CollectingData - B213753 assessment

June 18, 2022

1 Title: Collecting data using interactive Jupyter widgets

Author: B213753 , *Adapted from*: "CollectingDataUsingInteractiveJupyterWidgets.ipynb" by Mairead Bermingham (mairead.bermingham@ed.ac.uk)

Notebook and data info: This Notebook uses interactive jupyter-widgets to collect data for the length of stay (LOS_model) dataset from the test data observations, and save it to a working 'Data' folder. All capture data is saved to the 'RawData' folder.
Data: Data consists of numerical data and character data from NHSRdatasets package.

2 Data

The data used to input into the data collection tool is a generated test datafrom from the LOS_model dataset in the NHSRdatasets package in R. The selected variables include: patient ID, organisation/trust, patient age, length of stay, and death. The R script "./RScripts/load_dataset.R" was used to subset the full `LOS_model` data into test and training data.

3 Load pandas and test data

[1]:
```
#Load the 'pandas' package
import pandas as pd
testData=pd.read_csv("../Data/LOS_test.csv")
testData
```

[1]:

	ID	Organisation	Age	LOS	Death
0	60	Trust10	12	1	Alive
1	85	Trust5	90	12	Alive
2	87	Trust7	26	3	Alive
3	137	Trust7	78	10	Alive
4	159	Trust9	53	6	Alive
5	166	Trust6	18	2	Alive
6	175	Trust5	77	3	Died
7	251	Trust1	73	1	Died
8	293	Trust3	81	8	Died
9	300	Trust10	93	15	Alive

Check data types

[2]:
```
result = testData.dtypes
print("Output:")
print(result)
```

```
Output:
ID              int64
Organisation   object
Age             int64
LOS             int64
Death          object
dtype: object
```

The data type object is a string. The data types for the variables are:

- ID: integer 64 bits
- Organisation: string
- Age: integer 64 bits
- LOS: integer 64 bits
- Death: string

There are no abnormalities and the data types are all appropriate for the variables above.

3.0.1 Set up new empty data frame to store captured data

Use head() to look at the first entry in the dataset.

[3]:
```
testData.head(n=1)
```

[3]:

	ID	Organisation	Age	LOS	Death
0	60	Trust10	12	1	Alive

Use pd.DataFrame() to set up an empty data frame corresponding to the columns of the test data, and their data type. Add an extra column for consent - whether the data subject has consented to processing and sharing the data collected with the data capture tool.

[4]:
```
dfTofill = pd.DataFrame({'ID': [-999], # Integer
                         'Organisation': ['NA'], # String
                         'Age': [0], # Integer
                         'LOS': [0], # Integer
                         'Death': ['NA'], # String
                         'consent': [False]})  # Boolean

dfTofill
```

[4]:

	ID	Organisation	Age	LOS	Death	consent
0	-999	NA	0	0	NA	False

Save the empty data frame to your working 'Data' folder:

[5]:
```
#dfTofill.to_csv('../Data/CollectedData.csv', index=False)
```

Read the CollectedData.csv data frame to prepare for data collection.

[6]:
```
CollectData=pd.read_csv("../Data/CollectedData.csv")
CollectData
```

[6]:

	ID	Organisation	Age	LOS	Death	consent
0	4	Trust3	0	0	Alive	True
1	9	Trust9	51	7	Died	True
2	46	Trust6	49	7	Alive	True
3	112	Trust2	19	3	Alive	True
4	114	Trust4	23	3	Alive	True
5	135	Trust5	93	10	Alive	True
6	187	Trust7	33	4	Died	True
7	204	Trust4	27	2	Alive	True
8	213	Trust3	48	5	Alive	True
9	244	Trust4	76	9	Alive	True
10	260	Trust10	11	1	Alive	True
11	267	Trust7	79	10	Alive	True

4 Using Widgets to collect data

Widgets are interactive Python objects that have a representation in the browser and will be used by this tool as the method to collect data.

Import the *ipywidgets* Python package.

[7]:
```
#Load the 'ipywidgets' package
from ipywidgets import widgets, Layout
```

4.0.1 display()

The *IPython.display* package is used to display different objects in Jupyter.

[8]:
```
#Load the 'IPython.display' package
from IPython.display import display
```

4.0.2 Python code below to prepare and format the form

Consent input

[9]:
```
Consent_input = widgets.Checkbox(
        value=False,
        disabled=False,
        indent=False,
        layout=Layout(right='20%', border='solid red', width='200px',padding='0␣
↪0 0 45%', height='auto'))
```

```
consent_checkbox = widgets.Box(
    [widgets.Label('Consent:',
                   layout=Layout(left='25%', overflow='visible', padding='1px 0␣
 ↪0 0')),
    Consent_input]
)

consent_text = widgets.Textarea(
    value='I consent for the data I have provided to be processed and shared in␣
 ↪accordance with data protection regulations with the purpose of improving␣
 ↪care service provision across the UK.',
    disabled=True,
    layout=Layout(width='80%')
)

consent = widgets.HBox(
    [consent_text,consent_checkbox],
    layout=Layout(display='flex',align_items='flex-end',␣
 ↪justify_content='center' )
)
```

ID input

[10]:
```
ID_input= widgets.BoundedIntText(
    step=1,
    max=99999,
    description='ID :',
    disabled=False,
    style = {'description_width': '100px'}
)
```

Organisation input

[11]:
```
Organisation_input=widgets.Dropdown(
    options=['Trust1', 'Trust2', 'Trust3','Trust4', 'Trust5',␣
 ↪'Trust6','Trust7', 'Trust8', 'Trust9','Trust10'],
    value='Trust1',
    description='Organisation :',
    disabled=False,
    style = {'description_width': '100px'}
)
```

Age input

[12]:
```
Age_input=widgets.BoundedIntText(
    step=1,
```

```
    min=0,
    max=150,
    description='Age (Yrs) :',
    disabled=False,
    style = {'description_width': '100px'}
)
```

LOS input

[13]:
```
LOS_input=widgets.BoundedIntText(
    step=1,
    min=0,
    description='Length of Stay :',
    disabled=False,
    style = {'description_width': '100px'}
)
```

Death status input

[14]:
```
Dead_input=widgets.Dropdown(
    options=['Alive','Died'],
    value='Alive',
    description='Status :',
    disabled=False,
    style = {'description_width': '100px'}
)
```

Add a submit button to programmatically add a row onto collected data and reset the form

[15]:
```
button = widgets.Button(
    description='Submit',
    disabled=False,
    button_style='info',
    tooltip='Submit',
    icon='check',
    layout=Layout(left='150px')
)
```

[16]:
```
def reset_form():
    ID_input.value=0
    # Organisation value not reset as no real "default"
    Age_input.value=0
    LOS_input.value=0
    Dead_input.value='Alive'
    Consent_input.value=False
```

```
# button.on_click passes in an argument  which is not needed in this function␣
 ↪(so is given as _ here)
def add_row(_):
    global CollectData
    # deep copy of empty row
    new_row=pd.DataFrame.copy(dfTofill)

    # get the form inputs in a list ordered as the columns
    inputs=[ID_input.value,Organisation_input.value,Age_input.value,LOS_input.
 ↪value,Dead_input.value,Consent_input.value]

    # iterate over each column and value, and enter into new row
    for i,value in enumerate(inputs):
        new_row.iloc[0,i]=value

    # append onto the main dataframe
    CollectData  = pd.concat([CollectData, new_row])

    # reset form values
    reset_form()

    # Update displayed table (starting from index 1 to miss the initialising␣
 ↪row at index 0)
    display_collected.update(CollectData[1:])

# Add the add_row callback function onto button click
button.on_click(add_row)
```

5 Consent

Consent is a vital area for data protection compliance. Consent means giving data subjects genuine choice and control over how you process their data. If the data subject has no real choice, consent is not freely given, and it will be invalid. The General Data Protection Regulation sets a high standard for consent and contains significantly more detail than previous data protection legislation. Consent is defined in Article 4 as: "Consent of the data subject means any freely given, specific informed and unambiguous indication of the data subject's wishes by which he or she, by a statement or by a clear affirmative action, signifies agreement to the processing of personal data relating to him or her".

Before data is collected, the end-user must consent to the processing and sharing the data collected with this data capture tool.

[17]:
```
display(consent)
```

```
HBox(children=(Textarea(value='I consent for the data I have provided to be processed and share
```

5.1 Enter data below:

[18]:
```
#marker = CollectData=pd.read_csv("../Data/LOS_test_marker.csv")
#marker
```

[19]:
```
display(ID_input,Organisation_input, Age_input,LOS_input,Dead_input,button)
display_collected = display(CollectData[1:],display_id=True)
```

```
BoundedIntText(value=0, description='ID :', max=99999, style=DescriptionStyle(description_widt
```

```
Dropdown(description='Organisation :', options=('Trust1', 'Trust2', 'Trust3', 'Trust4', 'Trust
```

```
BoundedIntText(value=0, description='Age (Yrs) :', max=150, style=DescriptionStyle(description
```

```
BoundedIntText(value=0, description='Length of Stay :', style=DescriptionStyle(description_wid
```

```
Dropdown(description='Status :', options=('Alive', 'Died'), style=DescriptionStyle(description
```

```
Button(button_style='info', description='Submit', icon='check', layout=Layout(left='150px'), s
```

	ID	Organisation	Age	LOS	Death	consent
1	9	Trust9	51	7	Died	True
2	46	Trust6	49	7	Alive	True
3	112	Trust2	19	3	Alive	True
4	114	Trust4	23	3	Alive	True
5	135	Trust5	93	10	Alive	True
6	187	Trust7	33	4	Died	True
7	204	Trust4	27	2	Alive	True
8	213	Trust3	48	5	Alive	True
9	244	Trust4	76	9	Alive	True
10	260	Trust10	11	1	Alive	True
11	267	Trust7	79	10	Alive	True

5.1.1 Filter for consent status

After completing data entry: rows without consent are removed before saving to a folder.

[20]:
```
CollectData=CollectData[CollectData['consent'] == True]
```

5.1.2 Saving the CollectData data frame

Saving the data collected by your data-capture tool to the working data folder:

[21]:
```
CollectData.to_csv('../Data/CollectedData.csv', index=False)
```

That is the CollectData data frame saved to the working 'Data' folder. You need to iterate through this Notebook until you have collected all of your test data and then save the captured test data to your 'RawData' folder.

[22]:
```
# CollectData.to_csv('../RawData/CollectedDataFinal.csv', index=False)
```

That is the final CollectData data frame saved to the 'RawData' folder.

Commented out after completed data collection.