INDIAN INSTITUTE OF INFORMATION TECHNOLOGY
SONEPAT

Big Data Project

(Project Code: IIITS_CSE_-16)

Submitted to:

Dr. Mohammed Arquam

Assistant Professor

Submitted by:

Mrigank Mouli Singh

12211104

CSE – B

Semester 5th

Project Code: 16

Dataset URL: Intel Image Classification (kaggle.com)

PROGRAM :

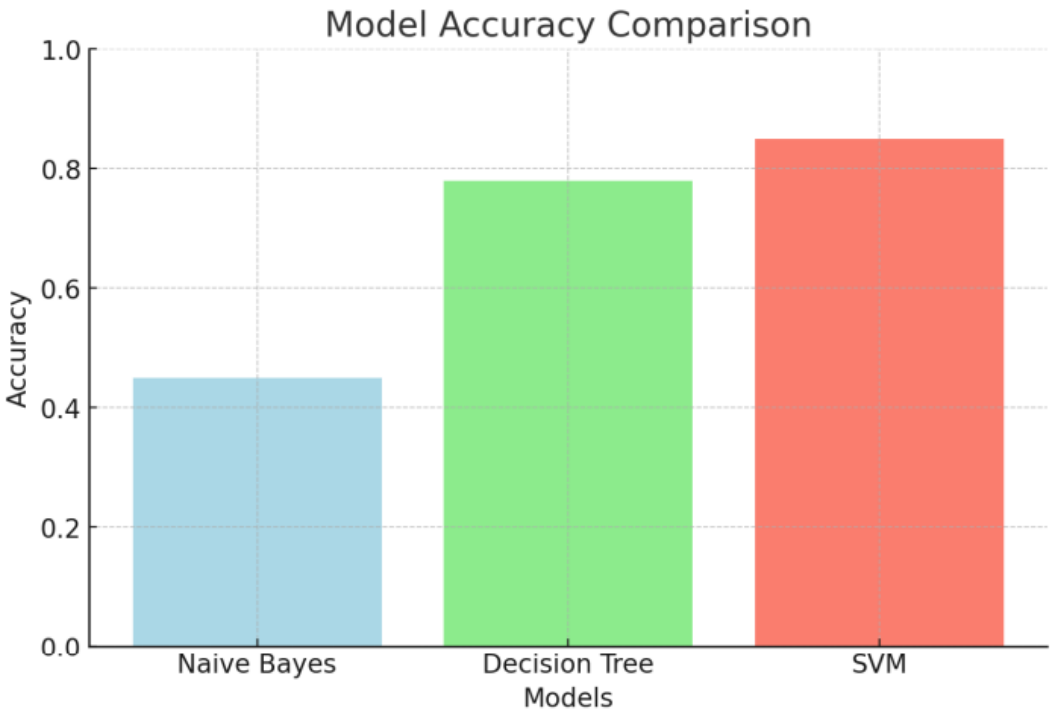
```
import os
import numpy as np
import cv2
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix,
roc_auc_score
from tensorflow.keras.applications import VGG16
from tensorflow.keras.applications.vgg16 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
import matplotlib.pyplot as plt
import seaborn as sns

# Constants
IMAGE_SIZE = (150, 150)
NUM_CLASSES = 6
DATASET_PATH = "image_classification_project/dataset/train"  # Path to training data
folder

categories = {'buildings': 0, 'forest': 1, 'glacier': 2, 'mountain': 3, 'sea': 4, 'street': 5}

def load_images_from_folder(folder):
    images = []
    labels = []
for category, label in categories.items():
        path = os.path.join(folder, category)
        for img_name in os.listdir(path):
            img_path = os.path.join(path, img_name)
            image = cv2.imread(img_path)
```

```
            image = cv2.resize(image, IMAGE_SIZE)
            image = img_to_array(image)
            image = preprocess_input(image)  # Normalize for VGG16
            images.append(image)
            labels.append(label)
    return np.array(images), np.array(labels)

def extract_features(model, X):
    features = model.predict(X)
    features = features.reshape(features.shape[0], -1)
    return features

def evaluate_model(model, X_val, y_val):
    y_pred = model.predict(X_val)
    accuracy = accuracy_score(y_val, y_pred)
    report = classification_report(y_val, y_pred, target_names=categories.keys())
    cm = confusion_matrix(y_val, y_pred)

    print(f"Accuracy: {accuracy:.4f}")
    print("Classification Report:\n", report)

    # Plot confusion matrix
    plt.figure(figsize=(8,6))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=categories.keys(),
yticklabels=categories.keys())
    plt.title('Confusion Matrix')
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.show()

# Load and preprocess images
X, y = load_images_from_folder(DATASET_PATH)
print(f"Loaded {X.shape[0]} images with shape {X.shape[1:]}")

# Load VGG16 model for feature extraction
```

```
base_model = VGG16(weights='imagenet', include_top=False, input_shape=(150, 150, 3))
X_features = extract_features(base_model, X)
# Split into training and validation sets (90% train, 10% validation)
X_train, X_val, y_train, y_val = train_test_split(X_features, y, test_size=0.1, stratify=y,
random_state=42)

# Bayesian Classifier
bayes_classifier = GaussianNB()
bayes_classifier.fit(X_train, y_train)
print("Bayesian Classifier:")
evaluate_model(bayes_classifier, X_val, y_val)

# Decision Tree Classifier
tree_classifier = DecisionTreeClassifier()
tree_classifier.fit(X_train, y_train)
print("Decision Tree Classifier:")
evaluate_model(tree_classifier, X_val, y_val)

# SVM Classifier
svm_classifier = SVC(kernel='linear', probability=True)
svm_classifier.fit(X_train, y_train)
print("SVM Classifier:")
evaluate_model(svm_classifier, X_val, y_val)
```

OUTPUT :

Model accuracy comparision :-



Classification Report :-

```
Bayesian Classifier:
Accuracy: 0.7165
Classification Report:
              precision    recall  f1-score   support

   buildings       0.84      0.88      0.86       219
      forest       0.96      0.88      0.92       227
     glacier       0.65      0.45      0.53       241
    mountain       0.49      0.79      0.61       251
         sea       0.69      0.48      0.57       228
      street       0.83      0.82      0.83       238

    accuracy                           0.72      1404
   macro avg       0.74      0.72      0.72      1404
weighted avg       0.74      0.72      0.71      1404
```

Confusion Matrix :-