Project #2: Fuzzy Decision Trees

Sruthikesh Surineni 10/27/2016 Instructor name: Erik Stone

Introduction

Crip sets and probability statistics have been the used for combined multiple features together. Recently use of Fuzzy logics had become popular because of the principle of least commitment and membership degree. Using soft membership will increase the flexibility to make a decision in later stages. In this project we implemented fuzzy decision tree for the automobile dataset and find the best car considering the given factors. Fuzzy decision tree is implemented using multiple membership functions and various operators.

Background

Introduction and advantages of fuzzy control systems

A fuzzy set is represented by a membership function defined on the universe of discourse. The membership function gives the degree, of membership within the set, of any element of the universe of discourse. The membership function maps the elements of the universe onto numerical values in the interval [0, 1]. A membership function value of zero implies that the corresponding element is definitely not an element of the fuzzy set, while a value of unity means that the element fully belongs to the set. A grade of membership in between corresponds to the fuzzy membership to set. We will look at the some of the membership functions and fuzzy operators below.

Membership functions

Membership functions are the input data mapping onto [0, 1] range. Below are 2 of the most used membership functions.

1. Triangular: This is a membership function in which only one of the value have a unity and all others are between [0,1). To represent this membership function we require three input values (a, b, c).

2. Trapezoidal: This is a membership function which allows multiple input values to be mapped to unity. To represent this we require four input values (a, b, c, d).

Below shown is the triangular and trapezoidal membership function.

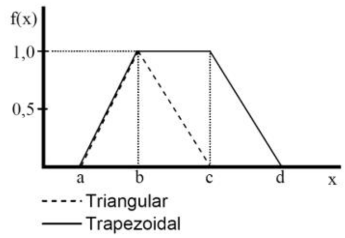


Figure: Triangular and Trapezoidal membership function mapping

Fuzzy operators

Fuzzy operations are the very basic operations AND, OR and NOT. Since the operations are not crisp sets we have various operators defined. Below are some of the operators we implemented in this project.

Zadeh operators: Min (T-norm) and Max (S-norm)

Complement operator: Below equation is used for NOT or complement of input membership.

$$A^c(x) = 1 - A(x)$$

Union and intersection: Equations below are the Zadeh OR and AND operators for 2 input memberships.

$$(A \cup B)(x) = \max\{A(x), B(x)\} = A(x) \vee B(x)$$

$$(A \cap B)(x) = \min\{A(x), B(x)\} = A(x) \wedge B(x)$$

Law of Contradiction and Law of excluded middle don't hold true for zadeh operators.

Below shown images represent

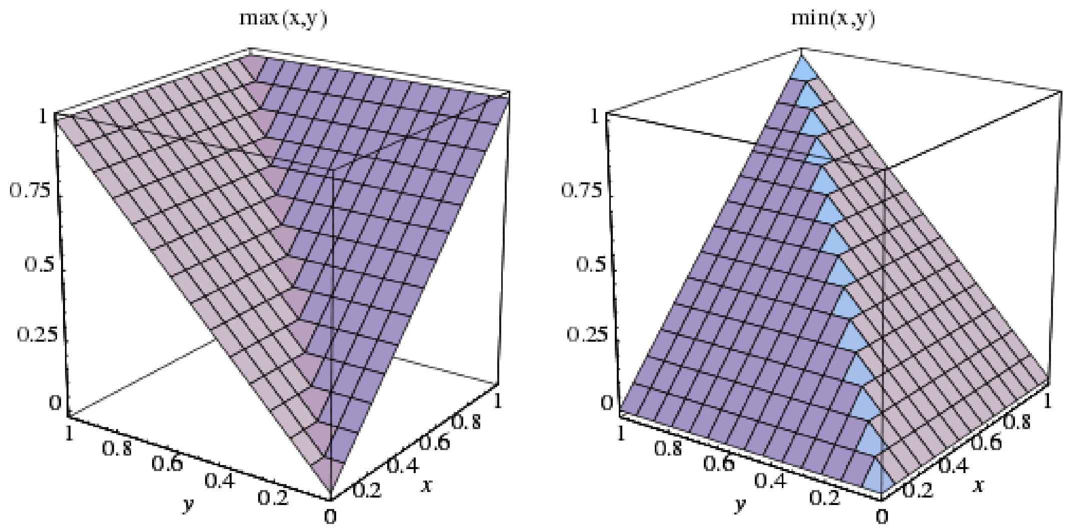


Figure: Max and Min operators

Bounded sum and Bounded difference

Equations for union and intersection are shown below.

$$(A \cup_b B)(x) = 1 \wedge (A(x) + B(x))$$

$$(A \cap_b B)(x) = 0 \vee (1 - (A(x) + B(x)))$$

Complement operator is same as Zadeh complement

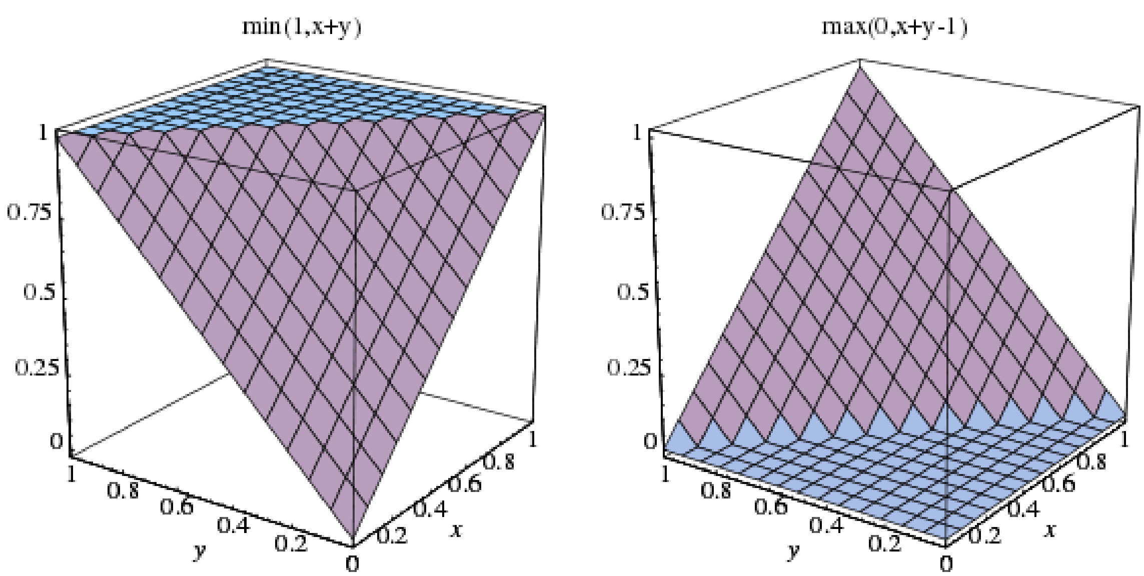


Figure: Bounded sum and Bounded difference operators

These operators have the valid 4 axioms (boundary conditions, associativity, commutativity, and monotonicity). But the bounded sum does not satisfy the idempotence and distributive properties.

Yager family of fuzzy operators

Union and Intersection operator:

$$(A \cup_w B)(x) = \min\{1, (A(x)^w + B(x)^w)^{1/w}\}, \quad w \in (0, \infty),$$

$$(A \cap_w B)(x) = 1 - \min\{1, ((1 - A(x))^w + (1 - B(x))^w)^{1/w}\}, \quad w \in (0, \infty)$$

Complement operator:

$$A^c(x) = (1 - A(x)^w)^{1/w}, \quad w \in (0, \infty)$$

All the 5 axioms are fully valid for these Yager operators.

Dataset

In this project we consider the UCI Automobile dataset. This automobile consists of 7 attributes which help in deciding to buy a best car based on the requirements. The 7 attributes are Car ID, Risk, Value Loss, Horse Power, City MPG, Highway MPG, and Price. Consider all these characteristics we will design a decision tree which gives us the best car to buy.

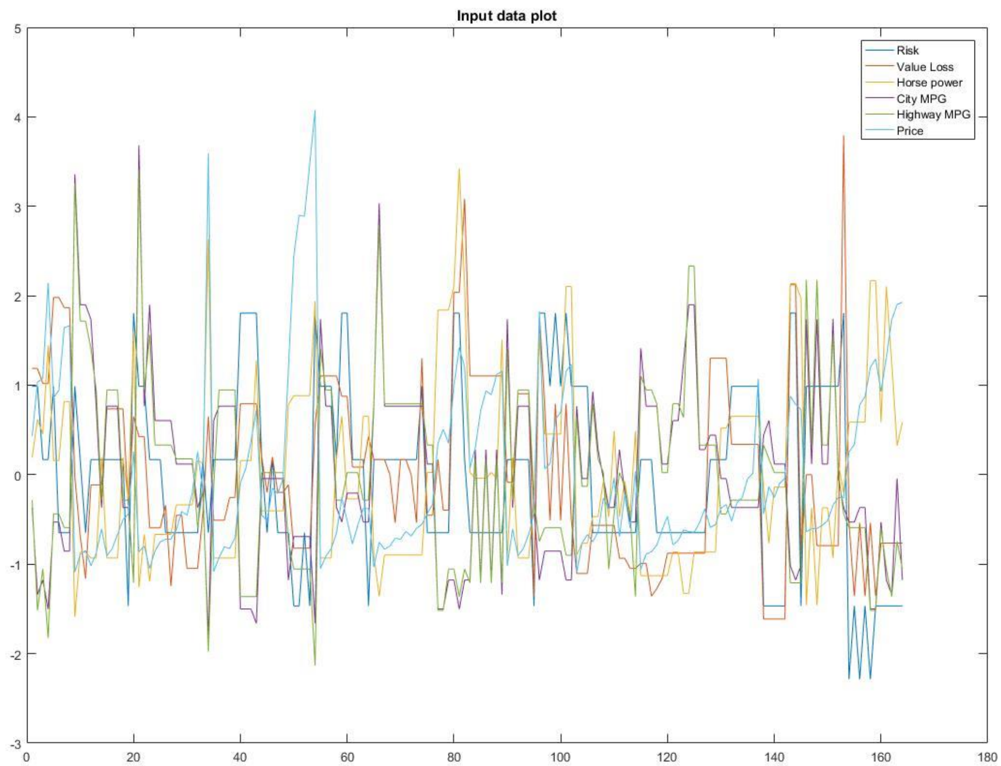


Figure. Normalized input data sets

From the above figure and histograms of each attributes we updated the membership functions. We use the histogram to decide the membership values so that we have a good rating as desired from the membership function.

Membership models:

Below shown in tables are the membership functions we used for this project. There are 2 membership functions the first is the default provided member functions and the second is the one designed for the decision tree 2. The 2nd membership function is created considering the histograms of the input data attributes.

Membership 1

Risk Low = Trap(x; -3, -3, -2, 0) Average = Tri(x; -2, 0, 2) High = Trap(x; 0, 2, 3, 3)	City MPG Poor = Trap(x; 0, 0, 20, 30) Average = Tri(x; 20, 30, 40) Good = Trap(x; 30, 40, 60, 60)
Value Loss Low = Trap(x; 0, 0, 100, 120) Average = Tri(x; 100, 120, 200) High = Trap(x; 120, 200, 300, 300)	Highway MPG Poor = Trap(x; 0, 0, 20, 30) Average = Tri(x; 20, 30, 40) Good = Trap(x; 30, 40, 60, 60)
Horsepower Low = Trap(x; 0, 0, 60, 100) Average = Tri(x; 60, 100, 140) High = Trap(x; 100, 140, 250, 250)	Price Cheap = Trap(x; 0, 0, 7000, 10000) Average = Tri(x; 7000, 10000, 20000) Expensive = Trap(x; 10000, 20000, 40000, 40000)

Membership 2

Shown in the below table is the updated membership function, the bolded text is the one which is updated for the decision tree 2.

risk.Low = trap, [-3 -3 -2 0];
risk.Average = tri, [-2 0 2];
risk.High = trap, [-3 1 3 3];
	cityMPG.Poor = trap, [0 0 20 30];
	cityMPG.Average = tri, [25 35 40];
	cityMPG.Good = trap, [30 40 60 60];

valueLoss.Low = trap, [0 60 100 140];
valueLoss.Average = tri, [100 120 200];
valueLoss.High = trap, [120 200 300 300];
	highwayMPG.Poor = trap, [0 0 20 30];
	highwayMPG.Average = tri, [20 30 40];
	highwayMPG.Good = trap, [30 45 60 60];

horsepower.Low = trap, [0 0 60 100];
horsepower.Average = tri, [50 85 125];
horsepower.Low = trap, [100 140 250 250];
	price.Cheap = trap, [0 0 7000 10000];
	price.Average = trap, [7000 10000 40000 40000];
	price.Expensive = trap, [10000 20000 40000 40000];

Fuzzy decision trees

Below are the 2 fuzzy decision trees which are implemented. First one is the default and second tree is the custom designed.

Tree 1

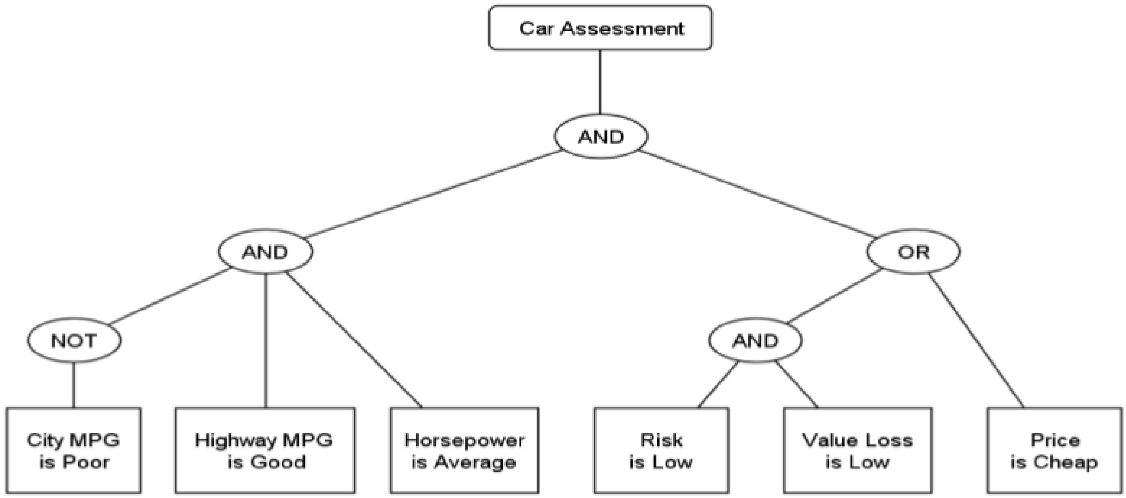


Figure: Fuzzy Decision Tree 1

Tree 2

This is the custom built tree designed considering my considerations for the car. To buy a car the important priority I would consider is to have good mileage, Low horse power because mileage is dependent on horse power, Low risk and value loss, and finally low price. Considering all these we built a graph. We OR the average city and good highway MPG to get a car which can either be good at city or highway MPG and we and this average horse power. This is because high horse power will result in low MPG. On the other end we want low value loss and we don't want a high risk car and finally the price should not be high.

We modified the membership functions to appropriately fit the requirements looking at the histograms of the corresponding input attributes.

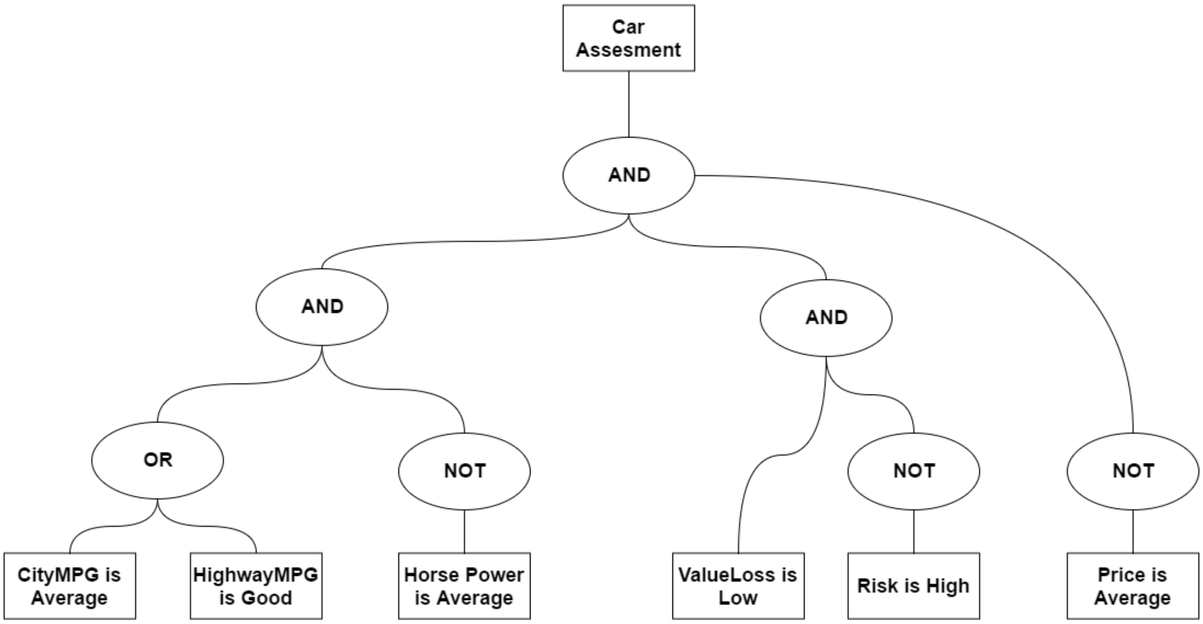


Figure. Fuzzy decision tree 2

Experiments and Results

Part A: In this section we run find the best car using decision tree 1, membership 1 and various operators we implemented. Below are the experiments changing the operators.

1. Zadeh operators: Here we use the Zadeh min & max operators in the decision tree.

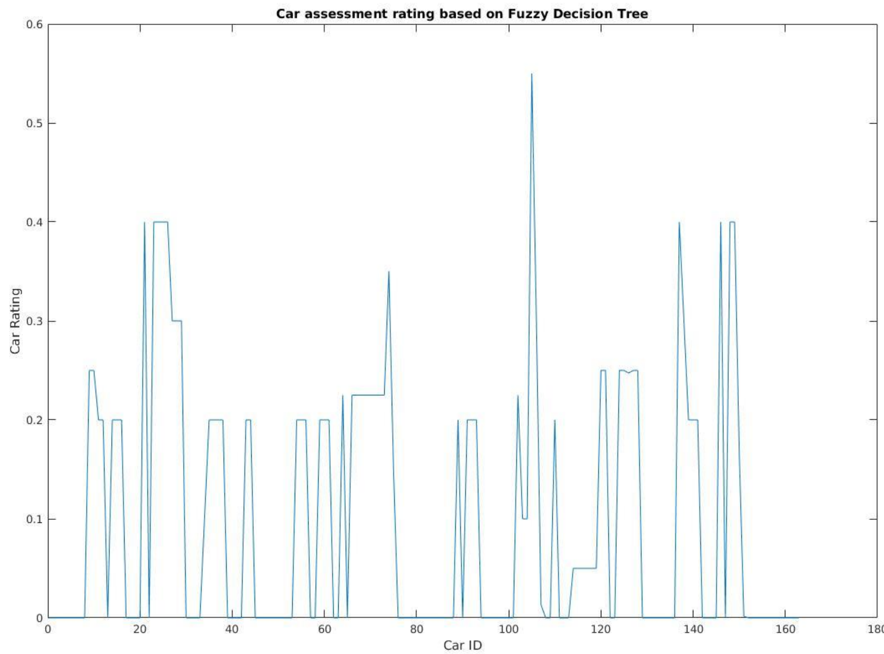


Figure: Fuzzy Decision Tree output car ratings

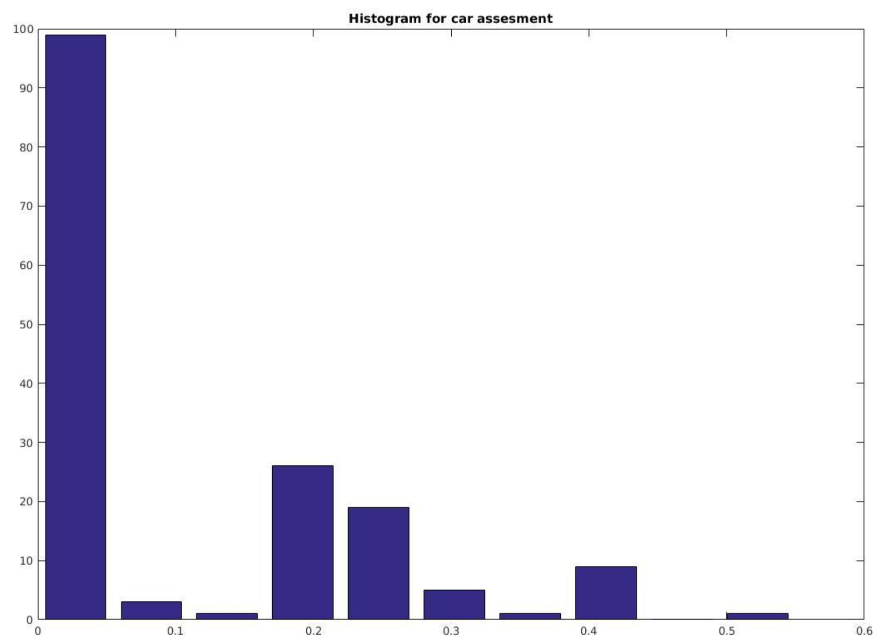


Figure: Car assessment Histogram

Bin	Bin Count	Bin Center
1	99	0.0275

2	3	0.0825
3	1	0.1375
4	26	0.1925
5	19	0.2475
6	5	0.3025
7	1	0.3575
8	9	0.4125
9	0	0.4675
10	1	0.5225

Table: Histogram Bin counts with centers

Best Car using the Zadeh operators and decision tree1 is 105 and its value is 0.55. If we look at Zadeh outputs since it is just min and max most of the cars have low rating. Only 1 car has a rating greater than 0.5 which is not interesting.

2. Bounded Sum and Bounded Difference: We use the bounded operators and rate the cars.

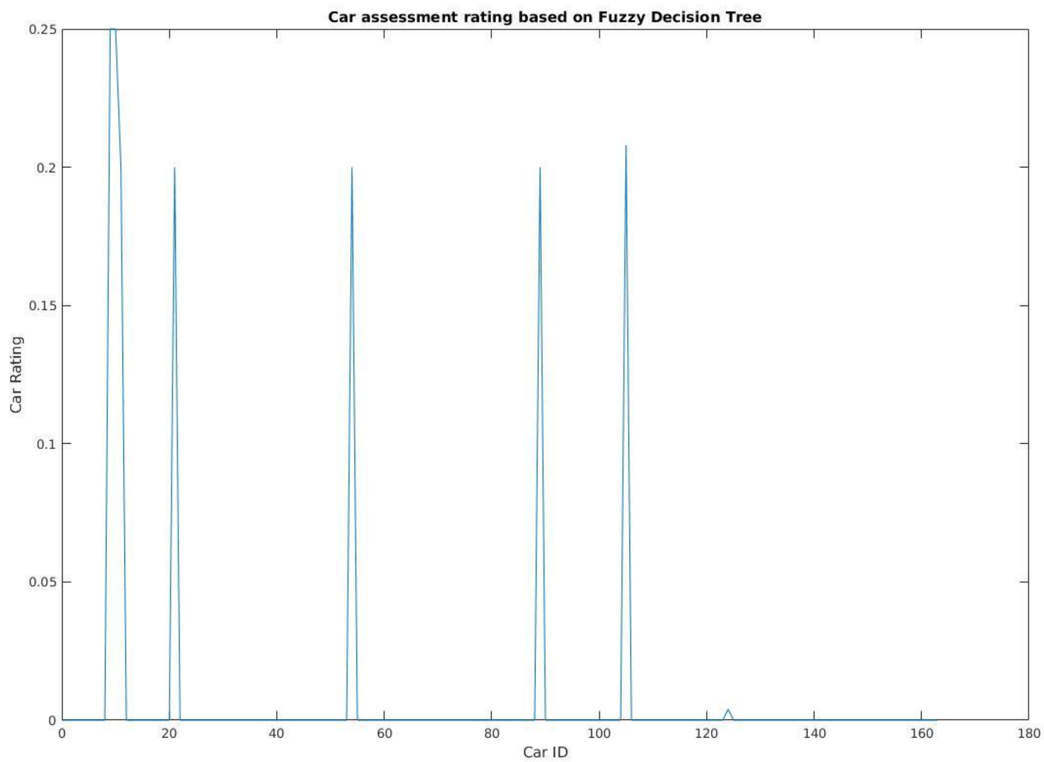


Figure. Car assessment considering Bounded Sum and Bounded Difference

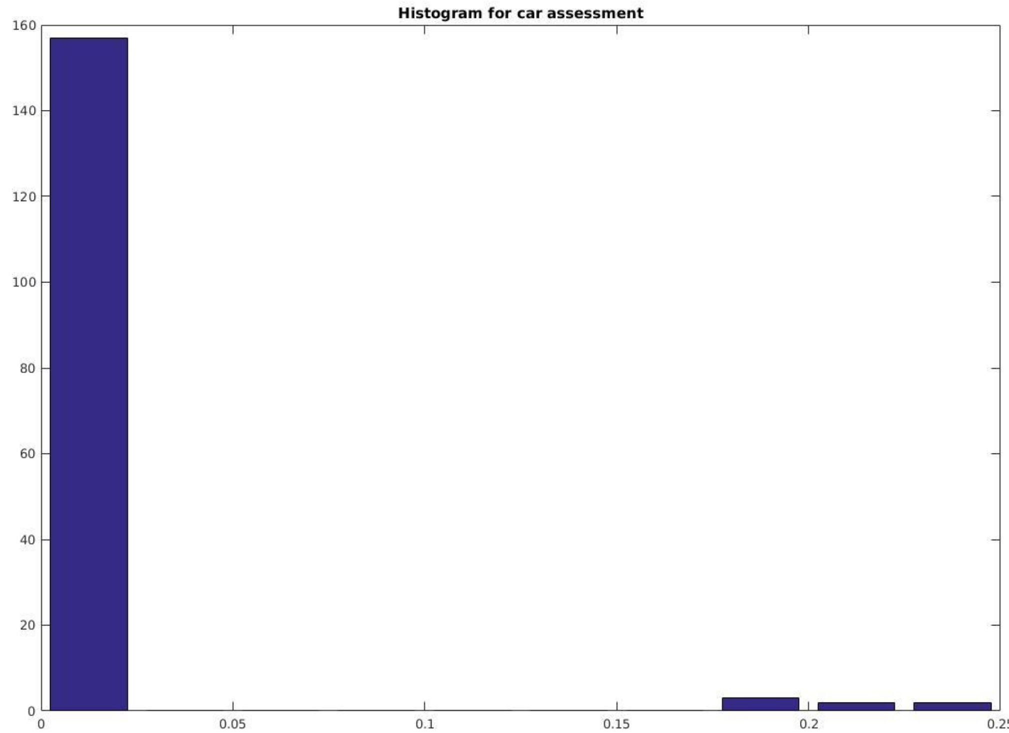
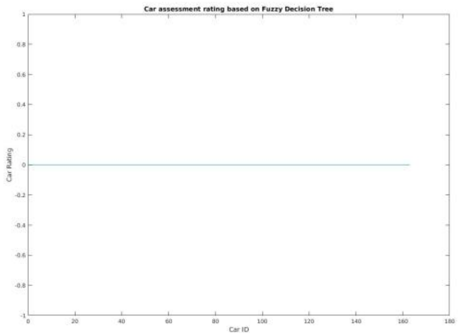


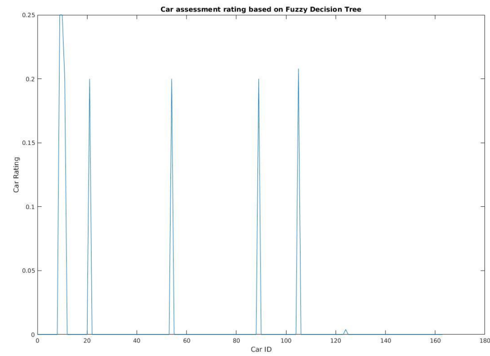
Figure. Histogram of Car Assessment outputs

Best Car with bounded operators is car ID 9 and its value is 0.25. Even with Zadeh the output is 0.25 for car ID 9, but since the other cars have more rating 9 was not the best car. Using the bounded operators is even not encouraging because most of the cars have very low rating and only few cars have non-zero rating.

3. Yager operators with Membership 1 functions: We use the Yager family of fuzzy operators to estimate the ratings for the cars. We vary the values of w as shown in below images. For 0.1 w all the ratings to cars are zero and with W=1 the output is just the bounded sum output. As we increase the values of w we have higher ratings for the cars. As we keep increasing the range of outputs are close to unity, but the output rating smooth out and there isn't a lot of variations we can see this from the below images. Which means that as we increase the w small variations are removed from the output ratings. Having a higher rating on the would be beneficial for having higher confidence.

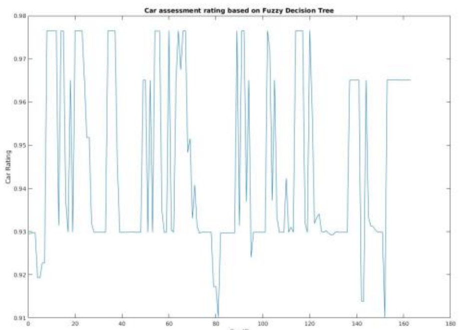
W=0.1 W=1

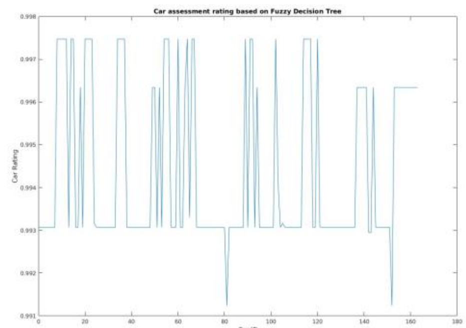




W=10

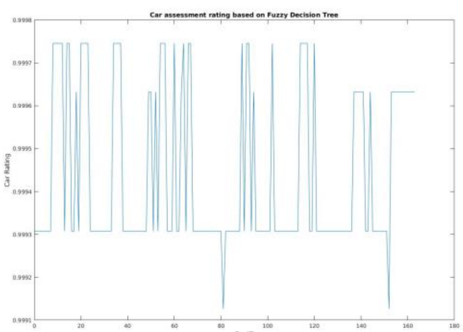
W=100

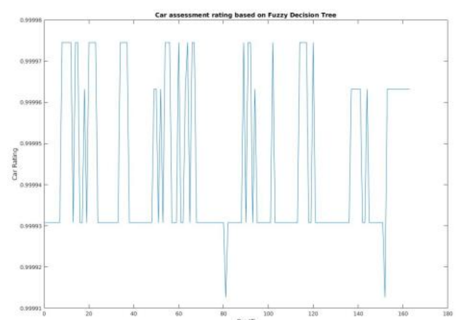




W=1000

W=10000





W	Best Car ID	Rating	Other Car ID	Rating
0.1	NA	0	NA	0
1	9	0.25	8	0
10	9	0.97651	8	0.9765
100	8	0.99747	9	0.99747
1000	8	0.99975	9	0.99975
10000	8	0.99997	9	0.99997

Table: Best car to choose with increasing W for Yager operator

As we increase the w the ratings for the cars we observe that most of the cars have same rating which is not beneficial in decision. As shown above in the table above we have 8 & 9 having the same rating since 8 is the first sample the max function in MATLAB returns the first index.

Decision Tree 2: Membership 2

1. Zadeh operators: Below shown are the outputs from decision tree using decision tree 2, membership function 2 and zadeh operators. Even though the total rating the car is reduced we have more cars having peak ratings which will give more choices.

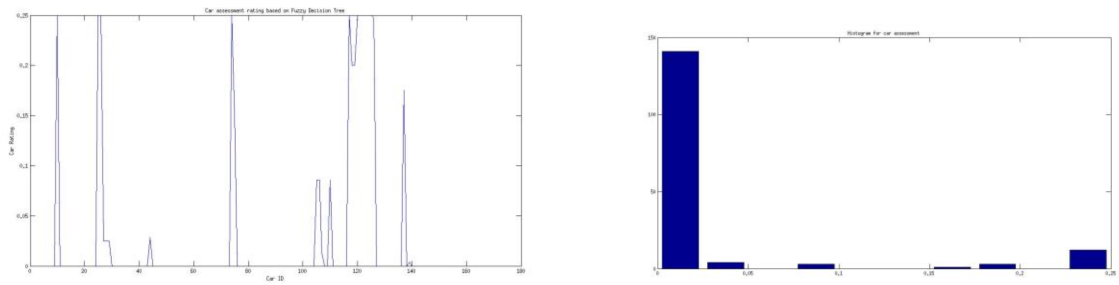


Figure. Car assessment output and Histogram of Car assessment using Zadeh operators

Best Car with decision tree 2, membership 2 and zadeh operators is the car with CarID 10 and rating for the car is 0.25.

2. Yager operators: Below are the results from decision tree 2 with membership 2 and Yager family of operations. For w=1 the ratings are all 0, so the bounded sum and difference won't be of any use for us.

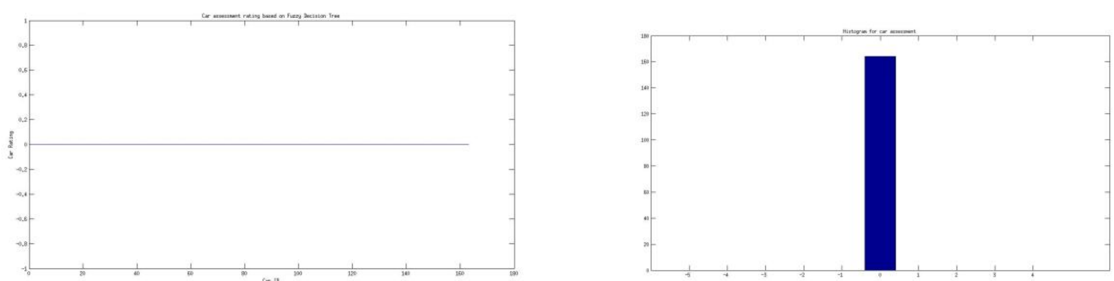


Figure. Car assessment output and Histogram for w=1

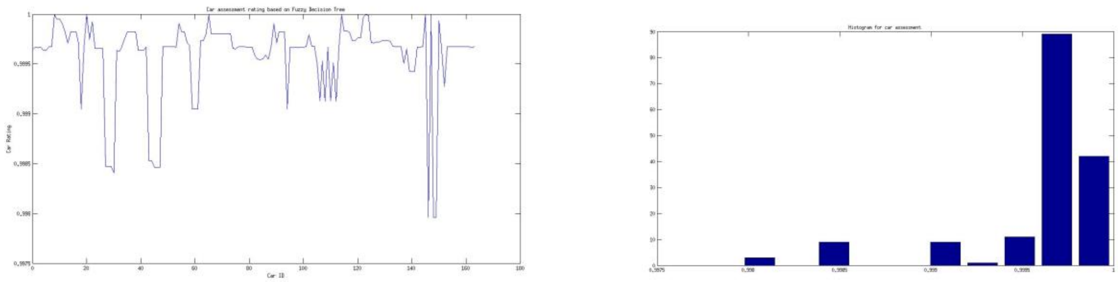


Figure. Car assessment output and Histogram for w=5

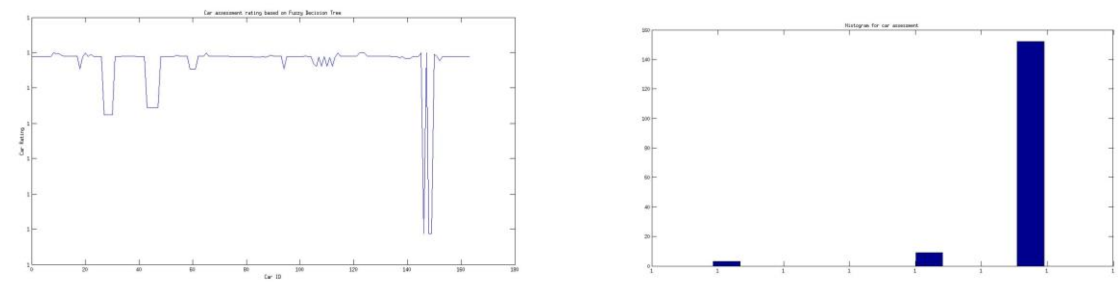


Figure. Car assessment output and Histogram for w=10

As we increase the w, we see that the outputs are close to unity. And the outputs have less variations with higher w.

For w =0, All the cars have zero rating.

For w=5, Car ID 145 has the highest rating 0.9999

For w=10, Car ID 145 has the highest rating 1

Discussion and Future work

From the experiments we conclude that having a Fuzzy logic system will help if we have to delay our decision for later stages. This is because the intermediate conclusion can have higher deterministic factors in later stages. So, delaying the decision and having the membership degree for each attribute will have help the decision making. Considering multiple decision trees with membership functions and various operators helped us with a good decision for rating the cars. Rather than having the crisp decision using soft decision was helpful in finding the best car for the requirement and the uncertainty is reduced in later stages.

As future work we could try implement a decision tree which combines various attributes multiple times. For example we could use Risk multiple times in the graph to give it a higher priority or combine 2 attributes. Also other things we could try is to implement more membership function like, Sigmoid, Gaussian etc., so that we have appropriate input mapping for the given dataset.

MATLAB Implementation code

Main function: Membership functions

```
function fdt(inFile, mf, op, decTree, w)
% Fuzzy decision trees

in = csvread(inFile);
% Input data has 7 variables
% Car ID, Risk, Value Loss, Horsepower, City MPG, Highway MPG, and Price


% Plot the input data for visualization
% Input data is zscore normalized to reduce the range of inputs
figure; plot(zscore(in(:,2:end), 1, 1));
title('Input data plot');
legend('Risk', 'Value Loss', 'Horse power', 'City MPG', 'Highway MPG', 'Price')


% Membership funtions for the input characteristics
if mf==1
    % Risk
    risk.Low = [-3 -3 -2 0];
    risk.Average = [-2 0 2];
    risk.High = [0 2 3 3];

    % Value Loss
    valueLoss.Low = [0 0 100 200];
    valueLoss.Average = [100 120 200];
    valueLoss.High = [120 200 300 300]; %#ok<*STRNU>

    % Horsepower
    horsepower.Low = [0 0 60 100];
    horsepower.Average = [60 100 140];
    horsepower.Low = [100 140 250 250];

    % City MPG
    cityMPG.Poor = [0 0 20 30];
    cityMPG.Average = [20 30 40];
    cityMPG.Good = [30 40 60 60];

    % Highway MPG
    highwayMPG.Poor = [0 0 20 30];
    highwayMPG.Average = [20 30 40];
    highwayMPG.Good = [30 40 60 60];
```

```
    % Price
    price.Cheap = [0 0 7000 10000];
    price.Average = [7000 10000 20000];
    price.Expensive = [10000 20000 40000 40000];
elseif mf==2    % 2nd membership function for 2nd decision tree
    % Risk
    risk.Low = [-3 -3 -2 0];
    risk.Average = [-2 0 2];
    risk.High = [-3 1 3 3];  % Modified

    % Value Loss
    valueLoss.Low = [0 60 100 140];  % Modified
    valueLoss.Average = [100 120 200];
    valueLoss.High = [120 200 300 300]; %#ok<*STRNU>

    % Horsepower
    horsepower.Low = [0 0 60 100];
    horsepower.Average = [50 85 125];  % Modified
    horsepower.Low = [100 140 250 250];

    % City MPG
    cityMPG.Poor = [0 0 20 30];
    cityMPG.Average = [25 35 40];   % Modified
    cityMPG.Good = [30 40 60 60];

    % Highway MPG
    highwayMPG.Poor = [0 0 20 30];
    highwayMPG.Average = [20 30 40];
    highwayMPG.Good = [30 45 60 60];    % Modified

    % Price
    price.Cheap = [0 0 7000 10000];
    price.Average = [7000 10000 40000 40000]; % Modified, membership is trapmf
    price.Expensive = [10000 20000 40000 40000];
end


% Decision Tree for evaluting the car for the membership function,
% operation and assumed decision tree
assesment = decisionTree(in, decTree, op, w, risk, valueLoss, horsepower, cityMPG,
highwayMPG, price);


[bestVal, bestIdx] = max(assesment, [], 1); % Finding the best Car
disp(['Best Car ID is ', num2str(in(bestIdx, 1)), ' and its value is ', num2str(bestVal)]);
close all;


% Plot output car ratings for each car
figure(1);
```

```
plot(in(:,1), assesment)
xlabel('Car ID')
ylabel('Car Rating')
title('Car assessment rating based on Fuzzy Decision Tree')


% Histogram plot all the car ratings
[counts, centers]=hist(assesment, 10);  % Histogram of output car ratings
figure(2);
bar(centers, counts)
title('Histogram for car assessment')

End
```

Membership function

```
% Membership function
%-------------------------------------
% mf is assumed to be of the form:
%    {[a b c d], val}
%-------------------------------------
function mv = eval_mf(mf, fn)


% 11 membership functions
% lines: traingular, trapezoidal
% TODO
% Smooth: gaussmf, gauss2mf, gbellmf
% Asymmetric functions: sigmf, dsigmf, psigmf
% Polynomial: zmf, pimf, smf


if strcmp(fn, 'trimf')
    % Triangular
    if mf{2} > mf{1}(3)
        mv = 0;
    elseif mf{2} > mf{1}(2)
        mv = (mf{1}(3) - mf{2}) / (mf{1}(3) - mf{1}(2));
    elseif mf{2} > mf{1}(1)
        mv = (mf{2} - mf{1}(1)) / (mf{1}(2) - mf{1}(1));
    else
        mv = 0;
    end
elseif strcmp(fn, 'trapmf')
    % Trapezoid
    if mf{2} > mf{1}(4)
        mv = 0;
    elseif mf{2} > mf{1}(3)
        mv = (mf{1}(4) - mf{2}) / (mf{1}(4) - mf{1}(3));
```

```
    elseif mf{2} > mf{1}(2)
        mv = 1;
    elseif mf{2} > mf{1}(1)
        mv = (mf{2} - mf{1}(1)) / (mf{1}(2) - mf{1}(1));
    else
        mv = 0;
    end
end
End
```

Fuzzy operators:

```
% Fuzzy operators
function out = fuzzyOp(in, op, model, w)


% Models
% zadeh, Bounded, and Yager


if strcmp(op, 'AND_F')    % Fuzzy intersection or conjuction
    if strcmp(model, 'zadeh')
        out = min(in, [], 2);
    elseif strcmp(model, 'bounded')
        out = max(0, sum(in, 2)-1);
    elseif strcmp(model, 'yager')
        out = 1 - min(1, sum((1-in).^w, 2).^1/w);
    end
elseif strcmp(op, 'OR_F')     % Fuzzy union
    if strcmp(model, 'zadeh') %#ok<*STCMP>
        out = max(in,[], 2);
    elseif strcmp(model, 'bounded')
        out = min(1, sum(in, 2));
    elseif strcmp(model, 'yager')
        out = min(1, sum(in.^w, 2).^1/w);
    end
elseif strcmp(op, 'NOT_F')    % Fuzzy complement
    if strcmp(model, 'zadeh')
        out = 1-in;
    elseif strcmp(model, 'bounded')
        out = 1-in;
    elseif strcmp(model, 'yager')
        out = (1-in.^w).^1/w;
    end
end

end
```

Decision Tree:

```
% Decision Tree for the given input
function assesment = decisionTree(in, decisionTree, opModel, w, risk, valueLoss, horsepower, cityMPG, highwayMPG,
price)

% Number of inputs
N = size(in, 1);
assesment = zeros(N,1);


for i=1:N % Loop for all the input data for the decision tree
    if decisionTree==1  % Default decision tree
        assesment(i) = fuzzyOp([...
                    fuzzyOp([...
                        fuzzyOp([eval_mf({highwayMPG.Good, in(i,6)}, 'trapmf'), eval_mf({horsepower.Average, in(i,4)},
'trimf')], 'AND_F', opModel, w),...
                        fuzzyOp(eval_mf({cityMPG.Poor, in(i,5)}, 'trapmf'), 'NOT_F', opModel, w)],...
                    'AND_F', opModel, w),...
                    fuzzyOp([...
                        fuzzyOp([eval_mf({risk.Low, in(i,2)}, 'trapmf'), eval_mf({valueLoss.Low, in(i,3)}, 'trapmf')],'AND_F',
opModel, w),...
                        eval_mf({price.Cheap, in(i,7)}, 'trapmf')],...
                    'OR_F', opModel, w)],...
                   'AND_F', opModel, w);
    elseif decisionTree==2
        assesment(i) = fuzzyOp([...
                    fuzzyOp([...
                        fuzzyOp([eval_mf({cityMPG.Average, in(i,5)}, 'trimf'), eval_mf({highwayMPG.Good, in(i,6)},
'trapmf')], 'OR_F', opModel, w),...
                        fuzzyOp(eval_mf({horsepower.Average, in(i,4)}, 'trimf'), 'NOT_F', opModel, w)],...
                    'AND_F', opModel, w),...
                    fuzzyOp([...
                        fuzzyOp([eval_mf({valueLoss.Low, in(i,3)}, 'trapmf'),...
                            fuzzyOp(eval_mf({risk.High, in(i,2)}, 'trapmf'), 'NOT_F', opModel, w)],...
                        'AND_F', opModel, w),...
                        fuzzyOp(eval_mf({price.Average, in(i,7)}, 'trapmf'), 'NOT_F', opModel, w)],...
                    'AND_F', opModel, w)],...
                   'AND_F', opModel, w);
    end
end

end
```