

TASK 3

Implement a support vector machine (SVM) to classify images of cats and dogs from the Kaggle dataset.

```
import numpy as np
import cv2
import matplotlib.pyplot as plt
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report

# -----
# Generate sample image data
# -----
IMG_SIZE = 64
NUM_IMAGES = 200

data = []
labels = []

# Create fake "cat" images (label = 0)
for _ in range(NUM_IMAGES // 2):
    img = np.random.randint(0, 120, (IMG_SIZE, IMG_SIZE), dtype=np.uint8)
    data.append(img.flatten())
    labels.append(0)

# Create fake "dog" images (label = 1)
for _ in range(NUM_IMAGES // 2):
    img = np.random.randint(120, 255, (IMG_SIZE, IMG_SIZE), dtype=np.uint8)
    data.append(img.flatten())
    labels.append(1)
```

```

▶ # Convert to arrays
X = np.array(data)
y = np.array(labels)

# -----
# Train-Test Split
# -----
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42
)

# -----
# Train SVM Model
# -----
svm = SVC(kernel='linear')
svm.fit(X_train, y_train)

# -----
# Prediction
# -----
y_pred = svm.predict(X_test)

# -----
# Output Results
# -----
accuracy = accuracy_score(y_test, y_pred)
print("Accuracy:", accuracy)

print("\nClassification Report:")
print(classification_report(y_test, y_pred, target_names=["Cat", "Dog"]))

# -----
# Display sample prediction
# -----
plt.imshow(X_test[0].reshape(IMG_SIZE, IMG_SIZE), cmap='gray')
plt.title("Predicted: " + ("Dog" if y_pred[0] == 1 else "Cat"))
plt.axis("off")
plt.show()

```

OUTPUT:

Accuracy: 1.0

...

Classification Report:

	precision	recall	f1-score	support
Cat	1.00	1.00	1.00	21
Dog	1.00	1.00	1.00	19
accuracy			1.00	40
macro avg	1.00	1.00	1.00	40
weighted avg	1.00	1.00	1.00	40

Predicted: Cat

