.995
XGBoost model accuracy for Testing set: 0.905
```

Subpart 3 is done in subpart 2 and 3

Subpart 4: LightGBM Model

- Training LightGBM model with different values of num_leaves
- Training the LGBM classifier with looping through the num_leaves
- Evaluating the performance on the testing set

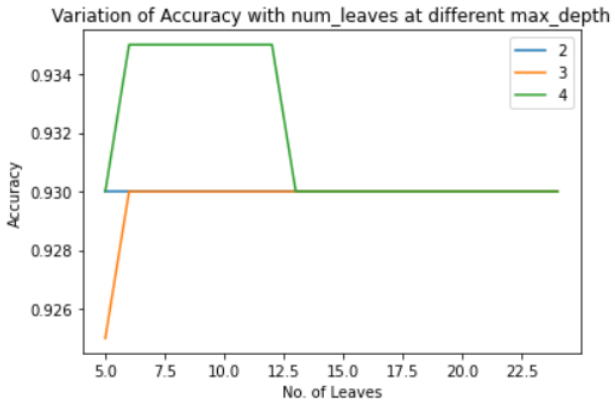
```
Accuracy with num_leaves=5: 0.93
Accuracy with num_leaves=6: 0.92
Accuracy with num_leaves=7: 0.935
Accuracy with num_leaves=8: 0.93
Accuracy with num_leaves=9: 0.935
Accuracy with num_leaves=10: 0.925
Accuracy with num_leaves=11: 0.93
Accuracy with num_leaves=12: 0.925
Accuracy with num_leaves=13: 0.915
Accuracy with num_leaves=14: 0.915
Accuracy with num_leaves=15: 0.91
Accuracy with num_leaves=16: 0.915
Accuracy with num_leaves=17: 0.9
Accuracy with num_leaves=18: 0.905
Accuracy with num_leaves=19: 0.91
Accuracy with num_leaves=20: 0.905
Accuracy with num_leaves=21: 0.9
Accuracy with num_leaves=22: 0.91
Accuracy with num_leaves=23: 0.905
Accuracy with num_leaves=24: 0.91
```

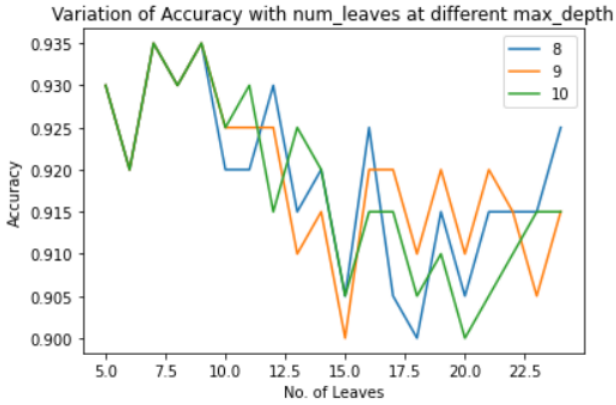
Subpart 5: Analysing the relation between max_depth and num_leaves

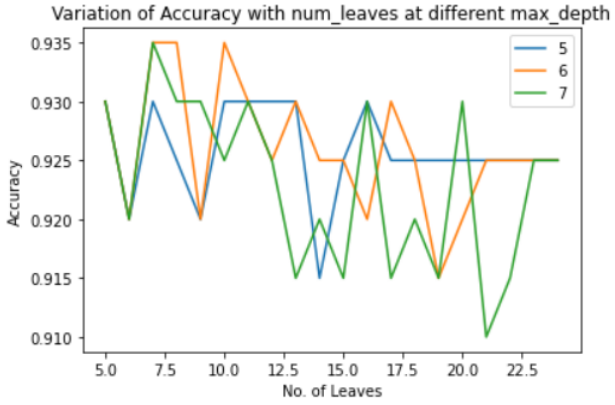
- Training LightGBM model with different values for max_depth
- Looping through them and calculating the accuracies

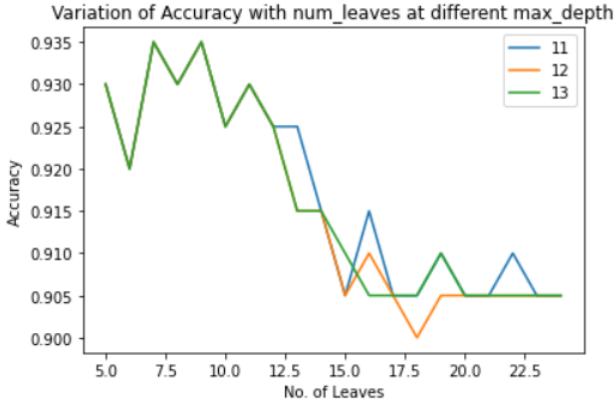
```
Accuracy with max_depth=2: 0.93
Accuracy with max_depth=3: 0.93
Accuracy with max_depth=4: 0.93
Accuracy with max_depth=5: 0.925
Accuracy with max_depth=6: 0.925
Accuracy with max_depth=7: 0.915
Accuracy with max_depth=8: 0.91
Accuracy with max_depth=9: 0.91
Accuracy with max_depth=10: 0.905
Accuracy with max_depth=11: 0.91
Accuracy with max_depth=12: 0.91
Accuracy with max_depth=13: 0.905
Accuracy with max_depth=14: 0.9
Accuracy with max_depth=15: 0.91
Accuracy with max_depth=16: 0.905
Accuracy with max_depth=17: 0.91
Accuracy with max_depth=18: 0.905
```

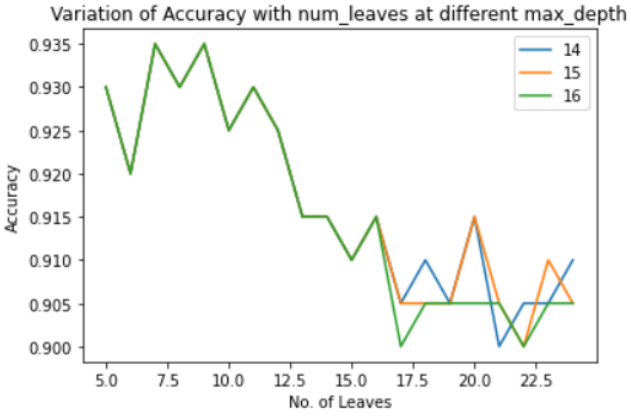
- Looping through the max_depth and num_leaves and calculating the accuracies fand anlysing the relation between max_depth and num_leaves

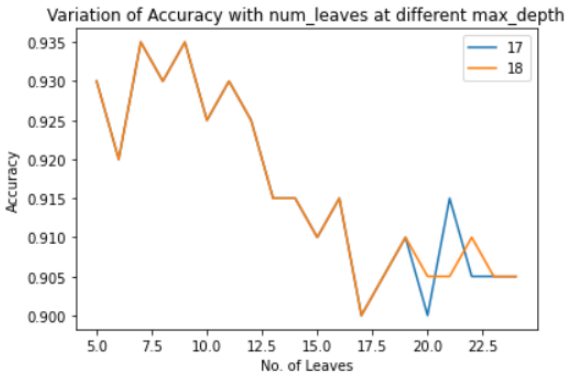










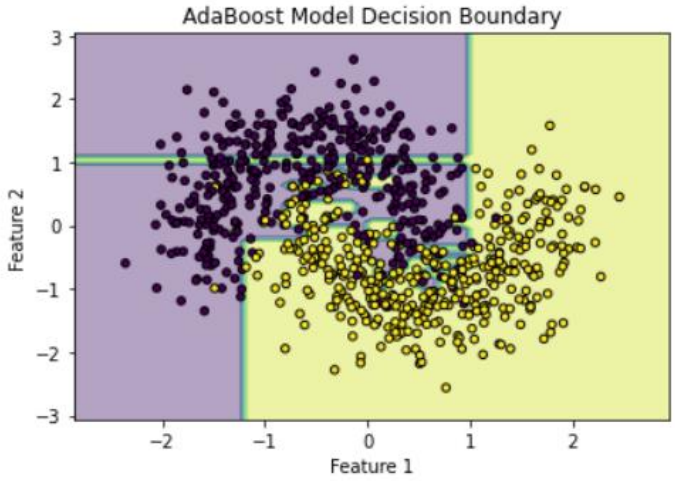


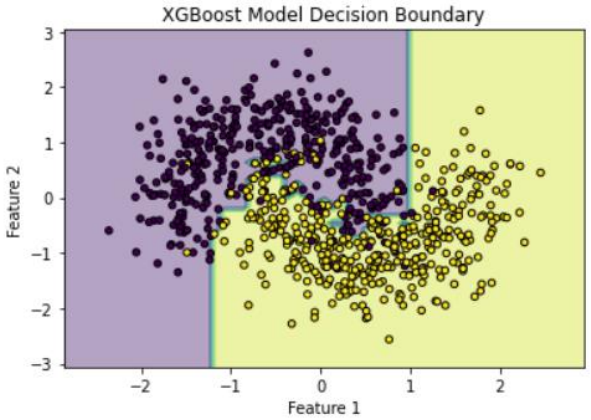
Subpart 6:

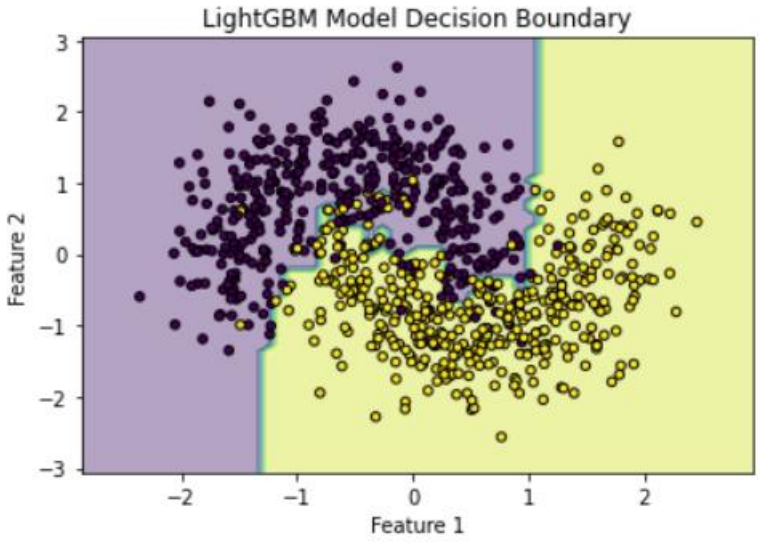
- With enough number of leaves, accuracy stops fluctuating and our model stops learning any new feature, for a given max_depth.
- When max_depth is low(less than 5), variation of accuracies with increase in num_leaves is rather small and is consistent.
- However, if max_depth is high, our accuracies first increase, reach their maxima and then take a local maximum.
- From the above analysis, max_depth hyperparameter is best for controlling and tweaking accuracy while, num_leaves hyperparameter is more suited for controlling ovrfitting of our model.

Subpart 6:

- Plotting the decision boundaries for AdaBoost, xgboost and LightGBM models







Question 3

Subpart 1:

- Training a Bayes classification on the dataset
- Tuning the hyperparameters using the GridSearchCV
- After finding the best parameters now training the model with the best parameters
- Evaluating the model on the test dataset

```
Best hyperparameters: {'var_smoothing': 1e-09}
Accuracy: 0.82
```

Subpart 2:

- We have already trained the individual models and stored in variables
    - AdaBoost: ada
    - XGBoost: xgb_model
    - LightGBM model: model_lgbm
- Now creating the Ensemble model using the VotingClassifier and using all the above classifiers as estimators
- Training the ensemble model with the train dataset
- Making predictions on the test dataset
- Evaluating the accuracy on the ensemble model
- Evaluating the accuracy of all the individual models

```
Ensemble model accuracy: 0.9050
Adaboost model accuracy: 0.9050
XGBoost model accuracy: 0.9050
LightGBM model accuracy: 0.9050
```