Project in Probabilistic Models

Antti Takalahti

Project report

UNIVERSITY OF HELSINKI

Department of Computer Science

Helsinki, May 4, 2015

Contents

1 Introduction

This is a report for the Project in Probabilistic Models (582637) course.

The task was to write a predictor that produces a probability distribution for 1000 rows of 303 column data. The source of the data is unknown and values are from 0 to 99 where 0 indicates that no measurement was made.

The task was split into three rounds, and a sample data was provided for each round. Total score was calculated by taking the natural logarithm for each probability per correct value and these were summed.

Initial algorithm was provided by Johannes Verwijnen, who was the teaching assistant for the course and the code is hosted on GitHub and can be accessed using the URL https://github.com/anttitakalahti/ProProMo2015 and any files mentioned are found there.

2 First round

I created a modular system where I could evaluate many different predictors to get an idea about their performance. I ended up using Java as a programming language as the example was written in Java. I added Apache Maven and set up Docker to have it installed on the system. I had not used Docker before so this took some time, but I feel that it was time well spent. I got the system set up pretty nicely and it helped with implementing and evaluating different ideas about possible predictions.

I took a look peak into the data and noticed that the initial guess to predict previously seen value and its neighbour was silly and I managed to double

the performance by removing that aspect. In the end I used a zero predictor for the first round with a score of −284 229 which is about 0.39 per correct value.

3 Second round

First I converted all non zero values to x and searched for patterns, but sadly there weren't any. I got 35 894 different patterns with two patterns having 5843 and 4189 rows and the rest had under 1000 rows each. The sample data had 67 785 rows total.

I then calculated the zero probabilities for each column and noticed that positions 16 (0.2718) and 50 (0.2636) look interesting. I started to predict these columns with zero probability < 0.5 and got better results. I ended up calculating smoothed probabilities for each value in each position and got a score of −89 150 which is close to 0.75 for each guess.

4 Third round

For the third round I had the system set up perfectly so that I could try out new ideas and get instant feedback on them. I added a Statistics class and calculated means and variances and noticed that means are around 60-80 and variances vary a lot.

I modelled a situation where values group together by adding a class called Peak where values in a row are grouped so that there is no gaps of more than 5 in values. It's not the greatest name in hindsight, but I got the feeling that

the following value is close to previous value but then there were jumps. as if there were two different peaks simultaneously. The file /data/peaks.txt contains more information about the peaks. Like max value - min value, where you can see that 1 has 35 176 appearances and 2 has 30 144 and so on.

I also calculated probabilities for each value and found that the value is never smaller than 36 or larger than 84. The item count is the most interesting data in peaks. You can see that a large number of Peaks have five items in them and I have no explanation for this phenomenon. It sounded like no one in the class did know what would cause this.

Table 1: Item counts for Peaks in rounds 1 and 2

Items	Round 1	Round 2
1	6092	6478
2	2418	2925
3	2493	2924
4	5014	5456
5	79 685	83 706
6	4533	4552
7	3106	3371
8	3130	3222
9	3699	3794
10	8369	7354

Finally I added a matrix to see how a value in each position predicts a value in the following positions and that data is in file predictor_matrix_3.txt under /src/main/resources/statistics/ but it did not yield a better score than using the updated probabilities from round 2.

I submitted the final solution with a score of 74 389, which is about 0.78 for each correct value.

5 Thoughts

I learned some real world skills in using Docker container. Test Driven Development approach is not at best when trying to explore options and the

final approach is unclear, but it helped in fixing bugs and I would use it again even though it was not perfectly suited for this kind of task. I had to keep an eye on not doing the things that were most comfortable for me like the actual programming and optimising the quality of the code instead of trying to solve the actual problem.

I tried to focus too much on trying to guess if there is a value in position and not enough on what that value might be. I failed to notice significance in patterns like a row with five values of 59 in the first round. The Peak might have helped and I got it close to my best solution but It did not exceed the score in any time.

I had fun in this course and I would recommend this to anyone. The setup seems strange and it is impossible to know how much time you will spend since there are no guaranteed paths to success like in other school problems, but I feel that it gives a nice taste about the real world problems.