NAME - NEGI AMIT SINGH
ENROLLEMENT - 23552006

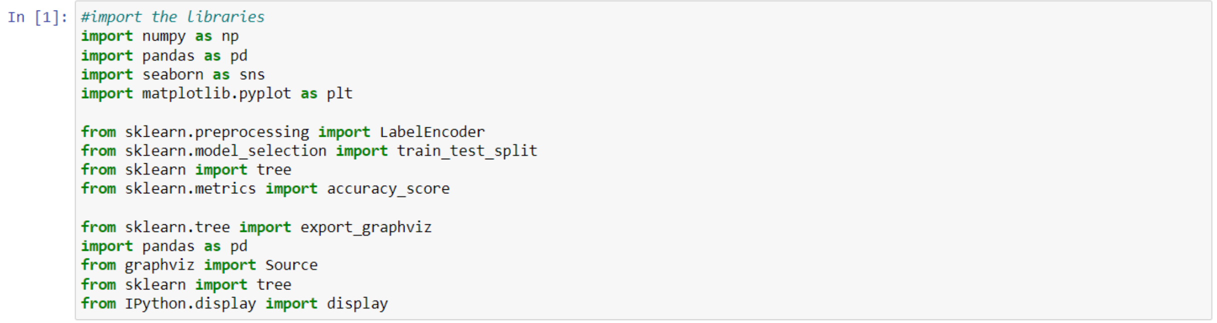
Problem statement: Iris Flower Classification Project using Machine Learning

Steps to Classify Iris Flower

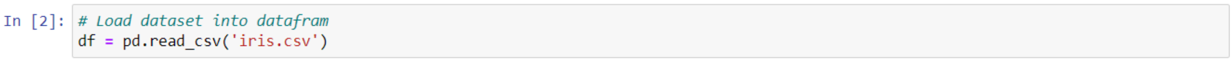
1. Load the data
2. Analyze and visualize the dataset
3. Model training.
4. Model Evaluation.
5. Testing the model.

Load the data.

- First, we've imported some necessary packages for the project.
- Next, we load the data using pd.read_csv().

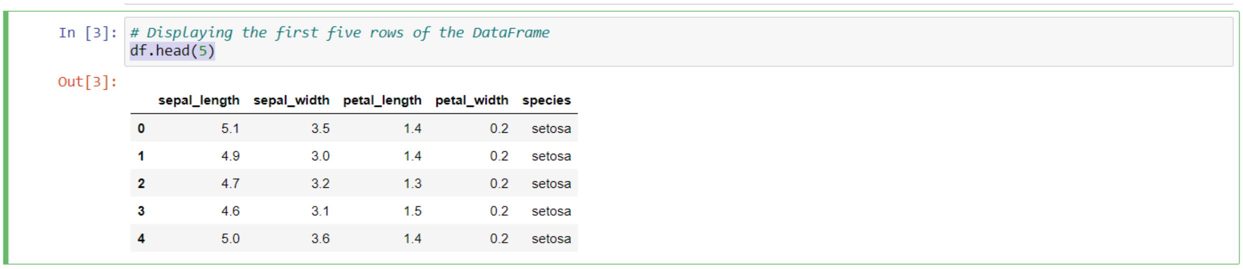


```
In [1]: #import the libraries
        import numpy as np
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt

        from sklearn.preprocessing import LabelEncoder
        from sklearn.model_selection import train_test_split
        from sklearn import tree
        from sklearn.metrics import accuracy_score

        from sklearn.tree import export_graphviz
        import pandas as pd
        from graphviz import Source
        from sklearn import tree
        from IPython.display import display
```



```
In [2]: # Load dataset into datafram
        df = pd.read_csv('iris.csv')
```

- df.head() only shows the first 5 rows from the data set table.



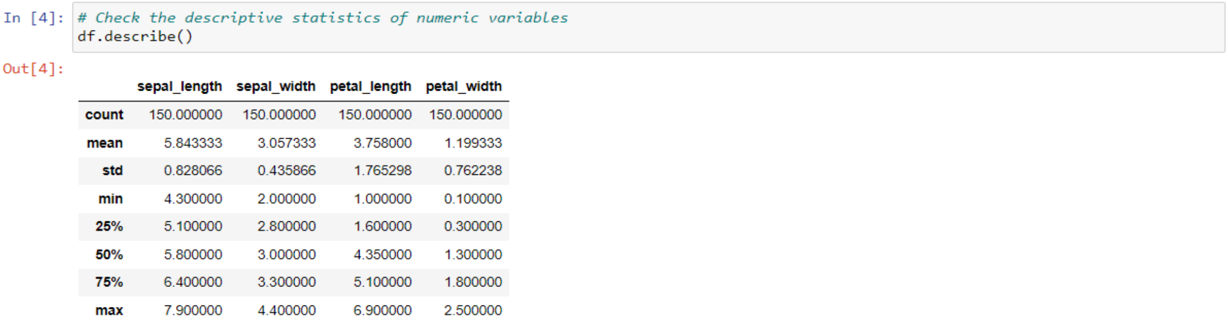
```
In [3]: # Displaying the first five rows of the DataFrame
        df.head(5)
```

Out[3]:

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

Step 2 – Analyze and visualize the dataset:

- From this description, we can see all the descriptions about the data, like average length and width, minimum value, maximum value, the 25%, 50%, and 75% distribution value, etc.

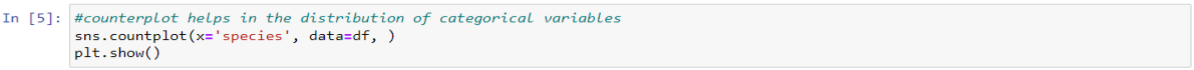


```
In [4]: # Check the descriptive statistics of numeric variables
        df.describe()
```

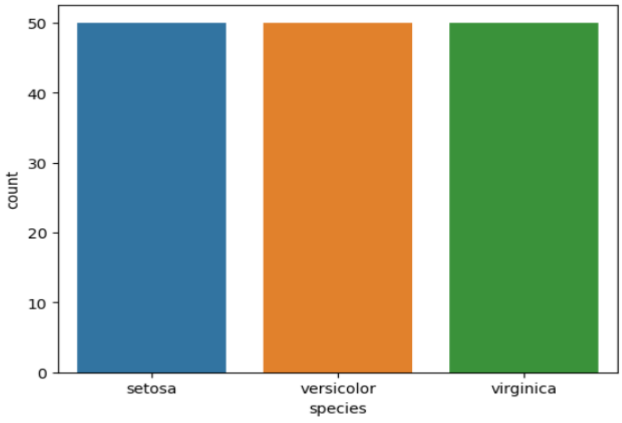
Out[4]:

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

- To visualize counting the occurrences of different categories in a dataset and visualizing them. We used seaborn libraries for count plot.

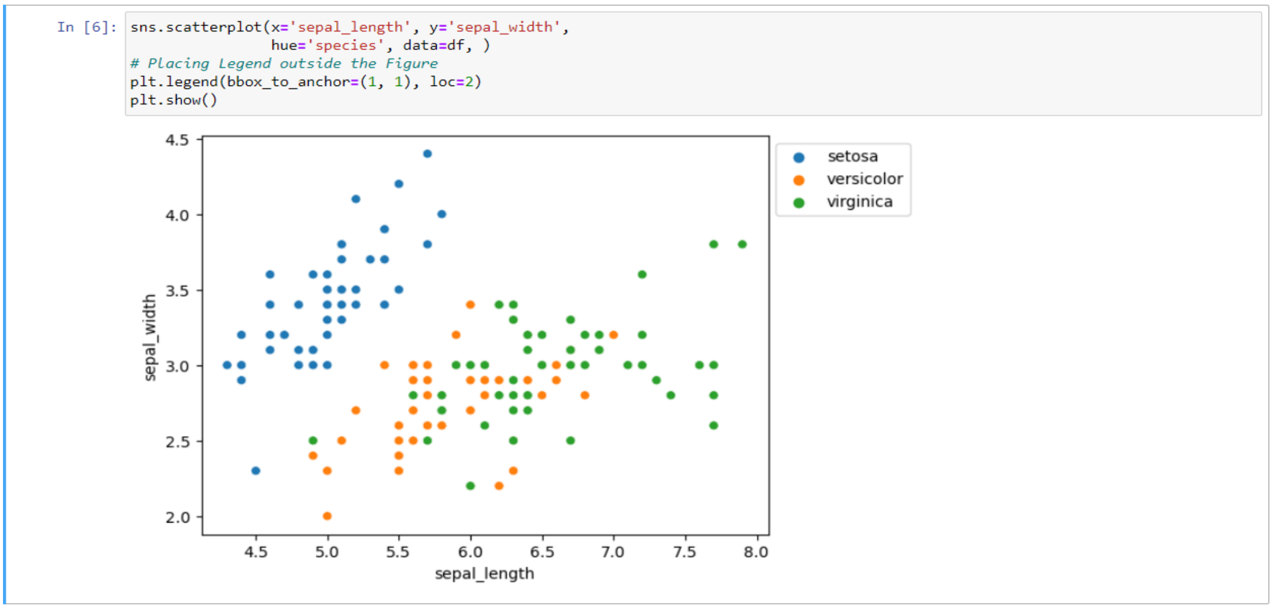


```
In [5]: #counterplot helps in the distribution of categorical variables
        sns.countplot(x='species', data=df, )
        plt.show()
```



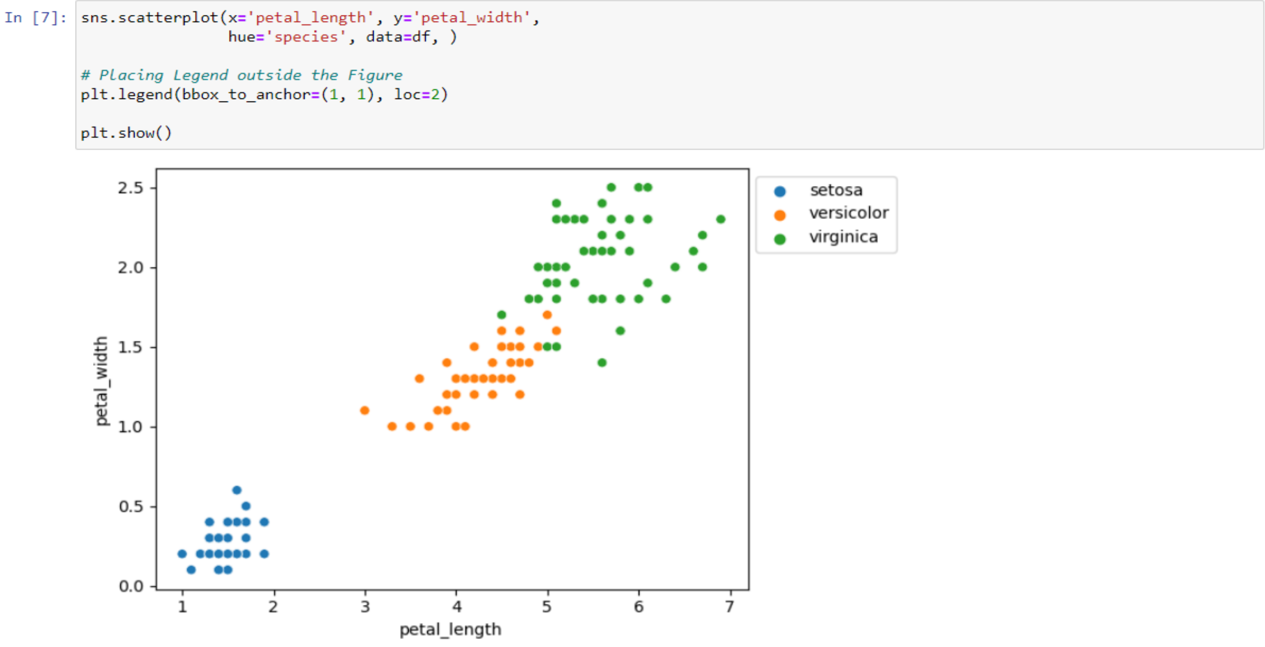
- we can see that all the species have equally balance.

- We will see the relationship between the sepal length and sepal width and also between petal length and petal width.



```
In [6]: sns.scatterplot(x='sepal_length', y='sepal_width',
                        hue='species', data=df, )
        # Placing Legend outside the Figure
        plt.legend(bbox_to_anchor=(1, 1), loc=2)
        plt.show()
```

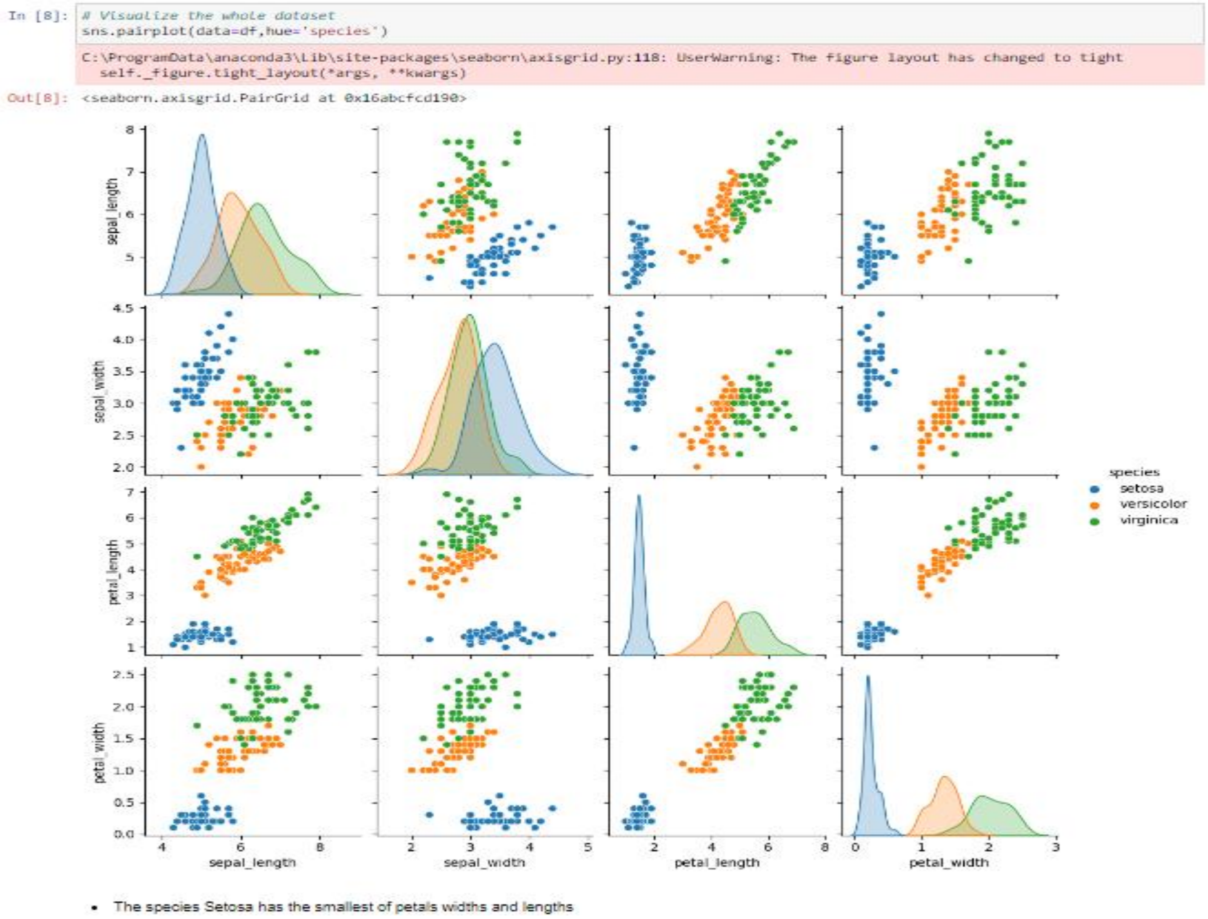
- Species Setosa has smaller sepal lengths but larger sepal widths.
- Versicolor Species lies in the middle of the other two species in terms of sepal length and width
- Species Virginica has larger sepal lengths but smaller sepal widths.



```
In [7]: sns.scatterplot(x='petal_length', y='petal_width',
                        hue='species', data=df, )

        # Placing Legend outside the Figure
        plt.legend(bbox_to_anchor=(1, 1), loc=2)

        plt.show()
```

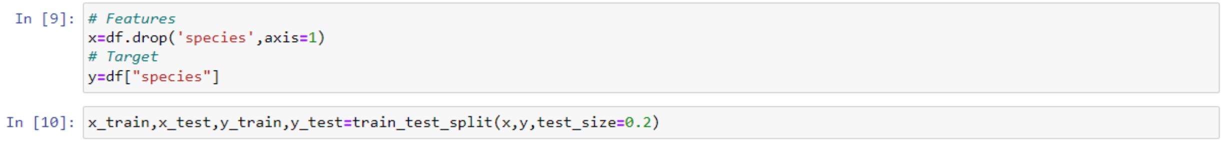
- Species Setosa has smaller petal lengths and widths.
- Versicolor Species lies in the middle of the other two species in terms of petal length and width
- Species Virginica has the largest of petal lengths and widths.

- To visualize the whole dataset, we used the seaborn pair plot method. It plots the whole dataset's information.



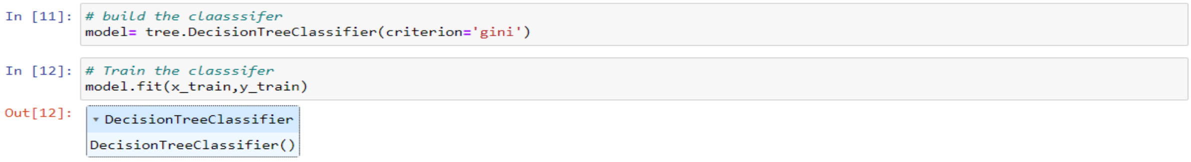
```
In [8]: # Visualize the whole dataset
        sns.pairplot(data=df,hue='species')

        C:\ProgramData\anaconda3\Lib\site-packages\seaborn\axisgrid.py:118: UserWarning: The figure layout has changed to tight
          self._figure.tight_layout(*args, **kwargs)

Out[8]: <seaborn.axisgrid.PairGrid at 0x16abcfcd190>
```

- The species Setosa has the smallest of petals widths and lengths



```
In [9]: # Features
        x=df.drop('species',axis=1)
        # Target
        y=df["species"]

In [10]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
```

Model training



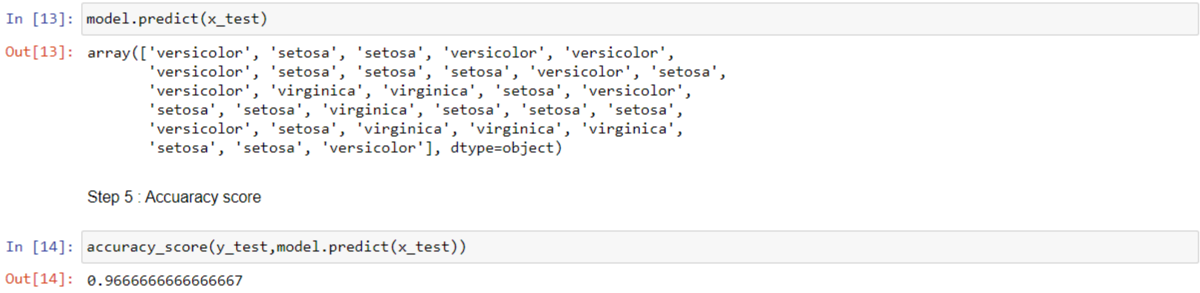
```
In [11]: # build the claasssifer
         model= tree.DecisionTreeClassifier(criterion='gini')

In [12]: # Train the classsifer
         model.fit(x_train,y_train)

Out[12]: ▸ DecisionTreeClassifier
           DecisionTreeClassifier()
```

- Using train_test_split we split the whole data into training and testing datasets. Later we'll use the testing dataset to check the accuracy of the model.

- we created an object and named it model(Decision tree classsifer).
- The training dataset into the algorithm by using the model. fit() method

Model Evaluation:

- we predict the classes from the test dataset using our trained model.



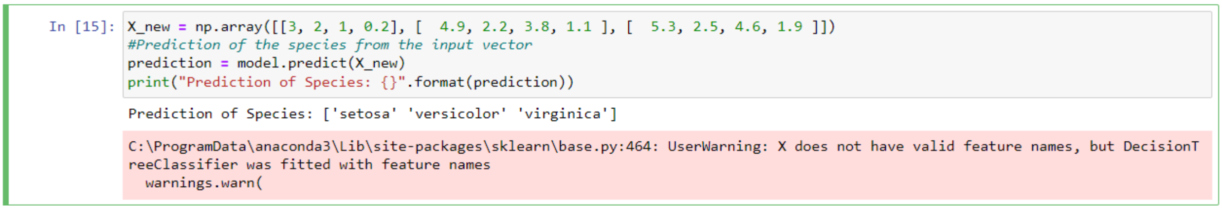
```
In [13]: model.predict(x_test)

Out[13]: array(['versicolor', 'setosa', 'setosa', 'versicolor', 'versicolor',
                'versicolor', 'setosa', 'setosa', 'setosa', 'versicolor', 'setosa',
                'versicolor', 'virginica', 'virginica', 'setosa', 'versicolor',
                'setosa', 'setosa', 'virginica', 'setosa', 'setosa', 'setosa',
                'versicolor', 'setosa', 'virginica', 'virginica', 'virginica',
                'setosa', 'setosa', 'versicolor'], dtype=object)
```

Step 5 : Accuaracy score

```
In [14]: accuracy_score(y_test,model.predict(x_test))

Out[14]: 0.9666666666666667
```

- Then we check the accuracy score of the predicted classes.
- accuracy.score() takes true values and predicted values and returns the percentage of accuracy.

- The accuracy is above 96%.
-



```
In [15]: X_new = np.array([[3, 2, 1, 0.2], [  4.9, 2.2, 3.8, 1.1 ], [  5.3, 2.5, 4.6, 1.9 ]])
         #Prediction of the species from the input vector
         prediction = model.predict(X_new)
         print("Prediction of Species: {}".format(prediction))

         Prediction of Species: ['setosa' 'versicolor' 'virginica']

         C:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X does not have valid feature names, but DecisionT
         reeClassifier was fitted with feature names
           warnings.warn(
```

Testing the model

Input: - we take some random values based on the average plot to see if the model can predict accurately.

Output:

Prediction of Species: [‘setosa’ ,‘versicolor’ ,‘virginica’]

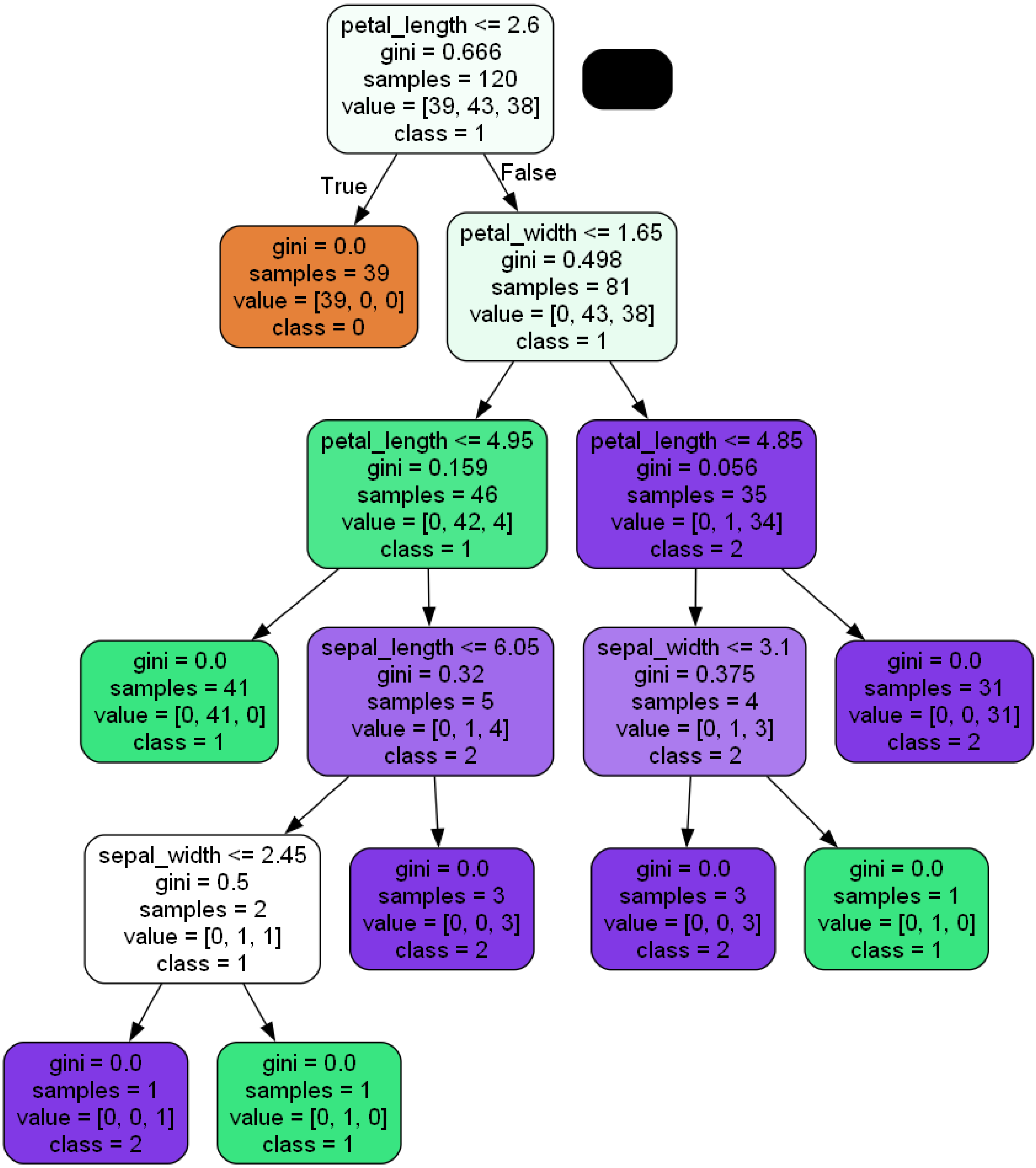
Summary

In this project, we learned to train our own supervised machine learning model using Iris Flower Classification Project with Machine Learning.

Through this project, we learned about machine learning, data analysis, data visualization, model creation.

NAME - NEGI AMIT SINGH
ENROLLEMENT - 23552006

Visualization of Decision tree using Graph Viz


petal_length <= 2.6
gini = 0.666
samples = 120
value = [39, 43, 38]
class = 1
[redacted]
True
False
gini = 0.0
samples = 39
value = [39, 0, 0]
class = 0
petal_width <= 1.65
gini = 0.498
samples = 81
value = [0, 43, 38]
class = 1
petal_length <= 4.95
gini = 0.159
samples = 46
value = [0, 42, 4]
class = 1
petal_length <= 4.85
gini = 0.056
samples = 35
value = [0, 1, 34]
class = 2
gini = 0.0
samples = 41
value = [0, 41, 0]
class = 1
sepal_length <= 6.05
gini = 0.32
samples = 5
value = [0, 1, 4]
class = 2
sepal_width <= 3.1
gini = 0.375
samples = 4
value = [0, 1, 3]
class = 2
gini = 0.0
samples = 31
value = [0, 0, 31]
class = 2
sepal_width <= 2.45
gini = 0.5
samples = 2
value = [0, 1, 1]
class = 1
gini = 0.0
samples = 3
value = [0, 0, 3]
class = 2
gini = 0.0
samples = 3
value = [0, 0, 3]
class = 2
gini = 0.0
samples = 1
value = [0, 1, 0]
class = 1
gini = 0.0
samples = 1
value = [0, 0, 1]
class = 2
gini = 0.0
samples = 1
value = [0, 1, 0]
class = 1