Harshit Mishra (19BCE0799)

CSE-3024 WEB MINING
LAB ASSIGNMENT 8

Aim: Classify the given network intrusion dataset into normal and anomaly using Decision Tree Classifier. Following things need to be printed along with the classification:

- Confusion Matrix
- Accuracy of model on Test data
- Decision Tree visualization.

Dataset Used: The network intrusion dataset from Kaggle.

Link to which is:
https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection?select=Train_data.csv

Procedure:

- Firstly, we import the necessary libraries of numpy, pandas, matplotlib and tree.
- Next, we import the dataset into our workspace. We also define the set of independent and dependent attribute.
- Next, we split the dataset into training set and test set using a ratio of 7:3.
- Then we train our decision tree model using DecisionTreeClassifier from sklearn.tree
- Next, we find the test set results as predicted by our model.
- Then we print our confusion matrix using predicted result and test set results.
- Similarly, we print the accuracy of our model using test set result and predicted result.
- Finally, using the tree of sklearn, we visualize our model.

Harshit Mishra (19BCE0799)

Code:

```
#Importing libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn import tree


#Importing dataset
dataset = pd.read_csv("Train_data.csv")
X = dataset.iloc[:, 4:41].values
y = dataset.iloc[:, -1].values


#Splitting the dataset
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)


#Fitting our model
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier(criterion = 'entropy' ,random_state = 0)
classifier.fit (X_train, y_train)


#Predicting the Test set Results
y_pred = classifier.predict(X_test)


#Printing the confusion matrix
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print(cm)


#Printing the accuracy of our model
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test, y_pred)
print(accuracy)


#Defining the labels of our dataset
classes = ["Anamoly", "Normal"]
```

#Printing the visualized decision tree
fig = plt.figure(figsize=(25,20))
_ = tree.plot_tree(classifier,
feature_names=dataset.columns,
class_names=classes,
filled=True)

#Printing the feature wise break points of our decision tree
test_representation = tree.export_text(classifier)
print(test_representation)

Code Snippet and Outputs:

```
In [1]: #Importing libraries
        import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd
        from sklearn import tree
```

Here we are importing our libraries. We import nupmy as np, pandas as pd, matplotlib's pyplot extension as plt and finally we import tree from sklearn.

```
In [2]: #Importing dataset
        dataset = pd.read_csv("Train_data.csv")
        X = dataset.iloc[:, 4:41].values
        y = dataset.iloc[:, -1].values
```

Here we are importing our Network Intrusion Dataset into our workspace using pandas. Then we are defining set of dependent and independent attributes. Set of independent attributes are labelled X and dependent ones are labelled y.

```
In [3]: #Splitting the dataset
        from sklearn.model_selection import train_test_split
        X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, rando
```

Here we are splitting our dataset into training set and test set. We are keeping 30% of the dataset into test set and 70% of it in training set.

```
In [4]: #Fitting our model
        from sklearn.tree import DecisionTreeClassifier
        classifier = DecisionTreeClassifier(criterion = 'entropy' ,random_state = 0)
        classifier.fit (X_train, y_train)

Out[4]: DecisionTreeClassifier(criterion='entropy', random_state=0)
```

Here we are training our model using training set data. We have used "entropy" as the decisive criteria for our decision tree classifier.

```
In [5]: #Predicting the Test set Results
        y_pred = classifier.predict(X_test)
```

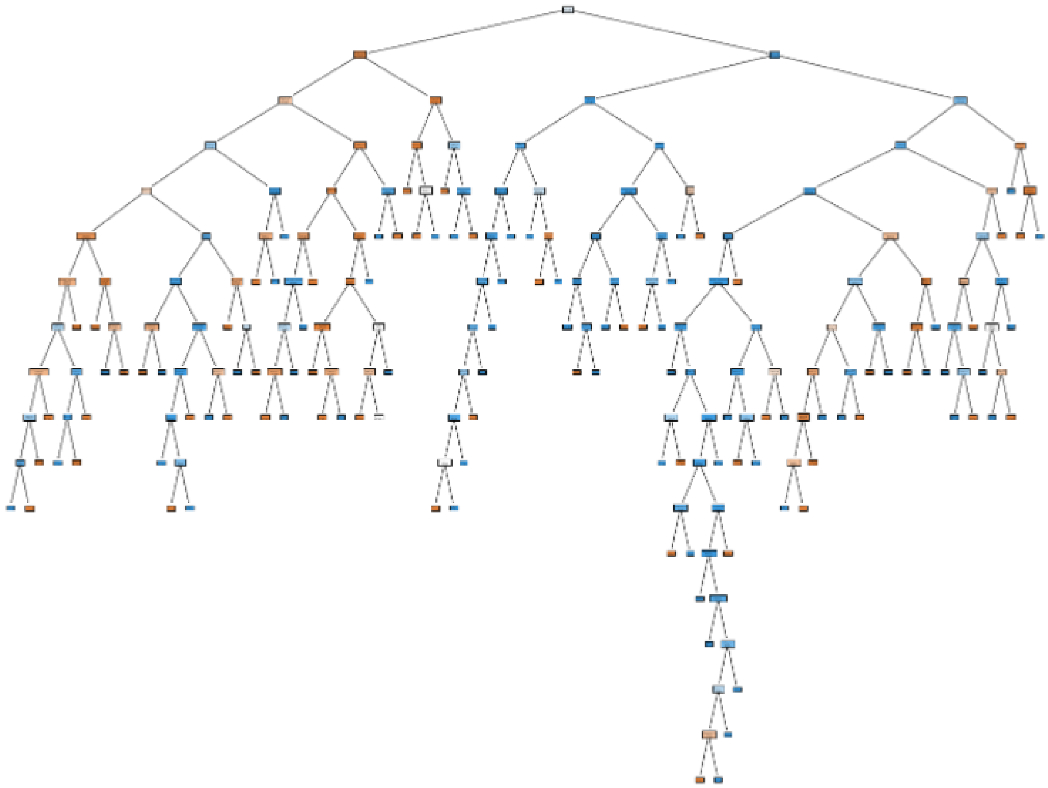
Here we are getting our predicted results of test set from the classifier and then are storing it in y_pred variable.

```
In [6]: #Printing the confusion matrix
        from sklearn.metrics import confusion_matrix
        cm = confusion_matrix(y_test, y_pred)
        print(cm)

        [[3488   17]
         [  21 4032]]

In [7]: #Printing the accuracy of our model
        from sklearn.metrics import accuracy_score
        accuracy = accuracy_score(y_test, y_pred)
        print(accuracy)

        0.9949722148716592
```

Here we are printing the confusion matrix and accuracy of our decision tree classifier. The accuracy of our model with test dataset is 99.49%.

Harshit Mishra (19BCE0799)

```
In [9]: #Printing the visualized decision tree
        fig = plt.figure(figsize=(25,20))
        _ = tree.plot_tree(classifier,
                           feature_names=dataset.columns,
                           class_names=classes,
                           filled=True)
```



Here we are visualizing our decision tree using sklearn's tree library

```
In [10]: #Printing the feature wise break points of our decision tree
         test_representation = tree.export_text(classifier)
         print(test_representation)
```

```
|--- feature_0 <= 28.50
|   |--- feature_18 <= 8.50
|   |   |--- feature_31 <= 0.50
|   |   |   |--- feature_29 <= 0.53
|   |   |   |   |--- feature_0 <= 5.50
|   |   |   |   |   |--- feature_35 <= 0.09
|   |   |   |   |   |   |--- feature_33 <= 0.96
|   |   |   |   |   |   |   |--- feature_28 <= 2.50
|   |   |   |   |   |   |   |   |--- feature_35 <= 0.01
|   |   |   |   |   |   |   |   |   |--- feature_18 <= 4.50
|   |   |   |   |   |   |   |   |   |   |--- feature_0 <= 0.50
|   |   |   |   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |   |   |   |   |--- feature_0 >  0.50
|   |   |   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |   |   |--- feature_18 >  4.50
|   |   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |   |--- feature_35 >  0.01
|   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |--- feature_28 >  2.50
```

Here we are printing the classification criterion of our decision tree. We can see that feature_0 lays as the root node for our classifier followed by several middle nodes.

Results:

Confusion Matrix:

```
[[3488   17]
 [  21 4032]]
```

This is our confusion matrix.

True Negatives: 3488

True Positives: 4032

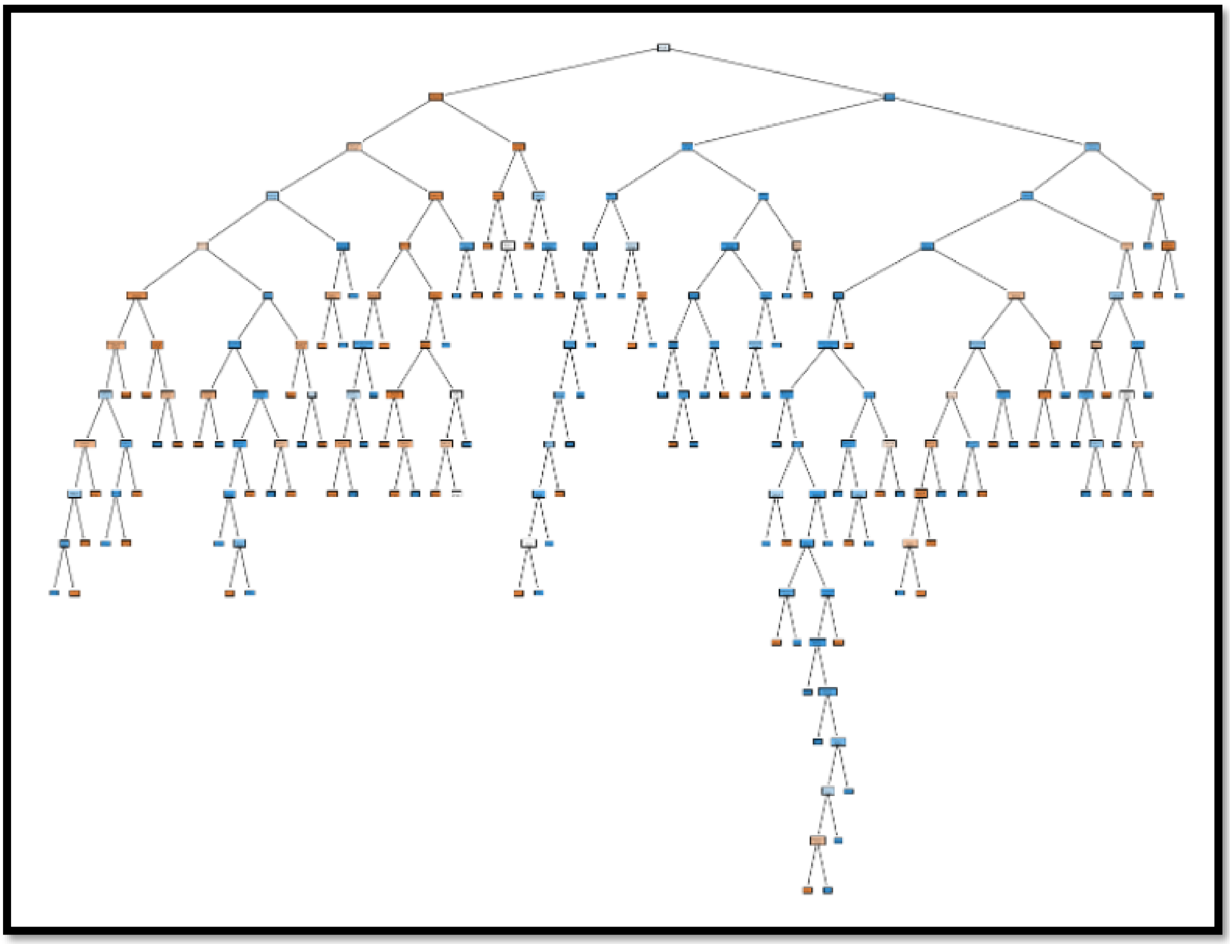
False Positives: 17

False Negatives: 21

Accuracy:

The accuracy of our model stands at 99.49%

Decision Tree Visualization:



Harshit Mishra (19BCE0799)

Classification Points:

```
|--- feature_0 <= 28.50
|   |--- feature_18 <= 8.50
|   |   |--- feature_31 <= 0.50
|   |   |   |--- feature_29 <= 0.53
|   |   |   |   |--- feature_0 <= 5.50
|   |   |   |   |   |--- feature_35 <= 0.09
|   |   |   |   |   |   |--- feature_33 <= 0.96
|   |   |   |   |   |   |   |--- feature_28 <= 2.50
|   |   |   |   |   |   |   |   |--- feature_35 <= 0.01
|   |   |   |   |   |   |   |   |   |--- feature_18 <= 4.50
|   |   |   |   |   |   |   |   |   |   |--- feature_0 <= 0.50
|   |   |   |   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |   |   |   |   |--- feature_0 >  0.50
|   |   |   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |   |   |--- feature_18 >  4.50
|   |   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |   |--- feature_35 >  0.01
|   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |--- feature_28 >  2.50
```

```
|   |   |   |   |   |   |   |   |--- feature_26 <= 0.26
|   |   |   |   |   |   |   |   |   |--- feature_0 <= 2.50
|   |   |   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |   |   |   |--- feature_0 >  2.50
|   |   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |   |--- feature_26 >  0.26
|   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |--- feature_33 >  0.96
|   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |--- feature_35 >  0.09
|   |   |   |   |   |   |--- feature_29 <= 0.21
|   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |--- feature_29 >  0.21
|   |   |   |   |   |   |   |--- feature_31 <= 0.18
|   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |   |--- feature_31 >  0.18
|   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |--- feature_0 >  5.50
|   |   |   |   |   |--- feature_0 <= 19.50
```

```
|   |   |   |   |   |   |--- feature_28 <= 1.50
|   |   |   |   |   |   |   |--- feature_30 <= 0.47
|   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |--- feature_30 >  0.47
|   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |--- feature_28 >  1.50
|   |   |   |   |   |   |   |--- feature_30 <= 0.72
|   |   |   |   |   |   |   |   |--- feature_1 <= 1039.50
|   |   |   |   |   |   |   |   |   |--- feature_26 <= 0.84
|   |   |   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |   |   |   |--- feature_26 >  0.84
|   |   |   |   |   |   |   |   |   |   |--- feature_29 <= 0.01
|   |   |   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |   |   |   |--- feature_29 >  0.01
|   |   |   |   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |   |   |--- feature_1 >  1039.50
|   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |   |--- feature_30 >  0.72
|   |   |   |   |   |   |   |   |--- feature_31 <= 0.30
```

```
|   |   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |   |   |--- feature_31 >  0.30
|   |   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |--- feature_0 >  19.50
|   |   |   |   |   |   |--- feature_29 <= 0.08
|   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |   |--- feature_29 >  0.08
|   |   |   |   |   |   |   |--- feature_3 <= 1.50
|   |   |   |   |   |   |   |   |--- class: normal
|   |   |   |   |   |   |   |--- feature_3 >  1.50
|   |   |   |   |   |   |   |   |--- class: anomaly
|   |   |   |--- feature_29 >  0.53
|   |   |   |   |--- feature_28 <= 4.00
|   |   |   |   |   |--- feature_30 <= 0.25
|   |   |   |   |   |   |--- class: anomaly
|   |   |   |   |   |--- feature_30 >  0.25
|   |   |   |   |   |   |--- class: normal
```