LAB # 05

SUPERVISED LEARNING (DECISION TREE)

Lab Tasks:-

1. Implement the Decision tree algorithm on the data given in the table. 1 and predict the new entry entered by the user.

Table. 1

	Gender	Height	Weight	Foot_Size
0	male	6.00	180	12
1	male	5.92	190	11
2	male	5.58	170	12
3	male	5.92	165	10
4	female	5.00	100	6
5	female	5.50	150	8
6	female	5.42	130	7
7	female	5.75	150	9

Code:-

```
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn import tree
import matplotlib.pyplot as plt

data = {
    'Gender': ['male', 'male', 'male', 'male', 'female', 'female', 'female', 'female'],
    'Height': [6.00, 5.92, 5.58, 5.92, 5.00, 5.50, 5.42, 5.75],
    'Weight': [180, 190, 170, 165, 100, 150, 130, 150],
    'Foot_Size': [12, 11, 12, 10, 6, 8, 7, 9]
}
df = pd.DataFrame(data)

le = LabelEncoder()
df['Gender'] = le.fit_transform(df['Gender'])  # male=1, female=0

X = df[['Height', 'Weight', 'Foot_Size']]
y = df['Gender']

model = DecisionTreeClassifier(random_state=0)
model.fit(X, y)

print("Enter details for prediction:")
h = float(input("Enter Height (in ft): "))
w = float(input("Enter Weight (in lbs): "))
f = float(input("Enter Foot Size: "))

new_entry = [[h, w, f]]
prediction = model.predict(new_entry)
predicted_gender = le.inverse_transform(prediction)[0]

print("\nPredicted Gender:", predicted_gender)
```

Output:-

```
Enter details for prediction:
Enter Height (in ft):  6
Enter Weight (in lbs):  197
Enter Foot Size:  8

Predicted Gender: male
```

2. Implement Decision Tree using table. 1 in such a way that the new entry becomes the part of the given dataset.

Code:-

```python
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn import tree
import matplotlib.pyplot as plt

data = {
    'Gender': ['male', 'male', 'male', 'male', 'female', 'female', 'female', 'female'],
    'Height': [6.00, 5.92, 5.58, 5.92, 5.00, 5.50, 5.42, 5.75],
    'Weight': [180, 190, 170, 165, 100, 150, 130, 150],
    'Foot_Size': [12, 11, 12, 10, 6, 8, 7, 9]
}
df = pd.DataFrame(data)

le = LabelEncoder()
df['Gender'] = le.fit_transform(df['Gender'])  # male=1, female=0

X = df[['Height', 'Weight', 'Foot_Size']]
y = df['Gender']

model = DecisionTreeClassifier(random_state=0)
model.fit(X, y)

print("Enter details for new entry:")
h = float(input("Enter Height (in ft): "))
w = float(input("Enter Weight (in lbs): "))
f = float(input("Enter Foot Size: "))

new_entry = [[h, w, f]]
prediction = model.predict(new_entry)
predicted_gender = le.inverse_transform(prediction)[0]

print("\nPredicted Gender:", predicted_gender)

new_row = {'Gender': prediction[0], 'Height': h, 'Weight': w, 'Foot_Size': f}
df = pd.concat([df, pd.DataFrame([new_row])], ignore_index=True)

df['Gender'] = le.inverse_transform(df['Gender'])

print("\nUpdated Dataset:")
print(df)
```

Output:-

```
Enter details for new entry:
Enter Height (in ft):  5
Enter Weight (in lbs):  120
Enter Foot Size:  7

Predicted Gender: female

Updated Dataset:
   Gender  Height  Weight  Foot_Size
0    male    6.00   180.0       12.0
1    male    5.92   190.0       11.0
2    male    5.58   170.0       12.0
3    male    5.92   165.0       10.0
4  female    5.00   100.0        6.0
5  female    5.50   150.0        8.0
6  female    5.42   130.0        7.0
7  female    5.75   150.0        9.0
8  female    5.00   120.0        7.0
```

3. Implement Decision Tree using table. 1 without the use of Pandas library. You can use numpy.

Code:-

```
import numpy as np
from sklearn.tree import DecisionTreeClassifier
from sklearn.preprocessing import LabelEncoder

genders = np.array(['male', 'male', 'male', 'male', 'female', 'female', 'female', 'female'])
heights = np.array([6.00, 5.92, 5.58, 5.92, 5.00, 5.50, 5.42, 5.75])
weights = np.array([180, 190, 170, 165, 100, 150, 130, 150])
foot_sizes = np.array([12, 11, 12, 10, 6, 8, 7, 9])

X = np.column_stack((heights, weights, foot_sizes))

le = LabelEncoder()
y = le.fit_transform(genders)

model = DecisionTreeClassifier(random_state=0)
model.fit(X, y)

print("Enter new entry for prediction:")
h = float(input("Enter Height (in ft): "))
w = float(input("Enter Weight (in lbs): "))
f = float(input("Enter Foot Size: "))

new_entry = np.array([[h, w, f]])
prediction = model.predict(new_entry)
predicted_gender = le.inverse_transform(prediction)[0]

print("\nPredicted Gender:", predicted_gender)

X = np.vstack([X, new_entry])
y = np.append(y, prediction)

print("\nUpdated Dataset:")
for i in range(len(X)):
    print(f"{i+1}. Height={X[i][0]}, Weight={X[i][1]}, Foot_Size={X[i][2]}, Gender={le.inverse_transform([y[i]])[0]}")
```

Output:-

```
Enter new entry for prediction:
Enter Height (in ft):  7
Enter Weight (in lbs):  200
Enter Foot Size:  9

Predicted Gender: male

Updated Dataset:
1. Height=6.0, Weight=180.0, Foot_Size=12.0, Gender=male
2. Height=5.92, Weight=190.0, Foot_Size=11.0, Gender=male
3. Height=5.58, Weight=170.0, Foot_Size=12.0, Gender=male
4. Height=5.92, Weight=165.0, Foot_Size=10.0, Gender=male
5. Height=5.0, Weight=100.0, Foot_Size=6.0, Gender=female
6. Height=5.5, Weight=150.0, Foot_Size=8.0, Gender=female
7. Height=5.42, Weight=130.0, Foot_Size=7.0, Gender=female
8. Height=5.75, Weight=150.0, Foot_Size=9.0, Gender=female
9. Height=7.0, Weight=200.0, Foot_Size=9.0, Gender=male
```