For this problem,

Input state is the exact round we reached including the information of the 2 parameters as and

The ouput action is the arm we pull from the selected slot machine

The reward is 1 or 0, if the slot machine returns twice our dollar invested, and 0 if we lose the dollar

The code worked fine in my environment . As per the model / script for each slot machine, we introduced 2 new variables, one for counting the no of times the machine returned a 0 reward and the other that counts no of times the slot machine returned 1 as reward

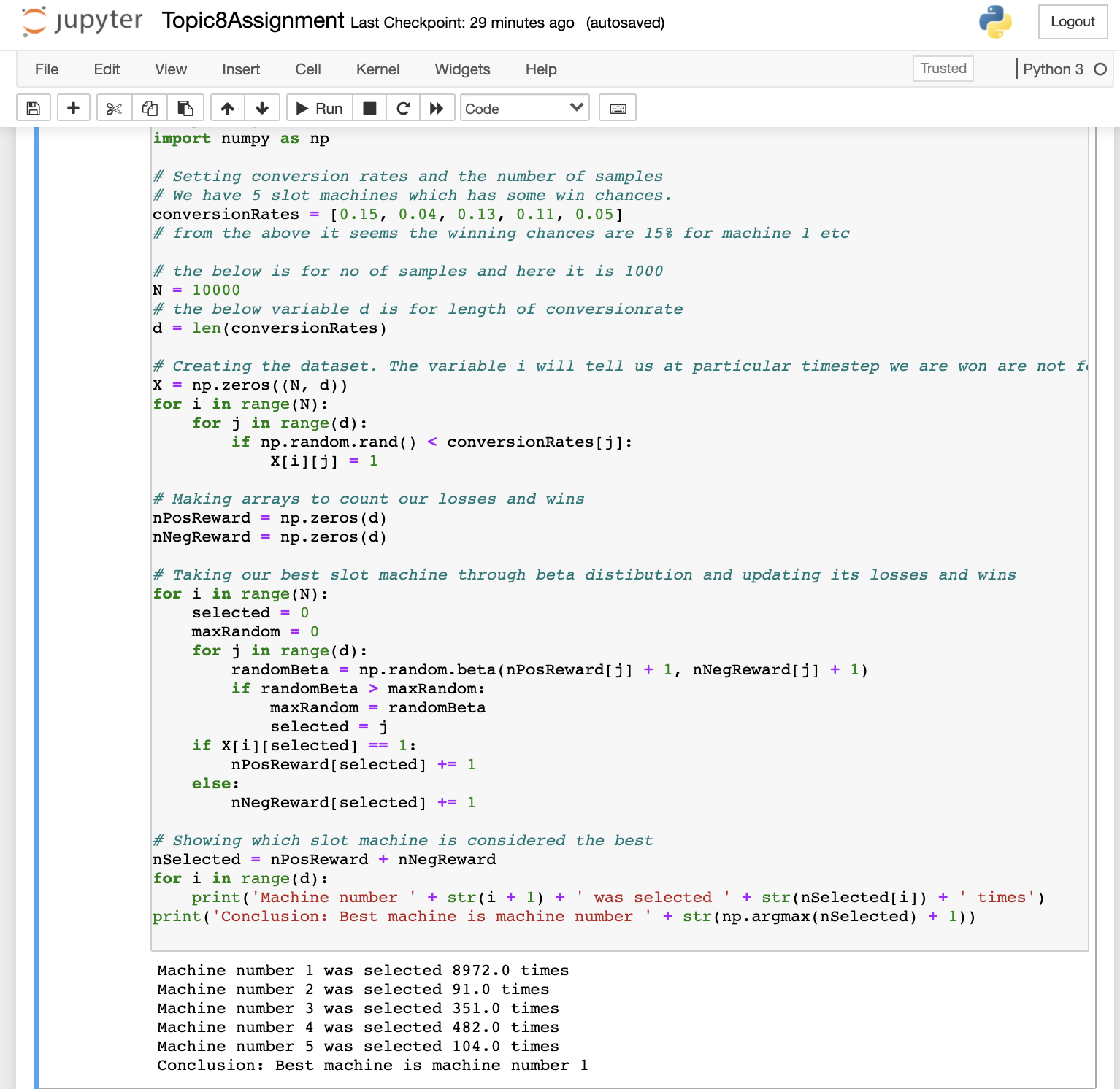
Let’s denote these varaibles as and , where the former is the no of times slot machine number I returned reward 0 up to round n and the later is the no of times slot machine number I returned reward 1 upto round n. These variables are denoted by nNegReward and nPosReward in the code. So based on values at round 5, let us try some values example for these variables

= 1 means that slot machine 1 has returned 1 loss over 1 round

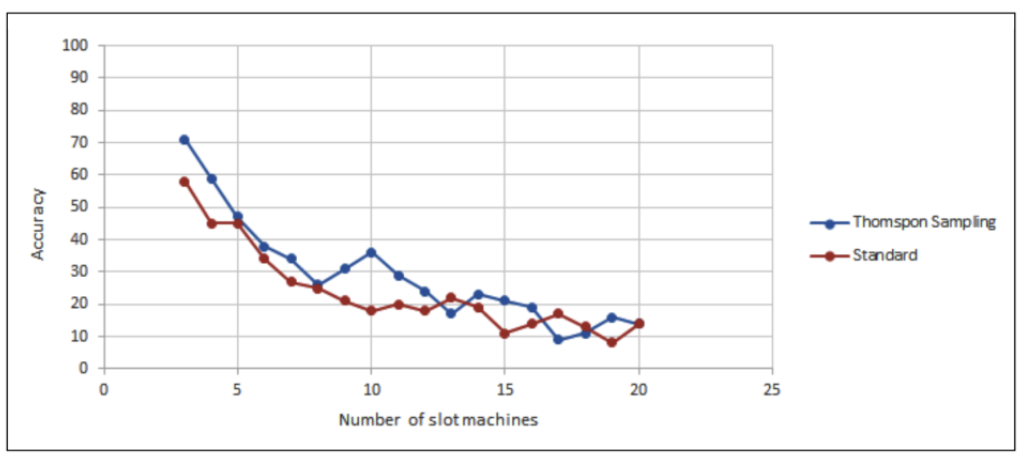
= 0 means that slot machine 1 has returned 0 wins over 1 round

Etc….

Code execution Screenshot :

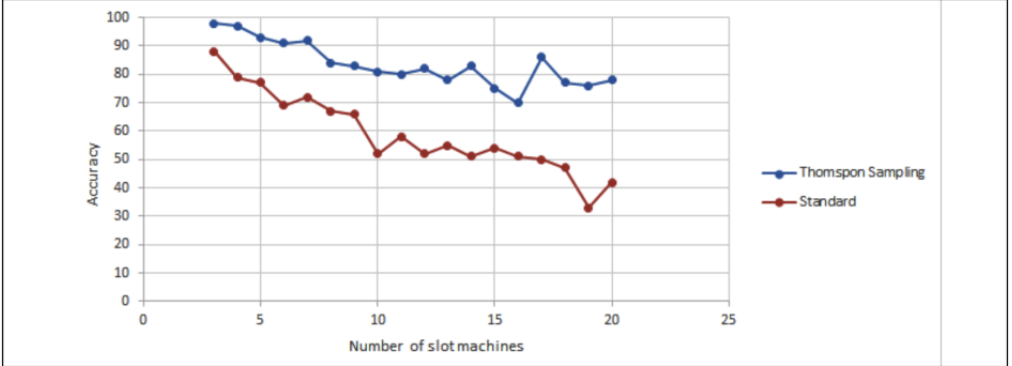


Accuracy (Vs) No of slot machines (200 samples) graphical output



The above graph illustrates the accuracy depending on the no of slot machines. The number of samples was set to 200 and the conversion rate ranges were set to 0-0.1. Overall this sampling performed better than standard model(22% better)

Accuracy (vs) no of slot machines (5000) samples



The above graph shows the performance for 5000 samples and the conversion rate ranges were set to 0-0.5 and it achieved the mean accuracy of 57%

Comparision .py execution output (Taken while executing):

Table

Description automatically generated with low confidence

Table

Description automatically generated with low confidence

**Results.xlsx**

