LEARNING OUTCOMES

By the end of this section, you should be able to:

9.4.1 Describe the insights produced by spatial heatmaps based on geospatial data.
9.4.2 Discuss the common features of GIS mapping.
9.4.3 Use a GIS mapper to produce a visualization of geospatial data.

Geospatial data refers to data that describes the geographic location, shape, size, and other attributes relative to a location on the Earth's surface. Geospatial data can be captured, stored, manipulated, analyzed, and visualized using various technologies and tools.

Geospatial data visualization using Python involves the representation and analysis of data that has a geographic component, such as latitude and longitude coordinates. Python offers several libraries and utilities for geospatial data visualization and analysis, including the `Pandas` library discussed in What Are Data and Data Science? `Pandas` is a Python library specialized for data manipulation and analysis. In a similar way, `Geopandas` extends the capabilities of `Pandas` to handle geospatial data. The `Geopandas` library provides the ability to read, write, analyze, and visualize geospatial data. `Geopandas` provides a GeoDataFrame object, which is similar to a `Pandas` DataFrame but includes geometry information for spatial operations.

Spatial and Grid Heatmaps

Spatial heatmaps are a data visualization method used to represent the density or intensity of data points within a geographical area using coloring and shading to represent densities of various attributes. These heatmaps provide a visual summary of the distribution and concentration of data points, highlighting areas of high and low density.

For example, a spatial heatmap of New York City might show crime rate information where higher crime rates in a certain location are represented by darker shades of red and lower crime rates are represented by lighter shades of red. The reader can quickly ascertain the differences in crime rates among various neighborhoods based on the shading in the heatmap.

Figure 9.6 provides an example of a heatmap showing differences in average temperature for January 2024 as compared to previous temperature data from 1991–2020 where the coloring scheme represents the differences (blue represents a reduction in temperature and red represents an increase in temperature).

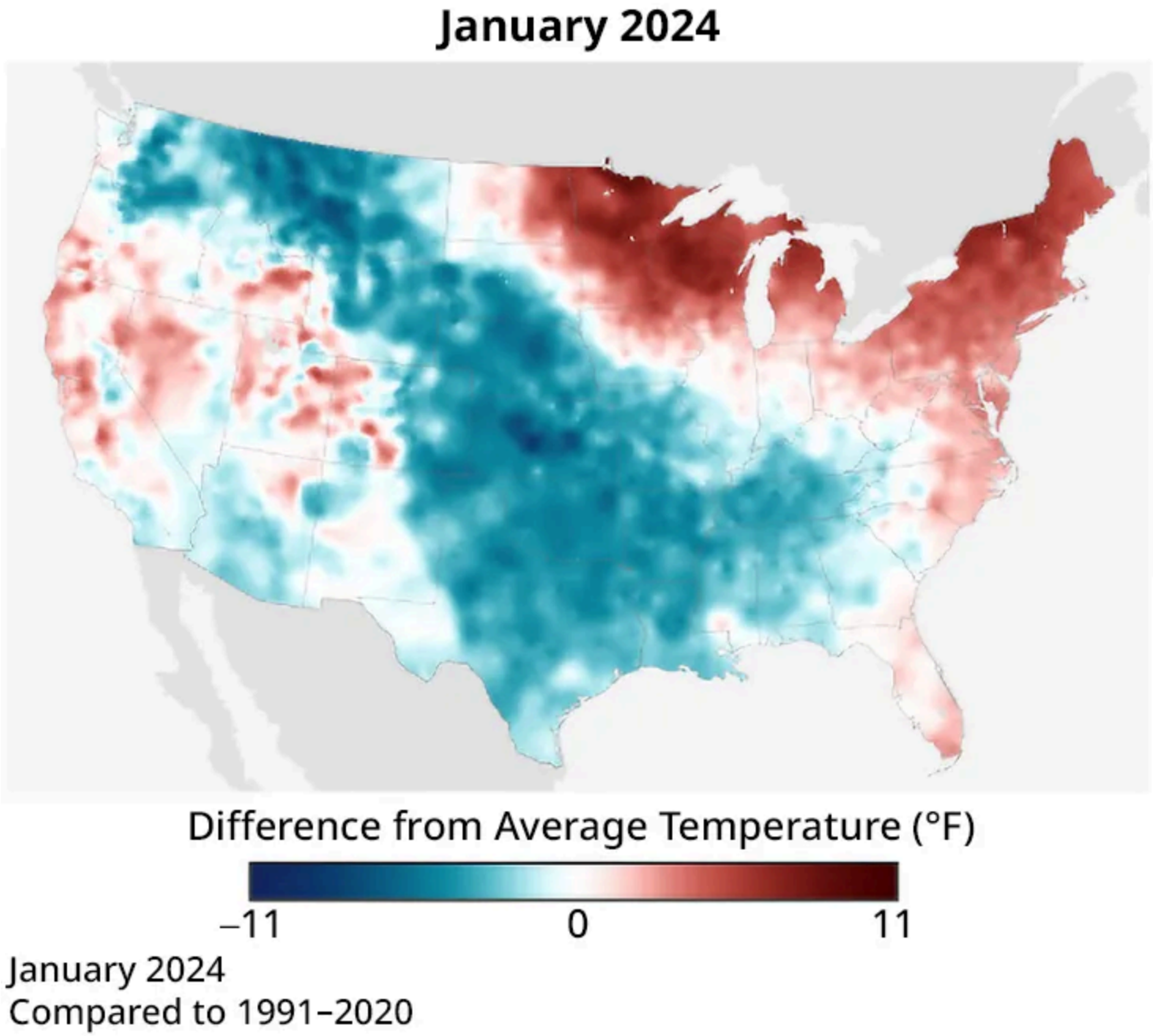

Figure 9.6 Example of Heatmap Showing Differences from Average Temperature for January 2024 by Colors (data source: NOAA Climate.gov maps based on data from NOAA National Centers for Environmental Information. [NCEI] https://www.climate.gov/media/15930, accessed June 23, 2024.)

Spatial heatmaps are generated by condensing the spatial coordinates of individual data points into a grid or a set of bins covering the geographic area of interest. The density of data points within each grid cell or bin is then calculated, and these density results are then mapped to specific colors using a color gradient map. Geographic areas with higher density values are assigned colors that represent higher intensity, while areas with lower density values are assigned colors that represent lower intensity. Finally, the colored grid cells can be overlaid on a map, and the result is a visual representation of the spatial distribution of data points.

Grid heatmaps display colors in a two-dimensional array where the x-axis and *y*-axis represent two differing characteristics and the color coding for a certain cell is based on the combined characteristics for that cell. For example, in Figure 9.7, a matrix is created where the *x*-axis represents different local

farmers and the *y*-axis represents harvest for various crops (in tons/year), and then each cell is coded based on the harvest value for that combination of farmer and specific crop. At a glance, the viewer can discern that the highest harvests appear to be for potatoes at BioGoods Ltd and for barley at Cornylee Corp. Note that in a grid heatmap, the grids are typically based on a fixed size since the intent is to detect clustering for the characteristics of interest. Also note that the color scheme used is where green and yellow shading represent higher harvest levels and blue and purple represent lower harvest levels.

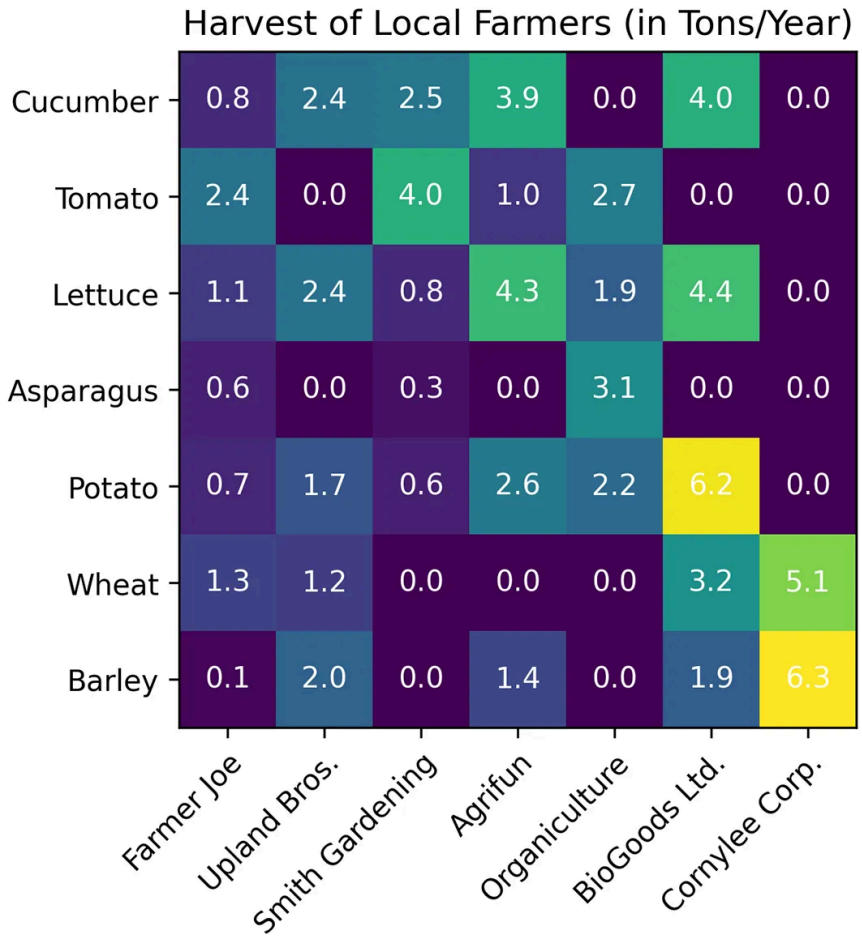

Figure 9.7 Grid Heatmap Showing Color Density for Crop Harvest by Local Farmer and Type of Crop
(source: created by author using Matplotlib by D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007. DOI: https://zenodo.org/records/13308876)

EXPLORING FURTHER

Interactive Heatmaps

Various interactive heatmaps are available that allow the user to preselect variables of interest and then view the corresponding geospatial map or heatmap. For example, see these

interactive heatmaps utilizing crash and demographic data as well as traffic safety improvement activities in California:

https://catsip.berkeley.edu/resources/crash-data/crash-data-tools-and-resources

https://safetyheatmap.berkeley.edu/map///

Using Python to Generate Heatmaps

There are several ways to generate heatmaps in Python. Two common methods include the following:

- `imshow()` function, which is part of `MatPlotlib` plotting library. The `imshow()` function can easily display heatmaps such as the two-dimensional heatmap shown in Figure 9.7.
- `heatmap()` function, which is part of the Seaborn library. Seaborn provides a high-level capability for statistical graphs.

EXPLORING FURTHER

Seaborn

Seaborn includes a wide variety of functions for visualizing statistical relationships. This user guide and tutorial provides an excellent introduction to its uses.

In the next example, Python is used to create a heatmap based on a heatmap function called `sns.heatmap()`, which is part of the Seaborn library. The example plots number of airline passengers for time-based data of month and year where month is plotted on the horizontal axis, year is plotted on the vertical axis, and the color coding of the heatmap represents the magnitude of the number of airline passengers.

EXAMPLE 9.10

Problem

Generate a heatmap of a dataset using the heatmap function that is part of the Seaborn library in Python.

[Show/Hide Solution]

Introduction to Geospatial Information System (GIS) Mapping

Geospatial Information System (GIS) mapping is a tool for visualizing, analyzing, and interpreting spatial data that makes use of various types of geographical data, such as maps and satellite images. The goal is to allow the user to visualize patterns and trends based on geographic attributes.

It is likely that you have already come across a heatmap overlaid on a geographic map in a news article or website. Heatmaps such as these are often referred to as a **choropleth graph**, which refers to the graphing of data according to geographic boundaries such as states, counties, etc. A choropleth graph is created by overlaying a grid on a geographic map and then typically coloring the map based on density or some other measurement for a specific grid location on the map. Choropleth graphs are commonly used to plot data overlaid on geographic maps for economic data, population data, housing data, medical-related data, etc.

These graphs provide a very natural and intuitive way to visualize data for certain geographic regions and thus identify patterns, trends, and correlations. These types of maps also allow the user to identify clusters and outliers to assist in data interpretation.

For example, a medical researcher might be interested in visualizing and tracking the spread of a virus over time across various counties, or a job seeker might be interested in visualizing the density of specific job openings in various cities.

Here are some other applications where the use of GIS mapping provides visualizations and insights, trends, and correlations:

- Telecommunications services and coverage
- Urban and transportation planning
- Environmental management
- Health care applications
- Disaster assessment and recovery
- Supply chain management

EXPLORING FURTHER

Interactive COVID-19 Choropleth Graph

Figure 9.8 provides an example of a choropleth graph for COVID-19 cases overlaid on a map of the world. Notice how higher numbers of cases of COVID-19 correlate to darker color shading on the map. Click on the “play” button in this interactive time-lapse map to see the change in time basis for the graph. Note that the time scale on the bottom of the graph can be used to show changes in data over time.

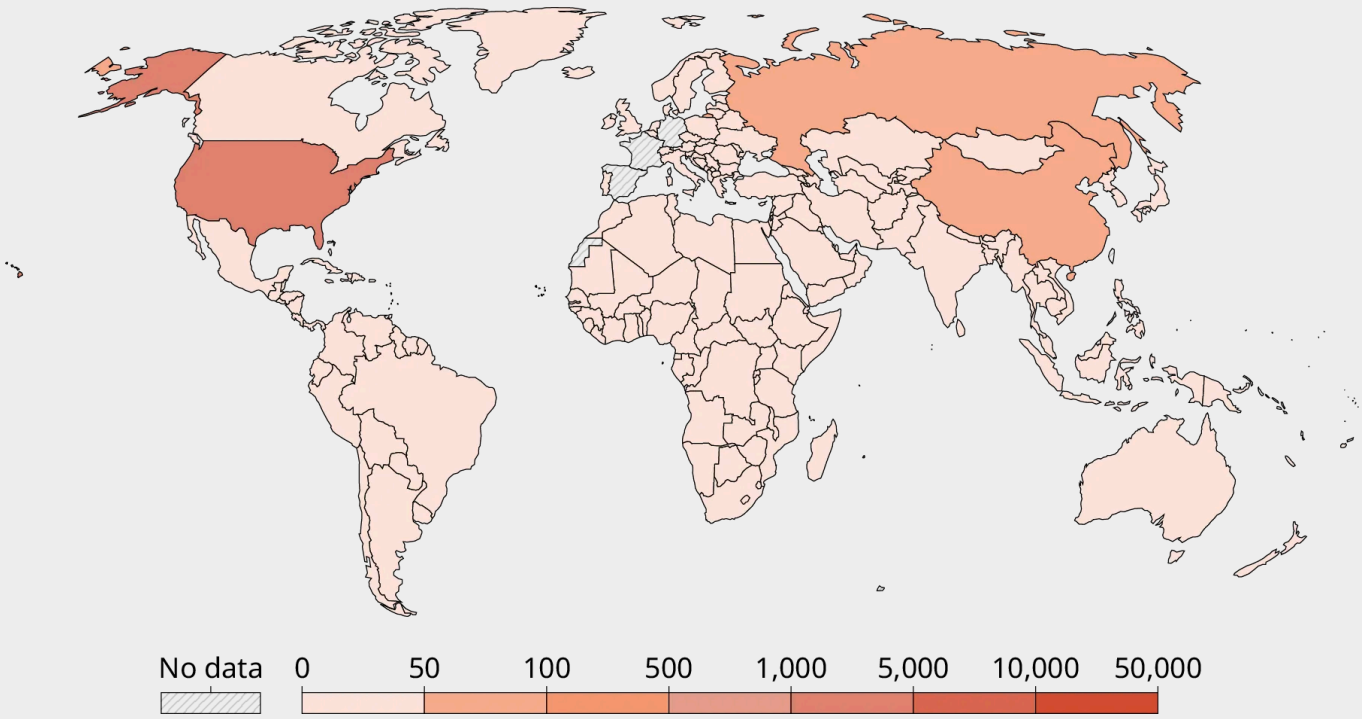

Figure 9.8 Choropleth Graph Showing Cases of COVID-19

(data source: Data from WHO COVID-19 Dashboard. Geneva: World Health Organization, 2020. Available online: https://data.who.int/dashboards/covid19/" via OurWorldinData.org. Retrieved from: https://ourworldindata.org/grapher/weekly-covid-deaths [Online Resource].)

Using Python for Geographic Mapping

As mentioned earlier, Python provides a number of libraries and tools to facilitate the generation of geographic mapping. There are several steps involved in the process:

1. Establish and collect geospatial data: The first step is to collect the geospatial data of interest. This type of data is readily available from a number of sources such as open data repositories, government agencies such as the Census Bureau, for example, etc. The data can be found in various formats such as shapefiles, GeoJSON files, etc. (a shapefile is a data format used for geospatial data that represents geographic data in a vector format).
2. Data processing: Once the data is collected, data processing or transformation may be needed, as discussed in Collecting and Preparing Data. For example, some form of coding might be needed to transform addresses into geographic coordinates such as latitude and longitude to facilitate the geographic mapping. Additional data processing might be needed to estimate or interpolate missing data to generate continuous temperature maps, for example.
3. Generate the map to visualize the geographic data using libraries such as `matplotlib` and `geopandas`. Various tools also provide interactivity to the map so that the user can zoom in, zoom out, rotate, or view data when hovering over a map feature.

Python provides an excellent environment to allow the user to create, manipulate, and share geographic data to help visualize geospatial data and detect trends or arrive at conclusions based on the data analysis.