W281 Final Project: Galaxy Zoo Classification

Ryan Chen, Immanuel Abdi, Qichen Liu

1 Problem Definition

1.1 Problem Introduction and Motivation

There are countless stars above our heads and each of them can have various differing morphologies that are arduous to manually classify due to their complexity. As neural network technology has become increasingly mature and popular in recent years, astronomy scientists and amateurs are able to investigate astronomical objects more deeply. Thus, our team is aiming for astronomical object classification by using classical and machine-learning computer vision methodologies. We believe this project helps develop an automatic method for identifying and tagging astronomical objects.

1.2 Dataset Description

We adopted Galaxy Zoo 1 dataset on Kaggle. This dataset was produced by the Galaxy Zoo project from July 2007 until February 2009, where volunteers classified images of Sloan Digital Sky Survey (SDSS) galaxies as belonging to one of the five categories below.

1. **Cigar-Shaped Smooth**: Galaxies that appear elongated and smooth without distinct features like spiral arms or a disk visible.
2. **In-between Smooth**: Galaxies that do not fit neatly into the completely round or cigar-shaped categories and have a smooth appearance without clear features.
3. **Complete Round Smooth**: Galaxies that appear completely round and smooth, with no visible features such as spiral arms or disks.
4. **Edge-on**: Galaxies viewed from the side, making their disk visible as a thin line or edge, often obscuring detailed features.
5. **Spiral**: Galaxies with visible spiral arms winding out from the center, often with a bulge at the center.

The dataset was preprocessed by Lintott et al. 2008, MNRAS, 389, 1179. All of the 28,793 images were corrected with redshifts and astronomical objects were centered. They all have a size of 424 x 424 and a resolution of 96 dpi. The images of galaxies are shown in Figure 1.

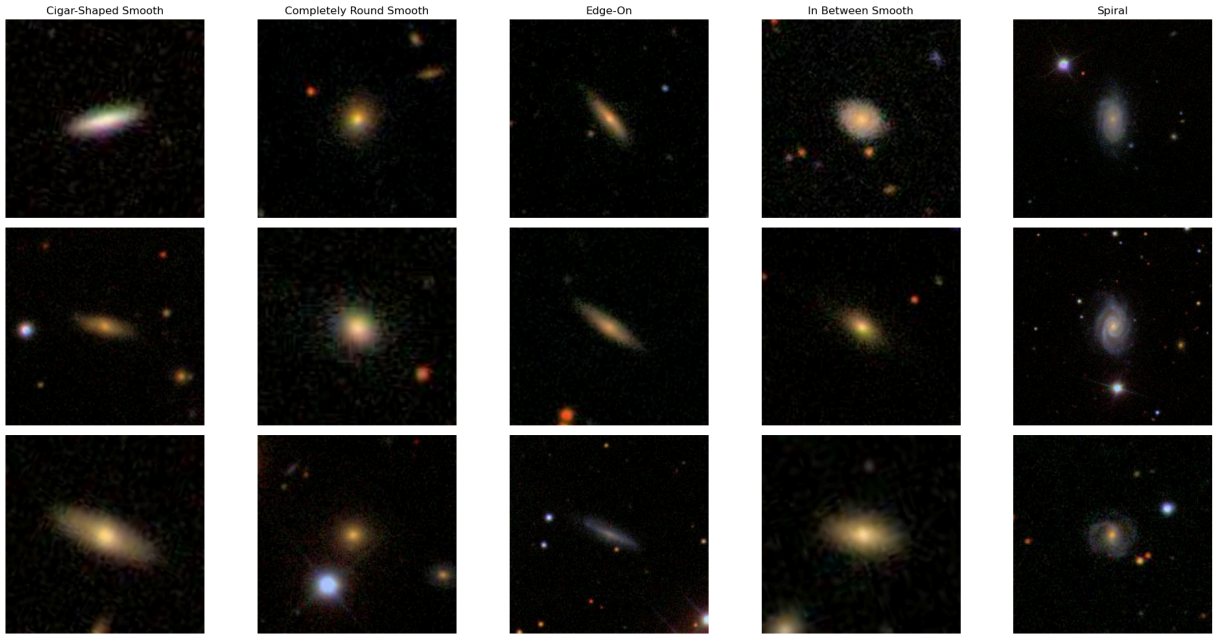


Figure 1. Images of five types of galaxies.

At first glance, we found it hard to distinguish between cigar-shaped and edge-on galaxies due to their similar morphology. Moreover, there is a strong grain effect around the object that can potentially lead to confusion in recognizing shapes.

Figure 2 addresses the number of images in each category, where Cigar-Shaped Smooth has 579 examples, Completely Round Smooth 8436 examples, Edge-On 3903 examples, In Between Smooth 8069 examples, and Spiral 7806 examples. There is a natural constraint on class Cigar-Shaped Smooth that it only has 579 examples so we might have to do the data augmentation later.

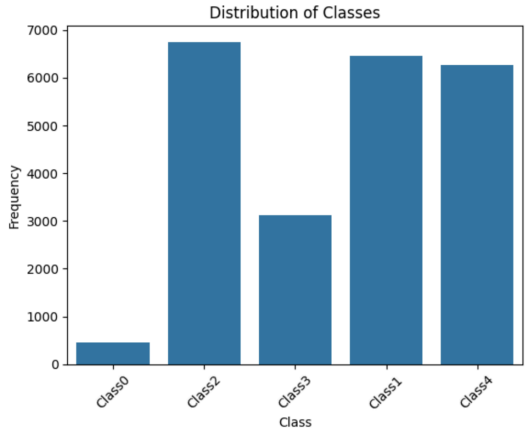


Figure 2. Data distribution across five classes of galaxies.

2. Data Preprocessing

2.1 Data Split

We divided our dataset with a ratio of 0.8, 0.1, and 0.1 for training, validation, and testing respectively, and mapped the label to each image for further processing.

2.2 Image Preprocessing

We took advantage of a variety of image processing methods to filter out the noise in the background and highlight key information about the galaxy we want to identify. Those methods were organized as filters in several pipelines. The filters included histogram clipping, image blurring, and Sobel edge detector.

A common practice in image classification is to extract features from RGB respectively. However, we found that the distributions of intensity across RGB were similar and heavily right-skewed. As a result, we decided to translate each image to grayscale before preprocessing it through pipelines 1 and 2.

Here are the reasons for applying each specific method.

- **Histogram Clipping**: Clipping in a histogram occurs when a region of a photo is too dark or too light for the sensor to capture detail in that region. We clipped histograms with an intensity larger than 30 for the purpose of eliminating surrounding noise in the background of an image, so that we could focus on more useful information of the galaxy object in the center.
- **Gaussian Blur**: A Gaussian blur makes the image less clear or distinct by a Gaussian function. It helps in noise removal, as there are grain effects that can potentially lead to confusion in recognizing the shapes of each galaxy. Since noise is considered a high-frequency signal, noise can be restricted by the application of low-pass filter kernels.
- **Bilateral Blur**: A bilateral filter is a non-linear, edge-preserving, and noise-reducing smoothing filter for images. It replaces the intensity of each pixel with a weighted average of intensity values from nearby pixels. This weight can be based on a Gaussian distribution. Thus, sharp edges are preserved while discarding the low-intensity ones.
- **Sobel Edge Detector**: A Sobel operator performs a 2-D spatial gradient measurement on an image and so emphasizes regions of high spatial frequency that correspond to edges. Typically it is used to find the approximate absolute gradient magnitude at each point in an input grayscale image.
- **Center Crop**: Center crop is cropping an image from the center which gives equal padding on both sides vertically and horizontally, so it helps data augmentation. We applied it as a final step of preprocessing pipelines.

We constructed three image preprocessing pipelines (see Figure 3) to cooperate with the following feature extractors respectively.

- **Pipeline 1 for SIFT**: Gray Scale Conversion, Histogram Clipping, Gaussian Blur, Bilateral Blur, Sobel Edge Detector, Center Crop
- **Pipeline 2 for HOG feature**: Gray Scale Conversion, Histogram Clipping, Gaussian Blur, Bilateral Blur, Center Crop
- **Pipeline 3 for ResNet-50**: Pristine Image (No Filters)

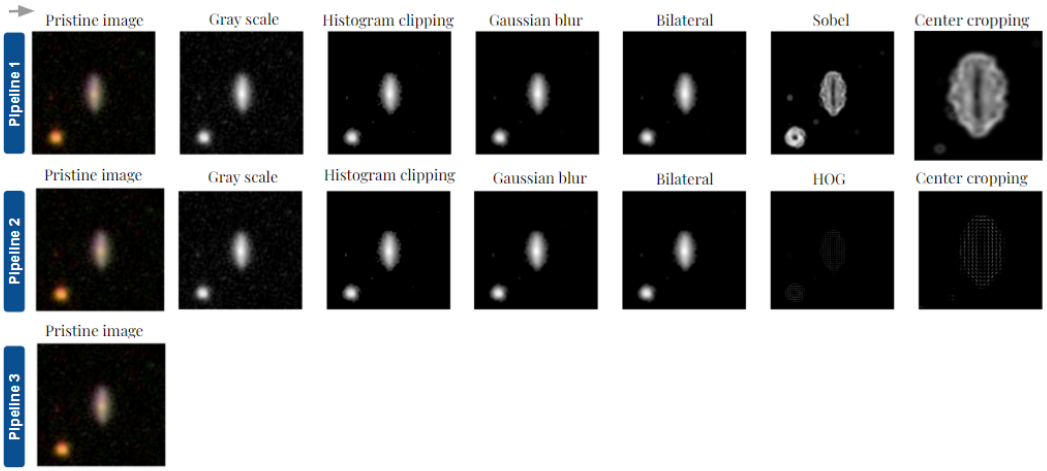


Figure 3. Pipelines of image preprocessing steps.

2.3 Feature Extraction

After many attempts, we decided to use the following three feature extractors as they managed to extract reasonably key information from galaxy images for shape recognition. Additionally, we applied different strategies to further process the features that we had extracted from the three pipelines.

2.3.1 Scale Invariant and Feature Transform (SIFT)

SIFT is an algorithm to detect and describe local features in images. The strength of SIFT is that it can extract distinctive invariant features from images that are used to perform reliable matching between different views of the galaxy object. The features are invariant to image scale and rotation, and robust to distortion, noise, and change in illumination. We treated the product from pipeline 1 with SIFT and applied K-means and bag-of-visual-word (BoVW) to get features. We applied K-means with 200 centroids and 1 iteration to SIFT features and got 200 codes of visual words. Next, we assigned the codes to the SIFT features so that each image was represented by visual words. Lastly, the BoVW method converted the features into a vector of 200 features.

2.3.2 Histogram of oriented gradients (HOG)

HOG captures local intensity gradients and their orientations within an image, which are essential for characterizing object shapes and structures. It is particularly useful in image detection where the outline of the shape matters more in our case. We applied HOG to the product from pipeline 2, followed by PCA to reduce the dimensions from 63,504 to 500 features, and a z-score normalization to introduce uniformity across features which is beneficial for the regression.

2.3.3 ResNet-50

ResNet-50 is a pre-trained convolutional neural network with 50 layers. We removed the classifier on top of it so we could take advantage of the embeddings from the pre-trained model. For each pristine image from pipeline 3, we did not preprocess it and directly fed it into the body of ResNet-50 instead, because we expected the model to capture meaningful information from each image without preprocessing. It is noteworthy that ResNet tends to capture high-level, abstract features and encapsulate non-linear relationships to its output features, which can be a good match with non-linear SVM. Lastly, we got 2,048 features from this pipeline.

2.4 Normalization and Dimensionality Reduction

We performed PCA only on the HOG features from pipeline 2 to reduce the feature dimensions from 63,504 to 500. As for the features from pipelines 1 and 3, we skipped the PCA process and got 200 features from SIFT and 2,048 features from ResNet-50. The cumulative variance for pipelines 2, 3 and integration of all three pipelines is shown in Figure 4. The top 500 PCA features of Pipeline 2 can explain around 60% of the variance. To check out the sparsity of our dataset, we also printed out the variance of Pipeline 3 and the top 300 features of Pipeline 3 can cover more than 90% of the variance. Lastly, we printed out the top 500 features of the integration of three pipelines before the classifier which can explain more than 80% of the variance.

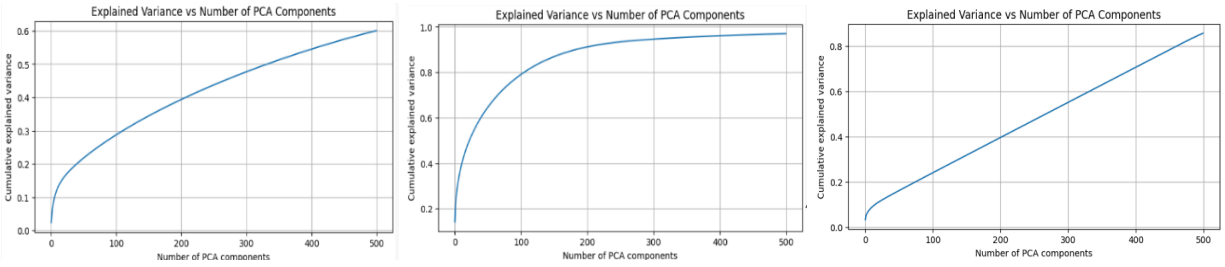


Figure 4. Cumulative variance of PCA for pipelines 2, 3 and integration of all three pipelines (from left to right).

As for the normalization, We applied min-max normalization to the features from pipeline 1, in case some features may have a wide range of output and influence the following classification. For HOG features after PCA, we treated them with z-score normalization to maintain the distribution of the data. As for the product of pipeline 3, since ResNet-50 has normalization and pooling layers, there is no need to do the normalization. We concatenated 200 features from Pipeline1, 500 features from Pipeline2, and 2048 features from Pipeline3 for the following classification step.

3. Classifiers and Data Balancing

3.1 Classifiers: Logistic regression with LDA and non-linear SVM

To begin with, we took 500 examples from each class, 2000 examples in total, and we adopted logistic regression with LDA and non-linear SVM as our classifiers. We intended to take this small dataset to test out the classifiers, check their performance, and conduct the grid search. Since logistic regression only works with linear cases, it may not perform well in high dimensional classification, so we reduced the number of features using LDA and set the number of components to four as we have five classes of galaxies. As for SVM, we would either choose ‘poly’ or ‘rbf’ kernel to introduce non-linearity to the model.

The computational time and performance for each pipeline and model are listed in Table 1. With the T4 GPU provided by Google Colab, processing Pipeline 1 takes 6 seconds, Pipeline 2 20 seconds, and Pipeline 3 27 seconds per 100 examples. Pipeline 1 looks the most efficient; however, the training accuracy of logistic regression is only 63%, while SVM is 71%. On the other hand, Pipeline 2 achieves a training accuracy of 80% for logistic regression and 95% for SVM. HOG captures local intensity gradients and their orientations within an image, which are richer in terms of information in our case compared to the features of SIFT. As a result, Pipeline 2 performs better than Pipeline 1 on both the logistic regression with LDA and non-linear SVM.

As for Pipeline 3, The ResNet-50 tends to capture high-level, abstract features and encapsulate non-linear relationships to the output features, so the linear logistic regression would have a hard time processing the features, on the contrary, for the non-linear SVM, it can better capture non-linearity and complex patterns so the training accuracy is up to 89%. The validation accuracy of all three approaches is around 50%, and there is an overfitting spotted in the case of SVM. Both the hyperparameter C and gamma of SVM can decide the margin of SVM and fix the overfitting, so next, we would integrate three pipelines and run the grid search with a small dataset.

Pipeline	Preprocess steps	Feature extractor and dimension reduction	Computational Time (T4 GPU)	Classifiers	Training accuracy	Validation accuracy
Pipeline 1	Pristine image Gray scale Histogram Clipping Gaussian blur Bilateral Sobel Center cropping	SIFT, Kmean, VBOW (200 features) Normalization	68 / 100 examples	LDA + logistic regression	63%	45%
				SVM	71%	51%
Pipeline 2	Pristine image Gray scale Histogram Clipping Gaussian blur Bilateral HOG Center cropping	HOG feature (63504 features) PCA (200 features) Normalization	208 / 100 examples	LDA + logistic regression	80%	65%
				SVM	95%	64%
Pipeline 3	Pristine image	Body of Resnet50 (2048 features) PCA (100 features)	278 / 100 examples	LDA + logistic regression	28%	5%
				SVM	89%	41%

Table 1. Comparison of pipelines in terms of computational time and performance with 2,000 examples.

After putting all three pipelines together and running the grid search, we have the performance of each classifier in Table 2. We noticed that there is a strong overfitting in SVM and it can stem from the fact that we only ran it with 2,000 examples, it might not incorporate the diversity of the dataset. Next, from the confusion matrix of validation on both logistic regression and non-linear SVM, both models seem like struggling with distinguishing between class 0 and class 3 which are the cigar-shape galaxies and edge-on galaxies. These two types of galaxies share a similar shape and the major difference is that edge-on galaxies are the side view of other galaxies so they have a bright spot in the center. As a result, It's hard to tell the difference even for a human. Besides class 0 and class 3, both models also have a hard time telling class 1 and 2 which are in-between galaxies and round galaxies. It's reasonable since the definitional boundary of these two is ambiguous and it's just the difference between an eclipse shape and a complete round shape.

Classifiers	Hyperparameters (grid search)	Training accuracy	Cross-Validation accuracy	Testing accuracy
LDA + logistic regression	Components = 4, L1 regularization, C=1	89.75%	84.25%	70.85%
SVM (non-linear)	Kernel = 'rbf', C = 5, gamma=0.01	98.45%	72.65%	71.25%

Table 2. Comparison of models in terms of performance.

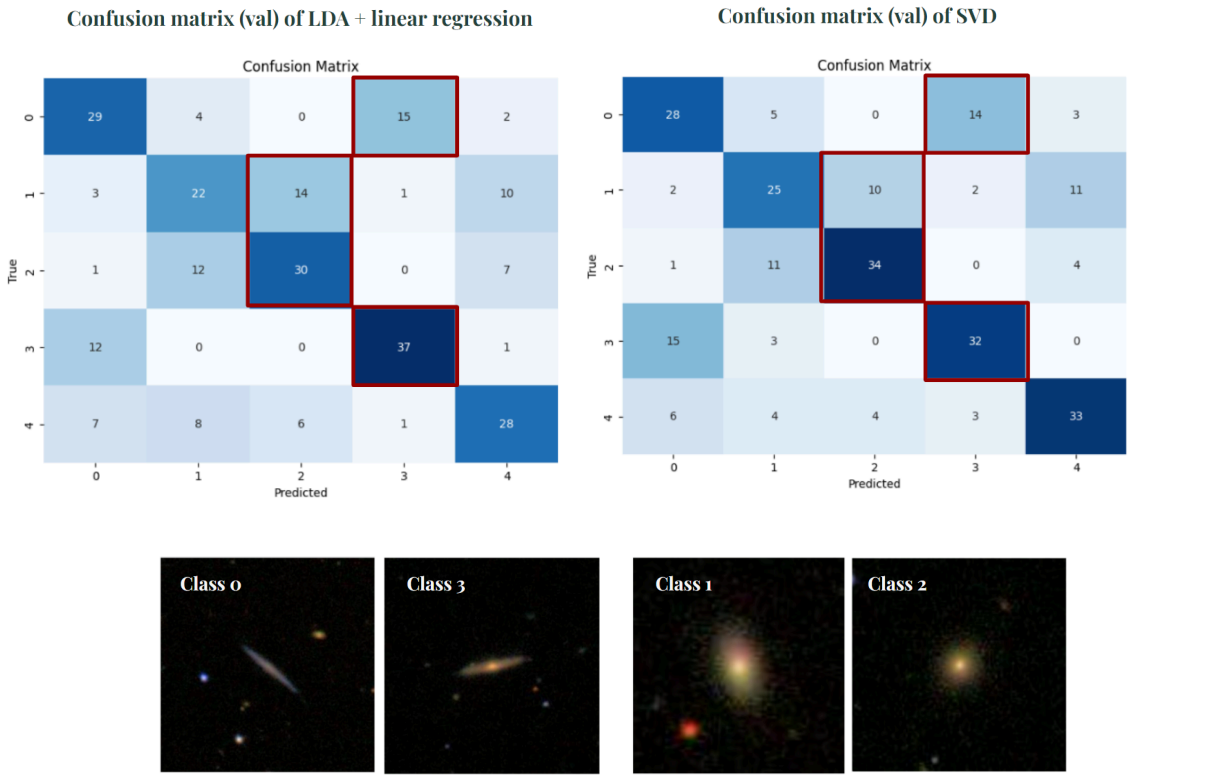


Figure 5. Confusion matrix of validation dataset for Logistic regression and non-linear SVM. Image of cigar-shaped, edge-on, in-between, and completed round galaxies from left to right.

3.2 Data Balancing

Initially, the precision levels across all the classes were from 50% to 60% and we intended to increase the number of examples to boost the performance of the models. Since we have a natural constraint that we only have 579 examples in class 0 and the model seems to have a struggle distinguishing class 0 and class 3, so we only increased the number of class 3 examples from 500 to 600. For other classes, we added the examples up to 1,000 and made it Dataset 2. As for Dataset 3, we increase the number of examples to 1,000 and 1,500 for other classes. The number of examples for each class and the performance across all three datasets in terms of precision are listed in Table 3 and Figure 6.

We can see that the precision across all classes of galaxies increases significantly after we feed more data to the models using Dataset 2. However, when we further increase the number of data, the precision of class 0 and 1 on both Logistic regression and non-linear SVM drops a bit. There is a trade-off between the performance of class 0 and other classes. In the next section, we will run the entire dataset and fix the overfitting by adjusting the hyperparameter of SVM through a grid search.

Dataset	Classifiers	Training accuracy	Cross-Validation accuracy	Testing accuracy
class 0: 500 class 1: 500 class 2: 500 class 3: 500 class 4: 500	LDA + logistic regression	89.75%	84.25%	70.85%
	SVM	98.45%	72.65%	71.25%
class 0: 579 class 1: 1000 class 2: 1000 class 3: 600 class 4: 1000	LDA + logistic regression	86.47%	86.48%	68.18%
	SVM	97.97%	77.21%	71.77%
class 0: 579 class 1: 1500 class 2: 1500 class 3: 1000 class 4: 1500	LDA + logistic regression	85.07%	84.92%	68.59%
	SVM	97.57%	80.75%	80.09%

Table 3. Comparison of Dataset after data balancing in terms of performance.

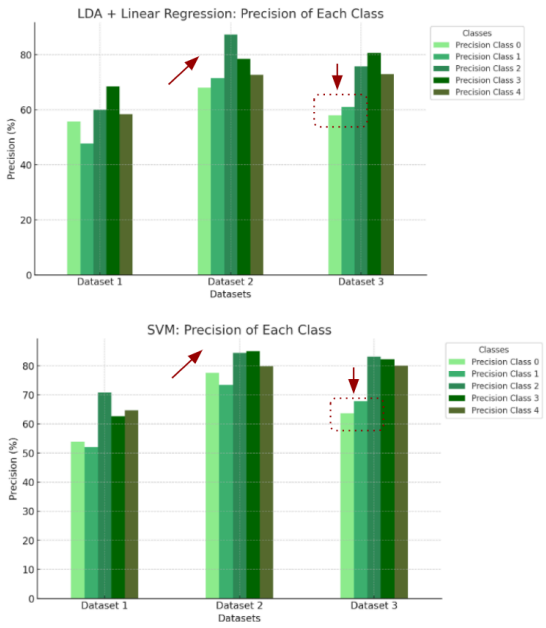


Figure 6. Precision of datasets with different data distribution across each class for Logistic regression and SVM

4. Model Evaluation

4.1 Models trained with the entire dataset

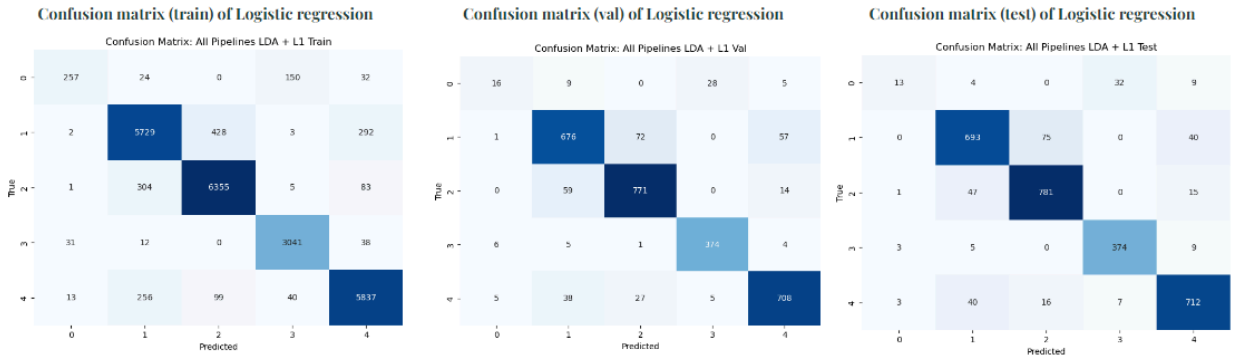
We integrated three pipelines, ran them with the entire dataset, and processed the grid search. Furthermore, we took away the PCA step for Pipeline 3 and raised the PCA feature up to 500 for Pipeline 2. The performances of pipelines 1 to 3 and the integration of three pipelines are shown in Table 4. Among pipelines, pipeline 2 seems to dominate the classification process in the case of non-linear SVM, it gets 98% of training accuracy and 81% of validation accuracy. On the other hand, the integration of SVM gets 99% training accuracy and 89% validation accuracy. The other two pipelines play a significant role in helping the model overcome the overfitting. Furthermore, we fine-tuned the hyperparameter of SVM to reduce overfitting.

As for the logistic regression, by combining the locational key points from SIFT and the gradient of orientation from HOG, we successfully boost the training accuracy by up to 92% and validation accuracy by up to 88% which is compatible with non-linear approaches. Lastly, By learning the pattern from the entire dataset, the test accuracies on both Logistic regression and non-linear SVM are improved by 20% from around 70% to nearly 90%.

Pipeline	Preprocess steps	Feature extractor and dimension reduction	Computational Time (T4 GPU)	Classifiers	Training accuracy	Validation accuracy	Testing accuracy
Pipeline 1	Pristine image Gray scale Histogram Clipping Gaussian blur Bilateral Sobel Center cropping	SIFT, Kmean, VBOW Normalization	6s / 100 examples	LDA + logistic regression	76.72%	76.74%	76.59%
				SVM	76.29%	76.36%	76.24%
Pipeline 2	Pristine image Gray scale Histogram Clipping Gaussian blur Bilateral HOG Center cropping	HOG feature PCA Normalization	20s / 100 examples	LDA + logistic regression	79.15%	77.99%	77.67%
				SVM	98.65%	81.11%	81.10%
Pipeline 3	Pristine image	Body of Resnet50 PCA	27s / 100 examples	LDA + logistic regression	58.57%	51.71%	52.28%
				SVM	74.48%	65.15%	65.09%
Pipeline 1 + 2 + 3	1 + 2 + 3	1 + 2 + 3	54s / 100 examples	LDA + logistic regression	92.13%	88.34%	89.4%
				SVM	99.96%	89.83%	89.96%

Table 4. Performance of pipelines 1 to 3 and the integration of three pipelines using the entire dataset.

The confusion matrixes of the training, validation, and testing dataset and the precision with various datasets for both logistic regression with LDA and non-linear SVM are listed in Figure 7. We successfully maintain the performance of precision on class 0 while improving the precision across all the other classes up to 90%.



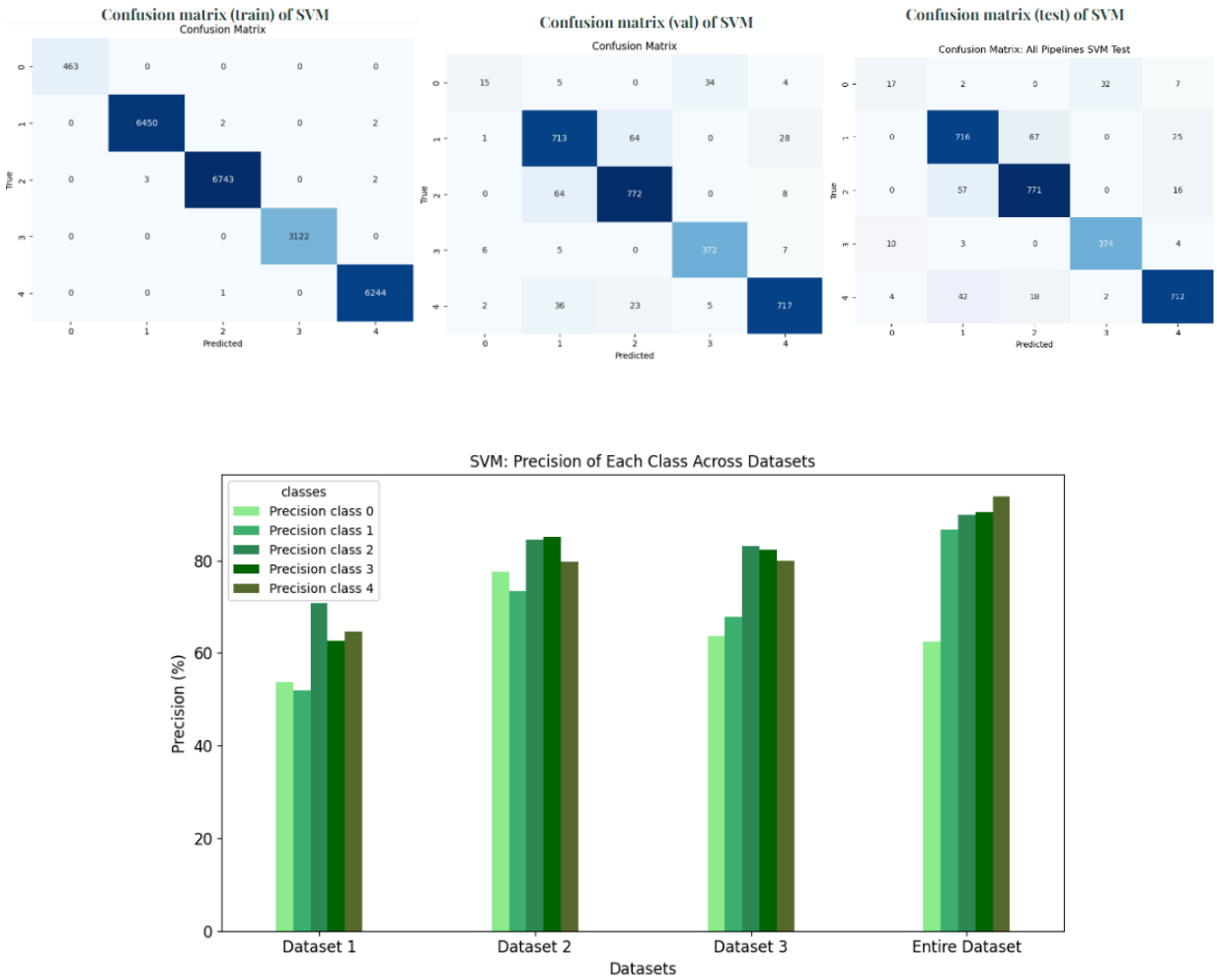


Figure 7. Confusion matrixes of the training, validation, and testing dataset and the precision with various datasets for both logistic regression with LDA and non-linear SVM

5. Conclusion

To wrap it up, We picked up the galaxy images and did the classification task on five classes of galaxies. In the preprocessing steps, we designed custom filters to get rid of the noise in the background and established pipelines for the feature extraction. For the feature extractor, we have SIFT with Kmean and BoVW to capture the locational information and key points of the galaxies' contour after the Sobel and we adopted HOG feature to get the information about the oriented gradient. For the complex feature, we applied the body of ResNet-50 and fed it with pristine images with the expectation that the ResNet-50 could capture meaningful information without filtering. After the feature extraction, we treated our data with PCA to reduce the number of features and did the row-wise normalization. We started with 2,000 examples and tested the

performance of each classifier. We noticed that there was a data imbalance between classes and overfitting in SVM. To deal with the issues, we trained the models with the entire dataset, ran the grid search, and adjusted the hyperparameter of SVM accordingly. Lastly, We fixed the overfitting on SVM with a training accuracy of 99% and a validation accuracy of 89% and successfully improved the precision across all classes. The testing accuracy of both classifiers attained 89% which is comparable with the state of the art performance 92% using end-to-end training on ResNet-50 on Kaggle.