Homework 2

PSTAT 134/234

Homework 2

Names of Collaborators (if any): William Mahnke

Part One: Analyzing the Weather

In this section, you will gain more practice working with public APIs, this time using a public weather API, WeatherAPI. The first thing you'll need to access the API is an API key. You can sign up for a key here: https://www.weatherapi.com/signup.aspx

Exercise 1

Use the http://api.weatherapi.com/v1/current.json URL to access the API and obtain real-time weather data. Note that you will want to specify three query parameters, at least – `key`, which should be set to your individual API key, `q`, which should equal the city name of a specified location – for example `q = "Isla Vista"` – and `aqi`, which indicates whether you want to obtain air quality data (`"yes"` or `"no"`).

Obtain current real-time weather data for **fifty randomly-selected cities**. I have saved a data file containing the names of fifty cities to `/data/cities.csv`. This ensures that you are all working with the same locations (although your results will still differ, depending on when you obtain the data).

```
# !pip install jupyter
# !pip install scikit-learn
# !pip install pandas
# !pip install statsmodels
# !pip install matplotlib
# !pip install seaborn
# !pip install flask
# !pip install requests
# !pip install bs4
import numpy as np
import sklearn as sk
import pandas as pd
import statsmodels.api as sm
```

```python
import matplotlib.pyplot as plt
import seaborn as sns
from flask import Flask, render_template, request  # Import Flask and related modules for web 
import requests  # Import requests library to make HTTP requests
from bs4 import BeautifulSoup # for webscraping

key = 'dd37f2e2fc62452a904195229241410'
cities = pd.read_csv('~/pstat-134-234/Homework/Homework 2/data/cities.csv')
aqi = "yes"
```

Exercise 2

Write code in R or Python (your choice) to extract and store the following data for each location:

- City name
- Country
- Whether or not it is currently daytime there
- Temperature (in Fahrenheit)
- Humidity
- Weather description (`condition` text; for example, "Mist", "Clear", etc.)
- Wind speed (in miles per hour)
- Precipitation (in millimeters)
- US EPA air quality index (ranges from 1 to 6, representing the 6 categories of air quality: https://www.airnow.gov/aqi/aqi-basics/)

```python
# Function to get current weather data from WeatherAPI
def get_current_weather(key, city, aqi):
    # Construct the API URL with the provided API key and location
    url = f"http://api.weatherapi.com/v1/current.json?key={key}&q={city}&aqi={aqi}"

    # Send a GET request to the API
    response = requests.get(url)

    # Check if the response was successful
    if response.status_code == 200:
        # Return the JSON data if the request was successful
        return response.json()
    else:
        # Return None if there was an error
        return None
```

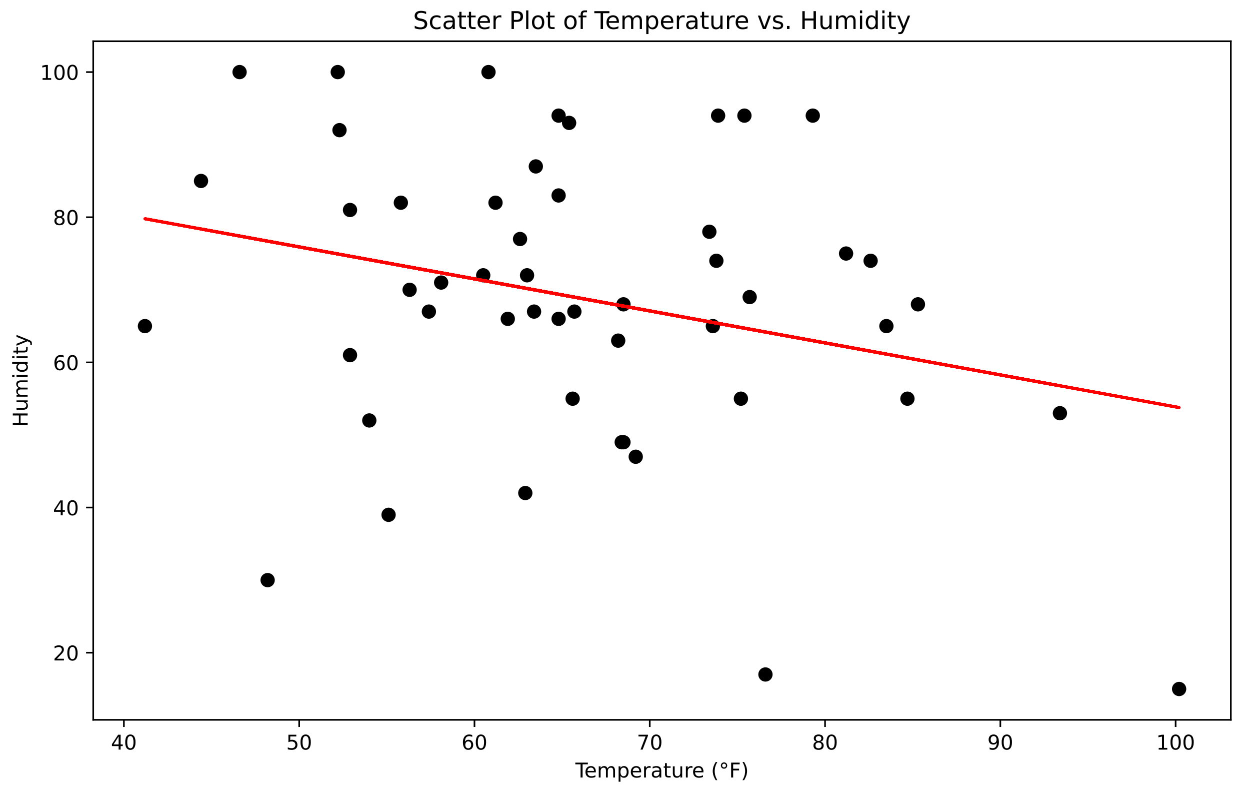
```
for i in range(len(cities)):
  weather_data = get_current_weather(key, cities['names'][i], aqi)

  if weather_data:
    cities.loc[i, 'Country'] = weather_data['location']['country']
    cities.loc[i, 'Daytime'] = weather_data['current']['is_day']
    cities.loc[i, 'Temperature'] = weather_data['current']['temp_f']
    cities.loc[i, 'Humidity'] = weather_data['current']['humidity']
    cities.loc[i, 'Description'] = weather_data['current']['condition']['text']
    cities.loc[i, 'WindSpeed'] = weather_data['current']['wind_mph']
    cities.loc[i, 'Precipitation'] = weather_data['current']['precip_mm']
    cities.loc[i, 'AirQuality'] = weather_data['current']['air_quality']['us-epa-index']
```

Exercise 3

Create a scatterplot of temperature vs. humidity. Add a linear regression line to the plot. What are the estimated intercept and slope values for this linear regression? Does there appear to be a significant relationship between temperature and humidity?

```
# in ChatGPT: is there a way to make a scatterplot, then add a line to it?

# Create X and y
X = cities['Temperature']
y = cities['Humidity']
X = sm.add_constant(X)  # Add a constant (intercept) to the model

# Fit the model
model = sm.OLS(y, X).fit()  # Fit the model

# Get slope and intercept
intercept = model.params[0]
slope = model.params[1]
y_predictions = model.predict(X)

# Plot the regression line
plt.figure(figsize=(10, 6))
plt.plot(cities['Temperature'], y, 'o', color='black') # scatterplot
plt.plot(cities['Temperature'], y_predictions, color='red') # line of best fit
plt.title('Scatter Plot of Temperature vs. Humidity')
plt.xlabel('Temperature (°F)')
plt.ylabel('Humidity')
plt.show()
plt.close()

print(f"Y-Intercept: {model.params[0]}")
print(f"Slopet: {model.params[1]}")
model.summary()
```

```
/tmp/ipykernel_5016/400319813.py:12: FutureWarning: Series.__getitem__ treating keys as positio
  intercept = model.params[0]
/tmp/ipykernel_5016/400319813.py:13: FutureWarning: Series.__getitem__ treating keys as positio
  slope = model.params[1]
/tmp/ipykernel_5016/400319813.py:26: FutureWarning: Series.__getitem__ treating keys as positio
  print(f"Y-Intercept: {model.params[0]}")
/tmp/ipykernel_5016/400319813.py:27: FutureWarning: Series.__getitem__ treating keys as positio
  print(f"Slopet: {model.params[1]}")
```



```
Y-Intercept: 97.95482783479632
Slopet: -0.440853876436222
```

Dep. Variable:	Humidity	**R-squared:**	0.075
Model:	OLS	**Adj. R-squared:**	0.056
Method:	Least Squares	**F-statistic:**	3.917
Date:	Sat, 26 Oct 2024	**Prob (F-statistic):**	0.0535
Time:	23:13:35	**Log-Likelihood:**	-217.97
No. Observations:	50	**AIC:**	439.9
Df Residuals:	48	**BIC:**	443.8
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P> |t|	[0.025	0.975]
const	97.9548	14.991	6.534	0.000	67.813	128.097
Temperature	-0.4409	0.223	-1.979	0.054	-0.889	0.007

Omnibus:	3.259	**Durbin-Watson:**	2.147
Prob(Omnibus):	0.196	**Jarque-Bera (JB):**	2.671
Skew:	-0.566	**Prob(JB):**	0.263
Kurtosis:	3.048	**Cond. No.**	369.

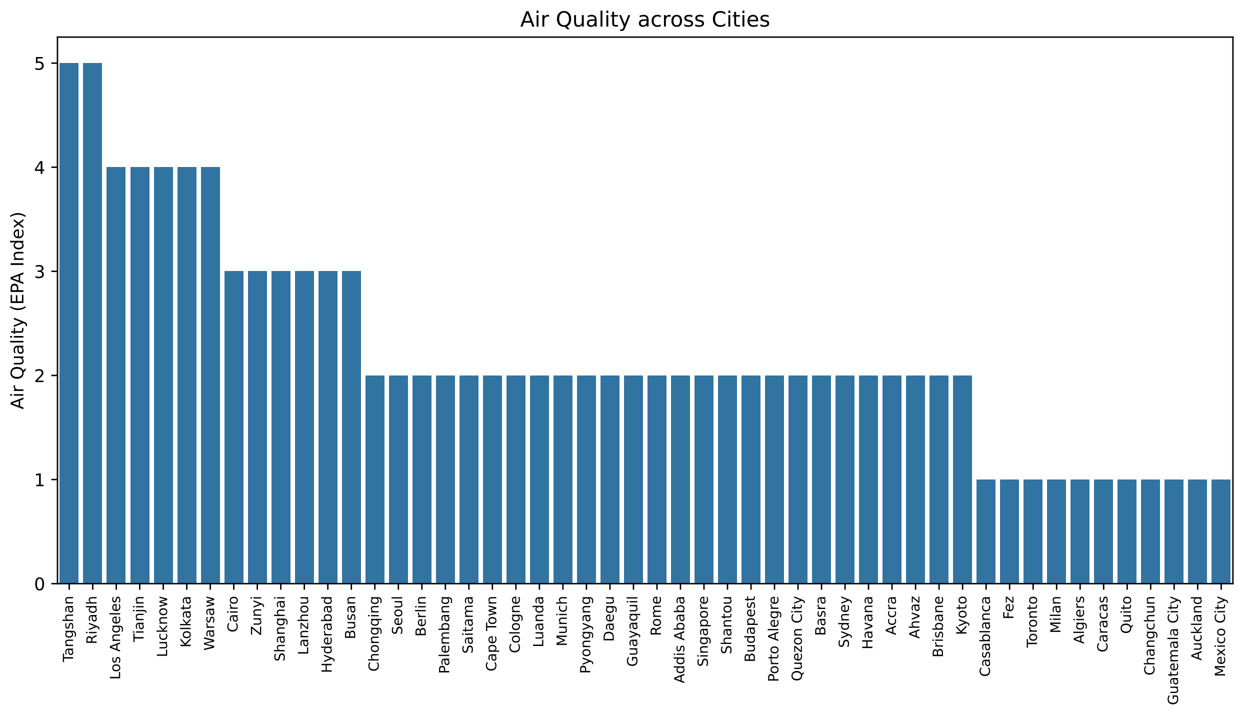
Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

There appears to be a negative relationship between temperature (in F) and humidity. As temperature increases, the humidity decreases.

Exercise 4

Create a bar chart of the EPA air quality index values. What does the distribution of air quality look like? Identify the location(s) with the best air quality and the worst air quality.

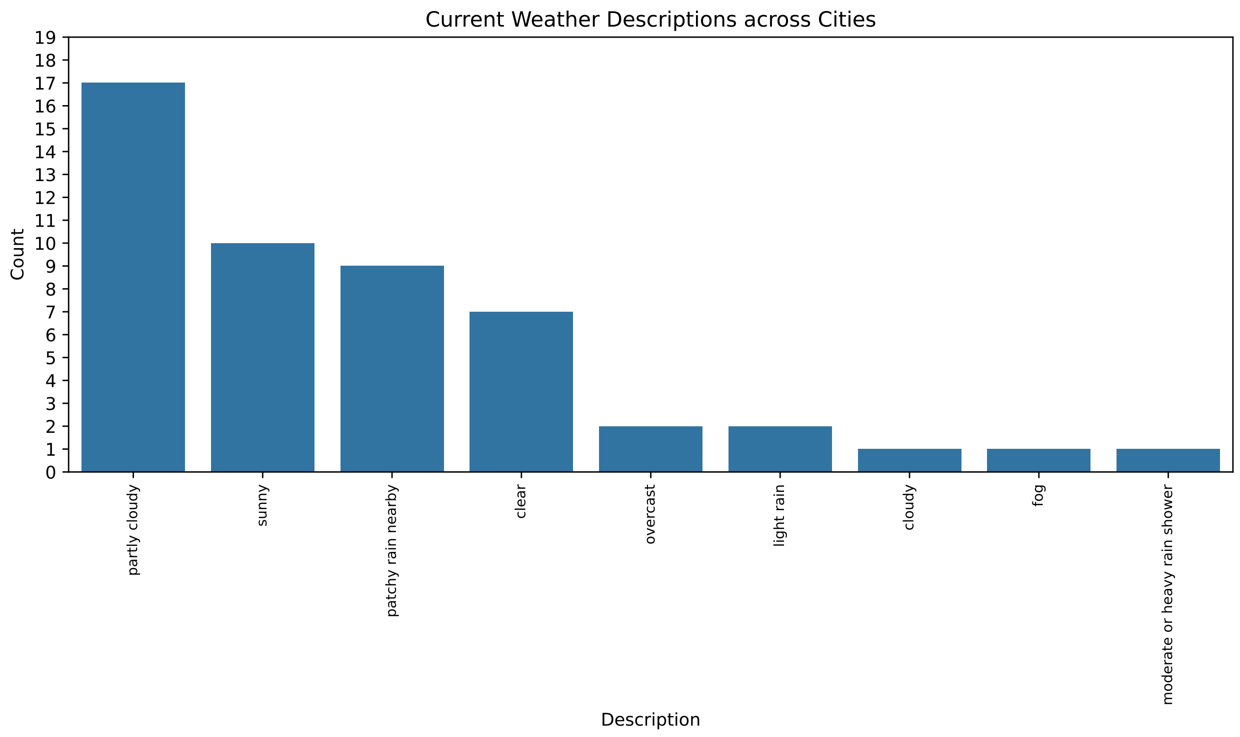
```
# in ChatGPT: "this is my code: plt.figure(figsize=(10, 6)) sns.barplot(data=cities, x='names'
plt.figure(figsize=(10, 6))
sns.barplot(data=cities, x='names', y='AirQuality',
order = cities.sort_values('AirQuality', ascending=False)['names'])
plt.title('Air Quality across Cities')
plt.ylabel('Air Quality (EPA Index)')
plt.xticks(rotation=90, fontsize = 8)
plt.tight_layout()
plt.show()
plt.close()
```



Exercise 5

Create a bar chart of the current weather description. Which conditions are the most common? Which are the least?

```
# in ChatGPT: "i have these descriptions column in my cities dataframe. can i convert all of th
plt.figure(figsize=(10, 6))
sns.countplot(data=cities, x=cities['Description'].str.lower(),
order = cities['Description'].str.lower().value_counts().index)
plt.title('Current Weather Descriptions across Cities')
plt.ylabel('Count')
plt.yticks(range(0, 20))
plt.xticks(rotation=90, fontsize = 8)
plt.tight_layout()
plt.show()
plt.close()
```



Partly cloudy is the most frequent weather condition, with 19 cities currently having that weather condition. Rain nearby and light rain are less frequent conditions, only currently occurring in 2 cities each.

(Because the data had differences in capitalization convention, I converted all descriptions to lowercase for comparison)

Exercises for 234 Students

Exercise 6

Do you think day vs. night cycles cause a significant difference in temperature? Test this hypothesis using a t-test.

Exercise 7

Create a table of the average temperature, humidity, wind speed, and precipitation broken down by weather description.

Exercise 8

Learn how to use the forecast API (http://api.weatherapi.com/v1/forecast.json).

Determine the chance of rain (in percentage) for Goleta, California tomorrow. *(Note that "tomorrow" may vary depending on when you do this assignment; that is fine.)*

Based on the percentage you obtained, do you think it will rain in Goleta tomorrow?

Part Two: Scraping Books

In this section, you'll practice your web scraping skills by experimenting with a fictional online bookstore located at https://books.toscrape.com/. Use the tools that we demonstrate in class to do the following, in either R or Python (your choice):

Exercise 9

Scrape the first 20 results from this site. Create a data frame (or tibble) that stores the following for each book:

- Title
- Price (excluding tax)
- Star rating
- Whether the book is in stock

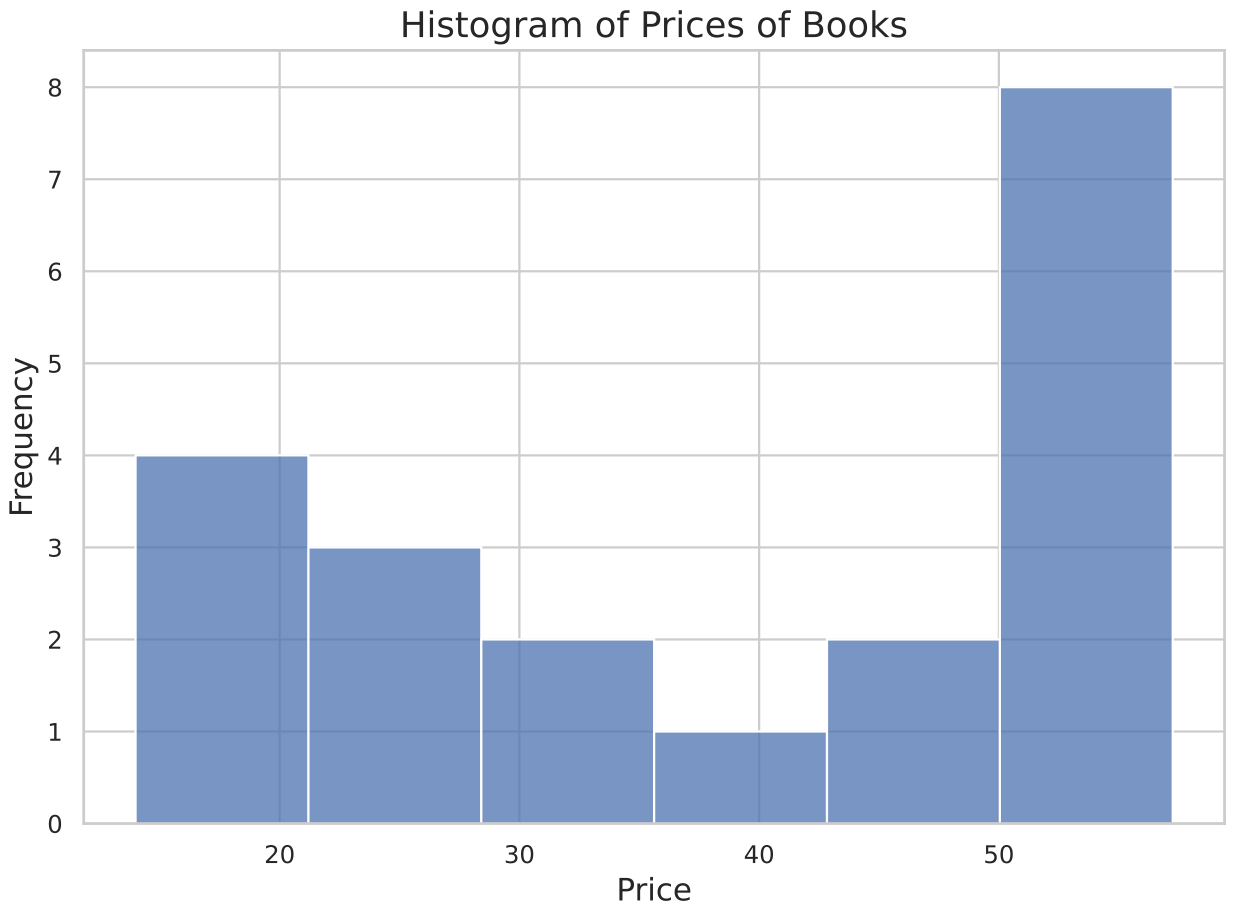
```
url = 'https://books.toscrape.com'

r = requests.get(url)

soup = BeautifulSoup(r.content, 'html.parser')
```

```
rows = soup.select('article.product_pod')

books = pd.DataFrame()

for i, row in enumerate(rows):
  books.loc[i, 'Title'] = row.h3.a['title']
  books.loc[i, 'Price'] = row.select_one('.product_price .price_color').text
  books.loc[i, 'Star rating'] = row.select_one('.star-rating')['class'][1]
  books.loc[i, 'Availability'] = row.select_one('.availability').text.strip()
```

Exercise 10

Create a histogram of prices for these 20 books. What is the average price?

```
for i in range(len(books['Price'])):
  books.loc[i, 'Price'] = pd.to_numeric(books['Price'][i][1:])

plt.figure(figsize=(8, 6))
sns.set(style="whitegrid")
sns.histplot(data=books, x='Price')
plt.title('Histogram of Prices of Books', fontsize=16)
plt.xlabel('Price', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.tight_layout()
plt.show()
plt.close()

books['Price'].mean()
```



```
np.float64(38.048500000000004)
```

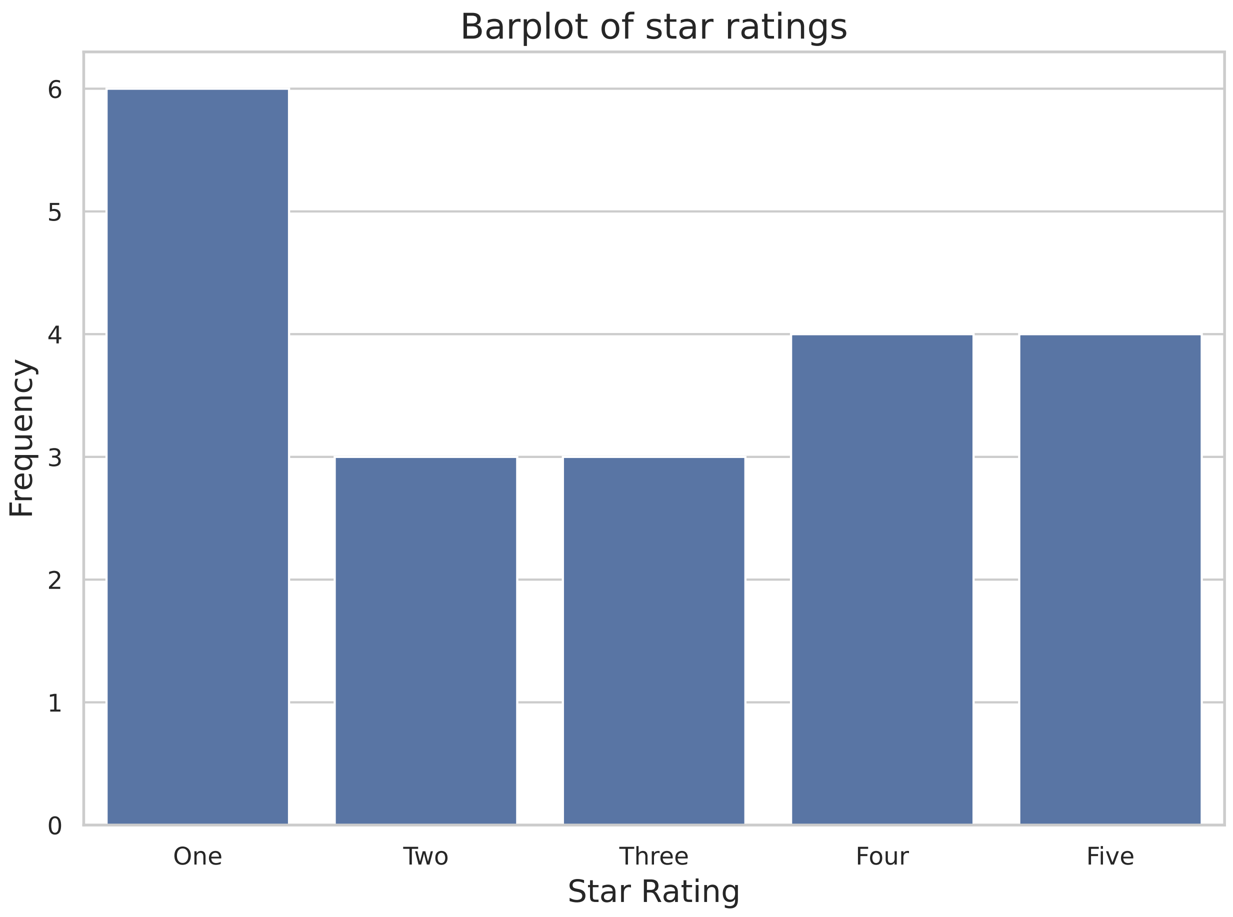
The average price of books is around 38.05 currency.

Exercise 11

Create a bar chart of star rating for these 20 books. Find the book(s) with the highest and lowest star ratings.

```
plt.figure(figsize=(8, 6))
sns.set(style="whitegrid")
sns.countplot(data=books, x=pd.Categorical(books['Star rating'],
categories = ['One', 'Two', 'Three', 'Four', 'Five'], ordered = True))
plt.title('Barplot of star ratings', fontsize=16)
```

```python
plt.xlabel('Star Rating', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.tight_layout()
plt.show()
plt.close()

books[books['Star rating'] == 'One']
books[books['Star rating'] == 'Five']
```



	Title	Price	Star rating	Availability
4	Sapiens: A Brief History of Humankind	54.23	Five	In stock
12	Set Me Free	17.46	Five	In stock
13	Scott Pilgrim's Precious Little Life (Scott Pi...	52.29	Five	In stock
14	Rip it Up and Start Again	35.02	Five	In stock

Tipping the Velvet, *Soumission*, *The Requiem Red*, *The Black Maria*, *Olio*, and *Mesarian* have the lowest rating of 1 star. *Sapiens, Set Me Free, Scott Pilgrim's Precious Little Life,* and *Rip it Up and Start Again* have the highest rating of 5 stars.

Exercises for 234 Students

Exercise 12

Extend your skills; instead of scraping only the first 20 books, scrape the first **two hundred books**.

For each book, in addition to the information we stored previously (title, price, star rating, etc.), figure out how to extract the **category** (i.e., Travel, Mystery, Classics, etc.).

Exercise 13

What is the most common category? What is the least common?