Charting cellphone data

Exercise

We know that Freddy Frequentist is the one who kidnapped Bayes the Golden Retriever. Now we need to learn where he is hiding.

Our friends at the police station have acquired cell phone data, which gives some of Freddie's locations over the past three weeks. It's stored in the DataFrame cellphone. The x-coordinates are in the column 'x' and the y-coordinates are in the column 'y'.

The matplotlib module has been imported under the alias plt.

Instructions

1-Display the first five rows of the DataFrame and determine which columns to plot

2-Create a scatter plot of the data in cellphone

In [ ]:

```
# Explore the data
print(cellphone.____)

# Create a scatter plot of the data from the DataFrame cellphone
plt.____(____, ____)

# Add labels
plt.ylabel('Latitude')
plt.xlabel('Longitude')

# Display the plot
plt.show()

#________________________#
#Solutions

# Explore the data
print(cellphone.head())

# Create a scatter plot of the data from the DataFrame cellphone
plt.scatter(cellphone.x,cellphone.y)

# Add labels
plt.ylabel('Latitude')
plt.xlabel('Longitude')

# Display the plot
plt.show()
```

Modifying a scatterplot

Exercise

In the previous exercise, we created a scatter plot to show Freddy Frequentist's cell phone data.

In this exercise, we've done some magic so that the plot will appear over a map of our town. If we just plot the data as we did before, we won't be able to see the map or pick out the areas with the most points. We can fix this by changing the colors, markers, and transparency of the scatter plot.

As before, the matplotlib module has been imported under the alias plt, and the cellphone data is in the DataFrame cellphone.

Instructions

1-Change the color of the points to 'red'

2-Change the marker shape to square.

In [ ]:

```
# Change the marker color to red
plt.scatter(cellphone.x, cellphone.y, ____=____)

# Change the marker shape to square
plt.scatter(cellphone.x, cellphone.y,color='red',____=____)

# Change the transparency to 0.1
plt.scatter(cellphone.x, cellphone.y,color='red',marker='s',____=____)

# Add labels
plt.ylabel('Latitude')
plt.xlabel('Longitude')

# Display the plot
plt.show()

#____________________________#
#Solutions

# Change the marker color to red
plt.scatter(cellphone.x, cellphone.y, color='red')

# Change the marker shape to square
plt.scatter(cellphone.x, cellphone.y,color='red',marker='s')

# Change the transparency to 0.1
plt.scatter(cellphone.x, cellphone.y,color='red',marker='s',alpha=0.1)


# Add labels
plt.ylabel('Latitude')
plt.xlabel('Longitude')

# Display the plot
plt.show()
```

Build a simple bar chart

Exercise

Officer Deshaun wants to plot the average number of hours worked per week for him and his coworkers. He has stored the hours worked in a DataFrame called hours, which has columns officer and avg_hours_worked. Recall that the function plt.bar() takes two arguments: the labels for each bar, and the height of each bar. Both of these can be found in our DataFrame.

Instructions

1-Display the DataFrame hours using a print command.

2-Create a bar chart of the column avg_hours_worked for each officer from the DataFrame hours.

3-Use the column std_hours_worked (the standard deviation of the hours worked) to add error bars to the bar chart.

In [ ]:

```
# Display the DataFrame hours using print
print(____)

# Create a bar plot from the DataFrame hours
plt.____(____, ____)

# Create a bar plot from the DataFrame hours
plt.bar(hours.officer, hours.avg_hours_worked,
        # Add error bars
        ____=____)

# Display the plot
plt.show()


#______________________#
#Solutions

# Display the DataFrame hours using print
print(hours)

# Create a bar plot from the DataFrame hours
plt.bar(hours.officer, hours.avg_hours_worked)

# Create a bar plot from the DataFrame hours
plt.bar(hours.officer, hours.avg_hours_worked,
        # Add error bars
        yerr=hours.std_hours_worked)

# Display the plot
plt.show()
```

Where did the time go?

Exercise

Officer Deshaun wants to compare the hours spent on field work and desk work between him and his colleagues. In this DataFrame, he has split out the average hours worked per week into desk_work and field_work.

You can use the same DataFrame containing the hours worked from the previous exercise (hours).

Instructions

1-Create a bar plot of the time each officer spends on desk_work. Label that bar plot "Desk Work"

2-Create a bar plot for field_work whose bottom is the height of desk_work.

3-Label the field_work bars as "Field Work" and add a legend.

In [ ]:

```
# Plot the number of hours spent on desk work
____

# Plot the hours spent on field work on top of desk work
____

# Add a legend
____

# Display the plot
plt.show()

#_________________________________________#
#Solutions

# Plot the number of hours spent on desk work
plt.bar(hours.officer, hours.desk_work,label='Desk Work')

# Plot the hours spent on field work on top of desk work
plt.bar(hours.officer, hours.field_work,bottom=hours.desk_work ,label='Field Work')

# Add a legend
plt.legend()

# Display the plot
plt.show()
```

Modifying histograms

Exercise

Let's explore how changes to keyword parameters in a histogram can change the output. Recall that:

range sets the minimum and maximum datapoints that we will include in our histogram.

bins sets the number of points in our histogram.

We'll be exploring the weights of various puppies from the DataFrame puppies. matplotlib has been loaded under the alias plt

Instructions

1-Create a histogram of the column weight from the DataFrame puppies.

2-Change the number of bins to 50.

3-Change the range to start at 5 and end at 35.

In [ ]:

```
#Solutions

# Create a histogram of the column weight
# from the DataFrame puppies
plt.hist(puppies.weight,  bins=50)

# Change the range to start at 5 and end at 35
plt.hist(puppies.weight,range=(5, 35))

# Add labels
plt.xlabel('Puppy Weight (lbs)')
plt.ylabel('Number of Puppies')

# Display
plt.show()
```

Heroes with histograms

Exercise

We've identified that the kidnapper is Fred Frequentist. Now we need to know where Fred is hiding Bayes.

A shoe print at the crime scene contains a specific type of gravel. Based on the distribution of gravel radii, we can determine where the kidnapper recently visited. It might be:

blue-meadows-park

shady-groves-campsite

happy-mountain-trailhead

The radii of individual gravel pieces has been loaded into the DataFrame gravel, and matplotlib has been loaded under the alias plt.

Instructions

1-Create a histogram of gravel.radius.

2-Modify the histogram such that the histogram is divided into 40 bins and the range is from 2 to 8.

3-Normalize your histogram so that the sum of the bins adds to 1.

4-Label the x-axis (Gravel Radius (mm)), the y-axis (Frequency), and the title(Sample from Shoeprint).

In [ ]:

```
#Solutions

# Create a histogram of gravel.radius
plt.hist(gravel.radius)

# Range is 2 to 8, with 40 bins
plt.hist(gravel.radius, bins=40, range=(2,8))

# Normalize to 1
plt.hist(gravel.radius, bins=40, range=(2, 8), density=True)

# Label plot
plt.xlabel('Gravel Radius (mm)')
plt.ylabel('Frequency')
plt.title('Sample from Shoeprint')

# Display histogram
plt.show()
```