Image Classifier using SVM

Data Science Project

Ardra P S, Hridey and Padmanabha

TIFR, Hyderabad

Abstract

In the modern world, data science is used in almost every application-based industry, and in this project we have classified various images in different groups. We have collected images of famous personalities, vehicles and animals, which act as the objects of classification and have tried to identify them from a test data set, after training our support vector model. In this process, we have used the machinery of scikit learn of python. We have used three different kernels: linear, poly and rbf for the analysis. The test scores achieved vary based on the number of labels used, from 0.6 to 0.9.

1 Introduction

Machine learning is one of the foundations for the development of artificial intelligence and also a path leading to the internet of things (IOT). One of the most encountered problems in this field is that of image classification. This has multiple uses in industry, ranging from identifying trespassing agents, finding lost people and pets, identifying criminals, to some of th more simple ones like auto-arrangement of pictures in gallery.

We have used three types of data sets. One set includes 10 different faces with ten different image of each person. Second data set includes ten different vehicles with ten different models of each, similarly a third set including the Flora and Fauna, ten for each type.

Other than these ten label, we further constructed different labels for the data set such as, for the face set was divided into three labels, to identify ethnicity, gender and the people with or without accessories. While for the vehicle set the new set of labels include in dividing the data into the ones which travel on land, air or water. Labeling was also done on the basis of the number of wheels each vehicle type has. For the third data set the classification was done by giving different label for wild and domestic animals. Similarly, another classification was to differentiate between flowers and animals.The sample images of the data can be seen in Fig. 1 The images were cropped well so that the background objects would not reduce the score giving a better model. To make the machine learning model, a part of the data was

Figure 1: Sample image of one from each data set is provided. All of these were first compressed down to either a 40 by 40 or 100 by 100 size, and then grayscaled.

used for training and rest for testing the model. For this we split the data into different ratios of test and training. The test had different fraction ranging from 0.01 to 0.99 depending on the instance.
For each such fraction plots different scores were obtained and plotted the same having the test ratio along the X-axis and scores along the Y-axis.This was done for the different labels chosen for each data set. getting a different score and there by different plots, So that its possible to compare each method and check the accuracy in the score predictions for different label provided. There are different classifier available for classifying the data set. Hence, we have used three different classifiers like linear classifier, rbf and poly classifier and generated scores for each and also the plots. In next section, we briefly look at the theory of the classifiers.

2 Theory of Support Vector Classifier (SVC)

It is a C-support vector classification whose implementation is based on **libsvm**. SVC is basically a subclass of a larger set of classifiers belonging to SVM(Support Vector Machine). SVC is basically meant to classify a small data-set with at most tens of thousands of data. SVC classifies the data set according to the given type of classification, into a number of subsets. As an example, if we use linear SVC, we will have hyperplanes separating the sets, while for poly SVC, we will have a hypersurface separating the data points. *Here all the data points have equal weights and hence the data set is balanced*

Types of classifiers :

Linear classifier :

It divides the set of data points in the data-spaces with the help of some hyperplanes, choosing the best support vectors separating the points. It is the fastest classification algorithm.

Polynomial classifier :

It divides the set of data points in the data-spaces with the help of some hypersurfaces, choosing the best support vectors separating the points. It is much slower that the linear SVC.

Radial Distribution Function(RDF) classifier :

.It the default classifier of SVC.It divides the data by a complex process of measuring the distances of the data points among each other and finding out the best hypersurfaces by scanning over all the best known surfaces. It the best known data classifier algorithm. It is much slower than the linear, and polynomial counterparts.

To help visualize the types of classifiers, we have included an image below :

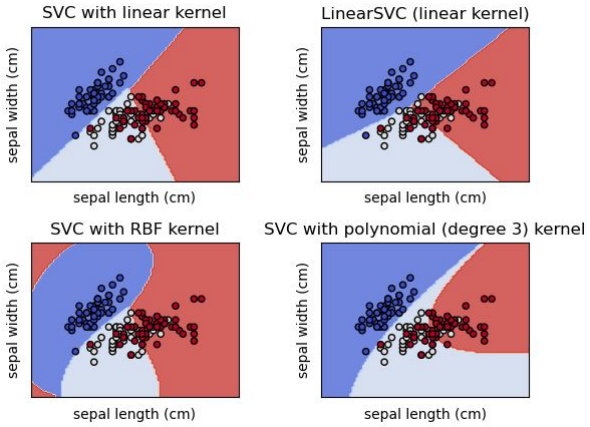

Figure 2: The types of classification. Source: [1].

Types of Scores

Scoring of the obtained results helps us in identifying how much the accuracy of the result is with the actual object. **Precision**: Precision is the fraction of retrieved results that are relevant to the query.
Recall: Recall is the fraction of the relevant results that are successfully retrieved.
F1 Score: The F1 score is the harmonic mean of the precision and recall. The highest possible value of an F-score is 1.0, indicating perfect precision and recall, and the lowest possible value is 0.

3 Observations

We will be using the SVC to analyze three image data-sets: Faces of Different People, Vehicles, and Flora and Fauna[1]. The data-sets can be found at https://tinyurl.com/HrideyFaceData,

[1]The data-sets are collected by Hridey, Ardra and Padmanabha respectively.

https://tinyurl.com/ArdraVehicleData and https://tinyurl.com/PadmanabhaAnimalData.
Essentially, we calculate the test and train score for different test ratio, each averaged over 10 different random states to rule out any fluctuations. The scheme of the classification is as follows: For each data-set, we use three SVC kernels: Linear, Poly[2] and RBF (Radial Basis Functions). We are using scikit-learn's library implementation of SVC, details can b found in ref. [1]. For each method, we choose two sizes of images, 40 by 40 and 100 by 100. Finally, for each data set we perform multiple analysis. For example, in the images data-set we construct classifier for identification of the person, for identification of gender etc. In case of vehicles, one of the classifier classifies based on the number of wheels the vehicle has and so on.
These are a lot of graphs[3], and we only present a sample of them here. Some sample results are shown in Fig. 3, 4, 5 (Accessories Identifier), Fig. 6, 7, Fig. 8 (Wheel Counter) and Fig. 9, 10, 11 (Gender Identifier).
All of the resulting graphs can be found at https://tinyurl.com/DSFinalGraphs. We state in brief the most important observations:

1. Having 10 images of 10 different things and classifying them does not yield good results (the test score hovers around 0.4-0.6 for reasonable test ratios). This is true of all methods.

2. The train score of most methods is around 1.0 always, as expected. This is not true for RBF method. This is because probably the RBF method is ill suited here.

3. Decreasing the number of labels, say by counting the number of wheels of a vehicle, or just classifying gender *always* improves the test score. This is because when the labels reduce, the classifier effectively has *more* data to work with.

4. Poly method yields better results than the simple linear kernel. This is again expected, because Poly has more freedom in fitting.

5. As the test ratio increases, the test score goes down, because the classifier now has less data to work with.

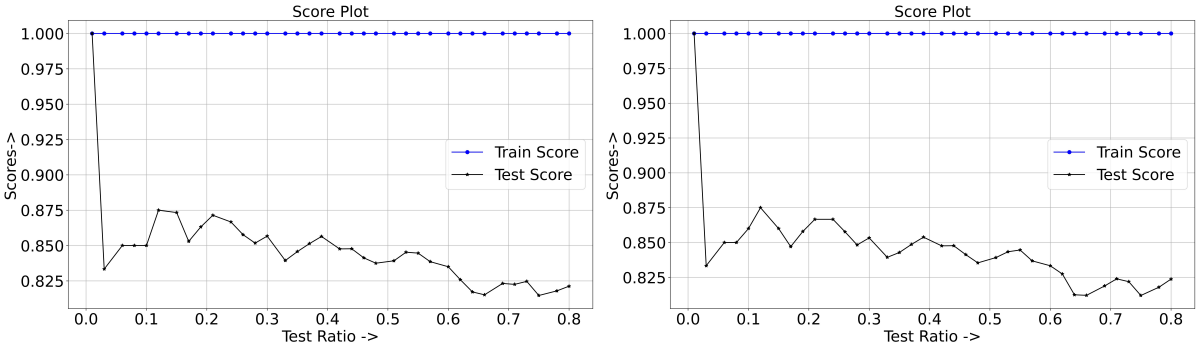

Figure 3: Results for ***Data-set 1: Accessory Identifier****, using* ***linear*** *kernel. 3 labels were used for no accessories, glasses and caps respectively. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

[2]By default, it employs a cubic polynomial.
[3]Perhaps we should classify these using Data Science.

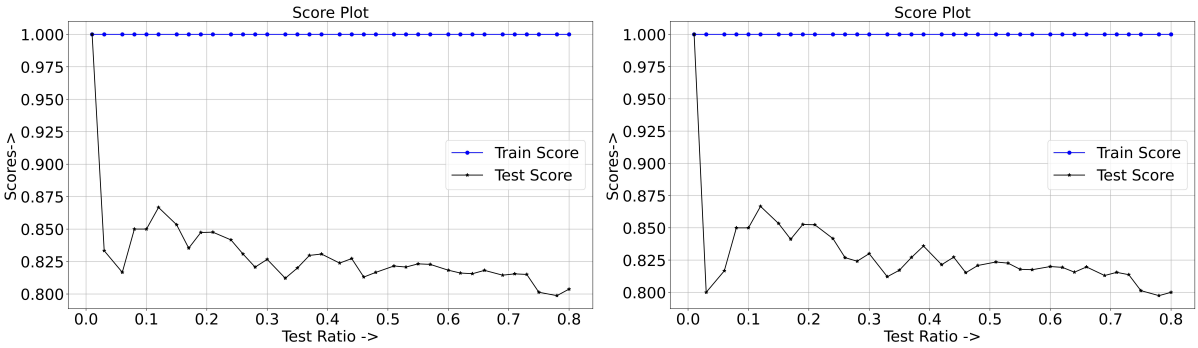

*Figure 4: Results for **Data-set 1: Accessory Identifier**, using **poly** kernel. 3 labels were used for no accessories, glasses and caps respectively. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

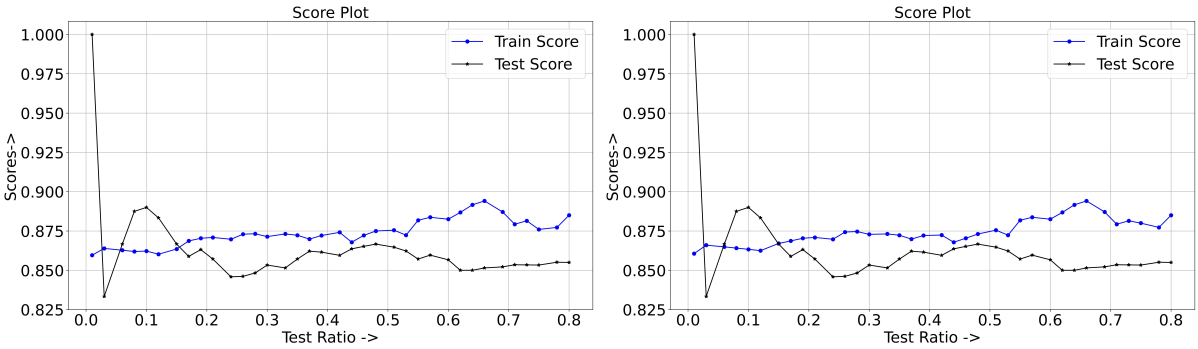

*Figure 5: Results for **Data-set 1: Accessory Identifier**, using **rbf** kernel. 3 labels were used for no accessories, glasses and caps respectively. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

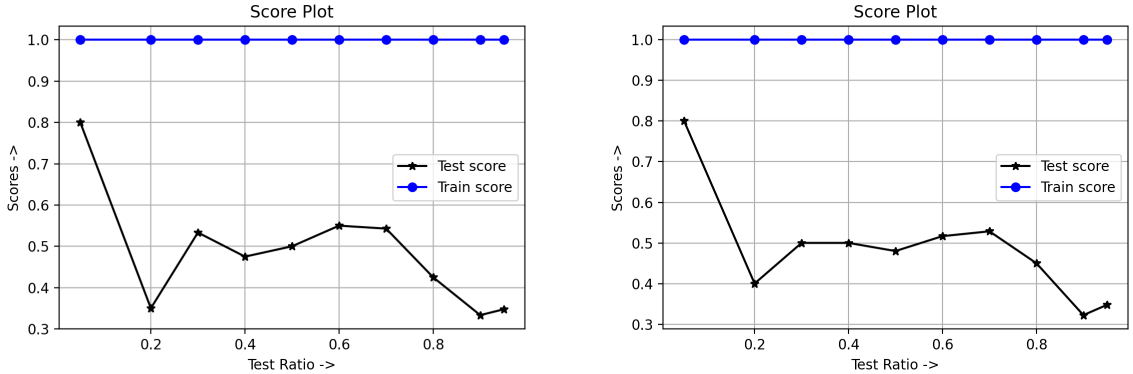

*Figure 6: Results for **Data-set 2: Wheels Counter**, using **linear** kernel. the different labels used were as per the number of primary wheels in the vehicle. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

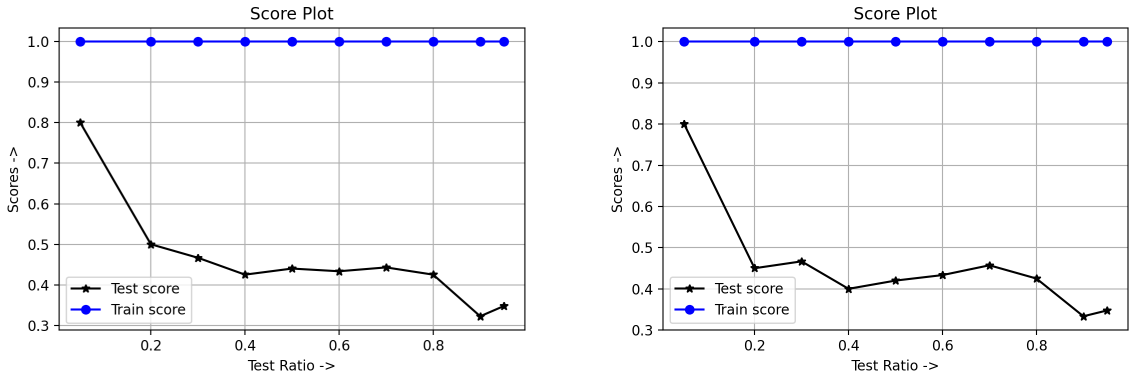

*Figure 7: Results for **Data-set 2: Wheels Counter**, using **poly** kernel. the different labels used were as per the number of primary wheels in the vehicle. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

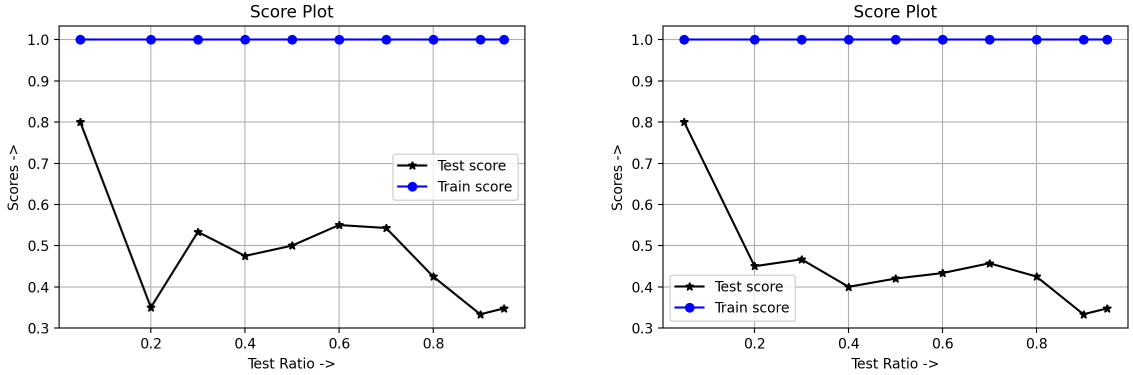

*Figure 8: Results for **Data-set 2: Wheels Counter**, using **rbf** kernel. the different labels used were as per the number of primary wheels in the vehicle. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

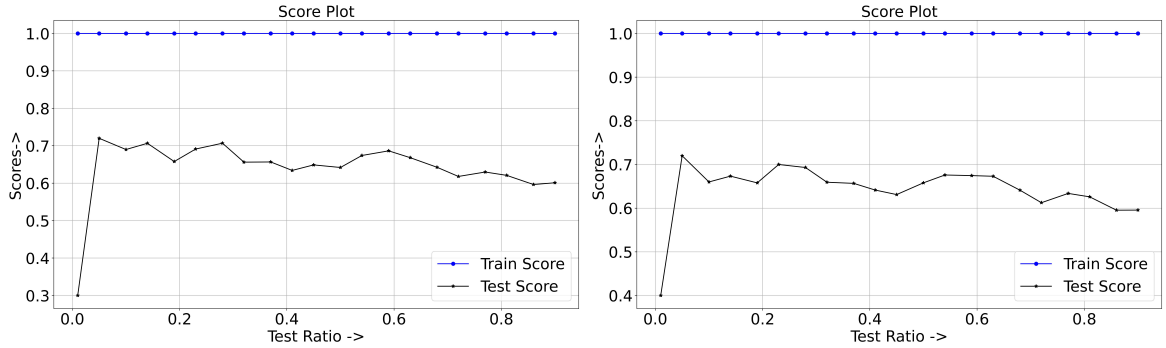

*Figure 9: Results for **Data-set 1: Gender Identifier**, using **linear** kernel. 2 labels were used for male and female gender. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

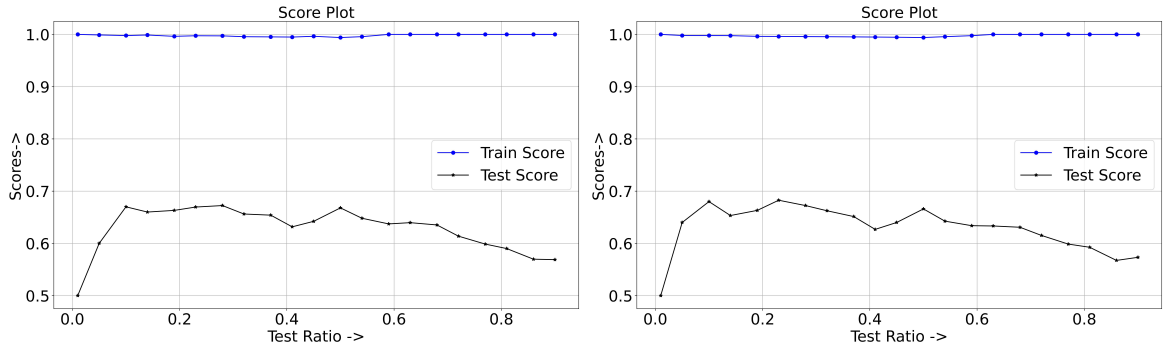

*Figure 10: Results for **Data-set 1: Gender Identifier**, using **poly** kernel. 2 labels were used for male and female gender. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

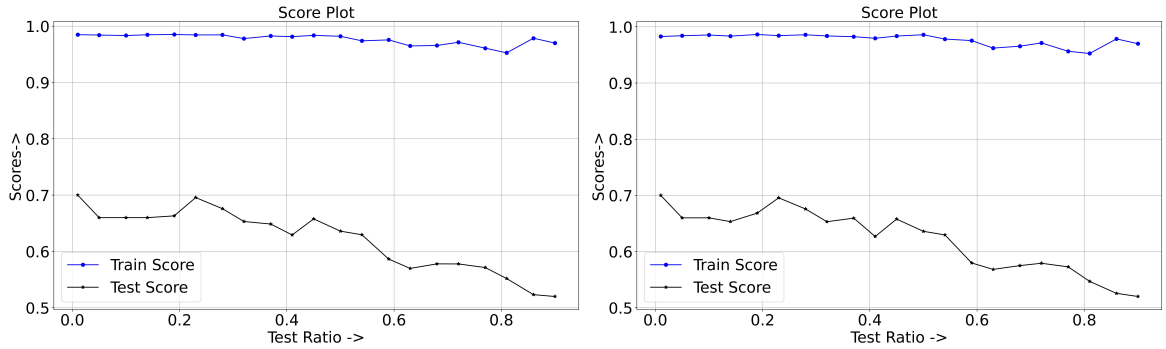

*Figure 11: Results for **Data-set 1: Gender Identifier**, using **rbf** kernel. 2 labels were used for male and female gender. The left side is for 40 by 40 image size and the right side for 100 by 100 image size.*

4 Results and Conclusions

In this study, we analyzed in total 3 data-sets, each composed of a hundred images. The first data-set had images of faces of famous people, second data-set that of vehicles and the final data-set the images of animals and flowers. The images were then compressed to 40 by 40 and 100 by 100 pixel size, followed by grayscaling. For these, we used imageio module in conjunction with numpy. Then we used Support Vector Classifier library function in scikit-learn. After cropping the images properly, we were able to get a score of 0.5-0.6 for small test fraction ratios for the main classifier. For classifiers which required fewer labels we managed to get a score of 0.7-0.9 which is very good. Note that these are better than the random scores expected in both the cases (scores when the classifier randomly classifies).

Potential Sources of Errors

Some of the potential sources of errors are:

- Bias by image cropping: If images of a certain person/vehicle/animal are cropped in a particular manner (say landscape), then the classifier might pick this feature up. This is in some sense cheating, because the classifier uses properties of the image rather than features of the object in question.
- Maximum Test Ratio: An error encountered initially was that if the number of labels is small, say 2, and one of those labels has few entries, then it may happen that all of those entries go into test set when the test set ratio is high. Hence, one has to adjust the maximum test size ratio.
- RBF: Radial Basis Function kernel often performed poorly compared to the other methods, because the problem is best suited to a linear classifier.

Acknowledgements

The authors acknowledge the support and help of the course instructor Raghunathan Ramakrishnan.

Author Notes

The contributions of different authors are as follows: Ardra wrote section 1, collected and analyzed data-set 2 (vehicles), Hridey wrote sections 3-4, collected and analyzed data-set 1 and Padmanabha wrote section 2, collected and analyzed data-set 3. All three authors proofread the entire report.

References

[1] Pedregosa et.al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.