DUCSTeach Workshop 08 – Machine Learning

Written by: Matthew Horger

mh3294@drexel.edu

2017-2018

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Time:** 60 Minutes

**People**: 10 - 15 People

**Materials:**

● 15 Arduino Unos with 170 pin breadboard

● 15 LM35 Temperature Sensors

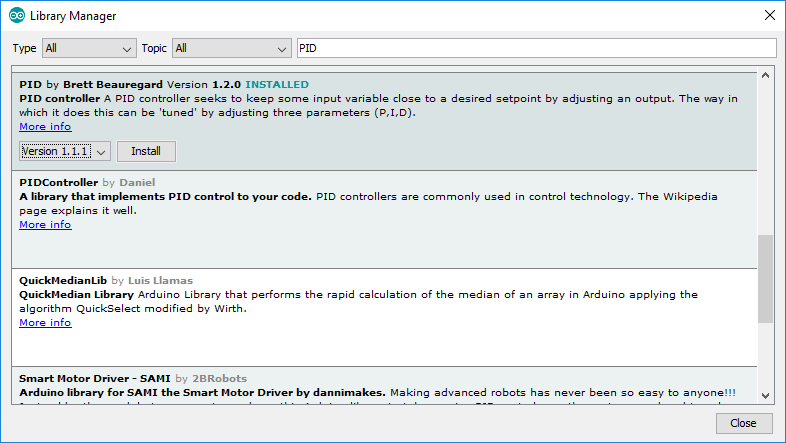
● USB Type B Cable

● Laptop with Arduino IDE installed

● PID.ino file

**Steps:**

1. Boot into Arduino IDE, click on **Tools** from the top navigation bar, and click on **Manage Libraries.**
2. Type PID in the filter search bar. Download the appropriate library, PID by Brett Beauregard. Please make sure it is the latest version.



1. Select close. If everything goes smoothly, we are now ready to program in our environment with this powerful library as a tool.

**But first? What’s a PID?**

From Wikipedia: "A PID controller calculates an 'error' value as the difference between a measured [Input] and a desired setpoint. The controller attempts to minimize the error by adjusting [an Output]."

So, you tell the PID what to measure (the "Input",) Where you want that measurement to be (the "Setpoint",) and the variable to adjust that can make that happen (the "Output".) The PID then adjusts the output trying to make the input equal the setpoint.

For reference, in a car, the Input, Setpoint, and Output would be the speed, desired speed, and gas pedal angle respectively.

In short, this is a simple machine learning tool for the Arduino to try and process large sets of data and adjust them.

**Back to coding!**

1. Follow each line by line of PID.ino to complete the program. There are descriptions and details for which each line does.