Report of Knowledge Discovery and Data Mining’

Data Mining

1/ Problem Understanding

The dataset I used was created by Stefano Leone and is composed of seven files. Indeed, each file represent a version of the game fifa by year from 15 to 20 and includes all the players data for the Career Mode. The one that interest us and that I choose is the fifa 20 file because it’s the most recent one. So, we end up with 104 variables for 18278 observations. The dataset has been done by scraping the website <https://sofifa.com>.

The main problem I wanted to tackle was a classification problem. From all the attributes of a player is it possible to find his position in a team. In order to do it I need to gather as much attributes and data to train a model on and do a classification task.

2/ Data Understanding and Preparation

The data needed to be clean and prepared in order to do my classification. Indeed, the first thing I did is selecting all the variables that will be useful for my classification. All the code can be found and execute in the file code.R

At that step I made the choice to don’t manage the goalkeeper because the way they are implemented in fifa needed me to take them apart, either to normalize their missing value (a lot missing) or adding them some them don’t have like the casual field’s players.

This was done by selecting all the columns of skill that I had and selecting all payer except goalkeepers. Then the other problem I had to solve was to change the datatype so I could exploit them later. All of this work was done by this function:

Une image contenant écran, moniteur, assis, tenant

Description générée automatiquement

Then I had the idea to add an attribute that could be easily compute thanks to the data’s that we had on a player. This was the Body Mass Index (or BMI) that is define by: **BMI** = weight (kg) / [height (m)]2

Finally, the last step was to create more general class to do my position classification job faster and easier. I decided to regroup all the soccer position in 3 groups:

|  |  |  |
| --- | --- | --- |
| 1: Defender | 2: Middle fielder | 3: Attacker |
| -Left Back  -Center Back  -Right Back  -Defense Mid | -Left Wing  -Right Wing  -Midfield | -Attack Mid  -Forward |

This job was done by the function mutate and here is the extract of the code:

Une image contenant assis, écran, table, noir

Description générée automatiquement

Finally, I decided to do a work on my attributes to make a selection on the best of them. This selection was really important for the next job of modelling, in order to avoid collinearity, speed up the computing time. To get an idea of the links between my variables I did the correlation matrix and heatmap: (pictures are available in bigger size at the end of the document or in the picture folder)

Figure 1: Heatmap and correlation matrix between the variables

As we can see in the graph below there are plenty of properties that correlate. To make my selection I choose the one that were the most correlated to the overall of a player. Indeed, the overall of a player is usually the first thing we see and check when looking to a player on FIFA. Here are the variables correlated to overall:

Une image contenant ordinateur

Description générée automatiquement

So this lead me to my unexpected and interesting choice of variables and final data set clean and ready to use for my models : model\_data composed of : mentality\_composure, skill\_ball\_control, power\_shot\_power, mentality\_vision, skill\_dribbling, skill\_curve, Class. This also surprise me to see that in general for a player his overall is mainly due to his mentality\_composture, I would have expected strength, speed or stamina in the first position.

4/ Modeling, Evaluation and Deployment

I did two approaches to tackle my classification problem, first one with the Knn algorithm and the second with support vector machines. All of these methods were implemented and deployed with the help of caret library which is a set of functions that attempt to streamline the process for creating predictive models. The package contains tools for: data splitting, pre-processing, feature selection, model tuning using resampling and variable importance estimation.

In my case I took an 80/20 split with the createDataPartition() that also try to balance the class distribution so that they are similar in both the training and test set. Then the first important thing to do was to define some basic control parameters for the training step with the function trainControl. The arguments are : “*method” that* defines the resampling method during training, “*number”* sets the “k” in k-fold and ”*repeats”* defines how often the cross validation should be repeated.

I used standard value in found on the internet and my courses for repears=”repeatdcv” to do the k-fold cross validation, number = 10 and repeat = 3. The next function is train() function is used to train our model, his arguments is preProcess that is used to scale and center the data, and tuneGrid() defines a grid of hyperparameters to be used in the training.

4.1/ Knn

Knn is a very strong algorithm where the only hyperparameter of this model is k, the number of considered neighbors. So, I did a lot of attempt with different K values to get the best result. I tried with a tunegrid, and the help of the function seq(), between 1 and 80 with a step of 5. Here are the results I get:

Une image contenant texte

Description générée automatiquement

We can see the that the k = 36 is the best result with an accuracy of 0,780 and a Kappa = 0,5965. Kappa statistic is a measurement of the agreement for categorical items.

After the training, The function predict() allow us to see the result. The resulting performance can then be viewed with the function confusionMatrix() :

Une image contenant texte

Description générée automatiquement

4.1/ SVM

I decided to use the package of caret that include a linear SVM here. The hyperparameter to be set is a cost value, which determines the margin around the separating hyperplanes (the bigger the value, the smaller the margin). So again we let the same trainControl() function but this time with an linear SVM

Une image contenant moniteur, capture d’écran, écran, télévision

Description générée automatiquement

Again, I tuned the cost parameters thanks to examples I saw on internet and from my courses.

After the computing I obtain these results:

Une image contenant capture d’écran

Description générée automatiquement

We can see the that the cost = 0.5 is the best result with an accuracy of 0,788.

After the training, again function predict() allow us to see the result. The result with the function confusionMatrix() is :

Une image contenant texte

Description générée automatiquement

5/ Conclusion

In my result we can see one major problem coming from the class 2, I think it come from my initial clean dataset that contains only a few examples of it and during the split sometimes create errors. I could fix it, by taking a look at my initial data set and see the proportion of each class in it. Otherwise for the other classes we can see that we had some satisfying result for the other class. If it was a competition between our models, we can see that the result of the SVM is a little bit more that the Knn. Finally, to go further, I only use the initial function of the possibility offered by the caret package, many tuning possibilities or hyperparameter could have been explored and tried. I choose this dataset and topic because I love soccer and all the possibility that it offers. It was interesting to see the videogames I played for years from another point of view. I can conclude from my study, that now I know what characteristic I will look for a player before buying him and also that data analysis linked to soccer is a very powerful and strategic advantage that professional club use a lot.

Appendix



