Assignment 1, Binary Classification problem with dense neural network

Lorenzo Famiglini - 838675 - lorenzofamiglini@gmail.com

Abstract: The assignment consists in the prediction of default payments using a neural network. There is a class imbalance problem and the aim is to find a model that reach a good F1 measure.

1. Introduction

The data set contains information on default payments, demographic factors, credit data, history of payment, and bill statements of credit card clients in Taiwan from April 2005 to September 2005.
The purpose of this study is to compare two different models, one complex and one simple in order to understand whether it is worth developing such a complex architecture compared to an elementary one.

2. Preprocessing

The data set is composed of 23 features and 1 target variable. After analyzing data types, I decided to apply a normalization for the continuous variable. I found the minimum and maximum points of the training data and then I applied them to the normalization phase for the test set. This was done in order to avoid bias if I had followed this procedure to all the data at once. Consequently, the column "Age" was discretized in three main levels: young, adult, old with respect to the quantile distribution. Finally, I used one hot encoding in order to obtain all non-continuous variables with levels 0 and 1.

2.1. Train, Test split

The training data set was split into train and test set with respect to 80% and 20% proportions. I used the test set (during training phase) for the validation part (also for tuning the parameters).

3. Data exploration

Figure 1: Visualization imbalanced target variable and age discretized

3.0.1. Analysis

After analyzing the correlation matrix and the Chi-squared test for the categorical data, I balanced the data (only the training set) with the oversampling technique called SmoteSVM. It utilizes the Support Vector Machine algorithm to generate new minority observations close to the border between the majority and minority classes. The result of the balanced data is:

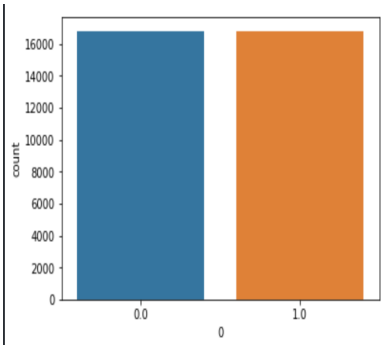

Figure 2: Target variable after oversampling

As a result of the oversampling the train set is balanced and prepared for the model.

4. Dense neural network

4.1. NN Architecture

Different optimization techniques were applied in order to build a reliable model: the first optimization step was based on the choice of the activation function and the loss function for the output layer. At the starting point, Sigmoid activation function was selected with a binary cross-entropy loss and 1 neuron (output layer). But at the end the results of the model were slightly poor. Therefor, I changed the Y target to categorical and then I applied a Softmax function with a categorical cross-entropy loss function and two neurons to the output layer. Two different models were built as to evaluate as many combinations as possible at the level of complexity of the mode. The former one is composed by (in order from the input to the output): [35,512,128,64,128,64,128,2] neurons for each layer. While the architecture of the latter model is as follows: [35,16,64,128,2].

4.2. Grid Search technique and choice of parameters

To choose the number of neurons, the learning rate, what kind of gradient optimization algorithm to use and how many layers to insert, I applied the technique of Grid Search for both of the models. At the end, the best activation function to use inside the hidden layers was the Rectified Linear Unit. Ultimately, the greatest optimization algorithm was the Nadam, which is based on the principle of correction of the gradient method but with the use of Nesterov Momentum. For the learning rate a value of 0.01 has been chosen for the most complex model and 0.001 for the simple model. Due to the complexity of the former model, I used the Dropout technique in order to avoid overfitting. In the previous paragraph, I mentioned the number of neurons and layers for each architecture; these values were obtained by a Grid Search phase on the principle of maximizing the F1 measure. In addition, I also used the Earlystopping method on the simplest neural network. This allowed me to anticipate the overfitting of the model and avoid the use of regularisation techniques. Conclusions and results are supported by empirical evidence (Grid Search techniques).

5. Metrics and results

Model	Acc. train	Acc. val	F1-w train	F1-w val	Loss Train	Loss val
Complex Model	0.81	0.79	0.81	0.79	0.43	0.53
Simple Model	0.80	0.79	0.82	0.79	0.41	0.49

Tab 1. Metrics and results

From table 1 we can conclude that there isn't a significant difference between the models. The precision is also quite similar: 0.70 for the complex model and 0.69 for the the other one (on validation set). The recall for the minority class on the validation set is slightly different: 0.44 for the simple model and 0.54 for the former. It is evident that there is a big difference in terms of computational expenditure, so we should find a trade off between the two models, but at the end the simplest one is better if we consider the overall results and the complexity. Either way, for the prediction task I chose the in-depth model because it produces a better recall for the minority class. Lastly, if we look at the validation loss and the training loss there is a slight overfitting, but this is normal due to the composition and dimension of the validation set.

5.1. ROC curve

When thinking about the classification problem, the ROC curve is an essential tool to estimate which model is the most suitable for our task. The results below refer to Validation set evaluation.

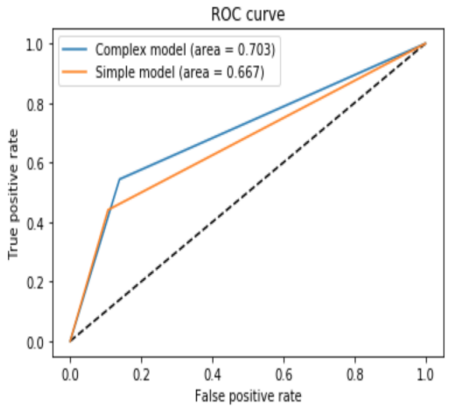

Figure 3: Roc curve

For a fixed value of the percentage of false positives the complex model is preferred when it reaches higher percentages compared to the other. For example, if the aim of the task is to achieve a percentage of false positive between 18% and 40%, we can conclude that the complex model reaches a better performance because its curve is higher up compared to the simple model curve.