Project 3: Machine Learning - What Is It Good For?

16:198:520

It is another day on the deep space vessel Archaeopteryx, and you are a tiny bot. You're responsible for the safety and security of the ship and crew. The aliens are back.

For this assignment, we are going to use machine learning to learn a model for controlling the security bot and, with any luck, improve on it. For the purpose of this assignment, we are considering Bot 1 as in Project 2, with one human and one alien. For the purpose of data collection and experimentation, you may take the ship layout as fixed, though the positions of the bot, the human, and the alien should be selected at random as usual. For α and k, use the values you found to be best in Project 2 (be clear in your writeup about which you are using).

1 Model 1: Modeling Selected Actions

Generate a dataset of the decisions that Bot 1 makes, by running repeated simulations and experiments and recording *the state of the experiment and/or knowledge base*, and *the action Bot 1 selected* at each point in time. You will then fit a model to this dataset to predict the action that Bot 1 will select for a given state of the simulation.

For the writeup, answer and be clear about the following (showing your math when necessary):

- How are you defining the input space? What information are you using as input to the model, and how are you representing it mathematically for the model?
- How are you defining the output space?
- How are you defining the model space - what kind of models are you considering to map from input to output?
- How are you defining the loss or error of your model? *Note: not all actions/directions can be selected at every point in time!*
- What are you using as your training algorithm to find the best possible model?

For full credit, your model needs to have features that are learned from the data, rather than ad hoc features by fiat.

> Do **not** try to write a fully generalizable neural network library for this. This is excessive, unnecessary, introduces many more points of failure than you potentially want to deal with for this project. I will take off points if you do this.

Your writeup should also include:

- Results from training, showing the model learning over time.
- An evaluation of your trained model to show that it does not overfit the data.
- A comparison between the actual performance of Bot 1, and the performance of a Bot that acts according to the selections of this trained model.

2 Model 2: Predicting Bot Performance

For this model, you want to predict the performance of a bot, namely, what is the probability that the bot successfully rescues the human. In this case, the dataset will be based again on simulations of Bot 1 - pairing current state/knowledge bases, with whether or not the bot was successful in rescuing the human.

For the writeup, answer and be clear about the following (showing your math when necessary):

- How are you defining the input space? What information are you using as input to the model, and how are you representing it mathematically for the model?
- How are you defining the output space?
- How are you defining the model space - what kind of models are you considering to map from input to output? *Note: We want to predict probabilities here.*
- How are you defining the loss or error of your model?
- What are you using as your training algorithm to find the best possible model?
- Given how successful Bot 1 is, how do you avoid success being overrepresented in your training?

For full credit, your model needs to have features that are learned from the data, rather than ad hoc features by fiat.

> I repeat: Do **not** try to write a fully generalizable neural network library for this. This is excessive, unnecessary, introduces many more points of failure than you potentially want to deal with for this project. I will take off points if you do this.

Your writeup should also include:

- Results from training, showing the model learning over time.
- An evaluation of your trained model to show that it does not overfit the data.

3 Model 3: Improvement

For this section, we will actually be training two models:

- **ACTOR** - the actor model takes as input a state / knowledge base, and gives as output a representation of what action to take in that state.
- **CRITIC** - the critic model takes as input a state / knowledge base *and a representation of what action will be taken*, and gives as output the probability that the actor network, taking the indicated action next, will successfully rescue the human.

The models will be trained separately, based on the following loops:

- **The CRITIC Loop**
  - Generate data based on the performance of the ACTOR network on the task of saving the human. Note, you may need to institute a time limit / threshold on the task.

  - Train the CRITIC network based on this data, to predict the success of the ACTOR network in saving the human.
  - Note, this parallels the structure of the training for Model 2.

- **The ACTOR Loop**
  - *Note:* The ACTOR network is initially trained to mimic Bot 1.
  - Generate data based on the performance of the ACTOR network on the task of saving the human.
  - For each state/knowledge base in the data set, use the CRITIC network to determine what action would have been best to take (according to the CRITIC's predictions).
  - For the state/knowledge bases in the data set, and the 'best' action as determined by the CRITIC network, train the ACTOR network to select these best actions.

The goal then is the following:

- Initialize the ACTOR network to mimic Bot 1.
- Train the CRITIC network to predict the performance of the ACTOR network.
- Train the ACTOR network to predict the better actions as predicted by the CRITIC network.
- Repeat, training the CRITIC network, then the ACTOR network based on it, etc.

Include in the writeup the following:

- The usual specifications for the model spaces, the loss functions, the learning algorithms.
- Can you show that the CRITIC network is successfully learning the performance of the ACTOR network?
- Can you show that the ACTOR network is successfully learning the CRITIC-specified actions?
- Can you show that the ACTOR network is improving based on this process?
- Can you beat the performance of Bot 1?

Bonus: Or, for partial credit - you may do all of the above using a machine learning framework of your choice (pytorch, tensorflow, jax, whatever you like).