Shape classification using supervised classifiers

Julen Etxaniz and Ibon Urbina

Abstract

In this document we report our proposal for the application of supervised classification methods to the shape classification problem. The datasets we have selected are [plane and car shapes]. The classifiers that we have selected for the classification tasks were: [Linear discriminant analysis, Logistic regression, and Decision Trees]. We have implemented the classification process using the [scikit-learn and pandas library]. We have learned the classifiers using the train data and computing the accuracy in the test data. From our results the best accuracy has been produced by the [Logistic Regression Classifier].

Contents

1 Description of the problem

The task we have to solve is the classification of the shape datasets, introduced in [1] and available from `http://biomecis.uta.edu/shape_data.htm`.

There are 4 shape datasets available. We have selected the plane and car datasets. In each dataset, Cartesian coordinates of each point on the perimeter of each shape are stored as MATLAB data files named as $Class\%d\_Sample\%d.mat$.

1.1 Plane dataset

The plane dataset has the following characteristics:

- ? attributes.
- 7 classes: (a) Mirage, (b) Eurofighter, (c) F-14 wings closed, (d) F-14 wings opened, (e) Harrier, (f) F-22, (g) F-15. See Figure 1.
- 210 instances.
- The data is perfectly balanced. There are 30 instances in each class.

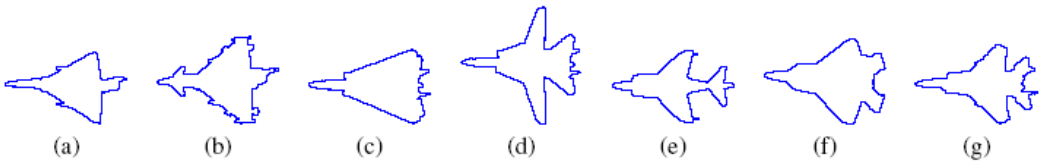

Figure 1: 7 plane classes: (a) Mirage, (b) Eurofighter, (c) F-14 wings closed, (d) F-14 wings opened, (e) Harrier, (f) F-22, (g) F-15.

1.2 Car dataset

The car dataset has the following characteristics:

- ? attributes.
- 4 classes: (a) Sedan, (b) Pickup, (c) Minivan, (d) SUV. See Figure 2.
- 120 instances.
- The data is perfectly balanced. There are 30 instances in each class.

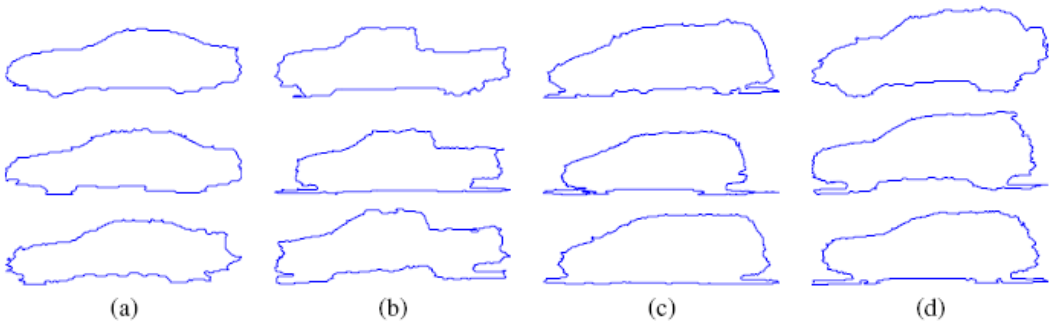

Figure 2: 4 car classes: (a) Sedan, (b) Pickup, (c) Minivan, (d) SUV.

2 Description of our approach

We organized the implementation of the project according to the tasks:

1. Preprocessing of the dataset.
2. Define and learn the classifiers using the training data.
3. Design the validation method to evaluate the accuracy of the proposed classification approaches.

2.1 Preprocessing

We used the pandas library to represent the dataset as a dataframe. We converted the values of the vehicle classes, that were represented as strings, to ordinal values between 0 and 4, because we have four classes.

The data was split into two different sets: train and test, for validation. We use the same train set to learn all the classifiers, and the same test set for evaluating their accuracy.

2.2 Classifiers

We use three different classifiers:

1. Linear discriminant analysis with parameters:
   - solver='svd'
   - shrinkage=None
   - priors=None
   - n_components=None
   - store_covariance=False
   - tol=0.0001
2. Logistic regression with parameters:
   - penalty='l2'
   - dual=False
   - tol=0.0001
   - C=1.0
   - fit_intercept=True
   - intercept_scaling=1
   - class_weight=None
   - random_state=None
   - solver='liblinear'
   - max_iter=100
   - multi_class='ovr'
   - verbose=0,
   - warm_start=False
   - n_jobs=1
3. Decision trees with parameters
   - criterion='gini'
   - splitter='best'
   - max_depth=None
   - min_samples_split=2
   - min_samples_leaf=1
   - min_weight_fraction_leaf=0.0
   - max_features=None

col_0	0	1	2	3
0	80	6	3	5
1	4	65	3	46
2	1	2	94	0
3	7	45	2	60

Table 1: Confusion matrix produced by the LDA classifier

col_0	0	1	2	3
0	83	3	2	6
1	3	74	3	38
2	1	3	93	0
3	5	46	1	62

Table 2: Confusion matrix produced by the Logistic regression classifier

- random_state=None
- max_leaf_nodes=None
- min_impurity_decrease=0.0
- min_impurity_split=None
- class_weight=None
- presort=False

For each classifier, these were the parameters by default of the scikit-learn library[1].

2.3 Validation

To validate our results we compute the classifier accuracy in the test data. Another possibility was to compute the cross-validation in the complete dataset but we used the split between train and test because it was simpler.

As an additional validation step we computed the confusion matrices for the three classifiers.

3 Implementation

All the project steps were implemented in Python. We used pandas for reading and preprocessing the dataset, and scikit-learn for the classification tasks. We illustrate how the implementation works in the Python notebook `Example_Notebook_Course_Project.ipynb`.

4 Results

The accuracies produced by the LDA, Logistic regression, and Decision tree classifiers were, respectively: 0.7068, 0.7376, and 0.6572. Therefore, the best classifier was logistic regression.

The results of the computation of the confusion matrices for LDA, Logistic regression, and Decision tree classifiers are respectively shown in Tables 1, 2, and 3.

The analysis of the confusion matrices shows that the most difficult discrimination, for all algorithms is between classes (1 and 3), which correspond to 'saab' and 'opel' vehicles. This occurs because the Saab 9000 and an Opel Manta 400 cars are more similar between them, than their difference to a double decker bus or a Cheverolet van. Therefore, even if the features are precise, they are not sufficient to make a good discrimination between these two vehicles.

[1]Hereyou should include the parameters of your implementation, not necessarily the default parameters of scikit-learn

col_0	0	1	2	3
0	74	8	5	7
1	6	50	8	54
2	0	6	87	4
3	8	40	9	57

Table 3: Confusion matrix produced by the Decision tree classifier

5 Conclusions

In our project we have applied LDA, Logistic regression, and Decision tree classifiers to the "Vehicle dataset", introduced in [1]. We have computed the accuracy of these classifiers and observed that Logistic regression produces the highest accuracy. We have also observed, from the analysis of the confusion matrices, that all classifiers make a poor discrimination between the Saab 9000 and an Opel Manta 400 cars.

References

[1] J. P. Siebert. Vehicle recognition using rule based methods. Research Memorandum TIRM-87-018, Turing Institute, March 1987.

[2] Catherine L Blake and Christopher J Merz. UCI repository of machine learning databases [http://www. ics. uci. edu/˜ mlearn/mlrepository. html]. irvine, ca: University of california. *Department of Information and Computer Science*, 55, 1998.