WEEK 2:

Name:Javeria Iqbal

1. 🧠 1. What is Artificial Intelligence (AI)?

Definition:
AI is the broadest concept — it refers to **machines or systems that mimic human intelligence** like learning, reasoning, problem-solving, and decision-making.

Examples:

- Self-driving cars
- Voice assistants (like Siri, Alexa)
- Chatbots
- Game-playing bots (like Chess AI)

📌 **AI includes everything from rule-based systems to learning algorithms.**

🤖 2. What is Machine Learning (ML)?

Definition:
ML is a **subset of AI** where machines **learn patterns from data** without being explicitly programmed.

Key Idea:
"Give data + labels → model learns rules."

Examples:

- Spam email filter
- Movie recommendation system
- Price prediction of a house
- Fraud detection in banking

📌 ML uses algorithms like:

- Linear regression
- Decision trees
- K-Nearest Neighbors
- Support Vector Machines (SVM)

🧬 3. What is Deep Learning (DL)?

Definition:
Deep Learning is a **subset of ML** that uses **neural networks with many layers** (hence "deep") to learn complex patterns.

Key Idea:
"Feed raw data → deep neural network figures out features & patterns automatically."

Examples:

- Face recognition
- Image classification
- Language translation
- ChatGPT, LLaMA, BERT (LLMs)

📌 DL uses architectures like:

- CNN (for images)
- RNN / LSTM (for sequences)
- Transformers (for language)

2.

```
import numpy as np
```

```
from sklearn.datasets import load_iris
```

```
from sklearn.linear_model import LinearRegression
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.metrics import mean_squared_error, r2_score
```

```
iris=load_iris()
```

```
X=iris.data
```

```
X
```

```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
       [5.7, 3.8, 1.7, 0.3],
       [5.1, 3.8, 1.5, 0.3],
```

```
y=iris.data[:,0]
```

```
y
```

```
array([5.1, 4.9, 4.7, 4.6, 5. , 5.4, 4.6, 5. , 4.4, 4.9, 5.4, 4.8, 4.8,
       4.3, 5.8, 5.7, 5.4, 5.1, 5.7, 5.1, 5.4, 5.1, 4.6, 5.1, 4.8, 5. ,
       5. , 5.2, 5.2, 4.7, 4.8, 5.4, 5.2, 5.5, 4.9, 5. , 5.5, 4.9, 4.4,
       5.1, 5. , 4.5, 4.4, 5. , 5.1, 4.8, 5.1, 4.6, 5.3, 5. , 7. , 6.4,
       6.9, 5.5, 6.5, 5.7, 6.3, 4.9, 6.6, 5.2, 5. , 5.9, 6. , 6.1, 5.6,
       6.7, 5.6, 5.8, 6.2, 5.6, 5.9, 6.1, 6.3, 6.1, 6.4, 6.6, 6.8, 6.7,
       6. , 5.7, 5.5, 5.5, 5.8, 6. , 5.4, 6. , 6.7, 6.3, 5.6, 5.5, 5.5,
       6.1, 5.8, 5. , 5.6, 5.7, 5.7, 6.2, 5.1, 5.7, 6.3, 5.8, 7.1, 6.3,
       6.5, 7.6, 4.9, 7.3, 6.7, 7.2, 6.5, 6.4, 6.8, 5.7, 5.8, 6.4, 6.5,
       7.7, 7.7, 6. , 6.9, 5.6, 7.7, 6.3, 6.7, 7.2, 6.2, 6.1, 6.4, 7.2,
       7.4, 7.9, 6.4, 6.3, 6.1, 7.7, 6.3, 6.4, 6. , 6.9, 6.7, 6.9, 5.8,
       6.8, 6.7, 6.7, 6.3, 6.5, 6.2, 5.9])
```

```
X_train, X_test, y_train, y_test = train_test_split(X[:, 1:], y, test_size=0.2, random_state=42)
```

```
model=LinearRegression()
```

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

```
LinearRegression
LinearRegression()
```

```
y_pred=model.predict(X_test)
```

```
print("mean Squared Error"),mean_squared_error(y_test,y_pred)
```

```
mean Squared Error
(None, 0.10212647866320382)
```

```
print("R^2",r2_score(y_test,y_pred))
```

```
R^2 0.8520477902310164
```

```
print("Mdel coefficients",model.coef_)
```

```
Mdel coefficients [ 0.66347568  0.75739488 -0.67418008]
```

```
print("model Intercept",model.intercept_)
```

```
model Intercept 1.7530468109297281
```

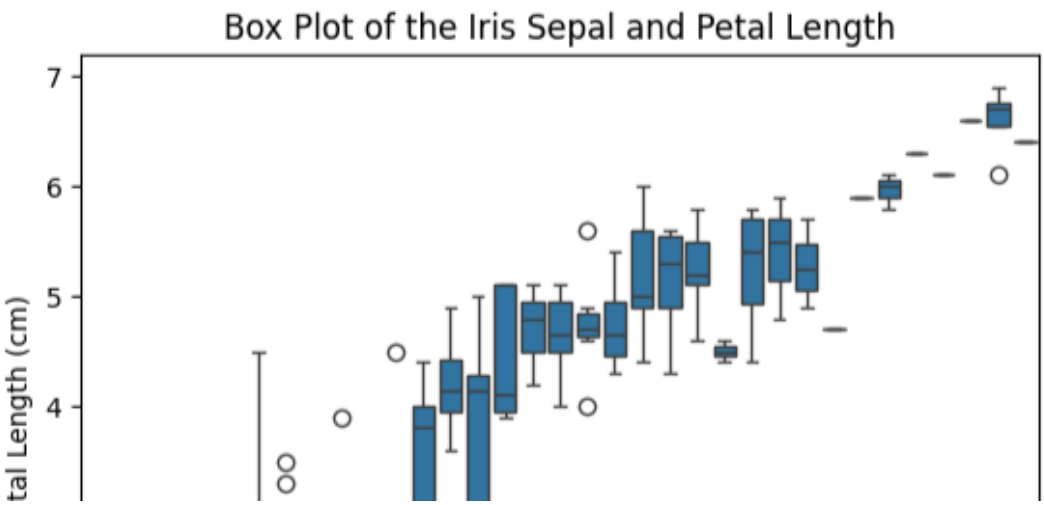
```
!pip install matplotlib
```

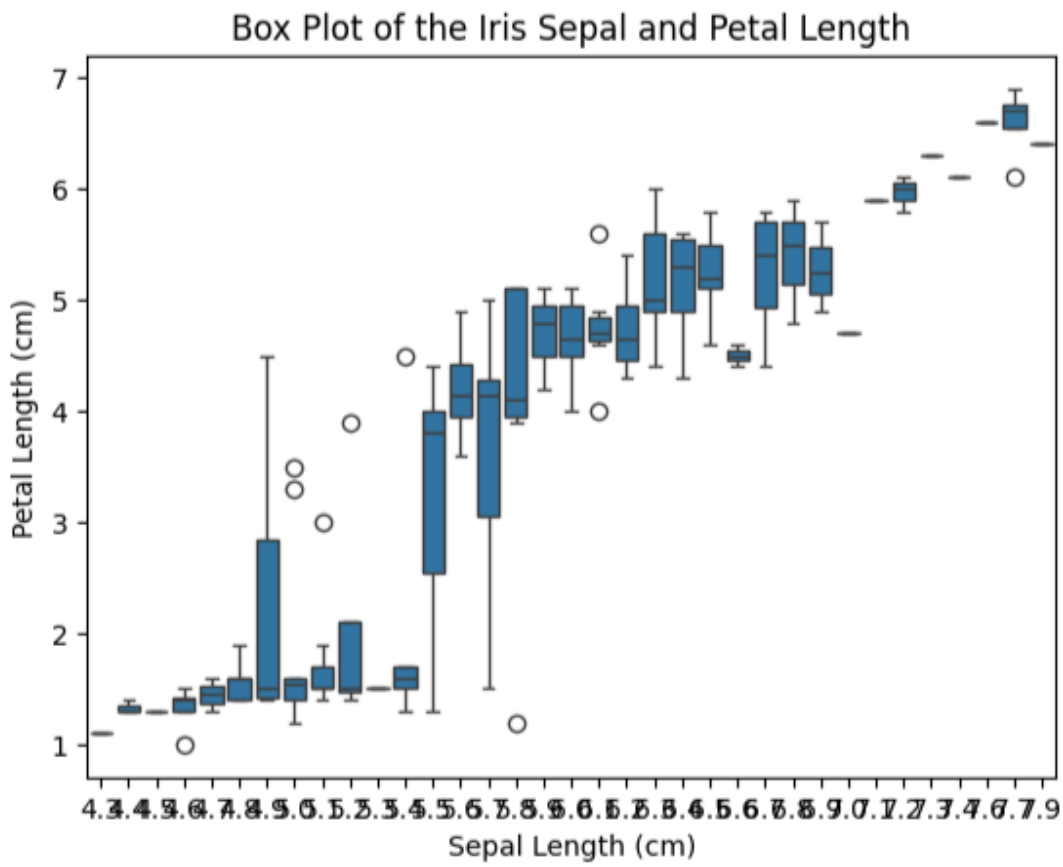
```
Requirement already satisfied: matplotlib in c:\users\administrator\appdata\local\programs\python\python313\lib\site-packages (3.10.3)
Requirement already satisfied: contourpy>=1.0.1 in c:\users\administrator\appdata\local\programs\python\python313\lib\site-packages (from matplotlib)
(1.3.2)
Requirement already satisfied: cycler>=0.10 in c:\users\administrator\appdata\local\programs\python\python313\lib\site-packages (from matplotlib) (0.1
2.1)
Requirement already satisfied: fonttools>=4.22.0 in c:\users\administrator\appdata\local\programs\python\python313\lib\site-packages (from matplotlib)
(4.58.0)
Requirement already satisfied: kiwisolver>=1.3.1 in c:\users\administrator\appdata\local\programs\python\python313\lib\site-packages (from matplotlib)
(1.4.8)
Requirement already satisfied: numpy>=1.23 in c:\users\administrator\appdata\local\programs\python\python313\lib\site-packages (from matplotlib) (2.2.
5)
Requirement already satisfied: packaging>=20.0 in c:\users\administrator\appdata\local\programs\python\python313\lib\site-packages (from matplotlib) (2
```

```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.datasets import load_iris

# Load iris data properly as DataFrame
iris = load_iris()
df = pd.DataFrame(iris.data, columns=iris.feature_names)
```

```
sns.boxplot(x="sepal length (cm)", y="petal length (cm)", data=df)
plt.title("Box Plot of the Iris Sepal and Petal Length")
plt.xlabel("Sepal Length (cm)")
plt.ylabel("Petal Length (cm)")
plt.show()
```





TASKS 3: DECISION TREE CLASSIFIER:

SOURCE CODE:

```
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier,plot_tree
from sklearn.metrics import accuracy_score,precision_score, recall_score,
f1_score, confusion_matrix, classification_report
import pandas as pd
import matplotlib as plt
data=pd.read_csv("marketing_campaign.csv")
X=data[["Age", "Income"]]
y=data["Subscribed"]
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.3,random_state=42)
model=DecisionTreeClassifier(criterion='gini',max_depth=3,random_state=42)
model.fit(X_train,y_train)
y_pred=model.predict(X_test)
evaluation={
    "accuracy":round(accuracy_score(y_test,y_pred),2),
```

```
    "precision": round(precision_score(y_test,y_pred),2),
    "Recall":round(recall_score(y_test,y_pred),2),
     "f1_score": round(f1_score(y_test,y_pred),2),
    "confusion matrix":confusion_matrix(y_test,y_pred).tolist(),
     "classification
Report":classification_report(y_test,y_pred,output_dict=True)}

plt.figure(figsize=(10,6))
plot_tree(model,feature_names=["Age","Income"],class_names=["Not
Subscribed","Subscribed"])
plt.title("DECISION TREE FOR MARKETING CAMPIANG")
plt.tight_layout()
plt.show()
```

OUTPUT: