Competition II: CS4786/5786-Machine Learning for Data Science (Fall 2016)

Find the bot Challenge

In-Class Kaggle competition for CS4786: Machine Learning for Data Science

Fall 2016

The second in-class Kaggle competition for our class:

This competition is based on the find the bot example we covered in class for HMM's. There is a 5x5 grid on which bots can move around. There are 3 different bots that each follow a different (probabilistic) pattern of movement. Each round/run, one of the 3 bots is selected at random according to some fixed distribution pi and is left to move around for 100 steps on the grid. While you never see which location in the grid the bot is at on any round, after each step you do get to observe how close the bot is to the top left corner of the grid (according to euclidean distance from the corner), think of this observation as you hearing how close the bot is. The observation matrix you are provided with are these observed distances for 3000 runs each of 100 steps. Your goal is to identify the location of the bot on round 100 in each of the runs. To help you with this, we have provided the locations of the bots on the 100th step for the first 200 runs. You will be tested on your predictions of the bots locations on the remaining 2800 runs.

Here is what you are provided with

- The **Observations of the bots**: You are given a 3000x100 matrix where each row of the matrix is observations made in one run. That is, row 20 column 40 specifies, how close the bot was on the 20th run and 40th step. This data is provided to you in the **obervations.csv** file in the comma separated values format.
- A few **labeled example**: You are also provided the location of the bot on step 100 for the the first 200 runs. The locations are labeled from 1-25. That is, each square in the 5x5 grid is labeled by a number from 1 to 25. by traversing left to right and top to bottom.

Task: For each of the remaining 2800 runs, predict where the bot is on the 100th step. The competition will be hosted on in-class-Kaggle.

Kaggle Link: https://www.kaggle.com/t/a159a375ad704dba8a233abd2340f729

Download the data below as zip file. When unzipped you will find the two files, obervations.csv and labels.csv

competition2_data

Group size: Group of size 1-4 students.

Due date The deadline is **11:59 pm, Monday, 5th December**. The due date for the report on CMS will be announced soon and is a couple days after the competition closes on Kaggle. Submit what you have at least once by an hour before that deadline, even if you haven't quite added all the finishing touches — CMS allows resubmissions up to, but not after, the deadline. If there is an emergency such that you need an extension, contact the professors.

> 1. Footnote: *The choice of the number "four" is intended to reflect the idea of allowing collaboration, but requiring that all group members be able to fit "all together at the whiteboard", and thus all be participating equally at all times. (Admittedly, it will be a tight squeeze around a laptop, but please try.)*

Deliverables:

1. **Report:** In the end of the competition each group should submit a 5-15 page writeup that includes visualization, clear explanation of methods etc. See grading guidelines for details about what is expected from the writeup. **(worth 50% of the competition grade)**
2. **Predictions:** Competition is held on Kaggle in-class as a competition. You can submit your predictions to kaggle to compete with your friends. You should also submit your predictions on CMS. **(worth 50% of the competition grade)**
3. **Code:** Submit the code you used for kaggle as a zip file.

Collaboration and academic integrity policy

Students may discuss and exchange ideas with students only within their group.

We distinguish between “merely” violating the rules for a given assignment and violating academic integrity. To violate the latter is to commit fraud by claiming credit for someone else’s work. For this assignment, an example of the former would be getting detailed feedback on your approach from person X who is not in your group but stating in your homework that X was the source of that particular answer. You would cross the line into fraud if you did not mention X. The worst-case outcome for the former is a grade penalty; the worst-case scenario in the latter is academic-integrity hearing procedures.

The way to avoid violating academic integrity is to always document any portions of work you submit that are due to or influenced by other sources, even if those sources weren’t permitted by the rules.[2]

> *2. Footnote: We make an exception for sources that can be taken for granted in the instructional setting, namely, the course materials. To minimize documentation effort, we also do not expect you to credit the course staff for ideas you get from them, although it’s nice to do so anyway.*

Grading Guidelines:

Grading:

- Clear explanation of your main model (20 points)
    - Explain any preprocessing you did, explain clearly what your model takes as input
    - Explain clearly what algorithm you used to train and not just the model
- How does your model fit the problem description? (10 points)
- How does model account for the fact that there were 3 bots? (10 points)
- How were parameters chosen in a principled fashion? (10 points)
- Failed attempts. (Have a clear flow of your reasoning for why you tried various models and how their failure guided you to pick next one) Give clear comparison of things you tried. Dont go for numbers but rather clear progression of thought and how each model guided the next. (15 points)
- Visualization (what did you learn from them and how they guided you). This includes tables, plots, graphs etc. (10 points)
- Supervision: How did you use the labeled examples given in your model. Did you use these to minimize kaggle submissions? (10 points)
- Unlabeled examples: How were the unlabeled data points part of you model (10 points)
- Understanding data, what did you learn from the observations and how was it used in your approach? (5 points)

Bonus (at the discretion of the graders):

- Tried new or more methods not necessarily covered in class
- Developed new algorithm or methods, tweaked existing methods to fit the problem better