Bachelor in Data Science and Engineering

Machine Learning

Departamento de Informática
Universidad Carlos III de Madrid
Course 2020-2021

Assignment 1: Classification and Prediction

April 01, 2022

Marta Balairón García
100451724

Gracia Estrán Buyo
100452014

Marta Balairón García 100451724
Gracia Estrán Buyo 100452014

PHASE 1:

In this first phase we are concerned in collecting the datasets for the classification part. For this, we had to modify our printLineData function to be able to collect the relevant information about the agent, such as the positions of Pacman and the ghosts, the distances between them and the legal actions for example. We may also be able to create arff files. Moreover, we needed to collect the current action of Pacman and the next tick score of the game.

For preparing the datasets, we took advantage of rstudio and we created two different groups of datasets. On the one hand, the datasets for exercise 2, which do not contain instances related to the future, therefore, they don't have the attribute of Next_score. On the other hand, the datasets related to exercise 3, which contain all the information, including the one related to the future (since we want to predict it). In this dataset, the instances which have next_score = 0 are deleted. This is because we know that they are the last instances of one game, so the information that they give us is not new and the scores (next and current) are from different games.

PHASE 2:

The objective of this phase is to train a model which can predict the direction that Pacman is going to take.

In the 6 .arff files generated in Phase 1, we have 21 attributes which are:

- **xP**: numeric
- **yP:** numeric
- **pacman_direction**: nominal
- **legal_north**: nominal
- **legal_south**: nominal
- **legal_east**: nominal
- **legal_west**: nominal
- **dist_g1**: numeric
- **dist_g2**: numeric
- **dist_g3**: numeric
- **dist_g4**: numeric
- **position_ghost1_x**: numeric
- **position_ghost1_y**: numeric
- **position_ghost2_x**: numeric
- **position_ghost2_y**: numeric
- **position_ghost3_x**: numeric
- **position_ghost3_y:** numeric
- **position_ghost4_x**: numeric
- **position_ghost4_y**: numeric
- **score**: numeric
- **direction**: nominal

What is more, this are the number of instances related to each data set:

- **Training keyboard.arff:** 503 instances
- **Test samemaps keyboard.arff:** 211 instances
- **Test othermaps keyboard.arff:** 311 instances
- **Training tutorial1.arff**: 510 instances
- **Test samemaps tutorial1.arff:** 220 instances
- **Test othermaps tutorial1.arff:** 197 instances

As some attributes are numeric, we cannot use some machine learning techniques to apply the classification model and it is obvious that, as we learned in Tutorial 2, we need to start with data-preprocessing before applying any classification methods.

Our target to be classified during this phase is "DIRECTION". The difference between our attributes pacman_direction and "DIRECTION" is that pacman_direction is the action that Pacman did in the previous stage and DIRECTION is the action that Pacman is going to perform in the current tick.

DATA PREPROCESSING:

Marta Balairón García 100451724
Gracia Estrán Buyo 100452014

To perform a good prediction is important to prepare well our datasets with a proper preprocessing. In this case, we have made all our approaches with the training_keyboard dataset performing cross validation (table 1).

Firstly, we applied some filtering deleting the attribute "score" since, in our opinion, it is not relevant for our classification. To continue with, we deleted the instances whose current decision is to take the direction STOP. Our criterion to perform this is that the action of stopping can be considered as noise since it is not a proper action and can disturb us in our prediction.

As a first approach, we apply some individual filters as for example resample, discretizing, normalizing and much more which are shown in the following table with the accuracy:

	Original dataset	Original ds without score	Standardized	Resampled	Randomized
J48	72.36	72.56	72.56	78.72	69.98

	Discretized	PKI Discretized	Centered	Principal components	Normalized
J48	67.79	65.4076	72.56	65.01	72.5

After performing this approach, we realized that the best filter is the resampling one with an accuracy of 78.72, but we want to also try a combination of different of this filter to see if our accuracy gets even better.

In the following table, a combination of some of the best filters above is shown with their accuracies:

	Resampled + PKI discretized	Resampled + Centered	Resampled + Standardized
J48	76.54	78.72	78.72

	Randomized + PKI discretized	Randomized + Centered	Randomized + Standardized
J48	66.20	69.78	69.98

As a final approach, we are going to try the combination of Resample center and standardize filters obtaining an accuracy near to 80%.

Therefore, we realized that the dataset which provides the best accuracy is the one with the attribute score deleted, with the instances of direction = Stop filtered and that has been resampled and centered.

Before performing the testing classifier, we want to compare the j48 decision trees with random forest. The results obtained are shown in the following table:

	Randomized + PKI	Randomized + Centered	Resampled +PKI	Resampled + centered
J48	66.20	69.78	76.54	78.72
Random forest	70.52	72.40	82.81	84.47

To continue with, we are going to perform the supplied test analysis with different datasets (all preprocessed as we just explained before) in order to obtain a much more precise approach. The result of this analysis is shown in the following tables:

Marta Balairón García 100451724
Gracia Estrán Buyo 100452014

KEYBOARD DATA SETS:

	Same maps	Different maps
J48	33.1754	39.5498

TUTORIAL 1 DATA SETS:

	Same maps	Different maps
J48	52.2727	40.6091

KEYBOARD DATA SETS:

	Same maps	Different maps
Random forest	29.8578	42.7653

TUTORIAL 1 DATA SETS:

	Same maps	Different maps
Random forest	56.3636	42.6396

It is easy to conclude that since our accuracies have decrease when performing the test supplied classifiers with respect from the cross validation, it is possible that our model was suffering of over-fitting or that our databases were not initially sufficiently good.

As we can see in the tables, we conclude that the best model is the one performed by random forest, with the dataset of the tutorial 1 preprocessed as explained before obtaining the highest accuracy over 50% in the testing process, as it has slightly higher accuracies than the j48 model.

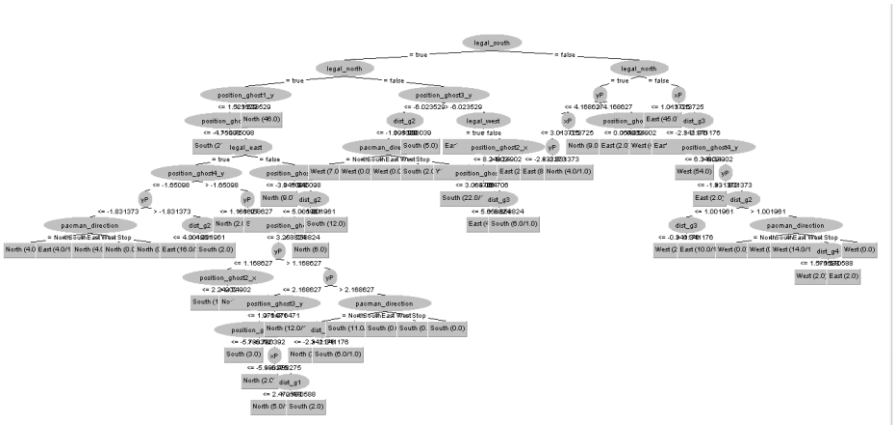

Figure 1: decision tree of the best model (j48)

PHASE 3:

The aim of this phase is to use regression models to predict the score given a current game state.

Similarly, as we did in phase 2, we are going to do all our initial approaches with the training keyboard dataset, in this case, maintaining all the attributes but applying the same filtering of instances. In one hand, we are going to compare some of our better preprocesses of phase 2 and on the other hand we are going to compare some different algorithms using regression. Finally, when we have achieved a reasonably good model, we will perform the test supplied analysis with different data sets.

Therefore, we are going to start to compare datasets with different filters applied individually, and some algorithms using cross validation.

For analyzing the regression model, we make use of two different parameters. On the one hand, we compare the correlations, and on the other hand we take advantage of the root mean square error, this is the standard deviation of the residual, which is the prediction errors. In terms of accuracy a best fit has a small RMSE and high correlation. We are also interested in the Root relative squared error which is a normalized value that represents similar insights as RMSE.

Table of RRSE	PKI Discretized	Centered	Randomized	Resampled
Gaussian Regression	26.80	20.36	20.58	18.87
Linear Regression	29.44	19.02	19.23	17.46
Super Vector Machine	26.78	14.87	14.71	13.79
K-Nearest Neighbors	17.48	33.26	36.68	26.75
Regression Trees	19.09	25.26	25.08	23.87

We can easily admit that the best data set was the resampled one followed by the randomized and the centered ones. What is more, the best regression algorithm was the super vector machine followed by the linear regression.

Furthermore, we also did a final approach combining some of the filters and observing again the best algorithms

	Randomized + Centered	Resampled + Centered
Linear Regression	24.6985	23.0199
Super Vector Machine	21.5902	20.1312

This table shows the best results of the best algorithms. We can easily admit that the results are quite good, but we must check now the test supplied validation with different datasets, all of them preprocessed previously by filtering the instances with direction STOP, resampled, and evaluated with linear regression and the algorithm based in Super Vector Machine.

In the following table the results are shown:

KEYBOARD DATA SETS:

- **Same maps:**

	Linear Reg	SVM
RRSE	46.4176%	21.6857 %
Correlation	0.9318	0.9737

- **Different maps:**

	Linear Reg	SVM
RRSE	53.255 %	19.0967 %
Correlation	0.8366	0.96

TUTORIAL 1 DATA SETS:

- **Same maps:**

	Linear Reg	SVM
RRSE	39.7244 %	13.3739 %
Correlation	0.8676	0.9837

- **Different maps:**

	Linear Reg	SVM
RRSE	51.2973 %	26.0235 %
Correlation	0.8667	0.9658

As a conclusion, we can easily admit that the best data set was the resampled one combined with the super vector machine, which provides always a quite low errors and a considerably high accuracy.

PHASE 4:

To build the automatic agent, we need to connect Pac-Man with Weka, so that the generated models indicate Pac-Man which action it should execute in each state. From the previous phases, we have selected the random forest model as we have proven it is the best for the prediction we want. We have also selected the tutorial1_training dataset with the filters resampled and centered, as they provided the best results to our desired prediction. Using javabridge and weka-wrapper packages in python and importing the class Weka,

we can connect Weka with the agent. We have included in our script some code to call the model we have selected (J48) and the selected dataset (keyboard_training) using the predict function. Before this, it was needed to have all instances collected in variable l, and to do so, we have used our printLineData method, splitting it in the choseAction method, inside the WekaAgent class, to collect each instance. By doing this, we are making Pac-Man follow the predictions of our model to move, that is, the model tells Pac-Man the direction it must take so that it plays correctly.

QUESTIONS:

1. What is the difference between learning these models with instances coming from a human-controlled agent and an automatic one?

The difference is that the decision taking of the automatic agent is supposed to be better because it takes many factors of the game into account, such as the distance to the ghosts. Whilst the human-controlled agent decision making it cannot take into account mathematical aspect which require quick calculations such as distances.

2. If you wanted to transform the regression task into classification, what would you have to do? What do you think could be the practical application of predicting the score?

To transform the regression into classification, we would have to discretize the attribute to define categories instead of numerical values. The practical application of predicting the score could be analyzing if the score gets worse or better to see if the direction taken is a good decision.

3. What are the advantages of predicting the score over classifying the action? Justify your answer.

The advantages of predicting the score over classifying the action are that by predicting the score we get a better prediction of the performance of the Pacman because the attributes have more correlation with the predicted score than the score attribute.

4. Do you think that some improvement in the ranking could be achieved by incorporating an attribute that would indicate whether the score at the current time has dropped?

We believe that there would be an improvement because we would have more information about the last tick and it would indicate if the agent's performance has been better or worse, knowing the distance to the ghosts and that the score increases when the agent eats a pacdot or a ghost.

CONCLUSIONS:

With this assignment, we have learned to put into practice and visualize all the theoretical concepts we have learned during the last year and a half about machine learning, data preprocessing, modeling, using data analysis to classify and many other things whose utility has been proven with this assignment. We have realized that it the best model to predict what we want depends on the dataset characteristics and attributes.

Although we have not achieved the best predictions, we have learned to look for the best model we could have. We have also learn the methodology of how this algorithms work.

We did not get the perfect results, but we consider we have learnt to understand machine learning processes and how to apply them to a dataset. Also, one thing we did not know, was how to connect a model such as j48 or random forest to a game like Pac-Man to make it play alone and to optimize it.