Decision Trees

Name: Aayush Karki

CS 460G- Machine Learning

In this assignment, I have implemented a decision tree classifier that will be used to classify four synthetic datasets and one real datasets.

Collaboration:

I collaborated with Sarthak Rijal.

Environment:

I used Juypter-notebook.

Classify Synthetic Data

For classifying the dataset, I have implemented ID3 algorithm which utilizes entropy and information gain to choose features in each node.

Tree:

I have implemented my own tree. I defined a class called node in the following manner.

```
class Node():
    feature = "no split"
    leaf = False
    prediction = ""
    child1 = None
    child2 = None
    child3 = None
    child4 = None
```

This is what I have done to manage my tree.

Discretize data:

I explicitly used 4 bins to classify my data being bin1, bin2, bin3, bin4. To do this I simply found range of the feature and divided it by total data. This is how I got width and bin would simply be formed based on width, min, and max. And based on the bins, I divided the data in four bins:

Entropy:

To find entropy, I used the following formula:

entropy = ((-1*p)/(p + n))*math.log(p/(p+n), 2) + ((-1*n)/(p + n))*math.log(n/(p+n), 2)

here, I have binary class label: p = positive examples and n is negative examples.

To prevent the log going to infinite, I add a case to check if p or n is 0 which simply means entropy is 0.

Gain and choosing features:

To calculate the gain of a feature I use:

gain = entropy of dataset – entropy of feature

For choosing the feature, I simply compared the gain and choose the one with highest gain!

ID3:

Pseudocode:

```
Took the base cases,
If positive cases and negative cases ==0, depth ==3 or len(dataframe) == 0
Else:
Make child recursively.
```

Prediction:

I traversed the tree to find the output for features.

Synthetic 1: 100% predicted value

Synthetic 2: 93.5% predicted value

Synthetic 3: 83.5% predicted value

Synthetic 4: 91% predicted value

Visualizing Classifier:

To visualize classifier, I took min (-1) and max (+1) of the feature. And for getting grid in the graph, I utilized mesh grid function from NumPy and used it to create a matrix. And, finally using the matrix, I predicted the value of the individual points in the matrix, I colored the respective grid to create a decision boundary.

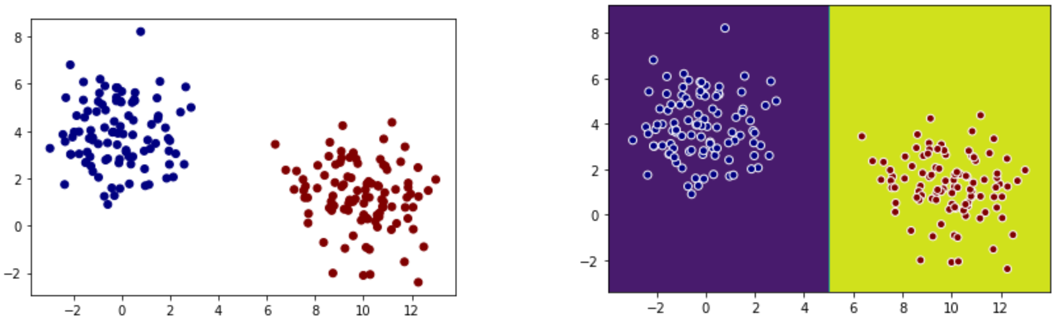

Figure 1: Decision boundary and visualizing the data for Synthetic data-1

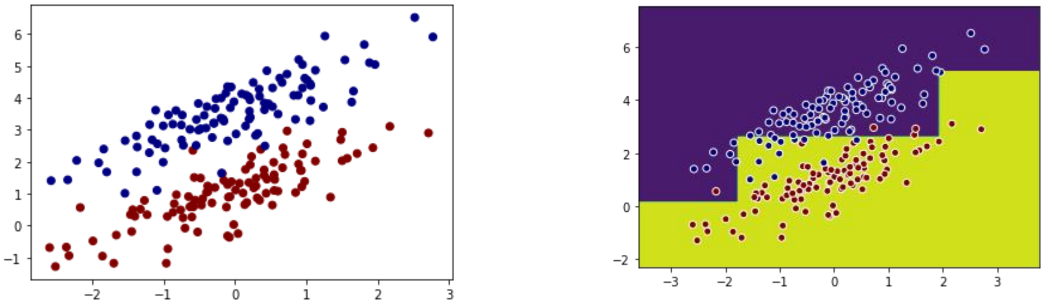

Figure 2: Decision boundary and visualizing the data for Synthetic data-2

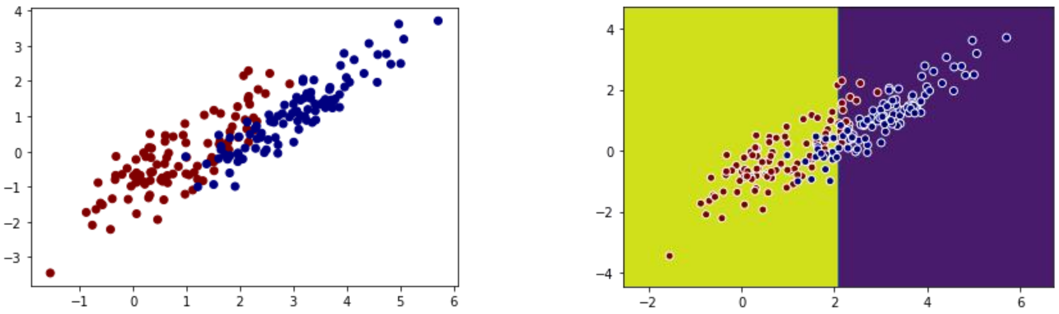

Figure 3: Decision boundary and visualizing the data for Synthetic data-3

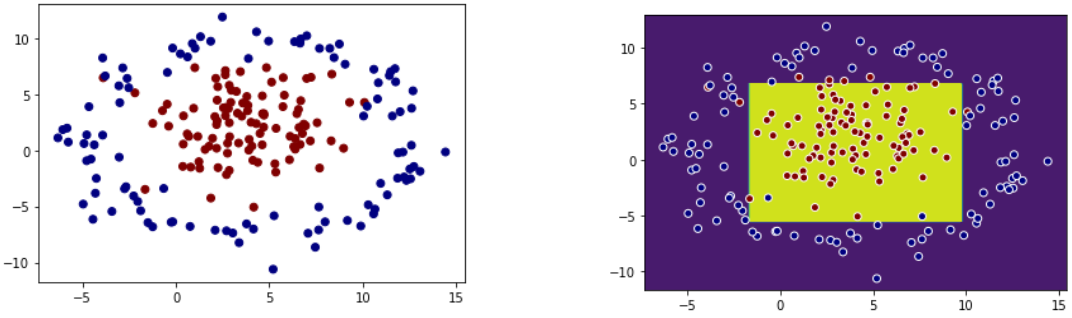

Figure 4: Decision boundary and visualizing the data for Synthetic data-4

Pokémon Dataset:

Most of my code have been reused in the Pokémon dataset. Just the main difference here is:

Main Difference:

- I discretized data for first 7 column:
- And my binning would change greater than 7th column chose created 2 bins while less created 4 bins.

Prediction:

My tree was able to gain **88.50%** accuracy for the dataset.