1 . **Aim:** Implement Linear Regression (Diabetes Dataset).

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.datasets import load_diabetes
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
diabetes = load_diabetes()
X = diabetes.data[:, np.newaxis, 2]
X_train = X[:-20]
X_test = X[-20:]
y_train = diabetes.target[:-20]
y_test = diabetes.target[-20:]
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
print("Mean Squared Error:", mse)
plt.scatter(X_test, y_test, color='black', label='Actual')
plt.plot(X_test, y_pred, color='blue', linewidth=3, label='Predicted')
plt.xlabel('Feature')
plt.ylabel('Target')
plt.title('Linear Regression on Diabetes Dataset')
plt.legend()
plt.show()
```

-

2. **Aim:** Implement Logistic Regression (Iris Dataset).

```
 import pandas as pd
 import numpy as np
 import matplotlib.pyplot as plt
from sklearn import datasets, linear_model
from sklearn.metrics import accuracy_score
 iris = datasets.load_iris()
 X = iris.data
 y = iris.target
clf = linear_model.LogisticRegression()
clf.fit(X, y)
y_pred = clf.predict(X)
accuracy = accuracy_score(y, y_pred)
print('Accuracy:', accuracy)
plt.scatter(X[:, 0], y, color='black')
plt.plot(X[:, 0], y_pred, color='red')
  plt.xlabel('Sepal length')
 plt.ylabel('Species')
 plt.show()
```

3 . Aim: Implement SVM classifier (Iris Dataset).

```
from sklearn.datasets import load_iris


from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score
iris = load_iris()
X = iris.data
y = iris.target
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = SVC(kernel='linear')
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

4. Aim: Train and fine-tune a Decision Tree for the Moons Dataset.

```
from sklearn.datasets import make_moons


from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
# Generate the Moons dataset
X, y = make_moons(n_samples=10000, noise=0.4, random_state=42)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,random_state=42)



tree_clf = DecisionTreeClassifier(max_depth=4, random_state=42)
tree_clf.fit(X_train, y_train)
y_pred = tree_clf.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)
```

5. Aim: Implement a Multi-Layer Perceptron (MLP) for the classification of handwritten digits using the MNIST dataset.

```
# pip install tensorflow
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.datasets import mnist
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout, Flatten

from tensorflow.keras.optimizers import RMSprop
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train = x_train.reshape(x_train.shape[0], 784).astype('float32') / 255.0
x_test = x_test.reshape(x_test.shape[0], 784).astype('float32') / 255.0
y_train = keras.utils.to_categorical(y_train, 10)
y_test = keras.utils.to_categorical(y_test, 10)
model = Sequential()
model.add(Dense(512, activation='relu', input_shape=(784,)))
model.add(Dropout(0.2))
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(10, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer=RMSprop(),
metrics=['accuracy'])
history = model.fit(x_train, y_train, batch_size=128,
```

```
epochs=20, verbose=1,
validation_data=(x_test, y_test))

score = model.evaluate(x_test, y_test, verbose=0)
print('Test loss:', score[0])
print('Test accuracy:', score[1])
```

6. Aim: Classification of images of clothing using TensorFlow with the Fashion MNISTdataset.

```
# pip install tensorflow keras
import tensorflow as tf
from tensorflow import keras
import numpy as np
fashion_mnist = keras.datasets.fashion_mnist
(X_train, y_train), (X_test, y_test) = fashion_mnist.load_data()

X_train = X_train.astype('float32') / 255.0
X_test = X_test.astype('float32') / 255.0
class_names = ['T-shirt/top', 'Trouser', 'Pullover', 'Dress', 'Coat',
'Sandal', 'Shirt', 'Sneaker', 'Bag', 'Ankle boot']
```

```
model = keras.models.Sequential([
keras.layers.Flatten(input_shape=(28, 28)),
keras.layers.Dense(128, activation='relu'), keras.layers.Dense(10,
activation='softmax')
])



model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy', metrics=['accuracy'])


model.fit(X_train, y_train, epochs=10)



test_loss, test_accuracy = model.evaluate(X_test, y_test)
print("Test Loss:", test_loss)
print("Test Accuracy:", test_accuracy)
test_example = X_test[0]
test_example = np.expand_dims(test_example, axis=0)
predictions = model.predict(test_example)
predicted_class = np.argmax(predictions[0])
print("Predicted Class:", class_names[predicted_class])
```