Naive Bayes, Decision Tree and Random Forest comparison on Fashion MNIST dataset

Marco Minarelli

Contents

Abstract

The target of this exercise is comparing the results of three basic (yet used) supervised learning algorithms over the Fashion MNIST dataset: Naive Bayes, Decision Tree and Random Forest. The desired outcome is to be able to correctly classify the 10 classes present in the dataset.

1 Algorithms Description

1.1 Naive Bayes

Naive Bayes Classifiers is a family of probabilistic classifiers that relies on Bayes probability theorem.
All the classifiers are based on the strong assumption that all the features are independent. Therefore, those algorithms need the number of parameters to be linear in the number of features.
Maximum Likelihood training is used in the experiments, so learning will take linear time.

1.2 Decision Tree

Decision tree is a predictive modeling approach for inductive inference. This method is used for approximating discrete-valued functions, being robust to noisy data. Moreover, decision trees are capable of learning disjunctive expression.
Classification is done by sorting the instances from the root to the leaf, which provides the classification of the instance. Every node in the tree specifies a test on some attribute on the instance; the branches descending from a node represent the possible value for the attribute being tested.

1.3 Random Forests

Random Forests is a learning method for classification that consists of constructing a multitude of decision trees during learning time and outputs the class that is the mode of the predicted classes of each tree that populate the forest.
Random Forests use a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features.

2 Dataset description

The dataset used can be obtained here: `https://github.com/zalandoresearch/fashion-mnist`.
It contains 60.000 training examples: 28x28 grayscale images, each associated with a label that represents the example class.
The possible classes are shown in Table 1. Moreover, the dataset gives 10.000 more couples image-label as a test set.

Class number	class
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

Table 1: Classes table

3 Approach

Scikit-Learn's Machine Learning training algorithms implementation are used.
To easily use and manipulate data the NumPy library is used.
When using Bernoulli Naive Bayes learning algorithm, to binarize the images, a threshold is set to 0.01, meaning that every non-white pixel contains information.

4 Obtained Results

Learning algorithm	Multiclass accuracy	f1-score accuracy	f1-score macro average
Bernoulli Naive Bayes	70.59%	0.71	0.69
Multinomial Naive Bayes	65.54%	0.66	0.63
Decision Tree	79.10%	0.79	0.79
Random Forests	87.85%	0.88	0.88

Table 2: Multiclass accuracy, f1 score accuracy and f1 score macro average of the various algorithms

Naive Bayes approaches are the techniques with the worst performances. This is (probably) because Naive Bayes assumes independence, which is not true (e.g.: if there is a black pixel, also the pixels around are likely to be black). Moreover, Multinomial Naive Bayes is based on counting the number of times a feature i appears in a sample of class y in the training set. In the Fashion-MNIST dataset, this means counting the number of times a certain pixel has a certain grayscale value.

4.1 More on Decision Tree and Random Forest

In Table 3 are shown the test results when entropy criterion is used (for decision trees) and a higher tree number is in the random forest (respectively, 500 and 1000 trees).
Using the Entropy criterion does not lead to a consistent performance improvement, probably because in both cases, no preprocessing was done. Not preprocessed pixel colors (even if in grayscale) does not contain most information regarding the target feature.
Augment the number of trees to 500 does not lead to a performance improvement, too. With 1000 trees, multiclass accuracy is even lower compared to the case with 100 trees. This loss is probably due to overfitting.

Learning algorithm	Multiclass accuracy	f1-score accuracy	f1-score macro average
Decision Tree with Entropy Criterion	80.19%	0.80	0.80
Random Forest with 500 trees	87.96%	0.88	0.88
Random Forest with 1000 trees	87.78%	0.88	0.88

Table 3: f1 scores of Decision Tree with entropy criterion and Random Forest with 500 and 1000 trees

5 Conclusion

The experiments have shown that the "best" algorithm is Random Forests. But it has to be noticed that no preprocessing was done in any case: this would lead to better performances.