The Book of Dash

Build Dashboards with Python and Plotly

by Adam Schroeder, Christian Mayer, & Ann Marie Ward

Errata updated to print 1

Page	Error	Correction	Print corrected

24	`import pandas as pd`
	
	`import pandas as pd`
		`import pandas as pd`
		
		`app = Dash(__name__, external_stylesheets=external_stylesheets)`
			Pending

27	**NOTE** Before we begin, you'll need to have **GitHub** installed.	**NOTE** Before we begin, you'll need to have **Git** installed.	Pending
37	This will print the data value in the third row (with index 2) and the second column (with index 1), which `'Carl'`.	This will print the data value in the third row (with index 2) and the second column (with index 1), which **is** `'Carl'`.	Pending
38	`Name: Cardio, dtype: bool`	`Name: cardio, dtype: bool`	Pending

38	`0 Alice`
	`1 Bob`
	`2 Carl`
	`Name: name, dtype: object`
	`'''`
		`0 Alice`
		`1 Bob`
		`2 Carl`
		`Name: name, dtype: object`
			Pending

39	`df['age'] = 16` `print(s)`	`df['age'] = 16` `print(df)`	Pending
45	URL replacement	You can download the data with the book's resources at *https://github.com/DashBookProject/Plotly-Dash*.	Pending
49	`df["date_time"] = pd.to_datetime(df["date_time"])`	`df["date_time"] = pd.to_datetime(df["date_time"], dayfirst=True)`	Pending

Page	Error	Correction	Print corrected

57	The following line of code in Python
	`<h1> Twitter Likes Analysis of Famous People </h1>`
	is roughly equivalent to the following line of HTML that is read by a web browser:
	`html.H1("Twitter Likes Analysis of Famous People")`
		The following line of code in Python
		`html.H1("Twitter Likes Analysis of Famous People")`
		is roughly equivalent to the following line of HTML that is read by a web browser:
		`<h1> Twitter Likes Analysis of Famous People </h1>`
			Pending

59	We can then insert `fig` into `dcc.Graph`'s figure prop to display the line chart on the page. Listing 4-13 shows **the code from the *twitter_app.py* file that does just that, assigned to `app.layout`.**	We can then insert `fig` into `dcc.Graph`'s figure prop to display the line chart on the page. Listing 4-13 shows **what the code would have looked like if the line chart was created above the layout. In the *twitter_app.py* file, the code is slightly different (`figure={}`) because the figure is built inside the callback and then returned to the layout.**	Pending
74	We'll also clean the data, taking out corrupt values, **taking out corrupt values,** extracting only the data we need, and merging it with another DataFrame to get the missing values.	We'll also clean the data, taking out corrupt values, extracting only the data we need, and merging it with another DataFrame to get the missing values.	Pending
99	New to this app are the `data_table`, `State`, `callback_context`, and `plotly.graph_objects` modules.	New to this app are the `dash_table`, `State`, `callback_context`, and `plotly.graph_objects` modules.	Pending
99	We use `data_table` to display results and source data.	We use `dash_table` to display results and source data.	Pending
106	Listing 6-4 shows the code for the Card component shown in Figure 6-**6**.	Listing 6-4 shows the code for the Card component shown in Figure 6-**8**.	Pending
116	`dtict=dict` is used to set the step between the labels on the x-axis.	`dtick=dtick` is used to set the step between the labels on the x-axis.	Pending
145	The function `serve_prediction_plot()` creates the Plotly graph object used to visualize the training and testing data and the contour plot (see Figure 7-**10**).	The function `serve_prediction_plot()` creates the Plotly graph object used to visualize the training and testing data and the contour plot (see Figure 7-**9**).	Pending
146	This code skeleton shows how we create the contour plots shown in Figure 7-**10** that visualize the SVM confidence levels, as well as the two scatter plots for the training and test data.	This code skeleton shows how we create the contour plots shown in Figure 7-**9** that visualize the SVM confidence levels, as well as the two scatter plots for the training and test data.	Pending
169	`X = 1 > 2`	`x = 1 > 2`	Pending
169	The keyword `not` evaluates to `True` in the expression `not x` when `x` is `False`.	The keyword `not` evaluates to `True` in the expression `not y` when `y` is `False`.	Pending

Page	Error	Correction	Print corrected
169			Pending
171	Deletion	~~The string method: `str(yes) == 'yes'` is `True`~~	Pending
172			Pending