Examen Aprendizaje Automático y Minería de Datos

Grado en Desarrollo de videojuegos, enero 2024

General instructions

The exam consists of two parts, a theoretical part worth 2 points and a practical part worth 8 points.

The theoretical part will be submit on campus with the student's name as file name without spaces and in one of the following formats: txt, docx, pdf. It will not be corrected if it is in a different format.

The practical part must be submit in a Jupiter Notebook inside a zip file with all the resources you need to execute. Everything, both the implementation and the possible explanation must be in the notebook. For this you can make use of the markdown cells in case you need to write any explanation or answer any question. The Jupiter notebook file must be executable by itself without the teacher's intervention, that is, make sure that what you submit runs and the zip has all the necessary to run. The only exception is the possible use of libraries, although initially you should not need any beyond those seen in class. If the submission is not correct and does not run (e.g. data is missing, the paths are absolute and it does not find the dataset, etc) the practical part of the exam will be penalized with up to -1 points.

In the zip file you must also include your library with the implementation of the generalized multilayer perceptron for multiple layers and neurons per layer. This library must be based on the code submmited in practice 5 and 6. It can have modifications, but if the code is radically different the exam will be failed and it will be treated as a copy.

It is allowed for the practical and theoretical part the use of the campus resources, notes and any support material that you bring with you. The use of ChatGPT or similar AI tools is not allowed.

Theoretical part

Time available 30 minutes.

Answer in a text document and submit it to the virtual campus with your name in the file name and also inside the file as the first line. Put the number of the question you are answering and then the answer.

Question 1 (0.5 points).

If I have a classification dataset with images of size 128x128 and using a multilayer perceptron I have not achieved good results in predicting the type of image involved, what transformations would you make to the data and/or the model to try to improve the results of the perceptron? Note, in the end the classification must be done by the perceptron, what is asked is what can we do before applying the perceptron. Justify your answer.

Question 2 (0.5 points).

Which of the following models has the ability to model nonlinear problems? Justify your answer.

- Decision tree based on ID3.
- A Deep Neural Network with 3 convolutional layers and a last layer Linear() the Pythorch.
- A multilayer Perceptron with activation function y=m*z+b where z is the value of the neuron.
- None of the above.

Question 3 (0.5 points).

We have a neural network model that gives us a training accuracy of 70% and a validation accuracy of 50%. Knowing that our baseline is 65% classification accuracy, what strategy would you follow to try to improve the results?

Question 4 (0.5 points).

We have 200 training data of a game saved by a player and with this data, we want to build a machine learning model that tries to imitate the player to develop an AI. The data that have been saved are: The position of the player (in which square he is), the input that has been performed (move left, right, shoot or jump), the position of the enemies in the scenario (square they are in) and the content of the 8 squares contiguous to the player (values that the squares can take: empty, ground, enemy, item). The game moves continuously, but it is logically divided in the form of a grid of size 1x1 game units.

If we want to use a multilayer perceptron, what number of input and output neurons will the perceptron have and briefly explain how you would train it? Justify the answer.

Practical part

Maximum time for the practical part is 2 hours and 30 minutes.

We have a dataset on dementia with the following fields:

- Subject ID: patient ID.
- MRI ID: identification of the visit to the doctor.
- Group: Group to which the patient belongs, which can be:
  - Demented: Has been diagnosed from the beginning.
  - Nondemented: The patient has not been diagnosed at any time.

    - Converted: 14 patients were initially misdiagnosed and ended up with dementia.

- Visit: visit number. If it is the first, second, ... visit of the patient.
- MR Delay: Deley since last visit.
- M/F: Gender.
- Hand: whether they are right-handed or not.
- Age: Age

And the following values that are medical tests performed to patients: SES,MMSE,CDR,eTIV,nWBV,ASF.

In the dataset there are several patients, each of them have visited the doctor at least 2 times. We want to build a model that predicts the patient's status between Demented, Nondemented and Converted. With this we will be able to identify in time especially those patients whose initial diagnostic tests indicated that they were not demented, but ended up being so.

Exercise 1 (1 point): Clean the dataset of errors and elements that are not relevant. Justify in a markdown cell the decisions you have made.

Exercise 2 (1 point): Graph the data once they have been cleaned.

Exercise 3 (2 points): Use your Multilayer Perceptron from practice 5 and 6 to build the prediction model. Calculate accuracy and the confusion matrix. The minimum accuracy result you should achieve is 65%. The model can have any layer, but **there must be a test performed with more than one hidden layer**, even if it is not the final model you set up. **Make it very clear which is the model you are finally going to keep.**

Exercise 4 (1.5 points): Use Sklearn Decision Tree algorithm to build a second model. Calculate accuracy and the confusion matrix. The minimum accuracy result you should achieve is 85% (random_state=42).

Exercise 5 (1 point): Compare by writing in a markdown cell both models and choose justifiably which one you think is the best to solve the problem presented in this exam.

Exercise 6 (1.5 points): Transform the Converted class into Demented and create a binary classification model with your multilayer perceptron with a single output neuron. Try to get the best performace as you can.

Delivery method:

In the virtual campus with a zip file where all the necessary content to run the jupiter notebook is. The zip file must have your name and in the first line of the notebook must also appear your name.

Attachment: API with some useful Python functions.

In Pandas: isna() : select null fields, combined with .sum() you can check the number of null fields in a dataframe.

.drop("column",axis=1) removes a column (same but with an index and axis 0 removes a row)

dropna(how='any'): deletes null fields.

to_numpy() converts a dataframe to a numpy array.

From numpy:

np.hstack((A,B)) merges two arrays A and B by adding B to the end of A. They must have a compatible size.

np.zeros_like(np_array) generates an array of the same size as np_array of zeros.

You have also the downloaded documentation of MLPClassifier and DEcission-TreeClassifier that you can consult.