Original Code:

1. Train - learnRate = 0.5
2. Train - learnRate = 0.5
   1. Implement: did not succeed. Immediately did a big loop then shot off to the left
3. Train - learnRate = 0.5
4. Train - learnRate = 0.3
   1. Implement: Did not succeed. Made it ⅔ of the way down then veered off to the left
5. Train - learnRate = 0.3
6. Train - learnRate = 0.3
   1. Implement: Success!
7. Train - learnRate = 0.1
8. Train - learnRate = 0.1
   1. Implement: Success!
9. Train - learnRate = 0.1
10. Train - learnRate = 0.1
    1. Implement:Success!
11. Train - learnRate = 0.1
12. Train - learnRate = 0.1
    1. Implement: Success!
13. Train - learnRate = 0.1
14. Train - learnRate = 0.1
    1. Implement: Success! But wow was it offset to the right
15. Train - learnRate = 0.1
16. Train - learnRate = 0.1
    1. Implement: Success! But even worse offset

Original Code:

1. Train - learnRate = 0.5
2. Train - learnRate = 0.5
   1. Implement: Did not succeed. Made it 10% of the way then cut hard right
3. Train - learnRate = 0.5
4. Train - learnRate = 0.3
   1. Implement: Did not succeed. Same as last one
5. Train - learnRate = 0.3
6. Train - learnRate = 0.3
   1. Implement: Did not succeed. Went 30% of the way then veered off right
7. Train - learnRate = 0.1
8. Train - learnRate = 0.1
9. Train - learnRate = 0.1
   1. Implement: Did not succeed. Went 10% of the way then veered off right same as last time
10. Train - learnRate = 0.1
11. Train - learnRate = 0.1
    1. Implement:Did not succeed. Went 20% of the way then veered off right same as last time
12. Train - learnRate = 0.5 <- Increased to get rid of a bad value
13. Train - learnRate = 0.5
14. Train - learnRate = 0.5
    1. Implement: Made it to the end! Stayed offset to the left for a while then swerved a
15. Train - learnRate = 0.1
16. Train - learnRate = 0.1
    1. Implement:

Ways to improve the learning rate:

* Keep track of number of times each action has been learned. Take the action with the least learning so far
  + Need new .txt file for that
* Calculate a learning rate function with the input of number of times we have learned
  + We want a ratio of previous noise to new noise
  + Possible range:
    - Min: 0.5, decrease rapidly, can’t converge if lower than this
    - Max: 1, Never decrease
  + Current value averages: 0.91
  + What factors determine what works best for that?
    - Low number makes it take forever to fix errors
    - High numbers take longer to converge
    - We need higher values to get rid of the influence of starting at zero
    - We need lower values to converge faster
* learnRate -> continuous scale
  + The rate is the change of doing the exploration instead of the optimal
* Decrease bin & action sizes
* Add discount

New Code take 1: Learning Rate Ratio 0.75

1. Train
2. Train
   1. Implement - failure - steered to the right, most likely due to it being the first implementation
3. Train
4. Train
   1. Implement - success reward:-109 Iterations:86
5. Train
6. Train
   1. Implement - success reward: -186.5 Iterations:87
7. Train
8. Train
   1. Implement - failure - circled left, no apparent reason
9. Train
10. Train
    1. Implement - failure - did the first half fine then spiraled off
11. Train
12. Train
    1. Implement - Success! CR: -99.5, 85 iterations
13. Train
14. Train
    1. Implement - Success! CR: -131.85, 85
15. Train
16. Train
    1. Implement - Success! CR: -124, 86 iterations

New Code Take 2: 0.75 learn rate ratio

1. Train
2. Train
   1. Implement - Failed, went really straight for the first part though!
3. Train
4. Train
   1. Implement - Failed, went really straight for the first part again then donuts
5. Train
6. Train
   1. Implement - Success! - CR: -329, iterations 88
7. Train
8. Train
   1. Implement - Success - CR: -289, Iterations 88
9. Train
10. Train
    1. Implement - Failed