

Fashion-MNIST: A Novel Image Dataset for Benchmarking Machine Learning Algorithms

Adilaraima Martínez Barrio
Informatics Department
PUC-Rio
Rio de Janeiro, Brazil
adilamtnz@gmail.com

Dayrene Frómeta Fonseca
Electrical Engineering
Department
PUC-Rio
Rio de Janeiro, Brazil
dayrene.ff@gmail.com

David Nuñez Cuadrado
Electrical Engineering
Department
PUC-Rio
Rio de Janeiro, Brazil
davidnunez2005@gmail.com

Reinier Morejón Novales
Informatics Department
PUC-Rio
Rio de Janeiro, Brazil
reinierm23@gmail.com

Abstract. Image classification has become a popular domain in machine learning in recent years. In this work we perform a classification task on the Fashion MNIST dataset using several classification algorithms of different nature. Fashion MNIST is intended as a drop-in replacement for the classic MNIST dataset—often used as the "Hello, World" of machine learning programs for computer vision. The Fashion MNIST dataset includes 70000 grayscale images whose size is 28x28 pixels. In order to respond to that problems are applied 9 different classifiers: Linear Discriminant Analysis (LDA), Naïve-Bayes, Decision Trees, SVM with cubic kernel, Multilayer Perceptron Neural Network (MLP), and Random Forest. After training, the SVM cubic model was selected for to process the test set, which reached an accuracy of 90.55%. This result improves the classification used in reference documents, which translates into an increase in performance.

Keywords: Fashion MNIST, Image classification, Machine Learning.

I Introduction

Recently, the researchers at Zalando, an e-commerce company, have developed a new image classification dataset called Fashion-MNIST in hopes of replacing MNIST dataset [1]. Fashion-MNIST, contains 28x28 grayscale images of 70,000 fashion products from 10 categories, with 7,000 images per category. It is intended to serve as a direct drop-in replacement for the original MNIST dataset for benchmarking machine learning algorithms, as it shares the same image size, data format and the structure of training and testing splits. Fashion-MNIST has a more challenging classification task than the simple MNIST digits data, whereas the latter has been trained to accuracies above 99.7% as reported in [2].

The objective of this work is to perform a classification task on the Fashion-MNIST dataset using several classification algorithms of different nature, specifically to develop a classifier that receives an image of a fashion product and returns as output the class label of the input image.

The remainder of this paper is organized as follows: next section describes the proposed methods for the dataset preprocessing and classification. Section III

presents the main features of the employed dataset. The results obtained for the different algorithms and methods applied are described in Section IV. Finally, Section V provides the conclusions of our work and possible future improvements.

II Methods

As stated in the Introduction, several classification algorithms were applied for the classification task. All the employed algorithms are briefly introduced in this section.

A. Decision Trees

Decision tree algorithm [3] is a data mining induction technique that recursively partitions a Dataset of records using depth-first greedy approach until all the data items belong to a particular class. A decision tree structure is made of root, internal and leaf nodes, and is used as a track way in classifying unknown data records. At each internal node of the tree, a decision of best split is made using impurity measure.

B. Random Forest

Random Forest [4] is a tree-based ensemble with each tree depending on a collection of random

variables. Fits many classification trees to a Dataset, and then combines the predictions from all the trees. The generalization error of a forest of tree classifiers depends on the strength of the individual trees in the forest and the correlation between them.

C. Naïve-Bayes

The Naive Bayes algorithm is an intuitive method that employs the probabilities of attributes belonging to each class to make a prediction based on Bayes Theorem. In its design, it simplifies the calculation of probabilities by assuming that the probability of each attribute belonging to a given class value is independent of all other attributes. The algorithm then predicts a class, given a priory probability of the set of features. This is a strong assumption but results in a fast and effective method [5].

D. KNeighbors

The K-Nearest Neighbors algorithm (k-NN) is a non-parametric method used for classification and regression. In k-NN classification, the output is a class membership. An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors (k is a positive integer, typically small). The best choice of k depends upon the data; generally, larger values of k reduces effect of the noise on the classification but make boundaries between classes less distinct [6].

E. MLP Neural Network

The MLP networks [7] are neural networks of feed-forward topology that contain an input layer, one or more hidden layers and an output layer. The layers are constituted by neurons (processors) that have computational capacities. The number of neurons in the output layer is determined by the required dimensionality of the desired response. For the choice of the number of hidden layers and the number of neurons of the same, there are no defined rules. These two aspects determine the complexity of the network. The activation functions of neurons must be non-linear and differentiable. Nonlinearity serves to separate patterns that are not linearly separable and differentiation allows the calculation of the decreasing gradient of the function, thus directing the adjustment of neuron weights during training. The specification of the synaptic weights that interconnect the neurons in the different layers of the network involves the use of supervised training algorithms. The most used training algorithm in these networks is the backpropagation algorithm, based on learning by error correction. In their implementation was defined the activation function as linear, and the number of neurons of the hidden layer were varied in order to find the best structure.

F. Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) [8] is based on the assumption that each probability density functions of the class can be addressed as a normal density and

have the same covariance. The objective of LDA is to determine a hyperplane that separates the data into their classes. This hyperplane is obtained by the search of the projection that maximizes the distance between the mean vector of each class and minimize the interclass variance. A new data is classified determining the highest probability density function.

G. SVM Cubic

SVM was developed by Vapnik and his coworkers in 1995, and it is based on the structural risk minimization. Is a classifier that uses a separation hyperplane that maximizes the margins between the data classes assigned in a higher-dimensional space by the "kernel trick". This maximization leads to an increase in the generalization capacity of the model, and the use of different cores helps to create non-linear decision limits. In addition, the SVM is known for its immunity to overfitting and the "curse of dimensionality". Is based on the structural risk minimization (SRM) principle which seeks to minimize an upper bound of the generalization error consisting of the sum of the training error and a confidence interval. Established on the unique principle, SVM usually achieves higher generalization performance than traditional neural networks that implement the ERM principle in solving many machine learning problems. Another key characteristic of SVM is that training SVM is equivalent to solving a linearly constrained quadratic programming problem so that the solution of SVM is always unique and globally optimal, unlike other networks' training which requires nonlinear optimization with the danger of getting stuck into local minimum. In SVM, the solution to the problem is only dependent on a subset of training data points which are referred to as support vectors. Using only support vectors, the same solution can be obtained as using all the training data points [9].

III Dataset

Zalando's website provided professional photographs of their products, showing the details of the appearance and the model. The original image has a light gray background and was stored with multiple resolutions. The Fashion-MNIST Dataset uses 70,000 thumbnail images of Zalando's products. These products come from different gender groups: men, women, children and neutrals. White products were excluded because they do not have good contrast. For the construction of the dataset, all the images went through a process of resizing to 28x28, cutting edges, converting the images to 8-bit gray-scale pixels.

The Fashion-MNIST dataset is freely available at [10]. To build it was used the front look thumbnail images of 70, 000 unique products. For the class labels, it was used the silhouette code of the product, and each one contains only one silhouette code. Table 1 gives a summary of all class labels in Fashion-MNIST with examples for each class, and Table 2 summarizes its main characteristics. Also, as can be seen in Figure 1, where is shown the class

distribution of Fashion-MNIST, all the classes are balanced.

Table 1 Class names and example images in Fashion-MNIST dataset [9].

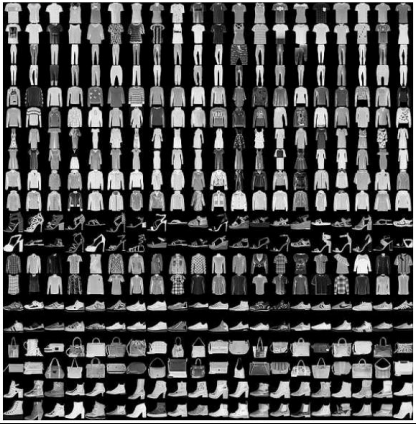
| Label | Description | Examples |
|-------|-------------|---|
| 0 | T-Shirt/Top |  |
| 1 | Trouser | |
| 2 | Pullover | |
| 3 | Dress | |
| 4 | Coat | |
| 5 | Sandals | |
| 6 | Shirt | |
| 7 | Sneaker | |
| 8 | Bag | |
| 9 | Ankle boots | |

Table 2 Characteristics of the selected Dataset.

| | |
|------------------------------|---------|
| Instance number | 70 000 |
| Number of attributes | 785 |
| Characteristic of attributes | Integer |

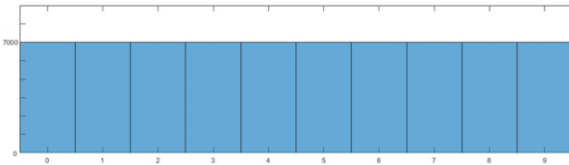


Figure 1 Class distribution of Fashion-MNIST

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test Datasets have 785 columns. The first column consists of the class labels (see above), and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image [11].

Finally, the dataset is divided into a training and a test set. The training set receives a randomly selected 6, 000 examples from each class. Images and labels are stored in the same file format as the MNIST Dataset, which is designed for storing vectors and multidimensional matrices [5].

IV Results and discussions

In order to test all the algorithms and validate their usefulness for this dataset the instances were divided as shown in Table 3.

Table 3 Dataset division.

| Groups | | Number of Instances |
|--------------------|----------------------|---------------------|
| Training Set (85%) | Training Set (70%) | 42 000 |
| | Validation Set (30%) | 18 000 |
| Test Set (15%) | | 10 000 |

The performance metric used to evaluate and compare the models is the percentage of correctly classified patterns:

$$P = 100 \frac{C_c}{N_p}$$

Where P is the performance given in percentage C_c, the number of correct classifications and C_t the total number of patterns.

Firstly, there were applied the Decision Tree algorithm (criterion = entropy, max_depth = 5, Splitter = best), in order to obtain our baseline, which resulted in an accuracy of 49%.

For the creation of our model, we rely on the configuration set by [9]. Initially we used the same parameters in each of the applied algorithms. After several iterations we obtained a better result. These results are shown in the Table 4.

Table 5 Accuracy obtained during validation.

| Classifier | Accuracy | |
|------------------------------|-------------|------------------|
| | Leaderboard | Training Results |
| SVM | 0.897 | 0.90 |
| Random Forest | 0.873 | 0.875 |
| MLP | 0.871 | 0.84 |
| KNeighbors | 0.854 | 0.862 |
| Linear SVC | 0.836 | 0.862 |
| Decision Tree | 0.798 | 0.805 |
| Naive Bayes-Gaussian | 0.511 | 0.593 |
| Linear Discriminant Analysis | | 0.832 |
| Naive Bayes-Bernouli | | 0.720 |
| Naive Bayes-Multinomial | | 0.66 |

Each of the methods used were adjusted until reaching 5 iterations for each one, as in the reference. One of the applied algorithms was: Naive Bayes-Bernoulli, Naive Bayes-Multinomial and Naive Bayes-Gaussian, when only the Gaussian was used in the reference. The result of Naive Bayes-Bernoulli improves the previously proposed up to a 0.720 precision in the prediction.

Another algorithm analyzed was Linear Discriminant Analysis, because the independent variable of our Dataset, follows a normal distribution, which allows to adjust the Gaussian density to each class and therefore shows a good accuracy.

Part of the process of building our model was to test the MLP Neural Network algorithm. The best obtained configuration that balances execution time and result is based on 2 hidden layers with a sigmoid

activation function and backpropagation for the training of fifteen iterations.

The algorithms of Decision Trees and Random Forest were also used, which according to the consulted sources are very efficient for classification tasks with the calculation of the Gini impurity coefficient and entropy. For its part, the KNN algorithm after plotting points, showed that the best prediction based on a training using close examples, is with a value of k equal to 4. K-NN is robust to data noise and sufficiently effective in large Datasets. It can be seen that by taking weighted averages of the nearest k -neighbors the algorithm can avoid the impact of isolated noise examples, but it is based on the memorization of the results and makes the training process somewhat slow.

The application of SVM Cubic made obtain the best result for our model with which it is inferred that it can adjust to the distribution of the data and improve the classification with an accuracy of 0.90. In the case of the results of references, they were based on a linear kernel, while our result is obtained by the modification to a cubic kernel.

As a tool that allowed us to visualize the performance of the model adopted in supervised learning, the confusion matrix is shown in Figure 2. Each column of the matrix represents the number of predictions of each class, while each row represents the instances in the real class. One of the benefits of confusion matrices is that they make easy to see if the system is confusing two classes.

As a positive point that sometimes tends to move away from the real values in the prediction process is to have balanced classes. In the main diagonal, the successes in the classification of each class are evident and the more intense the color, the fewer the failures.

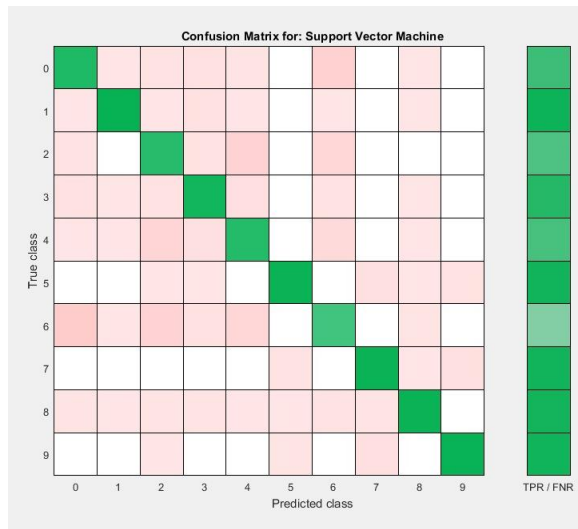


Figure 3 Confusion Matrix

The algorithms were implemented using the Matlab2016b and Python 3.6 with the libraries: scikit-learn v0.20.1, tensorflow 1.12, keras 2.4 distributions.

In general, compared to the reference results, the training process increased its accuracy in all the

algorithms. All the classifiers were relatively good, being the best result obtained for the cubic SVM algorithm. The naive Bayes-Bernoulli offered the worst performance. Once our model is adjusted, the next step is to check how effective the adjustments were. For them, the test set was used, which until now had not been used. This test set has 10,000 instances and were made available by the same creators of the Dataset for the training. In the iteration of the model's test with the cubic SVC algorithm, the result was 90.55 of accuracy.

Table 5: Result of the proposed method for the test set.

| | Classifier | Accuracy (%) |
|-----------------|------------|--------------|
| Proposed Method | Cubic SVM | 90.55 |
| [9] | | 89.7 |

As can be observed, if a superficial comparison is made, only with the observation of numbers, the reference used to create our model, was a little lower. Although many of the proposed algorithms were used in the documentation of references, in each iteration our model was adjusted. Something significant for the classification is to have balanced classes and the previous separation of data for training and testing.

V Conclusions

The objective of this work was to find a good model to classify images of fashion products, which can be very useful for comparing MNIST dataset in terms of prediction complexity. We performed a classification task in the Fashion MNIST Dataset using several classification algorithms of different nature. For this purpose, 10 classification methods were applied: Lineal SVC, LDA, Naive Bayes-Bernoulli, Naive Bayes-Multinomial, Naive Bayes-Gaussian, Decision Trees, SVM with cubic kernel, KNN, MLP and Random Forest each with their relevant adjustments in their parameters to offer the best possible model.

In this work, a model capable of predicting a given image, the class to which it corresponds, with a precision of 90.55% was obtained. With the selected model, the new results are recorded in the work with the Fashion MNIST Dataset a little better than those used as a reference. As future works, other joint methods may be applied, including a combination of classifiers of different nature.

It is recommended to improve the hyperparameters of neural networks that increase the precision in the prediction of this dataset. Apply other classification strategies to continue improving results, just as has been achieved with the MNIST dataset. Publish the current results as a sample of the progress in the use of the Fashion MNIST dataset.

References

- [1] “MNIST.” [Online]. Available: <https://www.kaggle.com/scolianni/mnistasjpg>. [Accessed: 05-Dec-2018].
- [2] D. Ciresan, U. Meier, and J. Schmidhuber, “Multi-column deep neural networks for image classification,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Providence, RI, 2012, pp. 3642–3649.
- [3] S. L. Salzberg, “Book Review: C4.5: Programs for Machine Learning by J. Ross Quinlan. Morgan Kaufmann Publishers, Inc., 1993.,” p. 6.
- [4] L. Breiman, “RANDOM FORESTS.” 2001.
- [5] J. Zhang, D.-K. Kang, A. Silvescu, and V. Honavar, “Learning accurate and concise naïve Bayes classifiers from attribute value taxonomies and data,” *Knowl. Inf. Syst.*, vol. 9, no. 2, pp. 157–179, Feb. 2006.
- [6] “sklearn.neighbors.KNeighborsClassifier — scikit-learn 0.20.1 documentation.” [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html>. [Accessed: 12-Dec-2018].
- [7] H. K. London, “A Systematic Introduction,” *Neural Netw.*, p. 509, 1996.
- [8] S. Mika, G. Ratsch, J. Weston, B. Scholkopf, and K. R. Mullers, “Fisher discriminant analysis with kernels,” in *Neural Networks for Signal Processing IX: Proceedings of the 1999 IEEE Signal Processing Society Workshop (Cat. No.98TH8468)*, Madison, WI, USA, 1999, pp. 41–48.
- [9] L. J. Cao and F. E. H. Tay, “Support vector machine with adaptive parameters in financial time series forecasting,” *IEEE Trans. Neural Netw.*, vol. 14, no. 6, pp. 1506–1518, Nov. 2003.
- [10] A *MNIST-like fashion product database. Benchmark :point_right: : zalandoresearch/fashion-mnist.* Zalando Research, 2018.
- [11] “Fashion MNIST.” [Online]. Available: <https://www.kaggle.com/zalando-research/fashionmnist>. [Accessed: 05-Dec-2018].
- [12] “TensorFlow,” TensorFlow. [Online]. Available: <https://www.tensorflow.org/>. [Accessed: 13-Dec-2018].