Fashion MNIST Project

First Author
chiara.colato@studenti.unipd.it

Second Author
flavia.gianfrate@studenti.unipd.it

Abstract

The project presented is part of the field of artificial intelligence and in particular deals with the problem of image classification. This process uses the information obtained from the pixel analysis, which represent training examples for the algorithms used. After an initial check of the dataset, the data were transformed into an understandable format, suitable for the needs of the work. Then, different automatic learning models were applied with the aim of identifying the one with the greater prevision accuracy.

1. Introduction

The images used in this project represent clothing sold on the Zalando e-commerce. The growth of the phenomenon of online shopping has led to an exponential increase in the number and variety of products available that can be ordered from the comfort of home. For this reason, the most important e-commerce companies are led to find solutions to facilitate customer research, for example through the possibility of doing filtered searches for certain product categories.

The classification of images performed in this project aims to direct each product towards the most suitable category. To achieve this goal, machine learning techniques based on the supervised learning paradigm were used. This approach makes it possible to identify a relationship between the data and its class of belonging with the help of models which, in the presence of similar future situations, will be able to perform the same task with different data.

2. Dataset

Fashion-MNIST is a dataset coming from the Kaggle Website of Zalando's article images—consisting of 70,000 examples. Each example is a 28x28 grayscale image, associated with a label between 10 classes: "T-shirt/top", "Trouser", "Pullover", "Dress", "Coat", "Sandal", "Shirt", "Sneaker", "Bag", "Ankle boot". These labels will constitute the target variable (Y), objective of the previsions made starting from the characteristics of the images, represented by pixels, that will constitute the features matrix (X).

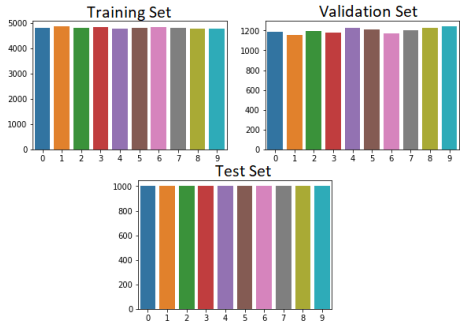
The complete dataset was provided divided into a Training Set containing 60,000 examples and a Test Set containing the remaining 10,000. In order to better evaluate the performance of the tested models, it was decided to extract 20% of the examples from the Training Set which will constitute the Validation Set. Each of the identified parts has a specific and different task: the Training Set is used to train the model that learns the relationships between the input data and the corresponding classes, which will be identified as "target" or "labels".

However, the model could run into "overfitting", i.e. it may be able to perfectly predict the data used in the training phase, but not be able to generalize on new data. To avoid the problem, it's useful to introduce the Validation Set consisting of data that the model does not know. The comparison between the previsions on these data and the real target values is important for evaluating the real predictive capacity of the model.

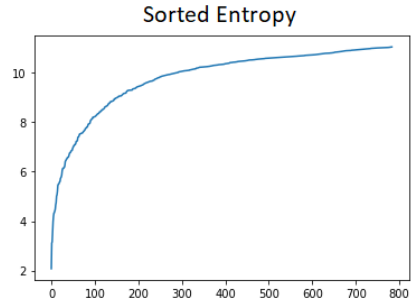
After finding a satisfactory result, the model is tested on the Test Set to confirm the performance of the classifier. In particular, the Test Set is also useful for comparing different algorithms, perfected on the Validation Set, whose performances can be judged and compared on a common situation.

After the subdivision, the data have been observed and in particular their integrity. Then they were preprocessed and a filtered, following the next steps.

- Verification of the presence of null values: it was looked if there were any null values that would later lead to biased results; the entire dataset didn't have any null value.
- Check the balance of the three different datasets: it was observed if the presence of the 10 different classes was balanced in each of the training, validation and test sets, in order to obtain a correctly trained model with an equal number of examples for the different categories.



- Observed the entropy values for each feature, corresponding to each images' pixel: since the entropy of a random variable indicates its "uncertainty" level, and it is known that the more uncertain an event is, the more information it contains, we want to consider those variables for which the entropy level is higher.



- Application of Principal Component Analysis (PCA): the dataset is very large, so PCA is a great way to have a less expensive algorithm and to reduce the number of variables to a smaller number of latent variables, limiting the loss of information as much as possible. The new variables capture 90% of the variance of the original data while reduce the effect of noise and minimize the risk of overfitting.
- Normalization: different scales of variables can influence the model with their own weight; to avoid this problem the method used is the MinMaxScaler which resizes all the features in an interval [0,1].

3. Method

To analyze and model the data, it was decided to follow the Supervised Learning paradigm which provides a "supervision" during the training of the model. The algorithm, in this case, learns the relationships existing in the Training Set, observing examples of data with their relative labels.

In order to be able to classify new images of articles, it was decided to test algorithms with non-parametric nature which have benefits like a better fit to data than parametric ones, they allow greater flexibility and they are not limited by the assumptions of some mapping function.

Each model has been valuated and perfected through the choice of hyperparameters that returned a better accuracy value on the validation set.

3.1. Support Vector Machines

Support Vector Machine aims to obtain the hyperplane with the greatest distance from the closest point (of the training set) of each of the classes. The greater the margin between these points, the smaller the generalization error made by the classifier.

3.2. K Nearest Neighbors

Using the kNN algorithm, it assumes that the training examples correspond to points in d-dimensional Euclidean space. The key idea is that the model assigns to the new example the label of the majority class among the k nearest neighbors.

3.3. Random Forest

A random forest is a classification algorithm consisting of many decisions trees. The fundamental idea behind this is to combine many decision trees into a single model in order to have more accurate predictions than of any individual tree.

3.4. Bagging Classifier

A bagging classifier is an ensemble estimator that fits more "base classifiers" each on random subsets of the original training data set. Then, it aggregates their individual predictions (by voting or averaging) in order to form a final forecast.

3.5. Stacking Classifier

Stacking is an ensemble machine learning algorithm that learns how to best combine the predictions from multiple well-performing machine learning models which can be non-homogeneus.

3.6. Neural Network

Neural networks are a typology of models suitable for manipulating high-dimensional data. These networks are composed of layers of neurons, where each neuron of one layer is connected to neurons of the previous layer. Neural network model multiplies the inputs by parameters and transforms the value obtained through an activation function to obtain the output of the neuron.

4. Experiments

All the models used were trained on the Train Set consisting of 48,000 examples, after which the predicted values for the train set and for the validation set were calculated. The choice of the hyperparameters of each algorithm

was evaluated by comparing these predicted values with real ones with the accuracy score function.

The accuracy is a measure that helps to understand how well the model explains the data it was trained with. The choice of this metric it's because it valorise the examples that have been predicted correctly.

Assuming the use in an e-commerce site, it was considered more useful to evaluate the model on the right predicted products, rather than penalizing the model for having also searched for articles not completely related to research.

$$Prec = \frac{TruePositive}{TruePositive + FalsePositive} \quad (1)$$

$$Prec = \frac{TruePositive}{AllPredicted} \quad (2)$$

4.1. Support Vector Machines (SVM)

The algorithm is used in the project in a non-linear way, using the default kernel function "rbf". We tried more values for the hyperparameter C between 1 and 10. Different values allowed to reach higher accuracy values but at the same time showed overfitting of the model. For this reason it was decided to keep the SVM as best with C = 1.

4.2. K Nearest Neighbors (kNN)

The algorithm was applied using different values of k that is a hyperparameter indicating the number of neighboring examples considered for classification. The highest accuracy value for both the train set and the validation set was obtained using k = 7, however the model with k = 10 was chosen because despite having a slightly lower accuracy, it showed less overfitting.

4.3. Random Forest

The Random Forest algorithm was used with the default number of estimators while trying different values of the "max_depth" and "criterion" hyperparameters. After several tests, the best configuration of the hyper-parameters was that which provided for a maximum depth equal to 10 and which exploited the entropy criterion to conduct the splits.

4.4. Bagging Classifier

After noting that the best performing model was SVM, it was proposed to use the Bagging Classifier, which would have trained 10 SVM models identical to the one defined above. However, it was noted that the improvement was not considerable and that the computational time of execution was about tripled.

4.5. Stacking Classifier

Subsequently an idea was to train the 2 best models obtained, which are SVM and kNN and, through a meta-model, use the two predictions together. The goal was to get a more accurate estimate, but instead the result was lower accuracy.

4.6. Neural Network

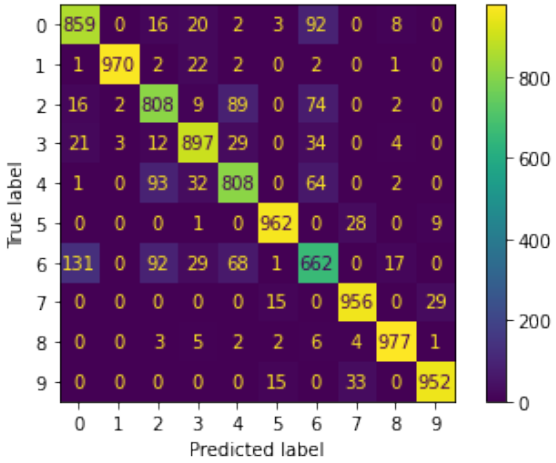
The neural network is a multi-faceted model with numerous hyper-parameters. In particular, for the initial choice of the best neural network, the number of hidden layers, the activation functions of the layers and the number of neurons were evaluated. After several tests and combinations of these hyperparameters, the best neural network was the one with only one hidden layer of 50 neurons, the ReLu activation function for the intermediate layer, where the amount of steps and calculations is greater, and the SoftMax activation function for output neurons. To reduce the probability of overfitting, a Dropout of 0.4 has been added. In compiling the model there are the "Stochastic gradient descent" ("sgd") as optimizer, the "categorical_crossentropy" as loss function and the "accuracy" as metric. Finally, to train the model there is a batch_size of 500 with 500 epochs which, however, could be stopped by early stopping. The early stopping monitors the loss on the validation set and will act when there are 3 consecutive epochs without improvements (without large reductions in the value_loss).

5. Results

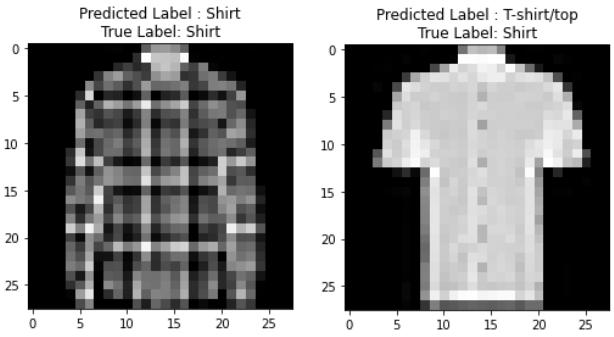
After choosing the best hyperparameters for each algorithm, a comparison of all models is required using the accuracy score on the test set to evaluate it once again on data he has never seen before. A table with the accuracy score values on the test set for each model analyzed can help to understand which model is the best.

SVM	**kNN**	**Random Forest**
0.8851	0.8584	0.8212
Bagging Classifier	**Stacking Classifier**	**Neural Network**
0.8855	0.8846	0.8064

The accuracy of SVM, Bagging Classifier and Stacking Classifier turns out to be very similar. For this reason, the choice of the best model is certainly the Support Vector Machine for being a less complex model than the other two and with a shorter execution time.



Looking at the confusion matrix we see that the most wrong label in the prediction is the number 6 corresponding to "Shirt". In particular, therefore, the model tends to confuse all similar items such as pullovers, shirts and t-shirts. An example of the error is shown here :



References

1. Scikit-Learn, url: https://scikit-learn.org/stable/
2. Kaggle, url: https://www.kaggle.com/c/image-classification-fashion-mnist/
3. Data Camp, url: https://www.datacamp.com/community