Sloan Digital Sky Survey (SDSS) Galaxy Classification using Machine Learning

1.Introduction

1.1 **Project overviews:**

This project involves the classification of galaxy images from the Sloan Digital Sky Survey (SDSS) into five morphological categories using a deep learning model. The goal is to accurately predict the type of galaxy from images by leveraging convolutional neural networks (CNNs).

1.2 **Objectives:**

- Build a CNN-based image classification model.
- Classify galaxies into one of five categories:
  - Cigar-shaped smooth
  - In between smooth
  - Completely round smooth
  - Edge-on
  - Spiral
- Deploy the model via a Flask web application.
- Provide a user-friendly interface for image upload and result display.

2.Project Initialization and Planning Phase

2.1 **Define Problem Statement:**

Astronomical galaxy classification is time-consuming and subjective when done manually. The goal is to automate this process using machine learning on image data.

2.2 **Project Proposal:**

We propose a CNN-based image classification solution that takes galaxy images as input and returns one of the five defined morphological types with associated confidence.

2.3 **Initial Project Planning:**

- Technology Stack: Python, TensorFlow/Keras, Flask, HTML/CSS
- Dataset Source: Galaxy Zoo dataset from Kaggle
- Phases: Data collection → Preprocessing → Model training → Evaluation → Deployment

3. Data Collection and Preprocessing Phase

3.1 Data Collection Plan and Raw Data Sources Identified:

- **Source**: Kaggle - Galaxy morphology images
- **Classes**: 5 galaxy types
- **Structure**: Image files in folders categorized by class

3.2 Data Quality Report:

- Images were checked for corruption.
- Resized uniformly to match model input (256x256).
- Class balance was verified.

3.3 Data Exploration and Preprocessing:

- Used ImageDataGenerator for augmentation.
- Normalized pixel values (0–1).
- Split into train, validation, test sets using splitfolders.

4. Model Development Phase

4.1 Feature Selection Report:

As it is an image classification task, raw pixel data was used as features. No manual feature engineering was needed.

4.2 Model Selection Report:

- CNN architecture was chosen for its superiority in image-based tasks.
- Alternatives like pre-trained VGG16 were considered but a custom CNN was used for experimentation and training efficiency.

4.3 Initial Model Training Code, Model Validation and Evaluation Report:

- CNN with Conv2D, MaxPooling2D, Flatten, Dense layers.
- **Accuracy**: ~76.6% on test data.
- **Loss**: Monitored using categorical cross-entropy.
- Early stopping and callbacks used during training.

5. Model Optimization and Tuning Phase

5.1 Hyperparameter Tuning Documentation:

- Tuned learning rate, number of filters, kernel size, batch size.
- Best results with Adam optimizer and batch size = 32.

5.2 Performance Metrics Comparison Report:

Model	Test Accuracy
CNN	0.7660804986953735
VGG16	0.294928752754983

5.3 Final Model Selection Justification:

The custom CNN was chosen for its generalization ability, smaller size, and easier deployment. Accuracy vs computational cost was optimized.

6. Results

Test Image:

Result:

```
WARNING:absl:Compiled the loaded model, but the compiled metrics have yet to be built. `model.compile_metrics` will be empty until you trai
Model 'SDSSmodel.h5' loaded successfully.
Model expects input dimensions: (256, 256)
Image 'demo.jpeg' loaded and preprocessed to shape: (1, 256, 256, 3)
Making prediction...
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 132ms/step

Prediction Results:
Raw predictions shape: (1, 5)
Raw predictions (first 5 values): [0.7297289  0.00169889 0.00177655 0.18002357 0.08677211]
Predicted class index: 0
Predicted class: Cigar-shaped smooth
Confidence: 0.7297

Testing complete.
```

7. Advantages & Disadvantages

Advantages	Disadvantages
• End-to-end pipeline from training to web app. • Accurate galaxy classification. • Clean, user-friendly interface.	• Moderate model accuracy (~76%). • Requires Advanced GPU for faster training. • Small dataset size may affect generalization.

8. Conclusion

The project demonstrates a successful application of CNNs for galaxy classification with competitive accuracy. A fully functional Flask app integrates the trained model, providing an intuitive frontend for users.

9. Future Scope

- Use transfer learning with larger datasets (e.g., VGG, ResNet).
- Extend to more galaxy classes.
- Improve UI/UX of the web application.
- Integrate real-time telescope data for classification.

Appendix

Available in my GitHub Repo,

Source code: GitHub