دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

گزارش ۹ درس هوش مصنوعی

پیاده‌سازی یک سیستم فازی به منظور کنترل یک مسئله با ورودی‌های زبانی

به قلم:
امیر بابامحمودی

استاد
دکتر مهدی قطعی

تیر ۱۴۰۰

مقدمه:

در این گزارش قصد داریم با گرفتن یک سری ورودی برای یک مسئله یک مدل کنترل فازی پیاده‌سازی کنیم.

تعریف مسئله:

در این مسئله قصد داریم تا با گرفتن درجه‌ی کثیفی لباس‌ها و وزن آنها حدودی برای میزان مصرف مواد شوینده بدست بیاوریم با توجه به اینکه این مسئله در کلاس هم بیان شد به بررسی جزئیات آن میپردازیم. ورودی میزان کثیفی لباس‌ها به صورت زبانی میباشد که در آن کاربر با دادن عددی بین ۱ تا ۵ درجه‌ی کثیفی لباس‌ها را بیان میکند. وزن لباس ها نیز که میتواند از ۱ کیلوگرم تا ۱۰ کیلوگرم باشد.

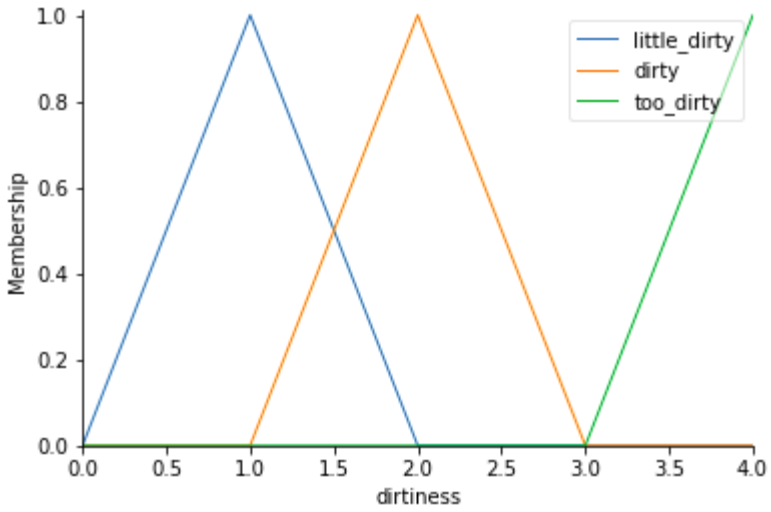
همانطور که میدانید نیاز میباشد که در ابتدا مقادیر ورودی fuzzy شده و بتوان یک حالت توصیفی برای آن‌ها ارائه داد که در ادامه به جزئیات آن میپردازیم.

درجه کثیفی :

بطور کلی در این مسئله از کتابخانه skfuzzy استفاده شده‌است. همانطور که در شکل بالا میبینند برای میزان کثیفی یک scale از ۰ تا ۵ برای میزان کثیفی تعریف شده است که به صورت گسسته و بافاصله‌ی ۱ درجه‌ها با هم فاصله دارند.

```
dirtiness = ctrl.Antecedent(np.arange(0, 5 , 1), 'dirtiness')
dirtiness['little_dirty'] = fuzz.trimf(dirtiness.universe, [0, 1, 2])
dirtiness['dirty'] = fuzz.trimf(dirtiness.universe, [2, 2, 3])
dirtiness['too_dirty'] = fuzz.trimf(dirtiness.universe, [3, 4, 5])
```

میزان کثیفی را بر اساس اینکه در چه بازه‌ای باشد به سه دست تقسیم کرده‌ایم. اگر درجه کثیفی از صفر تا دو بود توصیف «little_dirty» به آن نسبت داده شده است. اگر بین دو تا سه باشد توصیف «dirty» و در نهایت اگر بین سه و پنج بودی «too_dirty».
تمامی این سه دسته بندی در این گزارش به صورت مثلثی پیاده‌سازی شده‌اند که نموداری آن را در زیر میبینید.



این نمودار نسبت عضویت هر دسته لباس بر اساس درجه‌ی کثیفی به سه دسته‌ی تعریف شده را نشان میدهد.

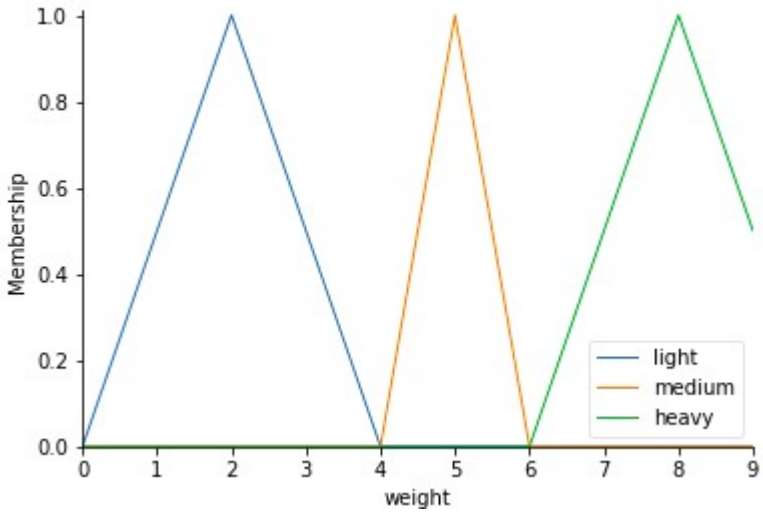
وزن لباس‌ها:

وزن لباس‌ها را در بازه‌ای بین صفر تا ده scale کردیم که برای آن سه طبقه توصیفی تعریف شده است.

```
weight = ctrl.Antecedent(np.arange(0, 10, 1), 'weight')
weight['light'] = fuzz.trimf(weight.universe, [0, 2, 4])
weight['medium'] = fuzz.trimf(weight.universe, [4, 5, 6])
weight['heavy'] = fuzz.trimf(weight.universe, [6, 8, 10])
```

همانگونه که در کد میبینید اگر بازه‌ی وزن لباس‌ها بین صفر تا چهار کیلوگرم باشد در دسته‌ی "light" اگر در بازه‌ی چهار تا ۶ کیلوگرم باشد "medium" و در نهایت اگر بین شش تا ۱۰ کیلو گرم باشد در دسته‌ی "heavy" قرار میگیرد.

نمودار میزان عضویت هر وزن به هردسته را میتوانید در شکل ببینید.



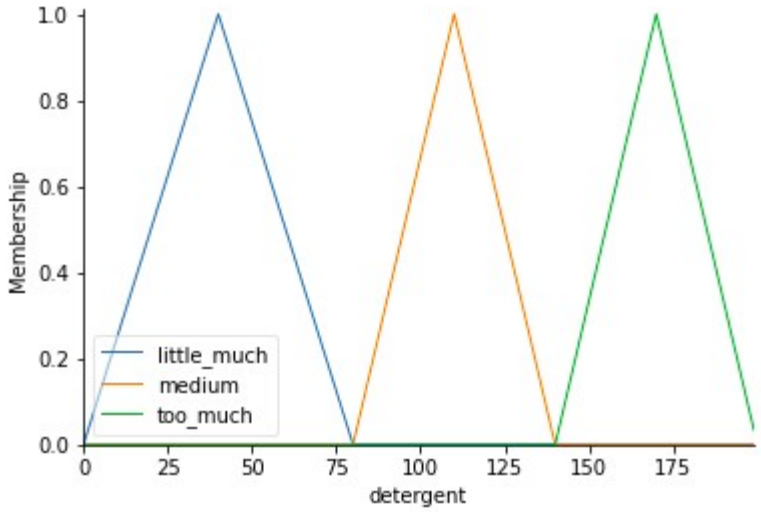
میزان مصرف مواد شوینده:

میزان مصرف مواد شوینده متغیری میباشد که میخواهیم به صورت عددی آن را خروجی بدهیم اما در ابتدا نیاز میباشد که به آن یک مدل فازی نسبت بدهیم که در انتها آن را defuzzicate کنیم.

همانگونه که میبینید میزان مصرف مواد شوینده که بازه‌ی آن از ۰ تا ۲۰۰ گرم قرار داده شده است را نیز به سه دسته تقسیم کردیم.

```
detergent = ctrl.Consequent(np.arange(0, 200, 1), 'detergent')
detergent['little_much'] = fuzz.trimf(detergent.universe, [0, 40, 80])
detergent['medium'] = fuzz.trimf(detergent.universe, [80, 110, 140])
detergent['too_much'] = fuzz.trimf(detergent.universe, [140, 170, 200])
```

بازه‌ی مصرفی مواد شوینده بین چهل تا هشتاد گرم در دسته‌ی "little_much" قرار گرفته و بازه‌ی هشتاد تا صد و چهل در دسته‌ی "medium" و بین صد و چهل و دویست گرم در دسته‌ی "too_much" قرار میگیرد که نمودار عضوت هر مقدار را میتونید در شکل زیر ببینید.



قوانین:

حال به قوانینی که میان مقادیر ورودی و خروجی که در بالا تعریف شده و فازی شدند میپردازیم

```
rule1a = ctrl.Rule(dirtiness['little_dirty'] | weight['light'], detergent['little_much'])
rule1b = ctrl.Rule(dirtiness['little_dirty'] | weight['medium'], detergent['little_much'])
rule1c = ctrl.Rule(dirtiness['little_dirty'] | weight['heavy'], detergent['medium'])

rule2a = ctrl.Rule(dirtiness['dirty'] | weight['light'], detergent['little_much'])
rule2b = ctrl.Rule(dirtiness['dirty'] | weight['medium'], detergent['medium'])
rule2c = ctrl.Rule(dirtiness['dirty'] | weight['heavy'], detergent['too_much'])

rule3a = ctrl.Rule(dirtiness['too_dirty'] | weight['light'], detergent['medium'])
rule3b = ctrl.Rule(dirtiness['too_dirty'] | weight['medium'], detergent['too_much'])
rule3c = ctrl.Rule(dirtiness['too_dirty'] | weight['heavy'], detergent['too_much'])
```

همانگونه که میبینید ۹ تا قانون ایجاد کردیم که در یک جدول خلاصه‌ی آن را میاوریم.

dirtiness		**wight**		**detergent**
little_dirty	AND	light	THEN	little_much
little_dirty	AND	medium	THEN	little_much
little_dirty	AND	heavy	THEN	medium
dirty	AND	light	THEN	little_much
dirty	AND	medium	THEN	medium
dirty	AND	heavy	THEN	too_much
too_dirty	AND	light	THEN	medium
too_dirty	AND	medium	THEN	too_much
too_dirty	AND	heavy	THEN	too_much

همانگونه که در جدول میبینید قوانین بطور فازی مشخص شده‌اند بطوری که برای مثال اگر میزان کثیفی لباس‌ها کم بوده و وزن لباس‌ها نیز کم باشد مقدار شوینده کمی نیاز میشود و میتوان ۸ قانون دیگ را به همین ترتیب در جدول دنبال کرد.

حال خروجی کد برای یکسری ورودی را حساب کرده و بعد از آن نحوه‌ی حساب شدن خروجی را توضیح میدهیم.

```
43 dirtiness.view()
44 x = 2
45 weight.view()
46 y = 5
47 detergent.view()
48
49 det_amount.input['dirtiness'] = int(x)
50 det_amount.input['weight'] = int(y)
51
52 det_amount.compute()
53 print(det_amount.output['detergent'])
54 detergent.view(sim=det_amount)
55
```

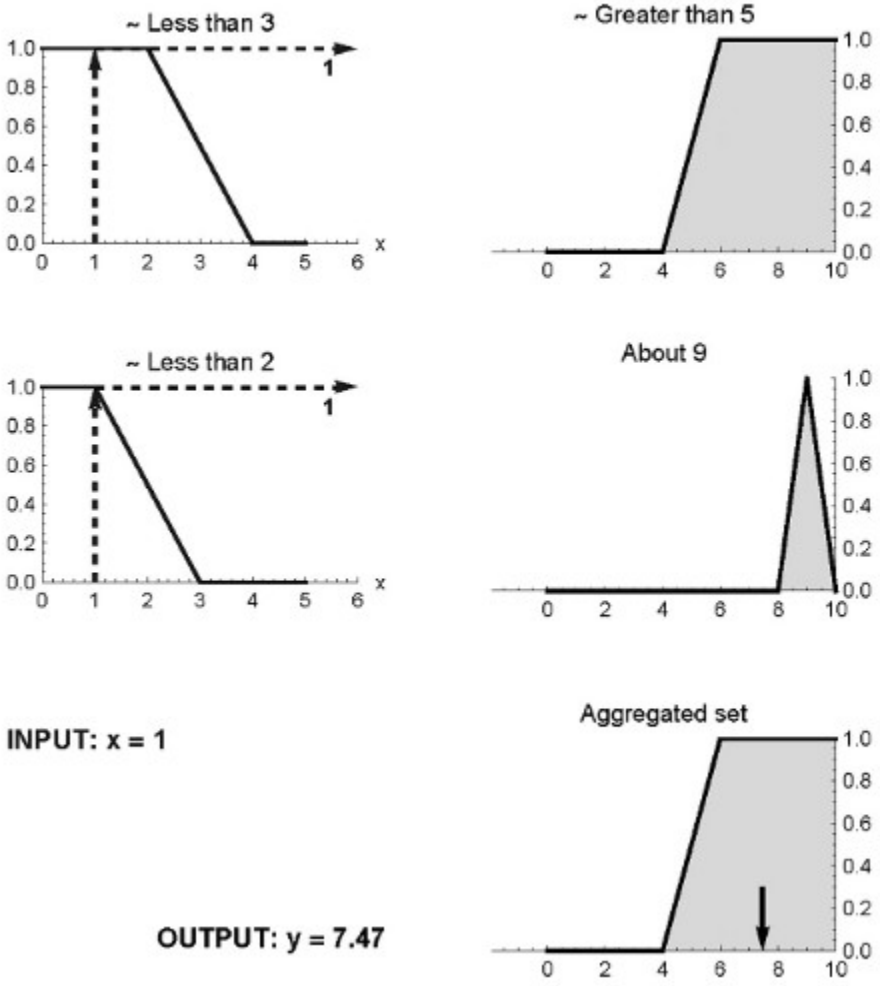
```
98.9832749902761
```

با توجه به جدول نیز میبینم که وزن در محدوده‌ی متوسط و درجه‌ی کثیفی نیز متوسط بوده که بازه‌ی مواد شوینده نیز باید متوسط بوده که ۹۹ در این بازه میباشد. حال میخواهیم نحوه‌ی بدست آمدن این خروجی رو به طور مختصر توضیح بدهیم.

تابع compute که جز توابع built_in کتابخانه skfuzzy میباشد به وسیله‌ی روش mamdani خروجی نهایی را محاسبه میکند. در این روش مقدار درجه‌ی عضویت ورودی‌ها در مجموعه‌های سمت antecedent ها محاسبه شده و مقدار عضویت آن در بخش consequence محاسبه میشود. به طوری این اتفاق میوفتد که از نقطه‌ی خاصی در بخش antecedent یک خط ممتد کشیده شده و در نمودار خروجی در هر لحظه مقدار مینیمم انتخاب میشود. این کار برای هر شرط انجام شده و در نهایت نمودار حاصل از اجتماع آن ها رسم میشود.حال برای بدست اوردن خروجی روش های مختلفی برای defuzzification وجود دارد که در روش مرکزگرایی یا centroid با محاسبه مساحت زیر نمودارهای انتخاب شده گرانیگاه آن محاسبه میشود و به عنوان مقدار عددی خروجی انتخاب میشود.

$$\mu outputk|x(y)=min(\mu Bk(y),\mu Ak(x))$$

همانگونه که توضیح داده شد در هر نقطه مقدار مینیمم به عنوان خروجی انتخاب میشود.
در شکل زیر نیز نمونه‌ی تصویری این روش قابل مشاهده میباشد



لینک کد در گیتهاب:

https://github.com/amirbabamahmoudi/AI-projects/tree/main/fuzzy_model

منابع:

۱ـ https://www.jasss.org/21/3/2.html

۲ـ https://towardsdatascience.com/a-very-brief-introduction-to-fuzzy-logic-and-fuzzy-systems-d68d14b3a3b8

3- https://github.com/FreakyHarsh/Fuzzy-controller/blob/master/fuzzy.py