```
import cv2
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split, cross_val_score, KFold
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, confusion_matrix, roc_curve, auc
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.datasets import load_digits

# Load Image
image_path = '/content/Screenshot 2025-01-30 112851.png'
image = cv2.imread(image_path)
image_rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

# Task 1.1: Image Resizing (Interpolation Methods)
linear_resized = cv2.resize(image, (300, 300), interpolation=cv2.INTER_LINEAR)
nearest_resized = cv2.resize(image, (300, 300), interpolation=cv2.INTER_NEAREST)
polynomial_resized = cv2.resize(image, (300, 300), interpolation=cv2.INTER_CUBIC)

# Display resized images
fig, axs = plt.subplots(1, 3, figsize=(15, 5))
axs[0].imshow(cv2.cvtColor(linear_resized, cv2.COLOR_BGR2RGB))
axs[0].set_title('Linear Interpolation')
axs[1].imshow(cv2.cvtColor(nearest_resized, cv2.COLOR_BGR2RGB))
axs[1].set_title('Nearest Neighbors')
axs[2].imshow(cv2.cvtColor(polynomial_resized, cv2.COLOR_BGR2RGB))
axs[2].set_title('Polynomial Interpolation')
for ax in axs:
    ax.axis('off')
plt.show()

# Task 1.2: Image Blurring
# Box Blurring
box_blur = cv2.blur(image, (5, 5))
# Gaussian Blurring
gaussian_blur = cv2.GaussianBlur(image, (5, 5), 0)
# Adaptive Blurring (Using Bilateral Filter)
adaptive_blur = cv2.bilateralFilter(image, 9, 75, 75)

# Display blurred images
fig, axs = plt.subplots(1, 3, figsize=(15, 5))
axs[0].imshow(cv2.cvtColor(box_blur, cv2.COLOR_BGR2RGB))
axs[0].set_title('Box Blurring')
axs[1].imshow(cv2.cvtColor(gaussian_blur, cv2.COLOR_BGR2RGB))
axs[1].set_title('Gaussian Blurring')
axs[2].imshow(cv2.cvtColor(adaptive_blur, cv2.COLOR_BGR2RGB))
axs[2].set_title('Adaptive Blurring')
for ax in axs:
    ax.axis('off')
plt.show()

# Task 2: Machine Learning on MNIST Dataset
digits = load_digits()
X = digits.images.reshape((len(digits.images), -1))
y = digits.target

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

# Model Selection (Naive Bayes and SVM)
models = {
    'Naive Bayes': GaussianNB(),
    'SVM': SVC(kernel='linear', probability=True)
}

kf = KFold(n_splits=5, shuffle=True, random_state=42)

for name, model in models.items():
    scores = cross_val_score(model, X_train, y_train, cv=kf, scoring='accuracy')
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    y_prob = model.predict_proba(X_test)[:, 1] if hasattr(model, 'predict_proba') else None

    # Metrics Calculation
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred, average='macro')
    recall = recall_score(y_test, y_pred, average='macro')
    f1 = f1_score(y_test, y_pred, average='macro')
    cm = confusion_matrix(y_test, y_pred)

    # ROC & AUC Calculation
```

```
        if y_prob is not None:
            fpr, tpr, _ = roc_curve(y_test, y_prob, pos_label=y_test.max())
            auc_score = auc(fpr, tpr)
        else:
            auc_score = 'N/A'

        print(f'\n{name} Model Results:')
        print(f'Accuracy: {accuracy:.4f}')
        print(f'Precision: {precision:.4f}')
        print(f'Recall: {recall:.4f}')
        print(f'F1 Score: {f1:.4f}')
        print(f'Confusion Matrix:\n{cm}')
        print(f'AUC Score: {auc_score}')

        if y_prob is not None:
            plt.plot(fpr, tpr, label=f'{name} (AUC = {auc_score:.4f})')

    if y_prob is not None:
        plt.xlabel('False Positive Rate')
        plt.ylabel('True Positive Rate')
        plt.title('ROC Curve')
        plt.legend()
        plt.show()
```


```
Naive Bayes Model Results:
Accuracy: 0.8472
Precision: 0.8650
Recall: 0.8476
F1 Score: 0.8437
Confusion Matrix:
[[31  0  0  0  0  1  0  1  0  0]
 [ 0 24  0  0  0  0  0  0  3  1]
 [ 0  2 20  0  0  0  1  0 10  0]
 [ 0  0  1 29  0  1  0  0  3  0]
 [ 0  0  0  0 38  0  1  7  0  0]
 [ 0  0  0  1  0 44  1  1  0  0]
 [ 0  0  0  0  1  0 34  0  0  0]
 [ 0  0  0  0  0  1  0 33  0  0]
 [ 0  2  0  0  0  0  0  2 26  0]
 [ 0  1  1  2  0  2  0  4  4 26]]
AUC Score: 0.522421875

SVM Model Results:
Accuracy: 0.9778
Precision: 0.9792
Recall: 0.9782
F1 Score: 0.9785
Confusion Matrix:
[[33  0  0  0  0  0  0  0  0  0]
 [ 0 28  0  0  0  0  0  0  0  0]
 [ 0  0 33  0  0  0  0  0  0  0]
 [ 0  0  0 32  0  1  0  0  0  1]
```