簡介

This is the whole algorithm of the gift ranking part. At the beginning, we fill in all the feature vector of each gift in step 0. Then, according to the occasion that sender choose, we calculate the min square distance from each gift to the occasion in step 1. Sort the gifts by distance, the gift in the top means the most suitable gift to the occasion. But some of the gifts have not measure up the baseline of some features. That means those gifts have some defects in this situation. So we keep going to step2, which filtered the gifts that have some feature vector values are not measure up.

After the expert system done its job. Here comes the commonsense part.

We use commonsense Divisi analyze tool to calculate the relation score in step 3. And calculate the spreading activation score in step 4.

Since step 3 has stronger relation than step 4. We give step 3 higher weight to stronger its characteristic.

細說為什麼要這樣做的部分

Sort by Min Square Distance

Occasion is the biggest issue when people pick a gift. So we calculate the distance between the occasion and gift in 8 feature vector space. If distance is short. That means this gift may suit in this occasion.

Filter With Preference Threshold

Sender can decide what features must need. Step2 help sender to filter the defect gifts. If a gift has one or more feature that not measure up the threshold, then it will be append to the inferior gift collection.

Commonsense

There are many factors may affect weather a gift is suitable or not. Sometimes, a few knowledge about will be written on the commonsense Knowledge base. Such as “rose” has a strong relation to Valentine’s Day, “Cake” has a strong relation to birthday. To find these kinds of relation, people usually come out SVD and wordNet. If two key word are commonly showed up together, then they possibly have strong relation. But we are using a better way, commonsense Knowledge base to find out their relations. Compare to the pure text analyze, commonsense has the detail information about two words’ relation. The assertion in commonsense Knowledge base are just like the thought in your mind.

For the reason below, we use commonsense Divisi to do analyze part. The function we call is called fast speed spreading activation. That means this function can calculate a category’s relation to a word. And this function can also help you to find out what are the same concepts between the category’s items.

In step 3, we want to filter the nonrelative items by commonsense. Since occasion and relationship are the most 2 important Information. We put them in to a category call scenario, to find out what kinds of gift has a strong relation with both of them. The strong relation has a harder condition to satisfy. This gift must be top 20 relative items by the category.

Because step 3’s condition is too hard to complete, we extend step 4 to find out more relative items by commonsense. This time, all the consideration form UI will be used to calculate spreading activation score. We put each consideration in to one category, and then give each score a weight which from the equalize score of the pre-questionnaire.

//main

===============================================================

//step0

InitialAndBulidGiftFeatureVector(AllTheInformations )

Main(AllTheInformations)

//step1

SortByMinSquareDistance(Occasion,Gifts[])

//step2

FilterWithPreferenceThreshold(Top20Gifts[],Features[])

//step3

FilterWithCommonSenseRelation(Occasion, Relationship, Inferior\_Gifts[])

FilterWithCommonSenseRelation(Occasion, Relationship, Better\_Gifts[])

//step4

FilterWithCommonSpreadingActivation(OtherInformations[],Inferior\_Gifts[])

FilterWithCommonSpreadingActivation(OtherInformations[],Better\_Gifts[])

//show result

Result = print Better\_Gifts[], then print Inferior\_Gifts[]

//step0

===============================================================

InitialAndBulidGiftFeatureVector(AllTheInformations )

Occasion = Occasion that sender choose

Relationship = Relationship that sender choose

OtherInformations = AllTheInformations Occasion Relationship

Gifts[] = Different 60 common gifts

For each Gift in all Gifts

Fill in 8 value of the vector

For each occasion in all occasions

Fill in 8 value of the vector

Return Gifts[],Occasions[]

Ex:

Gift -> Book [ 1.0, 0.6, 0.2, 0.4, 1.0, 0.0, 0.0, 0.6]

Occasion->Valentine’s day [ 0.2, 1.0, 0.6, 0.2, 0.0, 1.0, 0.2, 0.2]

紅色部分請務必擺上去

//step1

============================================================

SortByMinSqureDistance(Occasion,Gifts[])

For each gift in all gifts

=

Sort Gifts[] by

Top20Gifts [] = top 20 of Gifts[]

Return Top20Gifts []

//step2

=============================================================

FilterWithPreferenceThreshold(Gifts[],Features[])

For each Gift in all Gifts[]

For each Feature in all the Features[] that Sender Choose

If <

Then this Gift append to Inferior\_Gifts[]

Else if >=

Then this Gift append to Better\_Gifts[]

//step3

=============================================================

FilterWithCommonSenseRelation(Occasion, Relationship, Separate\_Gifts[])

For each Gift in all Separate\_Gifts[]

If is in top 20

Else is not in top 20

//step4

==============================================================

FilterWithCommonSpreadingActivation(OtherInformations[], Separate\_Gifts[])

For each Gift in all Separate\_Gifts[]

If

Else <

Sort Separate\_Gifts[] by