Activity: Explore probability distributions

Introduction ¶

The ability to determine which type of probability distribution best fits data, calculate z-score, and detect outliers are essential skills in data work. These capabilities enable data professionals to understand how their data is distributed and identify data points that need further examination.

In this activity, you are a member of an analytics team for the United States Environmental Protection Agency (EPA). The data includes information about more than 200 sites, identified by state, county, city, and local site names. One of your main goals is to determine which regions need support to make air quality improvements. Given that carbon monoxide is a major air pollutant, you will investigate data from the Air Quality Index (AQI) with respect to carbon monoxide.

Step 1: Imports

Import relevant libraries, packages, and modules. For this lab, you will need `numpy` , `pandas` , `matplotlib.pyplot` , `statsmodels.api` , and `scipy` .

In [1]:
```
# Import relevant libraries, packages, and modules
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.api as sm
from scipy import stats
```

A subset of data was taken from the air quality data collected by the EPA, then transformed to suit the purposes of this lab. This subset is a .csv file named `modified_c4_epa_air_quality.csv` . As shown in this cell, the dataset has been automatically loaded in for you. You do not need to download the .csv file, or provide more code, in order to access the dataset and proceed with this lab. Please continue with this activity by completing the following instructions.

In [3]:
```
# RUN THIS CELL TO IMPORT YOUR DATA.

### YOUR CODE HERE ###
data = pd.read_csv("modified_c4_epa_air_quality.csv")
# Display the first few rows of the dataset to understand its structure
data.head()
```

Out[3]:

	date_local	state_name	county_name	city_name	local_site_name	parameter_name	unit:
0	2018-01-01	Arizona	Maricopa	Buckeye	BUCKEYE	Carbon monoxide	P
1	2018-01-01	Ohio	Belmont	Shadyside	Shadyside	Carbon monoxide	P
2	2018-01-01	Wyoming	Teton	Not in a city	Yellowstone National Park - Old Faithful Snow ...	Carbon monoxide	P
3	2018-01-01	Pennsylvania	Philadelphia	Philadelphia	North East Waste (NEW)	Carbon monoxide	P
4	2018-01-01	Iowa	Polk	Des Moines	CARPENTER	Carbon monoxide	P

Hint 1

Hint 2

Hint 3

Step 2: Data exploration

Display the first 10 rows of the data to get a sense of how the data is structured.

In [4]:
```
# Display first 10 rows of the data.

### YOUR CODE HERE ###
# Display first 10 rows of the data
data.head(10)
```

Out[4]:

	date_local	state_name	county_name	city_name	local_site_name	parameter_name	unit
0	2018-01-01	Arizona	Maricopa	Buckeye	BUCKEYE	Carbon monoxide	P
1	2018-01-01	Ohio	Belmont	Shadyside	Shadyside	Carbon monoxide	P
2	2018-01-01	Wyoming	Teton	Not in a city	Yellowstone National Park - Old Faithful Snow ...	Carbon monoxide	P
3	2018-01-01	Pennsylvania	Philadelphia	Philadelphia	North East Waste (NEW)	Carbon monoxide	P
4	2018-01-01	Iowa	Polk	Des Moines	CARPENTER	Carbon monoxide	P
5	2018-01-01	Hawaii	Honolulu	Not in a city	Kapolei	Carbon monoxide	P
6	2018-01-01	Hawaii	Honolulu	Not in a city	Kapolei	Carbon monoxide	P
7	2018-01-01	Pennsylvania	Erie	Erie	NaN	Carbon monoxide	P
8	2018-01-01	Hawaii	Honolulu	Honolulu	Honolulu	Carbon monoxide	P
9	2018-01-01	Colorado	Larimer	Fort Collins	Fort Collins - CSU - S. Mason	Carbon monoxide	P

Hint 1

Hint 2

Hint 3

The `aqi_log` column represents AQI readings that were transformed logarithmically to suit the objectives of this lab. Taking a logarithm of the aqi to get a bell-shaped distribution is outside the scope of this course, but is helpful to see the normal distribution.

To better understand the quantity of data you are working with, display the number of rows and the number of columns.

In [5]:
```
# Display number of rows, number of columns.

### YOUR CODE HERE ###

# Display number of rows and columns
data.shape
```

Out[5]: (260, 8)

Hint 1

Hint 2

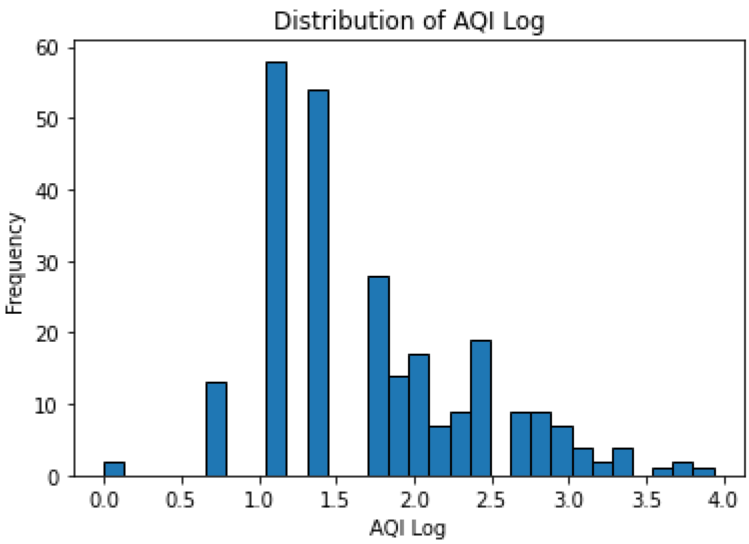
Hint 3

Now, you want to find out whether `aqi_log` fits a specific type of probability distribution. Create a histogram to visualize the distribution of `aqi_log`. Then, based on its shape, visually determine if it resembles a particular distribution.

In [6]:
```
# Create a histogram to visualize distribution of aqi_log.

### YOUR CODE HERE ###

# Create a histogram to visualize distribution of aqi_log
plt.hist(data['aqi_log'], bins=30, edgecolor='black')
plt.title('Distribution of AQI Log')
plt.xlabel('AQI Log')
plt.ylabel('Frequency')
plt.show()
```



Hint 1

Hint 2

Hint 3

Question: What do you observe about the shape of the distribution from the histogram?

[Write your response here. Double-click (or enter) to edit.]

Step 3: Statistical tests

Use the empirical rule to observe the data, then test and verify that it is normally distributed.

As you have learned, the empirical rule states that, for every normal distribution:

- 68% of the data fall within 1 standard deviation of the mean
- 95% of the data fall within 2 standard deviations of the mean
- 99.7% of the data fall within 3 standard deviations of the mean

First, define two variables to store the mean and standard deviation, respectively, for `aqi_log`. Creating these variables will help you easily access these measures as you continue with the calculations involved in applying the empirical rule.

In [7]:
```
# Define variable for aqi_log mean
mean_aqi_log = data['aqi_log'].mean()
print(f"Mean of AQI Log: {mean_aqi_log}")

# Define variable for aqi_log standard deviation
std_dev_aqi_log = data['aqi_log'].std(ddof=1)
print(f"Standard Deviation of AQI Log: {std_dev_aqi_log}")
```

```
Mean of AQI Log: 1.7669210929985577
Standard Deviation of AQI Log: 0.7147155520223721
```

In [8]:
```
# Define variable for aqi_log standard deviation
std_dev_aqi_log = data['aqi_log'].std(ddof=1)

# Print out the standard deviation
print(f"Standard Deviation of AQI Log: {std_dev_aqi_log}")
```

```
Standard Deviation of AQI Log: 0.7147155520223721
```

Hint 1

Hint 2

Hint 3

Now, check the first part of the empirical rule: whether 68% of the `aqi_log` data falls within 1 standard deviation of the mean.

To compute the actual percentage of the data that satisfies this criteria, define the lower limit (for example, 1 standard deviation below the mean) and the upper limit (for example, 1 standard deviation above the mean). This will enable you to create a range and confirm whether each value falls within it.

In [9]:
```
# Define variable for lower limit, 1 standard deviation below the mean
lower_limit = mean_aqi_log - std_dev_aqi_log

# Define variable for upper limit, 1 standard deviation above the mean
upper_limit = mean_aqi_log + std_dev_aqi_log

# Display lower_limit and upper_limit
print(f"Lower limit (1 SD below mean): {lower_limit}")
print(f"Upper limit (1 SD above mean): {upper_limit}")
```

```
Lower limit (1 SD below mean): 1.0522055409761855
Upper limit (1 SD above mean): 2.48163664502093
```

Hint 1

Hint 2

Hint 3

In [10]:
```
# Calculate the number of data points within 1 standard deviation of the me
an
within_1_sd = data[(data['aqi_log'] >= lower_limit) & (data['aqi_log'] <= u
pper_limit)]

# Calculate the percentage of data points within this range
percentage_within_1_sd = (within_1_sd.shape[0] / data.shape[0]) * 100

# Display the percentage
print(f"Percentage of data within 1 standard deviation of the mean: {percen
tage_within_1_sd:.2f}%")
```

```
Percentage of data within 1 standard deviation of the mean: 76.15%
```

Hint 1

Hint 2

Hint 3

Now, consider the second part of the empirical rule: whether 95% of the `aqi_log` data falls within 2 standard deviations of the mean.

To compute the actual percentage of the data that satisfies this criteria, define the lower limit (for example, 2 standard deviations below the mean) and the upper limit (for example, 2 standard deviations above the mean). This will enable you to create a range and confirm whether each value falls within it.

In [11]:
```
# Define variable for lower limit, 2 standard deviations below the mean
lower_limit_2sd = mean_aqi_log - 2 * std_dev_aqi_log

# Define variable for upper limit, 2 standard deviations above the mean
upper_limit_2sd = mean_aqi_log + 2 * std_dev_aqi_log

# Display lower_limit_2sd and upper_limit_2sd
print(f"Lower limit (2 SDs below mean): {lower_limit_2sd}")
print(f"Upper limit (2 SDs above mean): {upper_limit_2sd}")
```

```
Lower limit (2 SDs below mean): 0.33748998895381344
Upper limit (2 SDs above mean): 3.1963521970433018
```

Hint 1

Hint 2

Hint 3

In [12]:
```
# Display the actual percentage of data that falls within 2 standard deviat
ions of the mean.

### YOUR CODE HERE ###

# Calculate the number of data points within 2 standard deviations of the m
ean
within_2_sd = data[(data['aqi_log'] >= lower_limit_2sd) & (data['aqi_log']
<= upper_limit_2sd)]

# Calculate the percentage of data points within this range
percentage_within_2_sd = (within_2_sd.shape[0] / data.shape[0]) * 100

# Display the percentage
print(f"Percentage of data within 2 standard deviations of the mean: {perce
ntage_within_2_sd:.2f}%")
```

```
Percentage of data within 2 standard deviations of the mean: 95.77%
```

Hint 1

Hint 2

Hint 3

Now, consider the third part of the empirical rule:whether 99.7% of the `aqi_log` data falls within 3 standard deviations of the mean.

To compute the actual percentage of the data that satisfies this criteria, define the lower limit (for example, 3 standard deviations below the mean) and the upper limit (for example, 3 standard deviations above the mean). This will enable you to create a range and confirm whether each value falls within it.

In [13]:
```
# Define variable for lower limit, 3 standard deviations below the mean
lower_limit_3sd = mean_aqi_log - 3 * std_dev_aqi_log

# Define variable for upper limit, 3 standard deviations above the mean
upper_limit_3sd = mean_aqi_log + 3 * std_dev_aqi_log

# Display lower_limit_3sd and upper_limit_3sd
print(f"Lower limit (3 SDs below mean): {lower_limit_3sd}")
print(f"Upper limit (3 SDs above mean): {upper_limit_3sd}")
```

```
Lower limit (3 SDs below mean): -0.3772255630685586
Upper limit (3 SDs above mean): 3.911067749065674
```

Hint 1

Hint 2

Hint 3

In [14]:
```
# Calculate the number of data points within 3 standard deviations of the m
ean
within_3_sd = data[(data['aqi_log'] >= lower_limit_3sd) & (data['aqi_log']
<= upper_limit_3sd)]

# Calculate the percentage of data points within this range
percentage_within_3_sd = (within_3_sd.shape[0] / data.shape[0]) * 100

# Display the percentage
print(f"Percentage of data within 3 standard deviations of the mean: {perce
ntage_within_3_sd:.2f}%")
```

```
Percentage of data within 3 standard deviations of the mean: 99.62%
```

Hint 1

Hint 2

Hint 3

Step 4: Results and evaluation

Question: What results did you attain by applying the empirical rule?

[Write your response here. Double-click (or enter) to edit.]

Question: How would you use z-score to find outliers?

[Write your response here. Double-click (or enter) to edit.]

Compute the z-score for every `aqi_log` value. Then, add a column named `z_score` in the data to store those results.

In [15]:
```
# Compute the z-score for every aqi_log value
data['z_score'] = (data['aqi_log'] - mean_aqi_log) / std_dev_aqi_log

# Display the first 5 rows to ensure that the new column was added
data.head()
```

Out[15]:

	date_local	state_name	county_name	city_name	local_site_name	parameter_name	unit
0	2018-01-01	Arizona	Maricopa	Buckeye	BUCKEYE	Carbon monoxide	P
1	2018-01-01	Ohio	Belmont	Shadyside	Shadyside	Carbon monoxide	P
2	2018-01-01	Wyoming	Teton	Not in a city	Yellowstone National Park - Old Faithful Snow ...	Carbon monoxide	P
3	2018-01-01	Pennsylvania	Philadelphia	Philadelphia	North East Waste (NEW)	Carbon monoxide	P
4	2018-01-01	Iowa	Polk	Des Moines	CARPENTER	Carbon monoxide	P

Hint 1

Hint 2

Hint 3

Identify the parts of the data where `aqi_log` is above or below 3 standard deviations of the mean.

In [16]:
```
# Display data where `aqi_log` is above or below 3 standard deviations of t
he mean
outliers = data[(data['z_score'] > 3) | (data['z_score'] < -3)]
outliers
```

Out[16]:

	date_local	state_name	county_name	city_name	local_site_name	parameter_name	unit
244	2018-01-01	Arizona	Maricopa	Phoenix	WEST PHOENIX	Carbon monoxide	F

Hint 1

Hint 2

Hint 3

Question: What do you observe about potential outliers based on the calculations?

[Write your response here. Double-click (or enter) to edit.]

Question: Why is outlier detection an important part of this project?

[Write your response here. Double-click (or enter) to edit.]

Considerations

What are some key takeaways that you learned during this lab?

[Write your response here. Double-click (or enter) to edit.]

What summary would you provide to stakeholders? Consider the distribution of the data and which sites would benefit from additional research.

[Write your response here. Double-click (or enter) to edit.]

Reference

US EPA, OAR. 2014, July 8. [Air Data: Air Quality Data Collected at Outdoor Monitors Across the US](https://www.epa.gov/outdoor-air-quality-data) (https://www.epa.gov/outdoor-air-quality-data).

Congratulations! You've completed this lab. However, you may not notice a green check mark next to this item on Coursera's platform. Please continue your progress regardless of the check mark. Just click on the "save" icon at the top of this notebook to ensure your work has been logged.