

[**Data Visualization with Python**](https://www.coursera.org/learn/python-for-data-visualization)

**Cheat Sheet : Maps, Waffles, WordCloud and Seaborn**

| **Function** | **Description** | **Syntax** | **Example** | **Visual** |
| --- | --- | --- | --- | --- |
| Folium |  |  |  |  |
| **Map** | Create a map object with specified center coordinates and zoom level. | folium.Map(location=[lat, lon], zoom\_start=n) | world\_map = folium.Map()  canada =folium.Map(location=[56.130, -106.35], zoom\_start=4) |  |
| **Marker** | Add a marker to the map with custom icon, popup, and tiles  Tiles as Stamen Toner | folium.Marker(location=[lat , lon ], popup='Marker Popup', tiles='Stamen Toner').add\_to(map) | folium.Marker(location=[556.130, -106.35],  tooltip='Marker', tiles='Stamen Toner').add\_to(world\_map) |  |
|  | Tiles as Stamen Terrain | folium.Marker(location=[lat , lon ],  popup='Marker Popup', tiles='Stamen Terrain').add\_to(map) | folium.Marker(location=[556.130, -106.35],  tooltip='Marker', tiles='Stamen Terrain').add\_to(world\_map) |  |
| **Circle** | Add a circle to the map with specified radius, color, and fill opacity. | folium.features.CircleMarker(location=[lat, lon],  radius=n, color='red',  fill\_opacity=n).add\_to(map) | folium.features.CircleMarker(location=[56.130, -106.35],  radius=1000, color='red',  fill\_opacity=0.5).add\_to(world\_map) |  |
| **Chorpleth** | Create a choropleth map based on a GeoJSON file and a specified data column. | folium.Choropleth(geo\_data='path/to/geojson\_file', data=df, columns=['region', 'value\_column'], key\_on='feature.properties.id', fill\_color='YlGnBu',  fill\_opacity=0.7, line\_opacity=0.2,  legend\_name='Legend').add\_to(map) | world\_map.choropleth(geo\_data=world\_geo,  data=df\_can, columns=['Country', 'Total'], key\_on='feature.properties.name', fill\_color='YlOrRd', fill\_opacity=0.7,line\_opacity=0.2, legend\_name='Immigration to Canada') |  |
| PyWaffle |  |  |  |  |
| **Waffle** | Create a waffle chart based on values and categories. | plt.figure(FigureClass = Waffle,rows = 20, columns = 30, values = values)  waffle\_chart = waffle.Waffle(values=[value1, value2, ...],  rows=n, columns=n) | plt.figure(FigureClass = Waffle,rows = 20, columns = 30,  values = df\_dsn['Total'], cmap\_name = 'tab20', legend = {'labels': label,'loc': 'lower left', 'bbox\_to\_anchor':(0,-0.1),'ncol': 3}) |  |
| **Legend** | Add a legend to the waffle chart. | waffle\_chart.legend(loc='upper left', bbox\_to\_anchor=(1, 1)) |  |  |
| **Title** | Add a title to the waffle chart. | waffle\_chart.set\_title('Waffle Chart Title') |  |  |
| **Labels** | Add labels to the waffle chart. | waffle\_chart.set\_labels(['Label 1', 'Label 2', ...]) |  |  |
| WordCloud |  |  |  |  |
| **WordCloud** | Create a word cloud object based on text data. | wordcloud = WordCloud().generate(text\_data) | alice\_wc = WordCloud(background\_color='white', max\_words=2000, mask=alice\_mask, stopwords=stopwords) alice\_wc.generate(alice\_novel) plt.imshow(alice\_wc, interpolation='bilinear') |  |
| **Generate** | Generate the word cloud based on the text data. | wordcloud.generate(text\_data) |  |  |
| **Display** | Display the word cloud using matplotlib or other plotting libraries. | plt.imshow(wordcloud, interpolation='bilinear') |  |  |
| **Options** | Set various options for the word cloud, such as font, colors, mask, and stopwords. | wordcloud = WordCloud(font\_path='path/to/font\_file', background\_color='white', colormap='Blues', mask=mask\_image,  stopwords=stopwords).generate(text\_data) |  |  |
| Seaborn |  |  |  |  |
| **barplot** | Create a bar plot to visualize the relationship between a categorical variable and a numeric variable. | sns.barplot(x='x\_variable', y='y\_variable', data=dataframe) | sns.barplot(x='Continent', y='Total', data=df\_can1) |  |
| **countplot** | Create a count plot to display the frequency of each category in a categorical variable. | sns.countplot(x='category', data=dataframe) | sns.countplot(x='Continent', data=df\_can) |  |
| **regplot** | Create a scatter plot with a linear regression line to visualize the relationship between two numeric variables. | sns.regplot(x='x\_variable',  y='y\_variable', data=dataframe) | sns.regplot(x='year', y='total', data=df\_tot) |  |

**Author(s)**

Dr. Pooja

**Changelog**

| **Date** | **Version** | **Changed by** | **Change Description** |
| --- | --- | --- | --- |
| 2023-06-18 | 0.1 | Dr. Pooja | Initial version created |