

```

1  '''
2  Instruction: 6CCS3PRJ
3              Data Visualisation of Migration Data
4
5  Code Author: Yanpu Huang,
6              1725298,
7              K1763861,
8              Kings College London,
9              Computer Science,
10             3rd year student
11
12 supervisor: Prof.Dr.Rita Borgo
13
14 User guide: Install python 2.7
15             Install bokeh(set the path to python 2.7 site-packages)
16             Install pandas(set the path to python 2.7 site-packages)
17             Install numpy(set the path to python 2.7 site-packages)
18             Install openpyxl(set the path to python 2.7 site-packages)
19             open terminal, run: python -m bokeh serve --show web.py, wait for
the respond from browser
20 '''
21
22 import openpyxl
23 import numpy as np
24 import pandas as pd
25 from bokeh import events
26 from bokeh.plotting import figure, curdoc, output_file, show
27 from bokeh.models import ColumnDataSource, HoverTool, CustomJS
28 from bokeh.layouts import row, column, gridplot, widgetbox, layout
29 from bokeh.models.widgets import Button, RadioButtonGroup, Select, Slider,
Dropdown, Toggle, Tabs, Panel, CheckboxGroup
30 from bokeh.transform import linear_cmap, factor_cmap, dodge
31 from bokeh.io import export_png
32 import warnings
33 from bokeh.core.properties import value
34
35 output_file("LondonDataStoreDataVisualisation.html",title="Migration Data
Visualisation") #output file
36
37 # Make line chart of long term migration(London vs UK) by years
38 def plot_long_term_migration():
39     wb = openpyxl.load_workbook('data/Long term international
migration.xlsx') # Import datasets
40     ws = wb['Data']
41
42     x=[]
43     y1=[]
44     y2=[]
45     y3=[]
46     y4=[]
47     for row in range(32,72,4):
48         x.append(ws.cell(row = row,column = 1).value[:4]) # append dates by
years
49         y1.append(ws.cell(row = row,column = 8).value) # append number of
people migrated into London
50         y2.append(ws.cell(row = row, column = 2).value) # append numebr of
people migrated into UK
51         y3.append(ws.cell(row = row, column = 11).value) # append numebr of
people migrated out London

```

```

52     y4.append(ws.cell(row = row, column = 4).value) # append numebr of
people migrated out UK
53
54     #print(x)
55     #print(y1)
56     #print(y2)
57     #print(y3)
58     #print(y4)
59
60     source1 = ColumnDataSource(data = dict(dates = x, values = y1))
61     source2 = ColumnDataSource(data = dict(dates = x, values = y2))
62     source3 = ColumnDataSource(data = dict(dates = x, values = y3))
63     source4 = ColumnDataSource(data = dict(dates = x, values = y4))
64
65
66     p = figure(plot_width = 600, plot_height = 300,x_axis_label =
"dates",y_range = (0,1000),
67             y_axis_label = "Migration-input population",tools =
"hover,pan,box_zoom,save,reset,undo,zoom_in,zoom_out,wheel_zoom",title="Long-
term migration(London vs UK)") # Create a new figure
68
69     p.line(x = "dates", y = "values", line_width = 2, source = source1, color
= "black", legend = "London-in") # draw a line chart
70     p.line(x = "dates", y = "values", line_width = 2, source = source2, color
= "teal", legend = "UK-in") # draw a line chart
71     p.line(x = "dates", y = "values", line_width = 2, source = source3, color
= "chocolate", legend = "London-out") # draw a line chart
72     p.line(x = "dates", y = "values", line_width = 2, source = source4, color
= "darkred", legend = "UK-out") # draw a line chart
73     p.legend.location = "top_left"
74     p.legend.orientation = "horizontal"
75     p.circle(x = "dates", y = "values", fill_color = 'white',size = 3,source
= source1) # point of line chart
76     p.circle(x = "dates", y = "values", fill_color = 'white',size = 3,source
= source2) # point of line chart
77     p.circle(x = "dates", y = "values", fill_color = 'white',size = 3,source
= source3) # point of line chart
78     p.circle(x = "dates", y = "values", fill_color = 'white',size = 3,source
= source4) # point of line chart
79
80     return p
81
82
83 # Make line chart of employment (London vs UK) by years
84 def plot_employment_population():
85     wb = openpyxl.load_workbook('data/underemployment.xlsx') # Import
datasets
86     ws = wb.get_sheet_by_name('Data')
87
88     x = []
89     y1 = []
90     y2 = []
91     for row in range(8,19):
92         x.append(ws.cell(row = row,column = 1).value)
93         y1.append(ws.cell(row = row,column = 2).value)
94         y2.append(ws.cell(row = row,column = 7).value)
95
96     #print(x)
97     #print(y1)
98     #print(y2)
99

```

```

100     source1 = ColumnDataSource(data = dict(dates = x, values = y1))
101     source2 = ColumnDataSource(data = dict(dates = x, values = y2))
102
103     p = figure(plot_width = 600, plot_height = 300, x_axis_label = "dates",
104               y_axis_label = "Employees number", tools =
"hover,pan,box_zoom,save,reset,undo,zoom_in,zoom_out,wheel_zoom",
title="Total employees/ self-employed(16+) number(London vs UK)") # Create a
new figure
105
106
107     p.line(x = "dates", y = "values", line_width = 2,source = source1, color
= "black",legend = "London") # draw a line chart
108     p.line(x = "dates", y = "values", line_width = 2,source = source2, color
= "teal",legend = "UK") # draw a line chart
109     p.circle(x = "dates", y = "values", fill_color = 'white',size = 3,source
= source1) # draw point of line chart
110     p.circle(x = "dates", y = "values", fill_color = 'white',size = 3,source
= source2) # draw point of line chart
111
112     p.legend.location = "top_left"
113     p.legend.orientation = "horizontal"
114     p.y_range.range_padding = 1
115
116     return p
117
118 # Make line chart of underemployment rate(London vs UK) by years
119 def plot_underemployment_rate():
120     wb = openpyxl.load_workbook('data/underemployment.xlsx') # Import
datasets
121     ws = wb.get_sheet_by_name('Data')
122
123     x = []
124     y1 = []
125     y2 = []
126     for row in range(8,19):
127         x.append(ws.cell(row = row,column = 1).value)
128         y1.append(ws.cell(row = row,column = 5).value)
129         y2.append(ws.cell(row = row,column = 10).value)
130
131     #print(x)
132     #print(y1)
133     #print(y2)
134
135     source1 = ColumnDataSource(data = dict(dates = x, values = y1))
136     source2 = ColumnDataSource(data = dict(dates = x, values = y2))
137
138     p = figure(plot_width = 600, plot_height = 300, x_axis_label = "dates",
139               y_axis_label = "Percentage underemployed", tools =
"hover,pan,box_zoom,save,reset,undo,zoom_in,zoom_out,wheel_zoom",
title="Underemployment rate(London vs UK)") #Create a new figure
140
141
142     p.line(x = "dates", y = "values", line_width = 2,source = source1, color
= "black",legend = "London") #draw a line chart
143     p.line(x = "dates", y = "values", line_width = 2,source = source2, color
= "teal",legend = "UK") #draw a line chart
144     p.circle(x = "dates", y = "values", fill_color = 'white',size = 3,source
= source1) #point of line chart
145     p.circle(x = "dates", y = "values", fill_color = 'white',size = 3,source
= source2) #point of line chart
146

```

```

147     p.legend.location = "top_left"
148     p.legend.orientation = "horizontal"
149     p.y_range.range_padding = 1
150
151     return p
152
153 # Make line chart fo short-term migration(London areas) by years
154 def plot_linechart_areas_short_term_migration():
155     wb = openpyxl.load_workbook('data/Short term migration.xlsx') # Import
datasets
156     ws = wb.get_sheet_by_name('Data')
157     x = []
158     areas = []
159
160     y1,y2,y3,y4,y5,y6,y7,y8,y9,y10,y11,y12,y13,y14,y15,y16,y17,y18,y19,y20,y21,y
22,y23,y24,y25,y26,y27,y28,y29,y30,y31,y32,y33,y34 = [],[],[],[],[],[],[],[],
[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[],[]
160     #Rearrange data
161     for column in range (2,12):
162         areas.append(ws.cell(row = row,column = 1).value)
163         x.append(ws.cell(row = 5,column = column).value)
164         y1.append(ws.cell(row = 6,column = column).value)
165         y2.append(ws.cell(row = 7,column = column).value)
166         y3.append(ws.cell(row = 8,column = column).value)
167         y4.append(ws.cell(row = 9,column = column).value)
168         y5.append(ws.cell(row = 10,column = column).value)
169         y6.append(ws.cell(row = 11,column = column).value)
170         y7.append(ws.cell(row = 12,column = column).value)
171         y8.append(ws.cell(row = 13,column = column).value)
172         y9.append(ws.cell(row = 14,column = column).value)
173         y10.append(ws.cell(row = 15,column = column).value)
174         y11.append(ws.cell(row = 16,column = column).value)
175         y12.append(ws.cell(row = 17,column = column).value)
176         y13.append(ws.cell(row = 18,column = column).value)
177         y14.append(ws.cell(row = 19,column = column).value)
178         y15.append(ws.cell(row = 20,column = column).value)
179         y16.append(ws.cell(row = 21,column = column).value)
180         y17.append(ws.cell(row = 22,column = column).value)
181         y18.append(ws.cell(row = 23,column = column).value)
182         y19.append(ws.cell(row = 24,column = column).value)
183         y20.append(ws.cell(row = 25,column = column).value)
184         y21.append(ws.cell(row = 26,column = column).value)
185         y22.append(ws.cell(row = 27,column = column).value)
186         y23.append(ws.cell(row = 28,column = column).value)
187         y24.append(ws.cell(row = 29,column = column).value)
188         y25.append(ws.cell(row = 30,column = column).value)
189         y26.append(ws.cell(row = 31,column = column).value)
190         y27.append(ws.cell(row = 32,column = column).value)
191         y28.append(ws.cell(row = 33,column = column).value)
192         y29.append(ws.cell(row = 34,column = column).value)
193         y30.append(ws.cell(row = 35,column = column).value)
194         y31.append(ws.cell(row = 36,column = column).value)
195         y32.append(ws.cell(row = 37,column = column).value)
196         y33.append(ws.cell(row = 38,column = column).value)
197         y34.append(ws.cell(row = 39,column = column).value)
198
199
200     p_areas = figure(plot_width = 1000, plot_height = 500, x_axis_label =
"dates",
201                     y_axis_label = "Migration population",y_range = (0,7000),

```

```

202         tools =
"hover,pan,box_zoom,save,reset,undo,zoom_in,zoom_out,wheel_zoom",
title="Areas of London short-term migration") #Create a new figure
203     p2 = p_areas.line(x = x, y = y2, line_width = 1, color = "#000003", legend
= "City of London") #draw a line chart
204     p3 = p_areas.line(x = x, y = y3, line_width = 1, color = "#140D35", legend
= "Barking and Dagenham") #draw a line chart
205     p4 = p_areas.line(x = x, y = y4, line_width = 1, color = "#3B0F6F", legend
= "Barnet") #draw a line chart
206     p5 = p_areas.line(x = x, y = y5, line_width = 1, color = "#63197F", legend
= "Bexley") #draw a line chart
207     p6 = p_areas.line(x = x, y = y6, line_width = 1, color = "#8C2980", legend
= "Brent") #draw a line chart
208     p7 = p_areas.line(x = x, y = y7, line_width = 1, color = "#B53679", legend
= "Bromley") #draw a line chart
209     p8 = p_areas.line(x = x, y = y8, line_width = 1, color = "#DD4968", legend
= "Camden") #draw a line chart
210     p9 = p_areas.line(x = x, y = y9, line_width = 1, color = "#F66E5B", legend
= "Croydon") #draw a line chart
211     p10 = p_areas.line(x = x, y = y10, line_width = 1, color =
"#FD9F6C", legend = "Ealing") #draw a line chart
212     p11 = p_areas.line(x = x, y = y11, line_width = 1, color =
"#FDCD90", legend = "Enfield") #draw a line chart
213     p12 = p_areas.line(x = x, y = y12, line_width = 1, color =
"#FBFCBF", legend = "Greenwich") #draw a line chart
214     p13 = p_areas.line(x = x, y = y13, line_width = 1, color =
"#a6cee3", legend = "Hackney") #draw a line chart
215     p14 = p_areas.line(x = x, y = y14, line_width = 1, color =
"#1f78b4", legend = "Hammersmith and Fulham") #draw a line chart
216     p15 = p_areas.line(x = x, y = y15, line_width = 1, color =
"#b2df8a", legend = "Haringey") #draw a line chart
217     p16 = p_areas.line(x = x, y = y16, line_width = 1, color =
"#33a02c", legend = "Harrow") #draw a line chart
218     p17 = p_areas.line(x = x, y = y17, line_width = 1, color =
"#fb9a99", legend = "Havering") #draw a line chart
219     p18 = p_areas.line(x = x, y = y18, line_width = 1, color =
"#e31a1c", legend = "Hillingdon") #draw a line chart
220     p19 = p_areas.line(x = x, y = y19, line_width = 1, color =
"#fdbf6f", legend = "Hounslow") #draw a line chart
221     p20 = p_areas.line(x = x, y = y20, line_width = 1, color =
"#ff7f00", legend = "Islington") #draw a line chart
222     p21 = p_areas.line(x = x, y = y21, line_width = 1, color =
"#cab2d6", legend = "Kensington and Chelsea") #draw a line chart
223     p22 = p_areas.line(x = x, y = y22, line_width = 1, color =
"#6a3d9a", legend = "Kingston upon Thames") #draw a line chart
224     p23 = p_areas.line(x = x, y = y23, line_width = 1, color =
"#ffff99", legend = "Lambeth") #draw a line chart
225     p24 = p_areas.line(x = x, y = y24, line_width = 1, color =
"#b15928", legend = "Lewisham") #draw a line chart
226     p25 = p_areas.line(x = x, y = y25, line_width = 1, color =
"#e41a1c", legend = "Merton") #draw a line chart
227     p26 = p_areas.line(x = x, y = y26, line_width = 1, color =
"#377eb8", legend = "Newham") #draw a line chart
228     p27 = p_areas.line(x = x, y = y27, line_width = 1, color =
"#4daf4a", legend = "Redbridge") #draw a line chart
229     p28 = p_areas.line(x = x, y = y28, line_width = 1, color =
"#984ea3", legend = "Richmond upon Thames") #draw a line chart
230     p29 = p_areas.line(x = x, y = y29, line_width = 1, color =
"#ff7f00", legend = "Southwark") #draw a line chart
231     p30 = p_areas.line(x = x, y = y30, line_width = 1, color =
"#ffff33", legend = "Sutton") #draw a line chart

```



```

232     p31 = p_areas.line(x = x, y = y31, line_width = 1, color =
"#a65628", legend = "Tower Hamlets") #draw a line chart
233     p32 = p_areas.line(x = x, y = y32, line_width = 1, color =
"#f781bf", legend = "Waltham Forest") #draw a line chart
234     p33 = p_areas.line(x = x, y = y33, line_width = 1, color =
"#410967", legend = "Wandsworth") #draw a line chart
235     p34 = p_areas.line(x = x, y = y34, line_width = 1, color =
"#6A176E", legend = "Westminster") #draw a line chart

236
237     #Set legend location and legend orientation
238     p_areas.legend.location = "top_left"
239     p_areas.legend.orientation = "horizontal"
240     #Set callback function after click checkboxes
241     display_event = CustomJS(code="""
242         p2.visible = false;
243         p3.visible = false;
244         p4.visible = false;
245         p5.visible = false;
246         p6.visible = false;
247         p7.visible = false;
248         p8.visible = false;
249         p9.visible = false;
250         p10.visible = false;
251         p11.visible = false;
252         p12.visible = false;
253         p13.visible = false;
254         p14.visible = false;
255         p15.visible = false;
256         p16.visible = false;
257         p17.visible = false;
258         p18.visible = false;
259         p19.visible = false;
260         p20.visible = false;
261         p21.visible = false;
262         p22.visible = false;
263         p23.visible = false;
264         p24.visible = false;
265         p25.visible = false;
266         p26.visible = false;
267         p27.visible = false;
268         p28.visible = false;
269         p29.visible = false;
270         p30.visible = false;
271         p31.visible = false;
272         p32.visible = false;
273         p33.visible = false;
274         p34.visible = false;

275
276         if(cb_obj.active.includes(0)){
277             p2.visible = true;
278         }
279         if (cb_obj.active.includes(1)){
280             p3.visible = true;
281         }
282         if (cb_obj.active.includes(2)){
283             p4.visible = true;
284         }
285         if (cb_obj.active.includes(3)){
286             p5.visible = true;
287         }
288         if (cb_obj.active.includes(4)){

```

```
289         p6.visible = true;
290     }
291     if (cb_obj.active.includes(5)){
292         p7.visible = true;
293     }
294     if (cb_obj.active.includes(6)){
295         p8.visible = true;
296     }
297     if (cb_obj.active.includes(7)){
298         p9.visible = true;
299     }
300     if (cb_obj.active.includes(8)){
301         p10.visible = true;
302     }
303     if (cb_obj.active.includes(9)){
304         p11.visible = true;
305     }
306     if (cb_obj.active.includes(10)){
307         p12.visible = true;
308     }
309     if (cb_obj.active.includes(11)){
310         p13.visible = true;
311     }
312     if (cb_obj.active.includes(12)){
313         p14.visible = true;
314     }
315     if (cb_obj.active.includes(13)){
316         p15.visible = true;
317     }
318     if (cb_obj.active.includes(14)){
319         p16.visible = true;
320     }
321     if (cb_obj.active.includes(15)){
322         p17.visible = true;
323     }
324     if (cb_obj.active.includes(16)){
325         p18.visible = true;
326     }
327     if (cb_obj.active.includes(17)){
328         p19.visible = true;
329     }
330     if (cb_obj.active.includes(18)){
331         p20.visible = true;
332     }
333     if (cb_obj.active.includes(19)){
334         p21.visible = true;
335     }
336     if (cb_obj.active.includes(20)){
337         p22.visible = true;
338     }
339     if (cb_obj.active.includes(21)){
340         p23.visible = true;
341     }
342     if (cb_obj.active.includes(22)){
343         p24.visible = true;
344     }
345     if (cb_obj.active.includes(23)){
346         p25.visible = true;
347     }
348     if (cb_obj.active.includes(24)){
349         p26.visible = true;
```

```

350     }
351     if (cb_obj.active.includes(25)){
352         p27.visible = true;
353     }
354     if (cb_obj.active.includes(26)){
355         p28.visible = true;
356     }
357     if (cb_obj.active.includes(27)){
358         p29.visible = true;
359     }
360     if (cb_obj.active.includes(28)){
361         p30.visible = true;
362     }
363     if (cb_obj.active.includes(29)){
364         p31.visible = true;
365     }
366     if (cb_obj.active.includes(30)){
367         p32.visible = true;
368     }
369     if (cb_obj.active.includes(31)){
370         p33.visible = true;
371     }
372     if (cb_obj.active.includes(32)){
373         p34.visible = true;
374     }
375     """,args={'p2': p2, 'p3': p3, 'p4': p4,
376     p9,'p10': p10,
377     'p11': p11,'p12': p12,'p13': p13,'p14':
378     p14,'p15': p15,'p16': p16,
379     'p17': p17,'p18': p18,'p19': p19,'p20':
380     p20,'p21': p21,'p22': p22,
381     'p23': p23,'p24': p24,'p25': p25,'p26':
382     p26,'p27': p27,'p28': p28,
383     'p29': p29,'p30': p30,'p31': p31,'p32':
384     p32,'p33': p33,'p34': p34
385     })
386     #Set widgets checkboxes
387     selection_box = CheckboxGroup(labels= [
388         "City of London",
389         "Barking and Dagenham",
390         "Barnet",
391         "Bexley",
392         "Brent",
393         "Bromley",
394         "Camden",
395         "Croydon",
396         "Ealing",
397         "Enfield",
398         "Greenwich",
399         "Hackney",
400         "Hammersmith and Fulham",
401         "Haringey",
402         "Harrow",
403         "Havering",
404         "Hillingdon",
405         "Hounslow",
406         "Islington",
407         "Kensington and Chelsea",
408         "Kingston upon Thames",
409         "Lambeth",

```



```

406     "Lewisham",
407     "Merton",
408     "Newham",
409     "Redbridge",
410     "Richmond upon Thames",
411     "Southwark",
412     "Sutton",
413     "Tower Hamlets",
414     "Waltham Forest",
415     "Wandsworth",
416     "Westminster"],active =
[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28
,29,30,31,32,33])
417
418     selection_box.js_on_click(display_event)
419     row_1 = [p_areas, selection_box] #Make selection boxes located besides
line chart
420     return row_1
421
422
423 def plot_linechart_London_short_term_migration():
424     wb = openpyxl.load_workbook('data/Short term migration.xlsx') # Import
datasets
425     ws = wb.get_sheet_by_name('Data')
426     x = []
427     y = []
428     for column in range (2,12):
429         x.append(ws.cell(row = 5,column = column).value)
430         y.append(ws.cell(row = 6,column = column).value)
431
432     p_London = figure(plot_width = 600, plot_height = 300, x_axis_label =
"dates",
433                       y_axis_label = "Migration population",
434                       tools =
"hover,pan,box_zoom,save,reset,undo,zoom_in,zoom_out,wheel_zoom",
title="London short-term migration") #Create a new figure
435
436     p_London.line(x = x, y = y, line_width = 2, color = "black", legend =
"London")
437     p_London.circle(x = x, y = y, fill_color = 'white',size = 3)
438     p_London.legend.location = "top_left"
439     p_London.legend.orientation = "horizontal"
440     return p_London
441
442
443 def plot_shortterm_vbar():
444     wb = openpyxl.load_workbook('data/Short term migration.xlsx') # Import
datasets
445     ws = wb.get_sheet_by_name('Data')
446
447     #Rearrange data, append them into new lists
448     year2data = {}
449     areas_list = []
450     year_list = []
451     for column in range (2,12):
452         year = str(ws.cell(row = 5,column = column).value)
453         year_list.append(year)
454         data_dict = {}
455         for row in range(7, 40):
456             areas = ws.cell(row = row,column = 1).value
457             if areas not in areas_list:

```

```

458         areas_list.append(areas)
459         data_dict[areas] = ws.cell(row=row, column=column).value
460
461         year2data[year] = data_dict
462
463     default_year = year_list[0]
464     counts_list = []
465     display_data_dict = year2data[default_year]
466     for areas in areas_list:
467         counts_list.append(display_data_dict[areas])
468
469     source = ColumnDataSource(data=dict(areas_list=areas_list,
counts_list=counts_list))
470
471     #plot new figure
472     p_year = figure(plot_width=1000, plot_height=500,
473                     y_axis_label="Migration population", x_range=areas_list,
474                     y_range=(0, 6000),
tools="hover,pan,box_zoom,save,reset,undo,zoom_in,zoom_out,wheel_zoom",
475                     title="Areas of London short-term migration")
476     #set colour
477     colors = ['#000003', '#140D35', '#3B0F6F', '#63197F', '#8C2980',
478             '#B53679', '#DD4968', '#F66E5B', '#FD9F6C', '#FDCD90',
479             '#FBFCBF', '#a6cee3', '#1f78b4', '#b2df8a', '#33a02c',
480             '#fb9a99', '#e31a1c', '#fdbf6f', '#ff7f00', '#cab2d6',
481             '#6a3d9a', '#ffff99', '#b15928', '#e41a1c', '#377eb8',
482             '#4daf4a', '#984ea3', '#ff7f00', '#ffff33', '#a65628',
483             '#f781bf', '#410967', '#6A176E']
484     #implement vertical bar chart
485     p_vbar = p_year.vbar(x='areas_list', top='counts_list', source=source,
width=0.5, alpha=0.8, color=factor_cmap('areas_list', palette=colors,
factors=areas_list))
486     p_year.xaxis.major_label_orientation = 1.2
487     p_year.x_range.range_padding = 0.05
488     p_year.legend.location = "top_left"
489     p_year.legend.orientation = "horizontal"
490
491     #set select menu
492     select = Select(title="choose year", value=year_list[0],
options=year_list)
493
494     #set callback function when click select menu
495     def callback_select(attr, old, new):
496         year = select.value
497         counts_list = []
498         display_data_dict = year2data[year]
499         for areas in areas_list:
500             counts_list.append(display_data_dict[areas])
501
502         p_vbar.data_source.data['counts_list'] = counts_list
503
504
505     select.on_change('value', callback_select)
506     select.width = 100
507
508     return p_year, select
509
510
511
512
513 def plot_multi_stackvbar():

```

```

514     wb = openpyxl.load_workbook('data/LTIM reason (1).xlsx') # Import
datasets
515     ws = wb.get_sheet_by_name('Data')
516
517     #append data into lists
518     year2data = {}
519     years = []
520     index = ['work definite job', 'work looking for a job', 'accompany',
'formal study', 'other']
521     for row in range(4, 66):
522         if ws.cell(row=row, column=2).value:
523             year = str(ws.cell(row = row, column = 1).value)[:4]
524             if year not in years:
525                 years.append(year)
526             data_dict = year2data.get(year, {'In': [0,0,0,0,0], 'Out':
[0,0,0,0,0], 'Net': [0,0,0,0,0]})
527             data_in = data_dict['In']
528             data_out = data_dict['Out']
529             data_net = data_dict['Net']
530             ws_in = [ws.cell(row = row, column = 2).value, ws.cell(row =
row, column = 6).value,
531                     ws.cell(row=row, column=10).value, ws.cell(row =
row, column = 14).value,
532                     ws.cell(row=row, column=18).value]
533             ws_out = [ws.cell(row=row, column=3).value, ws.cell(row=row,
column=7).value,
534                     ws.cell(row=row, column=11).value, ws.cell(row=row,
column=15).value,
535                     ws.cell(row=row, column=19).value]
536             ws_net = [ws_in[0]-ws_out[0], ws_in[1]-ws_out[1], ws_in[2]-
ws_out[2], ws_in[3]-ws_out[3], ws_in[4]-ws_out[4]]
537
538             total_in = [data_in[0]+ws_in[0], data_in[1]+ws_in[1],
data_in[2]+ws_in[2], data_in[3]+ws_in[3], data_in[4]+ws_in[4]]
539             total_out = [data_out[0]+ws_out[0], data_out[1]+ws_out[1],
data_out[2]+ws_out[2], data_out[3]+ws_out[3], data_out[4]+ws_out[4]]
540             total_net = [data_net[0]+ws_net[0], data_net[1]+ws_net[1],
data_net[2]+ws_net[2], data_net[3]+ws_net[3], data_net[4]+ws_net[4]]
541
542             data_dict['In'] = total_in
543             data_dict['Out'] = total_out
544             data_dict['Net'] = total_net
545             year2data[year] = data_dict
546
547             #arrange data into new form
548
549             default_year = years[0]
550             data_dict = year2data[default_year]
551             df = pd.DataFrame(data_dict, index=index)
552             x_index = df.index.tolist()
553             type_work = df.columns.tolist()
554
555             data = {'index': x_index}
556             for type in type_work:
557                 data[type] = df[type].tolist()
558             print(data)
559
560             source = ColumnDataSource(data=data)
561
562             #Plot new figure

```

```

563     p = figure(plot_width=1000, plot_height=500, y_axis_label="Migration
population", x_range=x_index,
564                 y_range=(0, 1000),
tools="hover,pan,box_zoom,save,reset,undo,zoom_in,zoom_out,wheel_zoom",
title="reason for migration")
565
566     p_vbar_in = p.vbar(x=dodge('index', -0.25, range=p.x_range), top='In',
width=0.2, source=source, color="#ffff99", legend=value("In"))
567     p_vbar_out = p.vbar(x=dodge('index', 0.0, range=p.x_range), top='Out',
width=0.2, source=source, color="#b15928", legend=value("Out"))
568     p_vbar_net = p.vbar(x=dodge('index', 0.25, range=p.x_range), top='Net',
width=0.2, source=source, color="#e41a1c", legend=value("Net"))
569
570     p.legend.location = "top_left"
571     p.legend.orientation = "horizontal"
572
573     menu = years
574     dropdown = Dropdown(label=default_year, menu=menu) #Set new dropdown menu
575     def callback_dropdown(attr, old, new): #set call back function of
dropdown menu
576         year = dropdown.value
577         dropdown.label = year
578         display_data_dict = year2data[year]
579         p_vbar_in.data_source.data['In'] = display_data_dict['In']
580         p_vbar_out.data_source.data['Out'] = display_data_dict['Out']
581         p_vbar_net.data_source.data['Net'] = display_data_dict['Net']
582
583         dropdown.label = year
584
585     dropdown.on_change('value', callback_dropdown)
586     dropdown.width = 100
587     dropdown.height = 30
588
589     return p, dropdown
590
591
592 #set callback functions of interfaces switch buttons
593 def callback1():
594     layout_1.visible = True
595     layout_2.visible = False
596     layout_3.visible = False
597
598 def callback2():
599     layout_1.visible = False
600     layout_2.visible = True
601     layout_3.visible = False
602
603 def callback3():
604     layout_1.visible = False
605     layout_2.visible = False
606     layout_3.visible = True
607
608 #set interfaces switch buttons
609 button_1 = Button(label="short-term migration(London areas)")
610 button_2 = Button(label="migration vs employment(UK & London)")
611 button_3 = Button(label="reason for migration(UK)")
612 button_1.on_click(callback1)
613 button_2.on_click(callback2)
614 button_3.on_click(callback3)
615
616 p_year, select = plot_shortterm_vbar()

```

```
617 bt_row = row(button_1, button_2, button_3)
618 row_year = row(p_year, select)
619 row_1 =
    row(plot_linechart_London_short_term_migration(),plot_long_term_migration())
620 row_2 = row(plot_underemployment_rate(),plot_employment_population())
621
622 p_reason_mig, dropdown = plot_multi_stackvbar()
623 row_reason_mig = row(p_reason_mig, dropdown)
624
625 #set layout
626 bt_layout = layout(bt_row)
627 layout_1 = layout(row_year, plot_linechart_areas_short_term_migration())
628 layout_2 = layout(row_1,row_2)
629 layout_3 = layout(row_reason_mig)
630 layouts = layout(bt_layout, layout_1, layout_2, layout_3)
631
632 #show(layouts)
633 curdoc().add_root(layouts)
634
635
636 # python -m bokeh serve --show web.py
637 # By python 2.7
638
639
```