1 tableAgent: A chain-method table class in Matlab

> Author: linrenwen@gmail.com

1.1 objective

@tableAgent: create a class for applying chain method on table in Matlab

1.2 Install

All files you need are included in `tableAgent_test.zip` .

1. unzip the `tableAgent_test.zip` . and add the folder to `the bottom of` matlab path.
2. run `tableAgent_test.m` for examples.

1.3 usage

> See `tableAgent_test.m` for List of all features.

1.3.1 Prepare Data for futher test

```
load patients.mat
TRaw = table(LastName,Gender,Age,Height,Weight,Location,Smoker,Diastolic,Systolic,...
    SelfAssessedHealthStatus);
T = tableAgent(TRaw);
Ttable = T.table;
clear Age Diastolic Gender Height LastName Location SelfAssessedHealthStatus Smoker Systolic Wei
fCsv = 'data_patients.csv';
writetable(TRaw, fCsv,'Delimiter' ,'\t','QuoteStrings',true,'Encoding','UTF-8')
```

1.3.2 tableAgent Construction

Construction method 1

```
TB = tableAgent;
TB.name = ["Joan","Merry","Tom","Kate"]';
TB.sex = ["male","female","male","female"]';
TB.grade = [99,67,66,35]';
TB.G = [99,67,88,55]'+ 4;
```

Construction method 2

```
TB = table;
TB.name = ["Joan","Merry","Tom"]';
TB.grade = [99,67,35]';
TB.G = [99,67,35]'+ 4;
TB = tableAgent(TB);
```

Construction method 3

```
TB = readtableAgentRaw(fCsv);
TB = readtableAgent(fCsv);
```

1.3.3 Access data of tableAgent

Access Block of tableAgent

```
TB = T;
TB{1,8:9} = [1,125];
disp(TB{1,5});
TB{1,1} = {'SMITH'};
data1 = T{[1,3,6],'Age'}; % TODO: data1 = T{'Age<20','Age'};
data2 = TB{1,3};
```

dropcols keepcols, and droprows keeprows

```
TB = T.keepcol('LastName,Age');
TB = T.keepcol({'LastName','Age'});
TB = T.keepcol(["LastName","Age"]);
TB = T.keepcol([1, 3]);
TB = T.col([1,3]).keepcol();

TB = T.dropcol(3);
TB = T.dropcol('Age');

TB = T.droprow(3);
TB = T.row(3).droprow();
TB = T.row([1,10]).droprow('Age>50'); % drop
TB = T.keeprow([1,3,4]);
```

1.3.4 generate/update columns

generate new col of constant

```
TB = T.gen('Gmean1="good"');
TB = T.gen('Gmean2=1');
TB = T.gen('Gmean3=NaT');
TB = T.gen('No2=1:obj.height'); % nature number col

TB.No2 = (1:TB.height); % gen nature number col
TB.No2 = 1;
TB.No2 = 'good';

TB = T.row([1:10]).gen('Age3=pi');
TB = T(1:3,2:3).row(1).gen('Age4=Age+100');
```

generate new col from other cols

```
TB = T.row('ismember(LastName,{''Jones''})').gen('Age2=Age+100')...
    .row([1,2]).gen('Age2=Age+100');
TB = T.runCmdGen('Age2 = Age + 100');
```

generate new col or variable, by passing inline-function para

```
% % Test of passing anonymous-function-para
fnew = @(x)(x+3);
TB = T.row().gen('Age=fnew(pi)',fnew,'fnew');
```

generate new col by passing variable-para

```
para.x = [1,1]';
para.y = [10,10]';
TB = T.row([1,2]).gen('AgeB=Age + para.x',para);
```

generate new col by group operation

```
% % method 1
TB = T.row('Systolic>=120').groupby('Gender').genbygroup('Systolic_TF = Systolic - mean(Systolic
    .genbygroup('Diastolic_TF = mean(Diastolic)');
% % method 2
TB = T.groupby('Gender','AgeMean',@(x)mean(x),'Age');% coly = f(colx)
```

generate new col for each cols

```
TB = T.row([1,2]).gen_forEachCol('Age,Height','$x+4','$x_add');
```

generate new col by slicing or descrete assign

```
TB = T.row().gen_slice('HH', ...
    ["ismember(LastName,{'Smith','Jones'})","'NAME'"; ...
    "ismember(Gender,{'Female'})","'female'"
    "else","'ELSE'"]);
```

generate new col for dummy variable

```
TB = T.gen_dummy('Age');
```

change block of table by Exchange or Copy

```
rowsA = 1;
colsA = 2:8;
rowsTarget = rowsA +1;
colsTarget = colsA;
TB = T.blockExchange(rowsA,colsA,rowsTarget,colsTarget);
TB = T.blockCopy(rowsA,colsA,rowsTarget,colsTarget);
```

merge with other tableAgent

```
TB = T.gen('Age3 = Age+100');
TC = TB([1,2],[1,11]);
TM = T.merge(TC,'LastName');
```

query col value from other tableAgent

```
keyA = 'LastName';
keyB = keyA;
valsA = 'Age3';
valsB = 'Age3';

TM = T.queryTabAinTabB(keyA,valsA,TC,keyB,valsB);
```

stack col

```
TB = tableAgent;
TB.Month = {'October';'November';'December';...
    'January';'February';'March'};
TB.Year = [2005*ones(3,1); 2006*ones(3,1)];
TB.NE = [1.1902; 1.3610; 1.5003; 1.7772; 2.1350; 2.2345];
TB.MidAtl = [1.1865; 1.4120; 1.6043; 1.8830; 2.1227; 1.9920];
TB.SAtl = [1.2730; 1.5820; 1.8625; 1.9540; 2.4803; 2.0203];

TC = TB.stack('NE,MidAtl,SAtl');

TD = TC.unstack('NE_MidAtl_SAtl','NE_MidAtl_SAtl_Indicator');

% stackCell
TM = TD.gen_forEachCol('Year,NE,MidAtl,SAtl','num2str($x)');
TM{1,1:5} = TM.table.Properties.VariableNames;

TN = TM.stackCell('loc,Value',(1:6),(1:2),([3,4,5]));
% TM.stackCell(vnameRowandColval,rows,colsID,colsVal)
```

pivot

```
[TB,TBUnstacked] = T.pivot({'Gender','Smoker'},'Diastolic',@numel);
```

1.3.5 disp table

```
T.disp
disp(T);
disp(T.table);

T.dispclass;
dispclass(T);
T.gen('G=2').dispclass;

T.dispBasicProperties;
```

1.3.6 plotcols

```
hf = figure;
setfontdefault(11)
T.gen('No=1:obj.height').plotcols('No,Age,Weight');
```

1.3.7 a long example for chain method

```
TB = T.row('Age==40|Age<35').gen('Age = Age+1').gen('G = Age*2')...
    .row('Age<=99').gen('G = log(Age)*10')...
    .row([1,3]).gen('G=3')...
    .row().gen('G=pi');
```

1.4 TODO and FIXME

TODO:

1. about col
   (1) label properties supporting UTF8 colname/variable name.
   (2) collabel: select cols by label

FIXME:

1. the `T(:,:)` are not supported

1.5 requirement

Matlab 2018b