DANIELS
COLLEGE OF BUSINESS
UNIVERSITY of DENVER

Python for Data Analytics

ADV STAT Module: Lesson 3

Training Manual

ADVSTAT: Lesson 3

Lecture for ADVSTAT Lesson 3

Mini Assignment ADVSTAT L3

Objectives Lesson 3

- Lesson 3
  - Using Widgets to Control Output
  - iPython Navigation/Display of Images
  - Interactive Widgets to Dynamically Control Output
  - Import code from a python script file
  - Toggle (hide/view) code in iPython with HTML Java snipet
  - Using iPython Presentation Slides

DANIELS COLLEGE OF BUSINESS UNIVERSITY of DENVER

Widgets - ADVSTATL3WidgetsBasic.ipynb

Widgets allow you to give the user choices through a graphical interface in your code. We will need to import 3 items to work with the widgets in today's notes. We will also go ahead and import numpy, pandas, and matplotlib.

```
#Import Packages
from ipywidgets import *
from ipywidgets import interact
from IPython.display import display

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

DANIELS COLLEGE OF BUSINESS UNIVERSITY of DENVER

Slider Widget - Integer

Slider Widget

In [4]:
```
wSlid = widgets.IntSlider(
    value=5,
    min=0,
    max=40,
    step=5,
    description = 'Choose a value:'
)
display(wSlid)
```

Choose a value: 5

In [6]:
```
print(wSlid.value)
```

```
25
```

In [37]:
```
if wSlid.value == 5:
    print('You chose 5!!!')
```

```
You chose 5!!!
```

Slider Widget - Float

Float Slider

In [8]:
```
wSlidFlt = widgets.FloatSlider(
    value=5,
    min=0.0,
    max=5.0,
    description = 'Choose a value:'
)
display(wSlidFlt)
```

Choose a value: 3.5

In [40]:
```
print(wSlidFlt.value)
```

```
3.5
```

In [41]:
```
if wSlidFlt.value == 3.5:
    print('You chose 3.5!!!')
```

```
You chose 3.5!!!
```

Text Box Widget - Integer

'Text' Input Box

Text Box - Integer

```
In [11]: wIntText = widgets.IntText(
             value=7,
             description='Any:',
             disabled=False
         )
         display(wIntText)
```

Any: 7

```
In [12]: print('Your Integer Value was: ' + str(wIntText.value))
         Your Integer Value was: 7
```

Text Box Widget - Float

Text Box - Float

```
In [42]: wFloatText = widgets.FloatText(
             value=7.5,
             description='Any:',
             disabled=False,
             color='black'
         )
         display(wFloatText)
```

Any: 7.5

```
In [43]: print('Your Float Value was: ' + str(wFloatText.value))
         Your Float Value was: 7.5
```

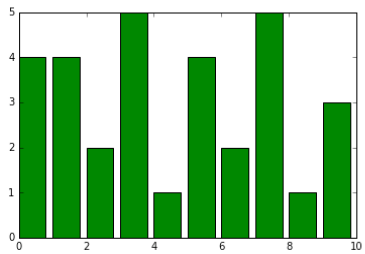
Radio Buttons - Widget

Radio Buttons

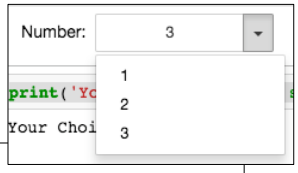
```
In [20]: wRadio = widgets.RadioButtons(
             options=['red', 'blue', 'green'],
             value='blue',
             description='Chart Color:'
         )
         display(wRadio)
```

Chart Color: ○ red
○ blue
◉ green

```
In [36]: y=np.random.randint(1,6,10)
         x = np.arange(10)
         print('Your Radio Button Choice Value was: ' + wRadio.value)

         plt.bar(x,y,color=wRadio.value)

         Your Radio Button Choice Value was: green

Out[36]: <Container object of 10 artists>
```



Drop Down – Widget – Version 1



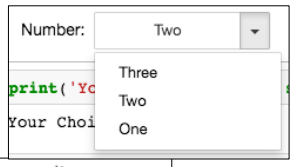
Version 1 - Choices are results

```
In [49]: wDropDwn1 = widgets.Dropdown(
             options=['1', '2', '3'],
             value='2',
             description='Number:'
         )
         display(wDropDwn1)
```

Number: 3

```
In [50]: print('Your Choice was: ' + str(wDropDwn1.value))

         Your Choice was: 3
```

Drop Down – Widget – Version 2



Version 2 - Use Dictionary so choices converted to results

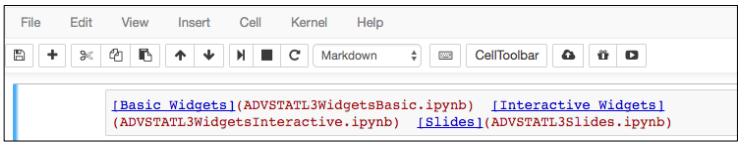
```
In [56]: wDropDwn2 = widgets.Dropdown(
             options={'One': 1, 'Two': 2, 'Three': 3},
             value=2,
             description='Number:',
         )
         display(wDropDwn2)
```

Number: Two

```
In [57]: print('Your Choice was: ' + str(wDropDwn2.value))
         Your Choice was: 2
```

Navigation

- You can link to other .ipynb files with the syntax: **`[text] (filename)`**

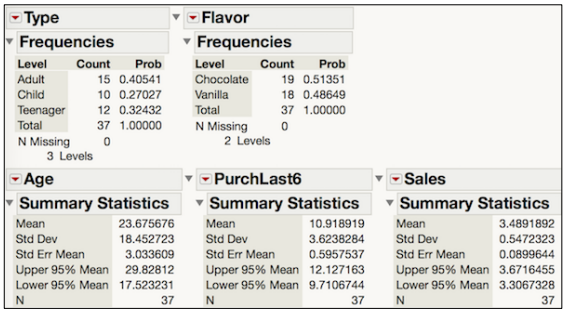


DANIELS COLLEGE OF BUSINESS UNIVERSITY of DENVER

Displaying Images with Broken Link Tag

You can use similar syntax to display an image. Note – we use `![text]` to indicate the text to be displayed if the image won't import. Note also this image is in an 'images' folder one level below the location of the .ipynb file.

```
![JMP Descriptive Ice Cream Data](images/ADVSTATL3DescStats.png)
```



Type

Frequencies

Level	Count	Prob
Adult	15	0.40541
Child	10	0.27027
Teenager	12	0.32432
Total	37	1.00000
N Missing	0	
3 Levels		

Flavor

Frequencies

Level	Count	Prob
Chocolate	19	0.51351
Vanilla	18	0.48649
Total	37	1.00000
N Missing	0	
2 Levels		

Age

Summary Statistics

Mean	23.675676
Std Dev	18.452723
Std Err Mean	3.033609
Upper 95% Mean	29.82812
Lower 95% Mean	17.523231
N	37

PurchLast6

Summary Statistics

Mean	10.918919
Std Dev	3.6238284
Std Err Mean	0.5957537
Upper 95% Mean	12.127163
Lower 95% Mean	9.7106744
N	37

Sales

Summary Statistics

Mean	3.4891892
Std Dev	0.5472323
Std Err Mean	0.0899644
Upper 95% Mean	3.6716455
Lower 95% Mean	3.3067328
N	37

JMP Descriptive Ice Cream Data

Interactive Widgets to Dynamically Control Output

The problem with the widgets is by default, you have to run each cell to see the results or use the results.

Instead, we can use `interact()` to both call code to run (a function) when the widget changes and also define the widget.

syntax:

```
def somefunction(x):
    do something

interact(somefunction, x=widget)
```

DANIELS COLLEGE OF BUSINESS UNIVERSITY of DENVER

Interact – Slider - ADVSTATL3WidgetsInteractive.ipynb

We can use *interact* to update output dynamically without having to 'rerun' cell. Based on the type of list of values you give the second parameter, it will choose the widget type. If you give it [0,40,5] it assumes this is min, max and step and will produce an integer slider. If you give it [0.0,40.0,0.5] it will produce a float slider.

Slider

```
def f(Choice):
    print(Choice)
interact(f, Choice=[0,40,5]);
```

Choice 20

```
20
```

Interact – Slider – explicit Widget

To be more specific and have access to additional widget options, we can explicitly define the widget with the widgets.name syntax.

```
def f(x):
    print(x)
    if x >= 20:
        print('You chose a value greater than or equal to 20')
    else:
        print('You chose a value less than 20')
interact(f, x=widgets.IntSlider(min=0,max=40,step=1,
                                value=10,
                                description = 'Enter Choice: '));
```

Enter Choice: 10

```
10
You chose a value less than 20
```

Enter Choice: 26

```
26
You chose a value greater than or equal to 20
```

Import code from a python script file

We have created a function myttest that will accept input for a t test and then output the results on a graph. Rather than have to recreate that code here, let's just read it in using:
%load ADVSTATFunctions.py.
Note - after it runs, it comments itself out so it doesn't rerun again.

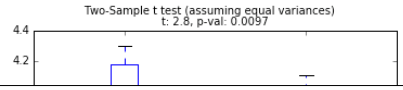
In [ ]:
```
# %load ADVSTATFunctions.py
"""
Created on Sat Nov 12 2016
Module: Advanced Statistical Analysis
t test function with graph output
@author: Kellie Keeling
"""

import pandas as pd
import numpy as np
from scipy import stats
import matplotlib.pyplot as plt

#t test for equal means - variances first
def myttest(data1,namedata1, data2, namedata2, alpha = .05):
    tvar, p_valvar = stats.bartlett(data1,data2)
    print("This is a test of equal variances with Ho: The variances are eq
    print("The t test statistic is " + str(round(tvar,3)) + " and the p-va
```

Interact – Drop Down – calling myttest

```
#Drop Down for Menu
def runttest(Choice):
    print(Choice)
    if Choice == 'Compare Choc vs. Vanilla':
        data1=df['Sales'][df['Flavor']=='Chocolate']
        data2=df['Sales'][df['Flavor']=='Vanilla']
        myttest(data1,'Chocolate', data2, 'Vanilla')
    elif Choice=='Compare Adult vs Teenager':
        data1=df['Sales'][df['Type']=='Adult']
        data2=df['Sales'][df['Type']=='Teenager']
        myttest(data1,'Adult', data2, 'Teenager')
interact(runttest, Choice=['Compare Choc vs. Vanilla',
                           'Compare Adult vs Teenager']);
```

Choice: Compare Adult vs Teenager

```
Compare Adult vs Teenager
This is a test of equal variances with Ho: The variances are equal
The t test statistic is 0.953 and the p-value is 0.329
Conclusion: Fail to Reject Ho: We can't reject that the variances are the
 same
This is a Two-Sample t test (assuming equal variances) of equal means wit
h Ho: The group means are equal
The t test statistic is 2.8 and the p-value is 0.0097
Conclusion: Reject Ho: The means are not equal
```



Multiple Inputs

Two Choices

Suppose you have a situation where you have a main drop down, but then you want to present the user with different choices based on what they originally choose. In this case, we define a choicelst dictionary where the keys are the 3 main options and then the subset of next variable options that they get are saved in a list. Note that if they choose Data, that there is no additional variable to choose. Once you make the two selections, the output will be updated.

```
df = pd.DataFrame({'John':[1,2,3,4],'Mark':[2,3,4,5],'Frank':[3,4,5,6]})

choicelst={'Bar Chart':['John','Frank'],
           'Pie Chart':['John','Mark', 'Frank'],
           'Data':['None to choose']}

def runmenu (var):
    print('The choice: ' + menuch.value + ' and the var: ' + var)
    if menuch.value=='Bar Chart':
        print('Bar' + ' with ' + var)
        y=df[var]
        x=np.arange(len(y))
        plt.bar(x,y)
    elif menuch.value=='Pie Chart':
        print('Pie' + ' with ' + var)
        y=df[var]
        plt.pie(y)
    elif menuch.value=='Data':
        print(df)
```

Multiple Inputs

Using `interactive` instead of `interact` removed extraneous output.

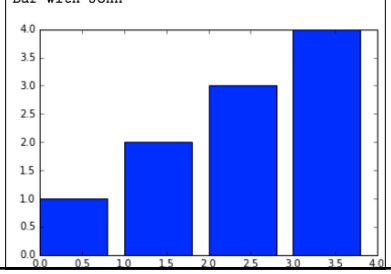
```
#Drop Down Menu of choices - returns 'choice'
menuch=widgets.Dropdown(options=choicelst.keys())
                        #['Bar Chart','Pie Chart','Data'])
init = menuch.value
#Secondary Drop Drown of variable choice - returns 'var'
choice2 = widgets.Dropdown(options=choicelst[init])

dispmenuch = widgets.interactive(select_var,choice=menuch)
dispchoice2 = widgets.interactive(runmenu,var=choice2)

display(dispmenuch)
display(dispchoice2)
```

choice: Data
var: None to choose

```
The choice: Data and the var: None to choose
   Frank  John  Mark
0      3     1     2
1      4     2     3
2      5     3     4
3      6     4     5
```

choice: Bar Chart
var: John

```
The choice: Bar Chart and the var: John
Bar with John
```



Toggling Viewing the Code - ADVSTATL3Slides.ipynb

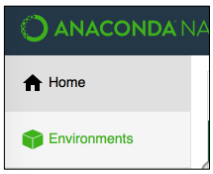
We can make the output look cleaner once we get it all to work by adding a Toggle button to view/hide the code.

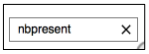
Click here to toggle on/off the raw code.

```
from IPython.display import HTML

HTML('''<script>
code_show=true;
function code_toggle() {
 if (code_show){
 $('div.input').hide();
 } else {
 $('div.input').show();
 }
 code_show = !code_show
}
$( document ).ready(code_toggle);
</script>
<form action="javascript:code_toggle()">
  <input type="submit"
     value="Click here to toggle on/off the raw code."></form>''')
```

iPython Presentation Slides

- Finally, we can use Notebooks to make presentations that we can save as html files to be viewed similar to a Powerpoint Presentation.
- In order to set up Jupyter to allow slides, we will use the nbpresent package. You will need to install it in the Anaconda Environments. Search for nbpresent and then click it to download the package. Once it is listed in your "installed" list, then open Jupyter to see new icons:





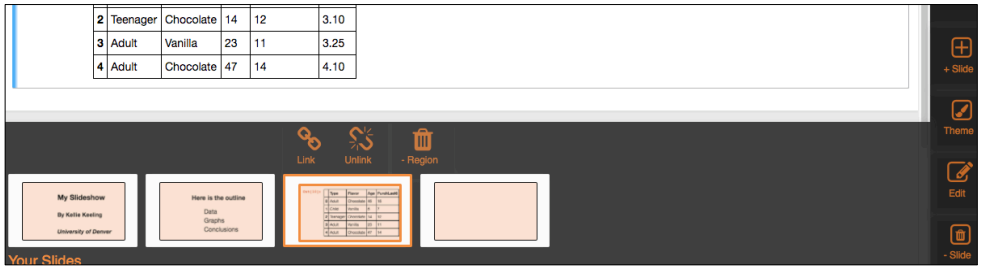
iPython Presentation Slides

- The easiest way to convert to slides is to design your Notebook so that each cell will be a slide in the presentation.
- Once you set up the slides
  - Choose View, Toggle Presentation. Note that the interface is a little strange. View, Toggle Presentation again will go back to the Notebook. Esc will also work.
  - Select the Slides icon and choose the Basic option and it will make each cell a slide. You can also delete any extra slides. Having the toggle on/off code is nice. This is something you will just have to play with.

DANIELS COLLEGE OF BUSINESS UNIVERSITY of DENVER

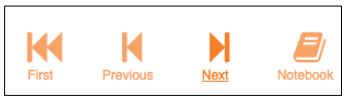
iPython Presentation Slides

View Presentation



iPython Presentation

Navigation:



My Slideshow

By Kellie Keeling

University of Denver

Here is the outline

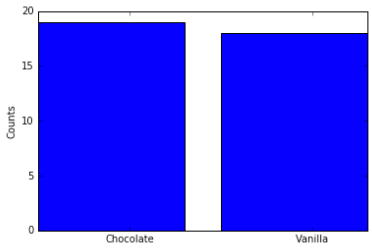
1. Data
2. Graphs
3. Conclusions

The Ice Cream Data Has 5 Columns

	Type	Flavor	Age	PurchLast6	Sales
0	Adult	Chocolate	45	15	4.25
1	Child	Vanilla	5	7	2.90
2	Teenager	Chocolate	14	12	3.10
3	Adult	Vanilla	23	11	3.25
4	Adult	Chocolate	47	14	4.10

Pick One: Flavor

Bar Plot of Counts by Ice Cream Flavor



Conclusions

From our analysis, we can see that for Person Type:

- Adults have the highest counts
- Children have the lowest counts

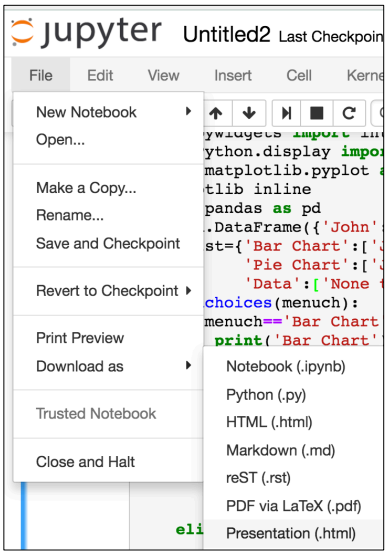
Therefore, we should market more to Adults

From our analysis, we can see that for Flavor:

- Chocolate and Vanilla have similar counts

iPython Presentation

You can save a static version of your Presentation slides by downloading them as .html.



DANIELS COLLEGE OF BUSINESS UNIVERSITY of DENVER

Help Online for Widgets

Widget Basics and Widget List:

https://github.com/ipython/ipywidgets/blob/master/docs/source/examples/Index.ipynb

Interact Help:

http://ipywidgets.readthedocs.io/en/latest/examples/Using%20Interact.html

DANIELS COLLEGE OF BUSINESS UNIVERSITY of DENVER

Answer the following questions:

1. What was the hardest content in this class for you to learn?
2. Is there any other content you would have liked to have covered in the class?
3. Was there any content you would have like to have covered less?

DANIELS COLLEGE OF BUSINESS UNIVERSITY of DENVER