Advanced Machine Learning

Assignment 1

LANTIERI Mirko
Matricola 858278

Table of Contents

Introduction

This report describes the approach used to predict the price of Airbnb hosting in New York City for the year 2019, within a Neural Network model. The dataset consists of three '.csv' files including the training sets and test set. In order to make the prediction, it is recommended to do some data exploration of the features available to the dataset. It is worth to mention that the available dataset is already preprocessed[1], but anyway irrelevant data can be found. Also, the source code is written in a Python Notebook, using Visual Studio Code as main editor but any other editor can be used. We will see in details as follows in the next sections.

Importing and exploration

We start our source by importing the relevant and important libraries, whom will be useful during the training and modelling of different neural networks:

- numpy – this library is useful when it comes to manipulation of array-size data inside a frame;
- tensorflow – as seen during laboratory this is the library used to create our NN models (with keras module)

1 This deduction is based to other similar dataset found on Kaggle

- pandas – will help during handling of data frames;
- matplotlib – useful for plotting the results during elaboration;
- sklearn – important library when it comes for splitting the data, scaling the training set and encode categorical data.

After importing the libraries and the dataset we start our exploration: we first begin by testing the shape of each training and test set, to assure that after downloading the dataset we do not have any missing feature. Then we proceed by reviewing different attributes, by watching for each file, the available columns, as we can see in airbnb_price_prediction.pdf for example. The X dataset (including training and test set) contains numerical data like 'id', 'latitude', 'minimum_nights' etc., boolean (binary) data like 'Entire_home/apt', 'Private_room' and also categorical data. While the y dataset contains the id and price feature, which is our objective of prediction. Still in the airbnb_price_prediction.pdf page 2 and 3 we can see in the plots the distribution of price comparing to its density. Other features of any importance can be the counting number of chosen 'Private_room' (over 17500 has chosen a private room, while the rest may have chosen the entire home). We can also see the distribution of minimum nights of permanence or the number of reviews for either private room or entire home. Another exploration we can do to our data is the correlation matrix, which can help explain further correlation between features, but as we can see in page 5 of airbnb_price_prediction.pdf depends mostly from features like private room, availability during the year, longitude and latitude (in case we explore by locations in NYC[2]) and the number of reviews.

Preprocessing

After concluding the exploration, we are ready to preprocess our data. Before cleaning the dataset we first check for potential missing values which in our case they were not missing. In case these data would miss we would fill the empty values with NaN values and then apply the mode or mean to each missing record. The exploration part helped us to understand which of the features could be removed in order to make the regression more accurate but we preferred to remove the id feature as would index each time the record, by accomplishing only redundancy to our set, so we drop this column in each training and test set[3]. We then can see for each set the columns and make some statistical description, which can turn really useful thus to see how our data is distributed.

Feature scaling and normalization

Before scaling our data we normalize the training set containing the price of each listing, by applying the logarithm to each price values: this technique is helpful because it reduce the computation task[4] and also reassemble the interval to $[0;1]$ input. After applying this kind of normalization, we scale the `X_train` under the variable `transformer`, available in the airbnb_price_prediction.pdf , by first applying the `MinMaxScaler()` method, then of course to fit after. After fitting the new data, we scale the train set.

2 New York City
3 We could remove also less important features but it would make unstable our dataset
4 By making it faster

Splitting the training and test data

The normalization and the scaling made our data ready to be split into the new training and test set. This time the chosen test size is of 0.2, because smaller size of splitting would lead to a possible over-fitting during the prediction. The random seed is 60 and is declared initially under the constant variable `RANDOM_SEED` at the very beginning. Now our entire set is ready to be inputted under a Neural Network.

Neural Network Model with 3 layers

For the initial version of the neural network, a relatively shallow three layer neural network will be created. It will consist of densely-connected layers, and use a `relu` activation function for the hidden layers and a linear activation function for the output layer, as it is being used for a regression task. The loss function will be mean squared error (again, because this is for regression).

Four-layer Neural Network and more epochs model

This second model of neural network, consists of dense connected layers, respectively four layers, with `sigmoid` and `relu` activation function for the hidden one. The loss function will be mean squared error, the optimizer Adam with 150 epochs.

Neural Network with SDG optimizer

This third model would identical to the previous in the mean of architecture (number of layers, units etc.) but the only difference is that the hidden layers would have a combination of `selu` and `relu` activation functions, while the optimizer used is `SDG`[5].

Neural Network with dropout regularization at 50%

In this model instead we use dropout regularization. A dropout rate of 50% will be used for the regularization. The number of epochs used are 150 as it does not seem to take more then 150 epochs to reach optimum loss minimisation.

XGBoost standard model

XGBoost[6] is a standard neural network model used for supervised learning problems, where we use the training data (with multiple features) x_i to predict a target variable y_i . It seems to be a good model as the result shows.

Final results

After comparing all the models implemented in the notebook , the final results are as above:

5 This optimizer as shown has bouncy performance.

6 https://xgboost.readthedocs.io/en/latest/tutorials/model.html

1. First model: MSE 0.2821209050290158; MAE 0.3934433364372566; RMSE 0.5311505483655419; R2-score 0.4299887470821828
2. Second model: MSE 0.2852159162253565; MAE 0.3893421514819327; RMSE 0.534056098387947; R2-score 0.42373543093873933
3. Third model: MSE 0.2863689202705159; MAE 0.3954222547228138; RMSE 0.5351344880219513; R2-score 0.4214058436281747
4. Forth model: MSE 0.323951555103329; MAE 0.424994437956569; RMSE 0.5691674227354628; R2-score 0.3454719997083088
5. XGBoost mode: Training MSE: 0.1296; Validation MSE: 0.2065; Training r2-score: 0.7326; Validation r2: 0.5828;

However the graphs and plots available in the notebook, gives a better view and understanding of the models.

Appendices

Appendix A

1. Mirko_Lantieri_858278_airbnb_price_prediction.pdf
2. Mirko_Lantieri_858278_airbnb_price_prediction.ipynb