SUS2024 Report

Camilla Brigandì, Francesco Lazzari, Matteo Pazzini, Riccardo Violano

Introduction

For the Adecco's statistics under the stars challenge we are required to understand and check which candidate is fit for a specific job. Given a job, among different candidates just one of them has to be selected as the "best fit". We propose a data cleaning process divided into different steps. Then we implement the LambaMART algorithm, a LTR (learning to rank) algorithm that is used to solve ranking problems. We finally visualize the features importance and other metrics deemed important.

Datasets preprocessing

Given 3 datasets (candidates, job positions and matches), we do the following operations to make it more understandable and useful for the data manipulation.
First of all we notice that Longitude and Latitude are not always present in the dataset but can be easily retrieved using localization via the Geopy python library. Using the same library we compute the geodesic distance between the candidate location and the work location, since we believe that this information can be crucial for selecting a candidate. Other than this new feature, we incorporate new attributes such as the "checks" (e.g. language, cards...), used to monitor whether a candidate has all the requirements for a job. In particular these are boolean values, 1 when the language/card requested by the job offer is present, 0 otherwise.
In those new columns "Percentual_skills" are used to evaluate the compatibility with the job of a candidate. It is computed as the ratio between the number of skills that the candidate already has and the number of skills needed for that particular job. Other preprocessing changes include substituting "NA's" with zeros in the "language" and "cards" columns and finally dropping all the features that we no longer need. Since we notice that there are few non-null values in the "Education" column and that the content of the this column in the candidates table isn't conformal to the format required in the job table, we decide to drop the column.

Implemented model - LambdaMART

LambdaMART is an ensemble model, so it uses multiple learning algorithms to obtain better predictive performance. It is a combination of LambdaRank and MART. MART uses gradient boosted decision trees for prediction tasks, and LambdaMART improves this by using gradient boosted decision trees with a cost function derived from LambdaRank to order any ranking situation. According to the original paper the model is the last iteration of the model (starting from RankNet), and performs better than the others. For the tuning part we opt for a grid search for the learning rate from 0.001 to 0.3, with 5 different values.

Benchmarks

To validate the new features added to the dataset, we firstly opt for a visualization of the features importance. As it can be seen from the plot, the distance is orders of magnitude more important than the other attributes, and that is expected. Indeed, if we were recruiting we would definitely prefer a candidate who is closer to the job site rather than selecting one who is further away. In our feature plot we also see that the two metrics regarding the job compatibility are less important but still necessary to be taken into consideration, since we wouldn't select a candidate who is not fit for the job.

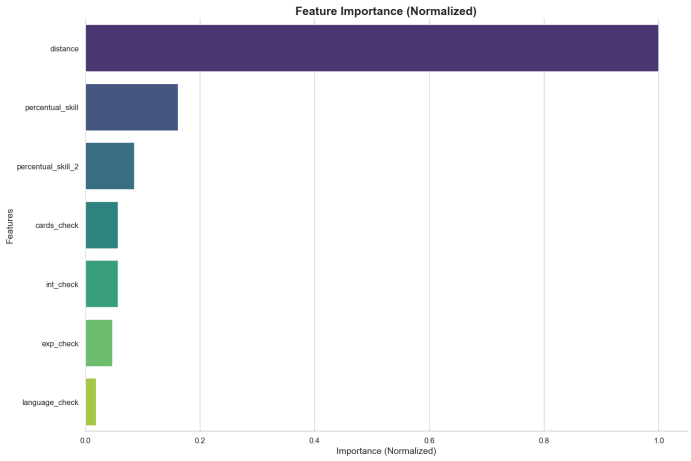

Figure 1: *Features importance visualization*

We then proceeded to plot the confusion matrix: Given that

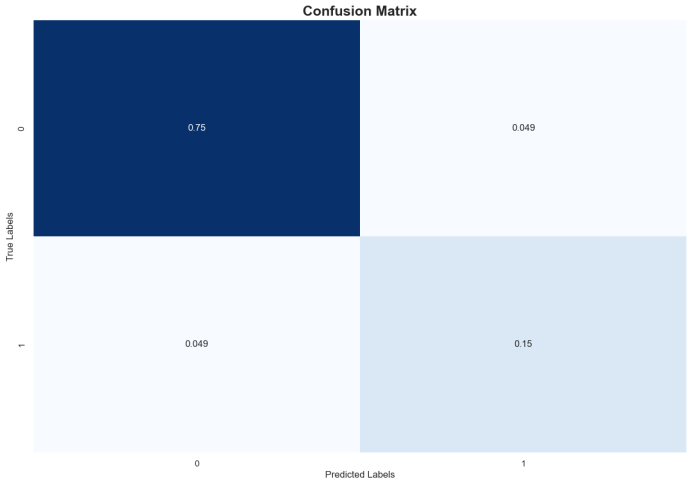

Figure 2: *Confusion Matrix on train after dropping the least important feature*

"language_check" is the least important feature, we decide to drop the column, improving the final accuracy score. The final accuracy on train is approximately 0.9.