Lincoln Isaque
Machine Learning

Assignment 1

First, I made the antecedents, ‘temperature’ and ‘wind’ and the consequent, ‘melt’.

I specified the boundaries for temperature, wind and melt. I put (0,51,01) for temperature because the temperature reaches up to 50 degrees and increments by 1. I set wind as (0,5,1) because the wind reaches and surpasses 3 m/s, so I made it so the graph shows up until 4 m/s wind. For melt, I made the boundary (0, 1.5,0.5) because the melt is from 0, 0.5 and 1, for ‘no melt’, ‘partially melted’ and ‘fully melted’, respectively.

Next, I set the number of functions for temperature and wind, temperature.automf(5) & wind.automf(3).

Then I created the triangular membership functions for ‘melt’. ‘no melt’ is set as [0, 0, 0.4], ‘partially melted’ as [0,0.5,1] and ‘fully melted’ as [0.6,1 ,1]. This creates the triangular functions for the 3 melt states.

Next is the rules. I made 15 rules in total. Rule1 -Rule5 has the 5 temperature states and low wind, and each respective ‘melt’ consequences.

Rule6 -Rule10 has all the 5 temperature states and average wind and the respective consequences

Rule11-Rule15 has all the 5 temperature states and high wind and the respective consequences.

This makes it so that each input is taken into account to produce the most accurate and specific representation of the algorithm possible.

For example, on rule3 is:

rule3 = ctrl.Rule(temperature['average'] & wind['poor'], melt['no melt']) # 12-25

and rule13 is:

rule13 = ctrl.Rule(temperature['average'] & wind['good'], melt['partially melted']) # 12-25

because the wind makes a difference despite the temperature being the same.

Next, I made the control system with all the rules and I made the input functions as well.

In addition, I also added the code for all the graphs of, ‘melt’, ‘wind’ and ‘temperature’, as well as the graph showcasing the output of the ‘melt’ state.

I tested the algorithm with different inputs and I determined that this is the best program I could do, rather than one with fewer rules.

```
#THIS RULE SET IS FOR ALL TEMPERATURES WHEN THERES LOW ('poor) WIND

rule1 = ctrl.Rule(temperature['poor'] & wind['poor'], melt['no melt']) #low wind, low temp
rule2 = ctrl.Rule(temperature['mediocre'] & wind['poor'], melt['no melt'])
rule3 = ctrl.Rule(temperature['average'] & wind['poor'], melt['no melt']) # 12-25
rule4 = ctrl.Rule(temperature['decent'] & wind['poor'], melt['partially melted']) # 23-40
rule5 = ctrl.Rule(temperature['good'] & wind['poor'], melt['fully melted']) # 38-50

#THIS RULE SET IS FOR ALL TEMPERATURES WHEN THERES AVERAGE WIND

rule6 = ctrl.Rule(temperature['poor'] & wind['average'], melt['no melt']) #low wind, low temp
rule7 = ctrl.Rule(temperature['mediocre'] & wind['average'], melt['no melt'])
rule8 = ctrl.Rule(temperature['average'] & wind['average'], melt['no melt']) # 12-25
rule9 = ctrl.Rule(temperature['decent'] & wind['average'], melt['partially melted']) # 23-40
rule10 = ctrl.Rule(temperature['good'] & wind['average'], melt['fully melted']) # 38-50

#THIS RULE SET IS FOR ALL TEMPERATURES WHEN THERES HIGH ('good) WIND

rule11 = ctrl.Rule(temperature['poor'] & wind['good'], melt['no melt']) #low wind, low temp
rule12 = ctrl.Rule(temperature['mediocre'] & wind['good'], melt['no melt'])
rule13 = ctrl.Rule(temperature['average'] & wind['good'], melt['partially melted']) # 12-25
rule14 = ctrl.Rule(temperature['decent'] & wind['good'], melt['fully melted']) # 23-40
rule15 = ctrl.Rule(temperature['good'] & wind['good'], melt['fully melted']) # 38-50
```

I noticed with my algorithm, that when there’s more wind, the ice cream is more melted.

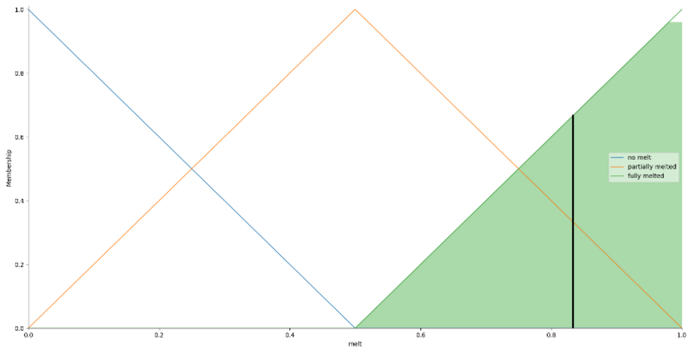
Input test 1: 38 temperature, 6 m/s wind

```
melting_ctrl = ctrl.ControlSystem([rule1, rule2, rule3,
melting = ctrl.ControlSystemSimulation(melting_ctrl)

melting.input['temperature'] = 38
melting.input['wind'] = 6
```

Input test 2: 38 temperature, 0 m/s wind

```
melting_ctrl = ctrl.ControlSystem([rule1, rule2, rule3, rule4,r
melting = ctrl.ControlSystemSimulation(melting_ctrl)

melting.input['temperature'] = 38
melting.input['wind'] = 0
```

Result 1 graph



Result 2 graph