09/04/2023

Practice 1

Classification and Prediction

MAHCHINE LEARNING I

Ángela María Durán Pinto 100472766
Paula Gutiérrez Arroyo 100472845

UNIVERSIDAD CARLOS III

PHASE 1

We have followed the following approach.
We add to 'BustersKeyboardAgent', the attribute self.line = “”, it is initialized as empty.
Our function printLineData() returns all the information related to the current state.
When we call it in chooseAction, in self.line we are saving the information about the current state calling to printLineData(), in order to have it when we are in the following tick.
Once we are in the next tick we write in the document the information of the previous tick (self.line) plus the current score which corresponds to the FutureScore attribute.

After that we have created our datasets training_keyboard.arff and test_samemaps_keyboard.arff we have using the maps: OpenHunt, sixHunt, openClassic, originalClassic, oneHunt

And for test_othermaps_keyboard.arff: mediumClassic, testClassic, smallClassic, capsuleClassic, contestClassic.

Finally, we have removed the observations whose value for the attribute Pacman_direction was ‘stop’ in order to avoid future problems. We have done it in Weka, using the filter unsupervised/instance/removeWithValues.

PHASE 2

First of all, as in the Weka class the last attribute is assumed to be the class, we have moved the Pacman-direction attribute to be the last one. For that, we have used Filter/unsupervised/Reorder.
Now, we start with the preprocessing step. We have removed the “FutureScore” attribute, as Pacman should not have future information at the moment of the decision.
In addition to that, we have balanced our attributes with respect to the Pacman_direction class using ”unsupervised/instance/Resample”. Then, we have normalized them, in order to avoid different value ranges “unsupervised/attribute/Normalize”.
Clearly, we have applied all these modifications to the three datasets.

Now, the next step is to “Choose the algorithm”. For that, we wanted to check the performance with our datasets of some different algorithms: J48, ID3 and OneR.
OneR algorithm, uses the minimum-error attribute for prediction, discretizing numeric attributes.

For each algorithm, we have evaluated it with 1) cross validation, 2) supplied test_samemaps_keyboard, 3) supplied test_othermaps_keyboard

For the ID3 algorithm, we have applied an additional filter, Filter/unsupervised/attribute/Discretize to discretize our numerical data in order to be able to apply it.

Cross validation

Algorithm	Accuracy	F-Measure	Mean absolute error
J48	77.3887 %	0,774	0.1019
ID3	77.6505 %	0,798	0.086
OneR	47.1204 %	0,469	0.2115

Use test_othermaps_keyboard

Algorithm	Accuracy	F-Measure	Mean absolute error
J48	40.0376 %	0,396	0.2391
ID3	25.3759 %	0,401	0.2371
OneR	16.5414 %	0,160	0.3338

Use test_samemaps_keyboard

Algorithm	Accuracy	F-Measure	Mean absolute error
J48	49.5763 %	0,494	0.201
ID3	92.9974 %	0,930	0.0346
OneR	27.8249 %	0,278	0.2887

In the first phase, we have generated our data taking account of a lot of attributes. In order to reduce dimensionality, we have applied different feature selection techniques. We have used as Attribute Evaluator: Information Gain, and OneR; and as Search Method: Ranker

We are saving the ACCURACY

Accuracy	Removed attributes
77.945 %	Number_ghosts, Ghost2_Distance,Living_Ghost2
77.8796 %	Number_ghosts, Ghost2_Distance,Living_Ghost2,Ghost2_Direction
77.8469 %	Number_ghosts, Ghost2_Distance,Living_Ghost2, Living_Ghost1, Living_Ghost3, Living_Ghost4
77.7487 %	Number_ghosts, Ghost2_Distance,Living_Ghost2,Living_Ghost3

Now, in order to get our final model we have removed the selected attributes since we consider (respect to the accuracies) that is the one that gives us the highest accuracy without losing much of the information. We removed Number_ghosts, Ghost2_Distance,Living_Ghost2, Living_Ghost1, Living_Ghost3, Living_Ghost4 .

PHASE 3

For this phase we have used the model that we have chosen in the previous step, called j48model.model. In order to manage building an automatic agent, we have started from the class BustersAgent and changed the chooseAction function. As we want our agent to be automatic, in each Tick that we call to the chooseAction function, we must use the predict function of Weka in order to get the movement the pacman should take.

We have obtained an error so that our pacman is not able to move.
Therefore, we are not able to make a comparison between the different agents.

The BasicAgentAA (the one from tutorial1), follows the shortest path in order to eat all the ghosts. The BustersKeyboardAgent, as is human-controlled, could follow a good path, but it is followed by view, and not using an efficient manner.Consequently, sometimes it could lead to a very good path, and other times to a disaster.

PHASE 4

We checked that the last attribute is “future score”, then we started the preprocessing step. In this case, we balanced our attributes by using ”unsupervised/instance/Resample”. Afterwards, we normalized them with “unsupervised/attribute/Normalize”.
Then, with our 3 datasets we have used 2 algorithms: “MultilayeredPerceptron” and “Gaussian Processes”. We did this with our training set with the cross validation and then using the 2 test sets (samemaps and othermaps)

Cross validation

Algorithm	Correlation coefficient	Mean absolute error	Relative absolute error
MultilayerPerceptron	0.9997	43.6202	2.2479 %
Gaussian Processes	0.9997	37.2941	1.9219 %

Use test_othermaps_keyboard

Algorithm	Correlation coefficient	Mean absolute error	Relative absolute error
MultilayerPerceptron	0.6089	17412.9201	1018.651 %
Gaussian Processes	0.9997	17095974.4294	1000109.7981%

Use test_samemaps_keyboard

Algorithm	Correlation coefficient	Mean absolute error	Relative absolute error
MultilayerPerceptron	0.9941	201.9596	7.407 %
Gaussian Processes	0.9945	257.0971	9.4291%

Regarding the algorithms, the MultilayerPerceptron is a neural network architecture that contains multiple layers of interconnected nodes or neurons, making it suitable for both classification and regression tasks.

On the other hand, Gaussian Processes are a type of probabilistic model that can be utilized for both regression and classification. They offer a probabilistic forecast of the output value, which can be beneficial for estimating uncertainty in the predictions.

Based on the performance metrics provided, it seems that the Gaussian Processes algorithm performs better than the MultilayerPerceptron algorithm for both sets of data, with higher correlation coefficients and lower mean and relative absolute errors.

Now, with our algorithm selected “Gaussian Processes”, we remove the attributes that don’t significantly affect its performance. For that we follow the same approach as in Phase2. In this case we use as Attribute Evaluator: PrincipalComponents; and as Search Method: Ranker.

Correlation coefficient	**Mean absolute error**	**Relative absolute error**	**Removed attributes**
0.9997	37.2941	1.9219 %	None
0.9997	37.1951	1.9168 %	Number_ghosts, Dist_nearest_pacDots, Ghost3_Distance, Ghost4_Distance
0.9997	36.3895	1.8753 %	Number_ghosts, Dist_nearest_pacDots, Ghost3_Distance, Ghost4_Distance, North_Legal,Ghost2_Posy, Ghost3_Posx
0.9997	37.0463	1.9091 %	Number_ghosts, Dist_nearest_pacDots, Ghost3_Distance, Ghost4_Distance, North_Legal,Ghost2_Posy, Ghost3_Posx, Living_Ghost3, Ghost3_Direction, Ghost4_Posx, Ghost4_Direction, Pacman_direction

QUESTIONS

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

   On one hand, the training instances provided by human-controlled agents have the goal of optimizing the performance of the machine learning model for a particular task, in our case, eating all ghosts. The instances may be hand-labeled with correct outputs or may be generated by humans following a specific set of rules or guidelines. These instances are typically of high quality and accuracy, but may not fully represent the range of possible inputs that the model may encounter in the real world.

   In contrast,automatic agents generate training instances without human intervention. These instances may be generated by the model itself or by other automated processes, such as data collection tools or simulations. These instances are often more diverse and representative of the real-world inputs that the model may encounter, but may also be noisier or of lower quality than the previous instances.

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 task into classification we would need to create categories based on the predicted score. We could divide the score range into intervals and assign them a name or class. For that we would use a discretize filter.

   Regarding applications, in general, predicting the score could help to improve the player's experience.

   One application could be to optimize the behavior of the ghosts to make the game more challenging and enjoyable. By predicting the score that the player is likely to achieve based on the information about the current state and the action performed, the game could adjust the behavior of the ghosts to make them more or less difficult to catch, strategic, or unpredictable.
   Another application could be to provide feedback to the player on their performance and strategy.

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

   Predicting the score provides a more precise prediction than classifying the action. Scores are numerical values which provide a range of numbers which is way more informative than a label (like in the classification). This means that in prediction/regression we achieve a more detailed and flexible decision-making.
   However, classification overall is easier and simpler to interpret than regression.

As a consequence of the flexibility, predicting the score allows the game to be able to adjust the difficulty level, rewards, in real-time to keep the player engaged and challenged.

Finally, in order to predict the score, more complex ML models can be used, which leads to a more immersive experience for the player compared to classification models.

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 consider that incorporating that attribute could potentially improve the ranking performance of the model since the drop of the score could be interpreted as a change of the behavior of the agent or it could change by far the predicted score value. For instance, if Pacman lost, we could not predict that the value of the score in the future would be really high, since it already lost/died.

CONCLUSIONS

Overall we have seen that Weka is an useful tool when talking about data, classification and regression. Along the course of this practice we have come to understand that it can achieve a lot of things, and we had to get used to the different features it provides (some of them were tricky because they were a little hidden).

Another thing that we found interesting was how much it changes the values of accuracy or other evaluation metrics depending not only on the algorithm but also on the dataset and the preprocessing that has been done to it.

The model is useful for, the obvious, getting to know what exactly is more relevant when playing the game itself. That is, if we want to optimize and know what the direction is going to go to next, the model gives insights on what variables or attributes are more useful or are more correlated to it than others. Decision trees in this case come in handy and are way more visually interpretable.

The problems encountered when doing this were, mostly, on the third phase. It was difficult since we had to download the virtual machine, then the code kept giving problems (even on parts that did work before). Also, to get to know where some of the features on weka were, since sometimes it is quite difficult to know.
Our opinion on the practice was overall good except for the third phase. The instructions were a bit vague, broad even, for our understanding on the exact task that was asked.