Intelligent System Assignment 1 (Group 6)

Questions:

1) A characteristic function has only binary values (zero and one) while a membership function has values between zero and one. The membership function describes the degree of affiliation of the input to a rule, while the characteristic function only states the belonging or absent of the input to a rule.
2)
   a) Yes, they are, all membership values are between zero and one. The Value of µ_A of the edge cases (x= -8, 8, 0) are: µ_A(8)=µ_A(-8)= 1/9 ; µ_A(0)=1.
   The same applies form µ_b. The Values of the Input x=8, -8, 0 for µ_b are: µ_b(8) = µ_b(-8)= 3/5 and µ_b(0)=1
   b)

$$\mu_{a,\alpha_{cut}}(x) = \frac{1}{1+|x|} \geq 0{,}3$$

$$\frac{7}{3} \geq |x|$$

$$-\frac{7}{3} \leq x \leq \frac{7}{3}$$

$$\mu_{b,\alpha_{cut}}(x) = 1 - \frac{|x|}{20} \geq 0{,}3$$

$$14 \geq |x|$$

$$-14 \leq x \leq 14$$

Fuzzy Model:

Problem 2) Report G06_A1

1) The first Step is to Import the libraries pyfume, sklearn and numpy.
2) Secondly, we load the given wine data which includes 3 Targets and 13 features, while normalizing the input data with a min-max normalization method. The names of the 13 features are also defined in this step as variable names.
3) Next, we split the x and y data into test and training data. We do this by using the hold-out method which is integrated into the DataSplitter() function from the pyfume library. By default, the data is split into 75% training data and 25% test data.
4) After this we cluster the training data (in input-output space) into 10 clusters. To accomplish this we use the Clusterer() function from the pyfume library, which uses FCM by default. The created clusterer object computes the cluster centers and the partition matrix, which we will need later.
5) The next step is to estimate the membership functions of the system. We do so by using the AntecedentEstimator() function from the pyfume library, which needs the x-training-data and the partition matrix as an input. By default, pyfume fits Guassian membership functions to the data. Through the membership functions we obtain the antecedent parameters.

6) Now we estimate the consequent parameters by using the ConsequentEstimator() function from the pyfume library through inputting the x- and y-training-data and the partition matrix. By default, global fitting of the consequent parameters is true.
7) After these steps we can now build the first order Takagi-Sugeno model through the SugenoFISBuilder() function from the pyfume library. The SugenoFISBuilder function needs the antecedent parameters, consequent parameters, and the variable names to build the first order Takagi-Sugeno model.
8) Finally, through the SugenoFISTester function from the pyfume library, we can now test our first order Takagi-Sugeno model and create a tester object. The Tester function needs the first order Takagi-Sugeno model, the x- and y-test-data and the variable names as an input. Through the tester object we compute the mean squared error and we predict the y-values. The predicted y-values are rounded to full integer values, and we take the absolute predicted y-values, so that they can be compared to the y-test-data.
9) Lastly, we compare our y-value-predictions with the test-y-data. We receive an accuracy value to evaluate our model-accuracy.
    a. Currently our accuracy values are between 0,88 and 0,98. This means our model is pretty accurate and can ensure pretty good predictions, although the variation in the prediction accuracy is relatively large. To improve the quality of the predictions we could optimize clustering (by changing the amount of clusters and by changing the Cluster method e.g. to GK or a Grid), change the used membership functions fitted to the data (e.g. double gaussian or sigmoid), normalize the input data through a different normalization method or try not using global optimization of consequent parameters.

Group 6:

Noah Ellenbogen

Maximilian Giebel