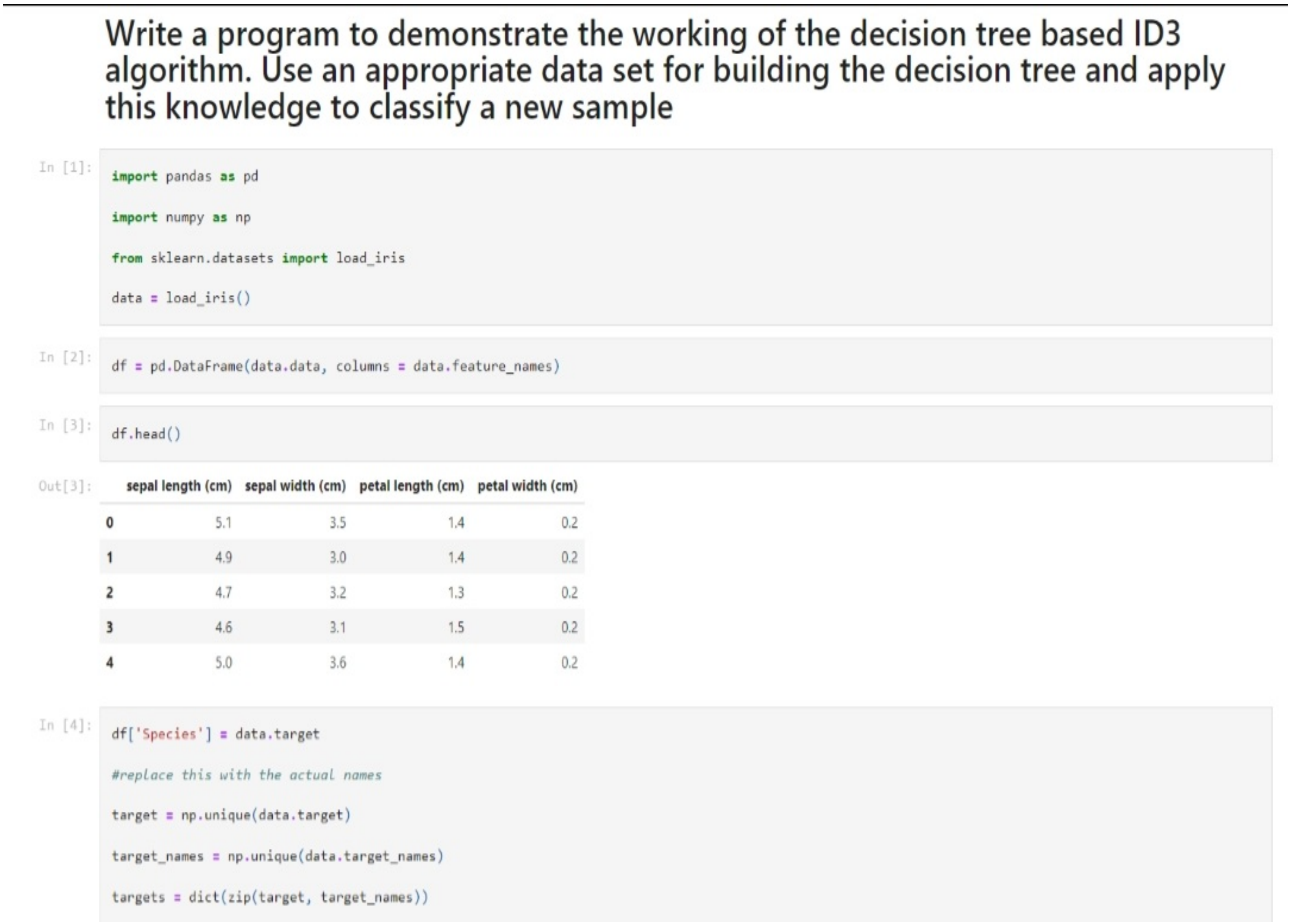
NEEHAL
1BM19CS097

LAB 3



Write a program to demonstrate the working of the decision tree based ID3 algorithm. Use an appropriate data set for building the decision tree and apply this knowledge to classify a new sample

In [1]:

```
import pandas as pd

import numpy as np

from sklearn.datasets import load_iris

data = load_iris()
```

In [2]:

```
df = pd.DataFrame(data.data, columns = data.feature_names)
```

In [3]:

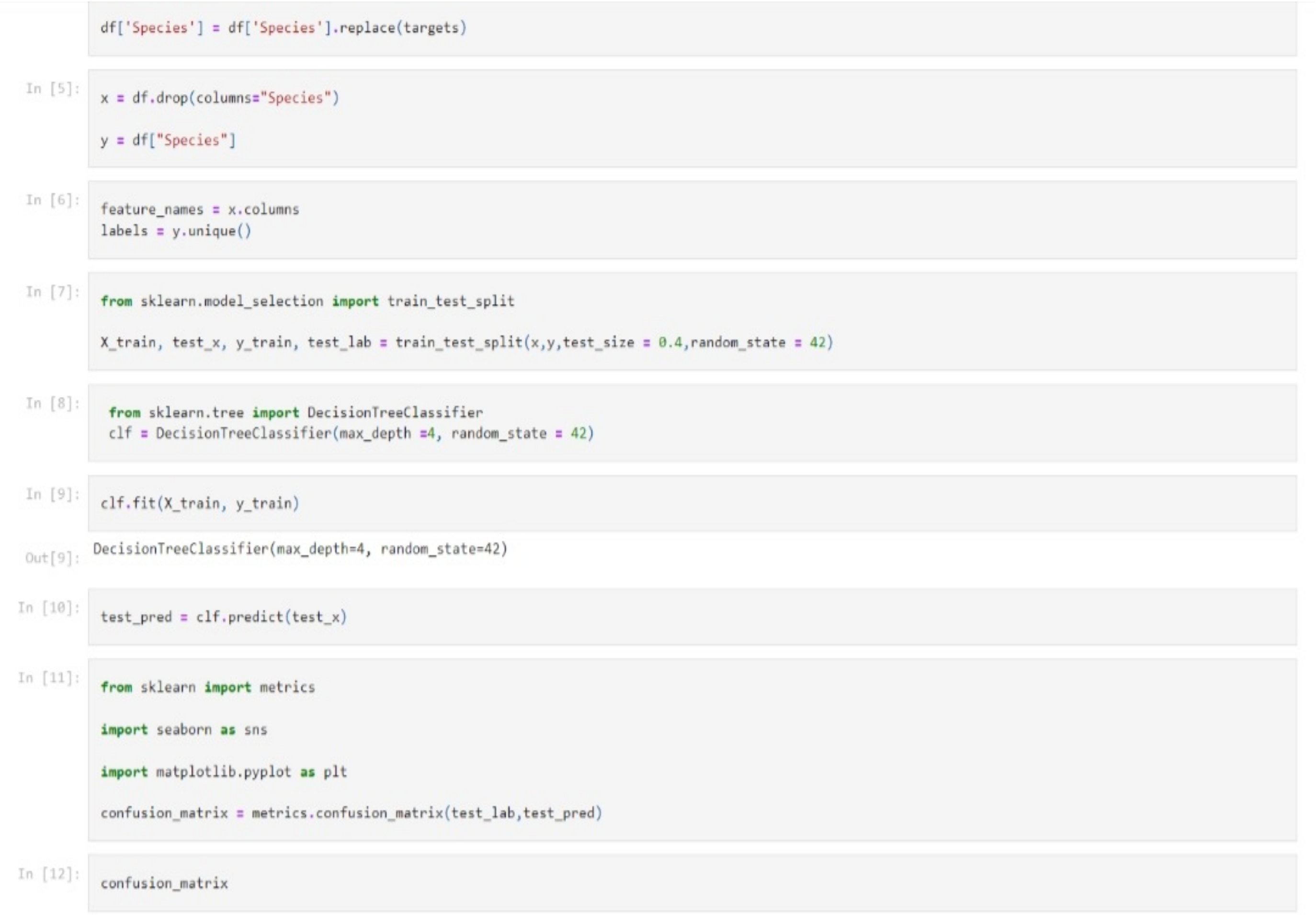
```
df.head()
```

Out[3]:

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [4]:

```
df['Species'] = data.target

#replace this with the actual names

target = np.unique(data.target)

target_names = np.unique(data.target_names)

targets = dict(zip(target, target_names))
```



```
df['Species'] = df['Species'].replace(targets)
```

In [5]:

```
x = df.drop(columns="Species")

y = df["Species"]
```

In [6]:

```
feature_names = x.columns
labels = y.unique()
```

In [7]:

```
from sklearn.model_selection import train_test_split

X_train, test_x, y_train, test_lab = train_test_split(x,y,test_size = 0.4,random_state = 42)
```

In [8]:

```
 from sklearn.tree import DecisionTreeClassifier
 clf = DecisionTreeClassifier(max_depth =4, random_state = 42)
```

In [9]:

```
clf.fit(X_train, y_train)
```

Out[9]:

```
DecisionTreeClassifier(max_depth=4, random_state=42)
```

In [10]:

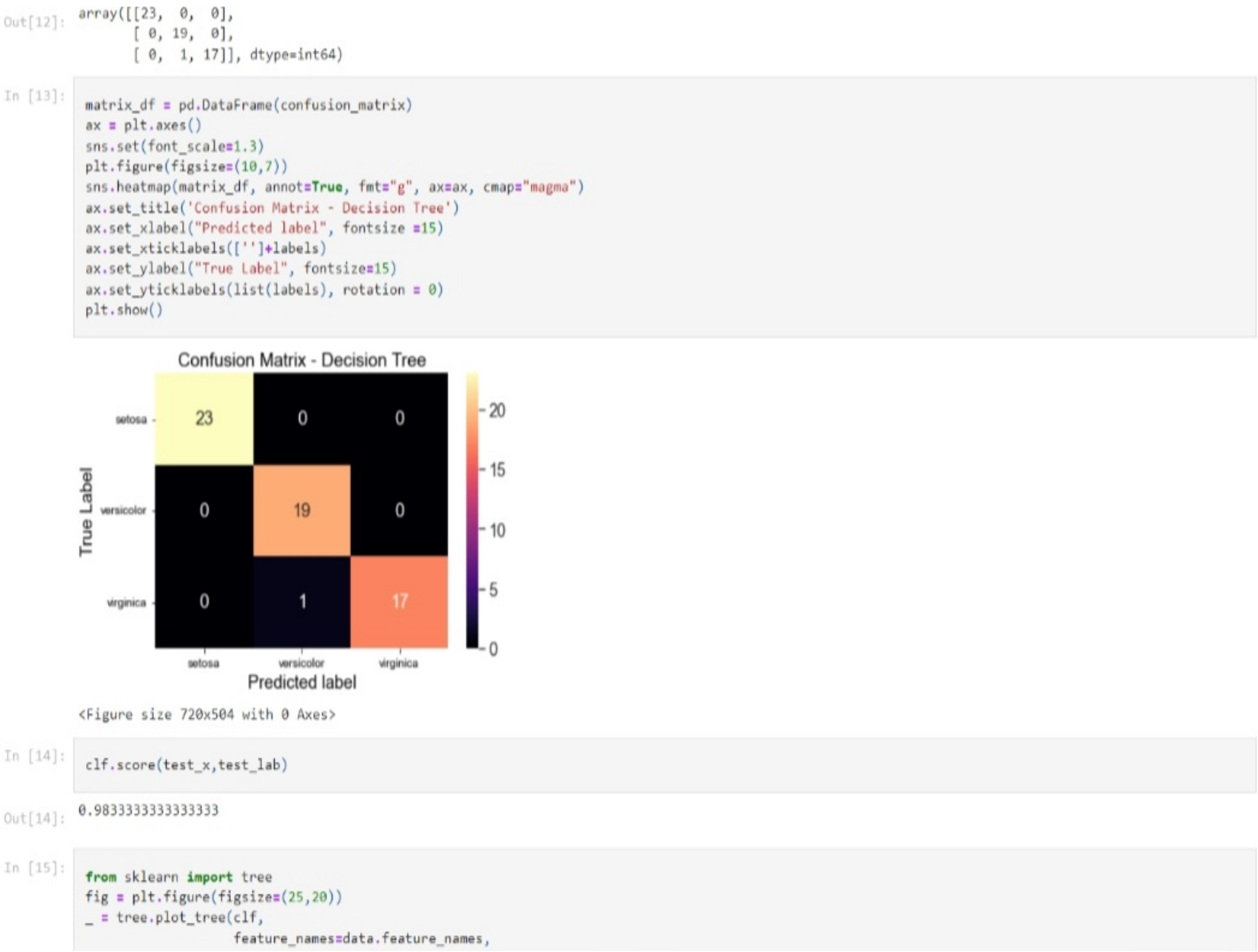
```
test_pred = clf.predict(test_x)
```

In [11]:

```
from sklearn import metrics

import seaborn as sns

import matplotlib.pyplot as plt

confusion_matrix = metrics.confusion_matrix(test_lab,test_pred)
```

In [12]:

```
confusion_matrix
```



```
Out[12]: array([[23,  0,  0],
                [ 0, 19,  0],
                [ 0,  1, 17]], dtype=int64)
```

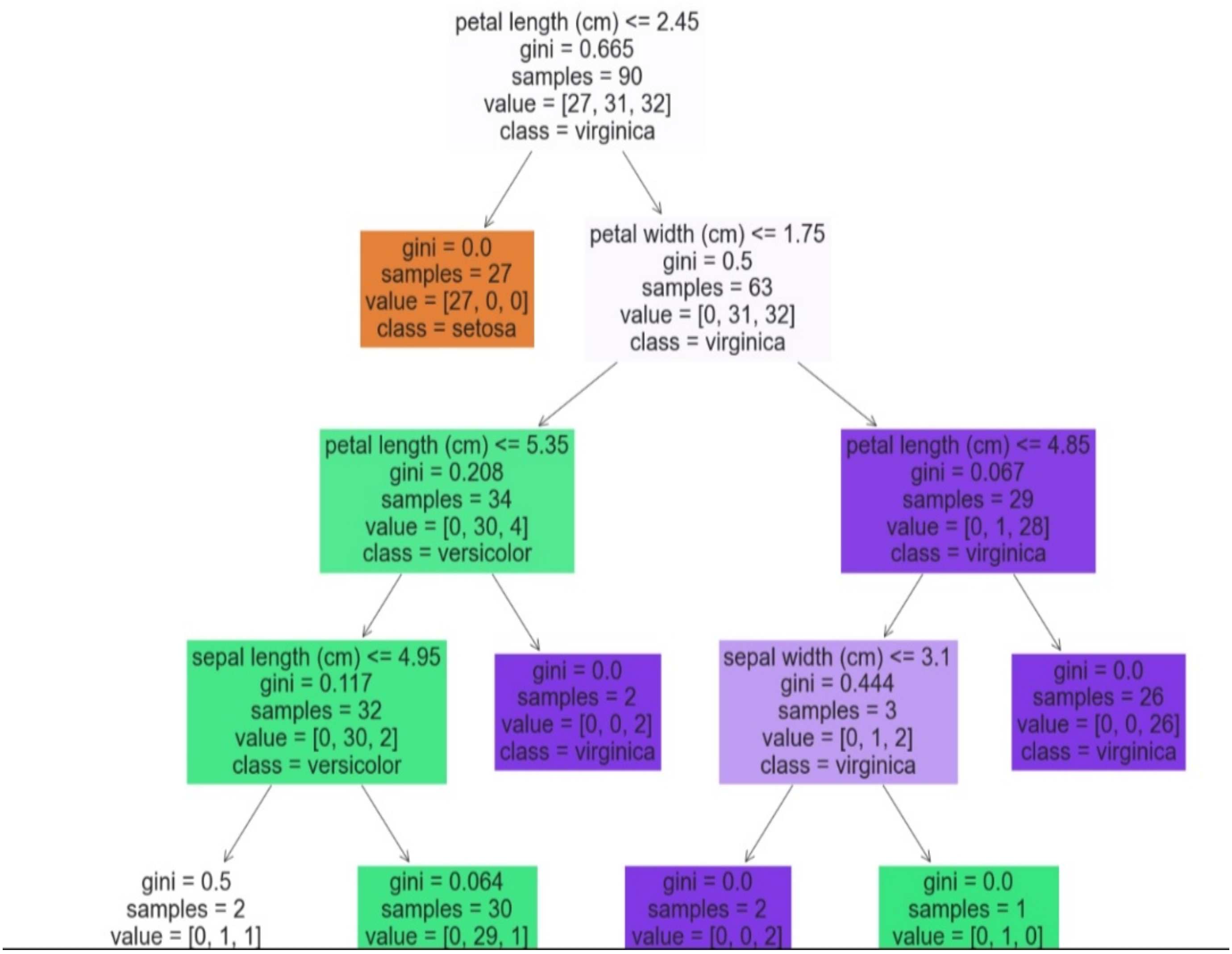
```
In [13]:
matrix_df = pd.DataFrame(confusion_matrix)
ax = plt.axes()
sns.set(font_scale=1.3)
plt.figure(figsize=(10,7))
sns.heatmap(matrix_df, annot=True, fmt="g", ax=ax, cmap="magma")
ax.set_title('Confusion Matrix - Decision Tree')
ax.set_xlabel("Predicted label", fontsize =15)
ax.set_xticklabels(['']+labels)
ax.set_ylabel("True Label", fontsize=15)
ax.set_yticklabels(list(labels), rotation = 0)
plt.show()
```

```
<Figure size 720x504 with 0 Axes>
```

```
In [14]:
clf.score(test_x,test_lab)
```

```
Out[14]: 0.9833333333333333
```

```
In [15]:
from sklearn import tree
fig = plt.figure(figsize=(25,20))
_ = tree.plot_tree(clf,
                   feature_names=data.feature_names,
```


petal length (cm) <= 2.45
gini = 0.665
samples = 90
value = [27, 31, 32]
class = virginica
gini = 0.0
samples = 27
value = [27, 0, 0]
class = setosa
petal width (cm) <= 1.75
gini = 0.5
samples = 63
value = [0, 31, 32]
class = virginica
petal length (cm) <= 5.35
gini = 0.208
samples = 34
value = [0, 30, 4]
class = versicolor
petal length (cm) <= 4.85
gini = 0.067
samples = 29
value = [0, 1, 28]
class = virginica
sepal length (cm) <= 4.95
gini = 0.117
samples = 32
value = [0, 30, 2]
class = versicolor
gini = 0.0
samples = 2
value = [0, 0, 2]
class = virginica
sepal width (cm) <= 3.1
gini = 0.444
samples = 3
value = [0, 1, 2]
class = virginica
gini = 0.0
samples = 26
value = [0, 0, 26]
class = virginica
gini = 0.5
samples = 2
value = [0, 1, 1]
gini = 0.064
samples = 30
value = [0, 29, 1]
gini = 0.0
samples = 2
value = [0, 0, 2]
gini = 0.0
samples = 1
value = [0, 1, 0]