Data Collection and Exploratory Data Analysis (EDA) Laboratory Session

Welcome to our comprehensive laboratory session on Data Collection and Exploratory Data Analysis (EDA) using Python. Over the course of 2 hours, you'll dive deep into the world of data, learning how to gather information from various sources, clean and prepare datasets, and uncover valuable insights through statistical analysis. Get ready to explore the power of Python libraries and apply your knowledge to real-world datasets!

Laboratory Work Plan Overview

1

Objective

Master data collection techniques and perform Exploratory Data Analysis using Python, covering everything from data gathering to visualization and basic statistical analysis.

2

Duration

2 hours total, for hands-on practical work to ensure a balanced learning experience.

3

Materials

Python environment (Jupyter Notebook or IDE), essential libraries (pandas, numpy, matplotlib, seaborn, requests, beautifulsoup4, selenium, ydata-profiling), and sample datasets for practice.

Data Collection

1 Data Sources

Explore diverse data sources including APIs, web scraping techniques, and reputable online repositories, providing a comprehensive understanding of data acquisition methods.

2 Essential Libraries

Master the use of powerful Python libraries such as requests for API interactions and beautifulsoup4 for efficient web scraping, enhancing your data collection toolkit.

3 Practical Application

Gain hands-on experience by working with a real-world public API, applying theoretical knowledge to collect and process actual data sets.

Data Cleaning and Preparation

Cleaning Techniques

Learn essential methods for data cleaning, including strategies for handling missing values, eliminating duplicates, and ensuring proper data types across your dataset.

Pandas Functions

Harness the power of pandas with key functions like dropna(), fillna(), and astype() to efficiently clean and transform your data, preparing it for analysis.

Practical Exercise

Apply your newfound skills to a real-world dataset, experiencing firsthand the challenges and solutions in data cleaning and preparation.

Exploratory Data Analysis (EDA) Overview

1

Purpose and Steps

Understand the crucial role of EDA in uncovering patterns, detecting anomalies, and forming hypotheses. Learn a systematic approach to exploring your data effectively.

2

Visualization Techniques

Master the art of data visualization using matplotlib and seaborn. Create insightful histograms, scatter plots, and box plots to reveal hidden trends in your data.

3

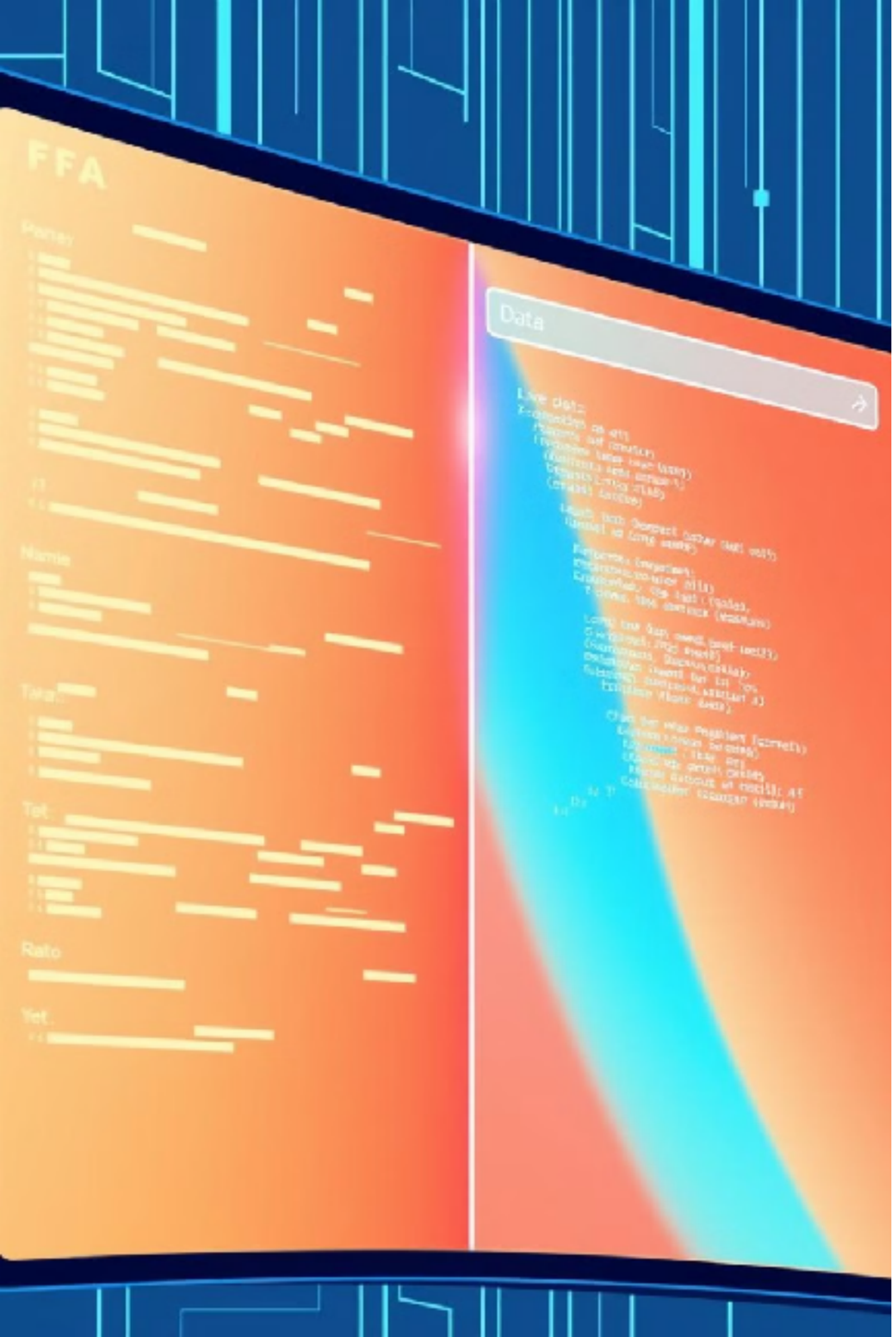
Statistical Analysis

Dive into basic statistical measures such as mean, median, mode, and standard deviation. Learn how these metrics provide valuable insights into your dataset's characteristics.

4

Practical Application

Apply EDA techniques to a sample dataset, gaining hands-on experience in interpreting visualizations and deriving meaningful conclusions from your analysis.

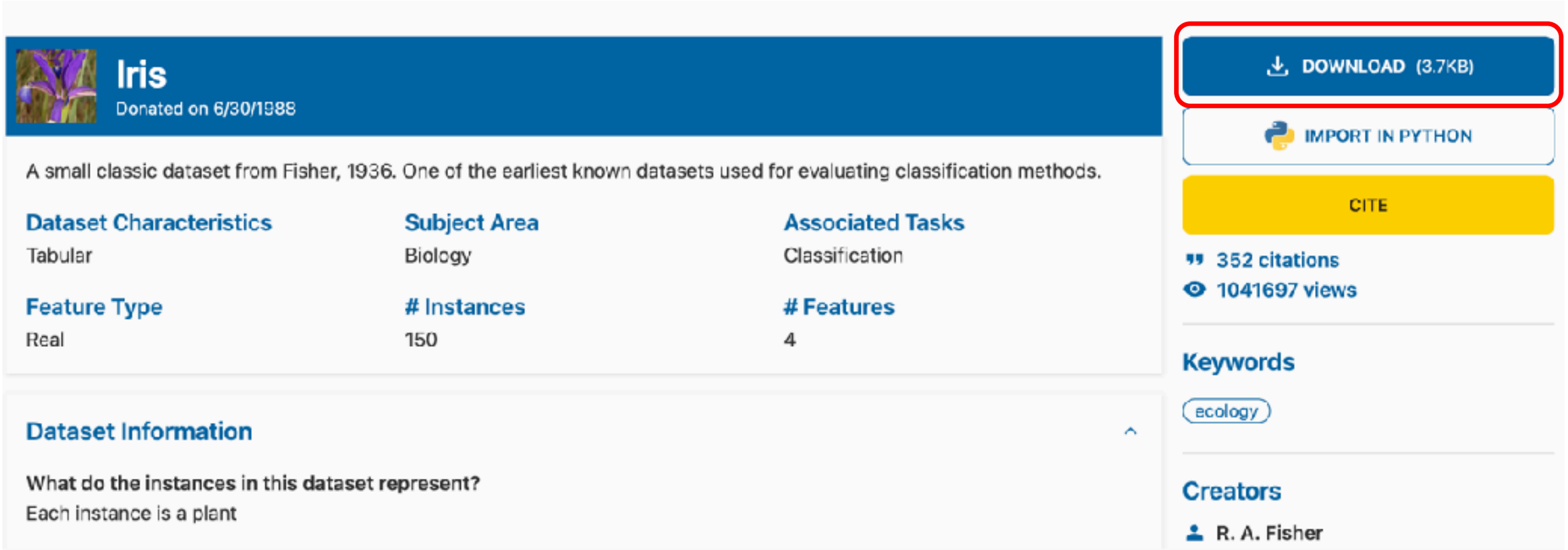


Practical Work: Data Collection

Task	Description	Tools
API Data Collection	Fetch COVID-19 or public dataset API data	requests library
Web Scraping	Collect news headlines or product prices	beautifulsoup4 library
Data Parsing	Convert collected data into DataFrames	pandas library

Download Data in csv format

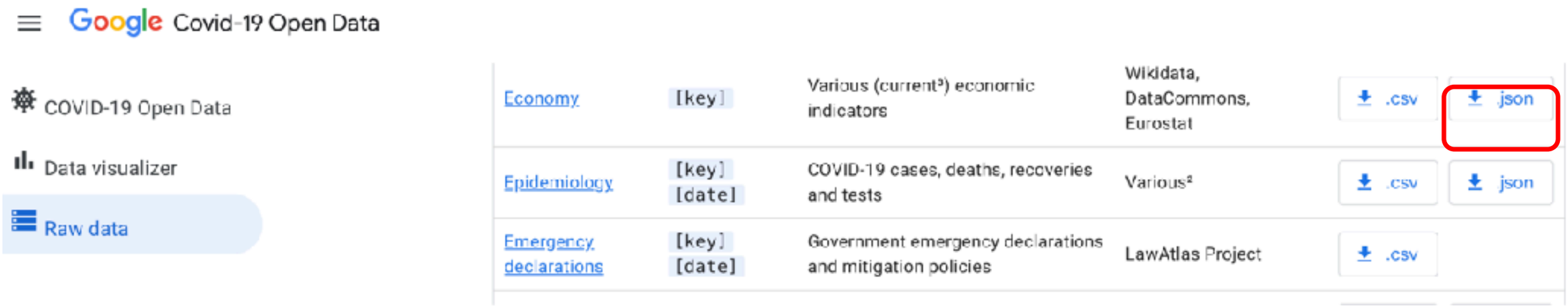
https://archive.ics.uci.edu/dataset/53/iris




```
import pandas as pd
data = pd.read_csv('iris.csv')
```

Download Data in json format

https://health.google.com/covid-19/open-data/raw-data




```
import pandas as pd
import requests

url = 'https://storage.googleapis.com/covid19-open-data/v3/epidemiology.json'

response = requests.get(url)
data_json = response.json()

data_df = pd.json_normalize(data_json)
```

Call hourly forecast data

How to make an API call

You can search weather forecast for 4 days with data every hour by geographic coordinates.

All weather data can be obtained in JSON and XML formats.

API call

```
https://pro.openweathermap.org/data/2.5/forecast/hourly?
lat={lat}&lon={lon}&appid={API key}
```

Parameters

`lat`	required	Latitude. If you need the geocoder to automatic convert city names and zip-codes to geo coordinates and the other way around, please use our Geocoding API
`lon`	required	Longitude. If you need the geocoder to automatic convert city names and zip-codes to geo coordinates and the other way around, please use our Geocoding API
`appid`	required	Your unique API key (you can always find it on your account page under the "API key" tab)
`mode`	optional	Data format. Possible values are `json` and `xml`. If the `mode` parameter is empty the format is JSON by default. Learn more
`cnt`	optional	A number of timestamps in response. Learn more
`lang`	optional	Language code. Learn more

Download Data in via API

OpenWeather

https://openweathermap.org/

```
#Importul bibliotecii requests
import requests

# Atribuirea valorii de identificare și parametrilor
appid = "875efb8e7aac89b96cc755c0cfa2ed1b"
URL = "http://api.openweathermap.org/data/2.5"
PARAMS={'q':s_city_name, 'type': 'like','units':'metric', 'lang': 'ro', 'APPID': appid}

# Verificarea prezenței informației despre localitatea

def get_city_id(s_city_name):
    try:
        res = requests.get(URL+"/find",params=PARAMS)
        data = res.json()
        cities = ["{} ({})".format(d['name'], d['sys'']['country'])
        for d in data['list']]:
            print("city:", cities)
            city_id = data['list'][0]['id']
            print('city_id=', city_id)
    except
        Exception as e:
            print("Exception (find):", e)
            pass
    assert isinstance(city_id, int)
    return city_id
```

```
# Prognoza meteo în localitatea cu id
def request_forecast(city_id):
    try:
        res = requests.get(URL +"forecast",params=PARAMS)
        data = res.json()
        print('city:', data['city']['name'], data['city’]['country'])
        for i in data['list']:
            print( (i['dt_txt'])[:16], '{0:+3.0f}'.format(i['main']['temp']),
                   '{0:2.0f}'.format(i['wind']['speed'])+"m/s", i['wind']['deg’],
                   i['weather'][0]['description'] )
    except
        Exception as e:
        print("Exception (forecast):", e)
        pass

#main body
city_id=None
s_city_name = 'Bucharest’
city_id = get_city_id(s_city_name)

request_forecast(city_id)
```

Task 1

Download data and load to pandas dataframe

1. csv format

https://www.sistemulenergetic.ro/
https://storage.googleapis.com/covid19-open-data/v3/epidemiology.csv

2. json format

World Air Quality – OpenAQ for Moldova (json format) from https://public.opendatasoft.com/

3. API

Weather forecast for Chisinau from openweathermap.org

Web scraping

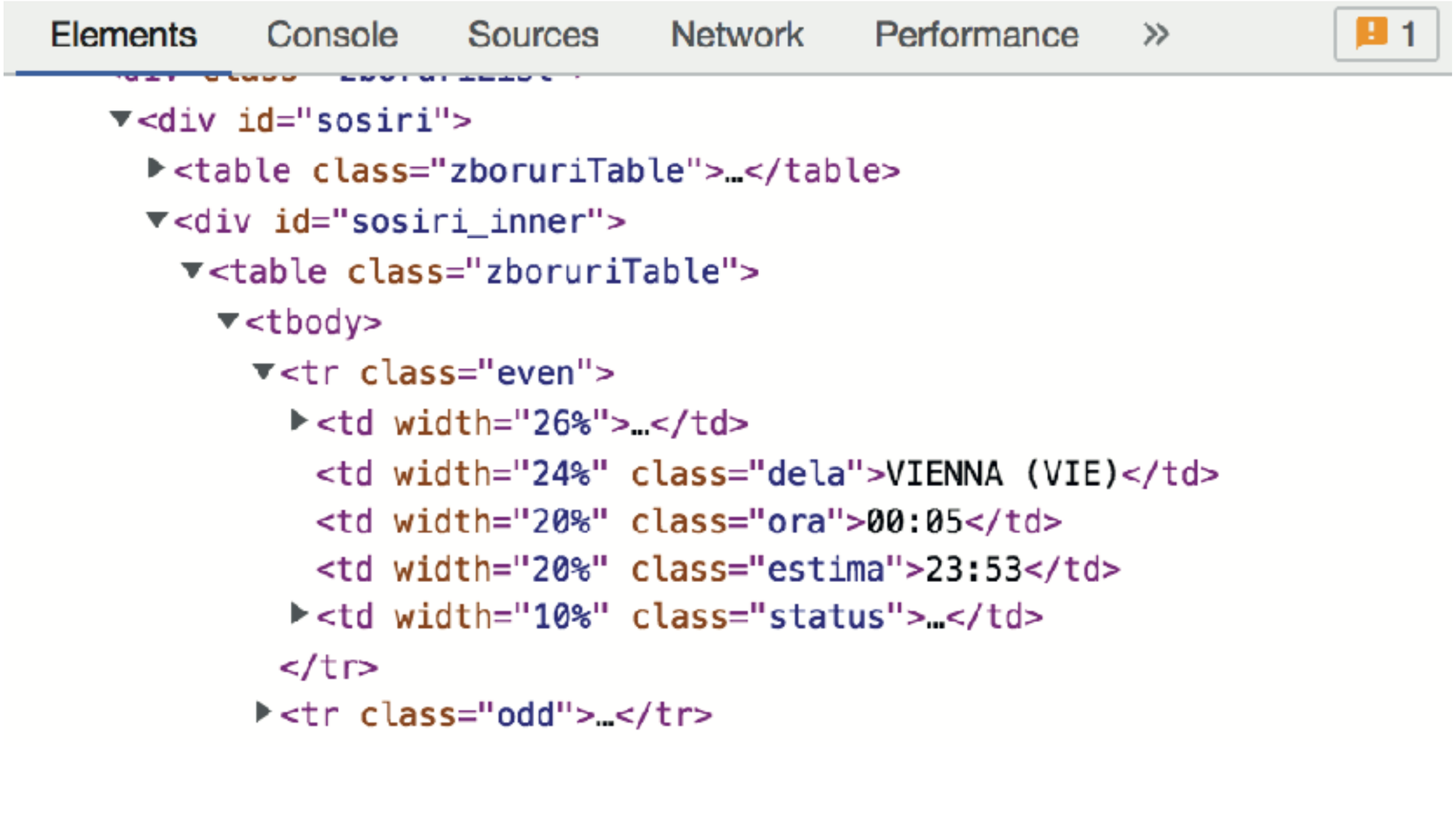
https://www.bucharestairports.ro/

Sosiri | Plecări | CĂUTARE ZBOR

NUMĂR CURSĂ	ORIGINE	ORA PROGRAMATĂ	ORA ESTIMATĂ	STATUS
AZ5041	SATU MARE (SUJ)	20:10	20:06	ATERIZAT
AF6612	SATU MARE (SUJ)	20:10	20:06	ATERIZAT
RO374	BRUSSELS (BRU)	20:35	23:23	ÎNTÂRZIAT
BT5653	BRUSSELS (BRU)	20:35	23:23	ÎNTÂRZIAT
RO238	BUDAPEST (BUD)	20:35	21:00	ATERIZAT
FR1007	LONDON (STN)	20:40	20:40	ESTIMAT
TK1045	ISTANBUL (IST)	20:40	20:22	ATERIZAT
RO9160	ISTANBUL (IST)	20:40	20:22	ATERIZAT
RO274	ATHENS (ATH)	20:45	20:46	ATERIZAT

Web scraping

https://www.bucharestairports.ro/



```
from bs4 import BeautifulSoup
import requests
import pandas as pd

page = requests.get('https://www.bucharestairports.ro/’)

bs = BeautifulSoup(page.content)
print(bs.title)

body = bs.find('div',{'id':'sosiri_inner'})
table = body.find('table', {'class':'zboruriTable'})

for row in rows:
    elements = row.find_all('td’)
    try:
        aa = elements[0].find('img')['title’]
    except:
        aa = None
     flight = {'airline': aa, 'flight': elements[0].text,
               'from': elements[1].text, 'schedule': elements[2].text,
               'estimation': elements[3].text, 'status': elements[4].text
              }
df = pd.DataFrame(result)
print(df.head())
```

Task 2

Scrape date from Bucharest airport for departure flights and download them to pandas dataframe. Add attribute date and time of scraping

Practical Work: Data Cleaning

Handle Missing Values

Learn techniques to identify and address missing data points, ensuring the integrity and completeness of your dataset for accurate analysis.

Eliminate Duplicates

Master methods to detect and remove duplicate entries, maintaining data quality and preventing skewed analysis results.

Outlier Detection

Detect outliers based on statistical methods, using standard deviation and interquartile range

Data Transformation

Explore advanced data manipulation techniques, including renaming columns, splitting complex fields, and converting data types for optimal analysis.

1. Download data

https://archive.ics.uci.edu/dataset/2/adult

```
df = pd.read_csv('adult.data', sep=',')
print(df.head())
```

2. Data structure analysis

```
df.shape
df.columns
df.info()
df.isnull().sum()
df.duplicated().sum()
df.value_counts()
```

3. Univariate analysis

```
df.describe(include ='all').T

Import seaborn as sns
sns.boxplot(x=df[”age"])
```

4. **Exclude duplicates**

```
df.drop_duplicates(inplace=True)
```

5. **Complete missing values**

```
moda = df['occupation'].mode()[0]
df['occupation'] = df['occupation'].fillna(moda)
```

6. **Outliers detection**

```
data_mean, data_std = df['age'].mean(), df['age'].std()
cut_off = data_std * 3
lower, upper = data_mean - cut_off, data_mean + cut_off
print('Number of outliers: ',df[~df['age'].between(lower, upper)].count())
df = df[df['age'].between(lower, upper)]
```

7. **Transform data from categorial to numeric**

```
from sklearn.preprocessing import OneHotEncoder
ohe=OneHotEncoder()
transformed = ohe.fit_transform(df[['workclass']])
print(ohe.categories_)
df[ohe.categories_[0]]=transformed.toarray()
```

Task 3

- Download data from https://archive.ics.uci.edu/dataset/222/bank+marketing

- Make Exploratory Data Analysis

- Identify outliers in “campaign” and “duration”

- Transform categorial data from “maritial” feature

Practical Work: YData-profiling

```
!pip install ydata-profiling

from ydata_profiling import ProfileReport
import pandas as pd

data = pd.read_csv(”data.csv")
profile = ProfileReport(data,title="Profiling Report")

profile.to_file("profiling_report.html")
```

Overview

Overview | Alerts 43 | Reproduction

Dataset statistics

Number of variables	30
Number of observations	2208
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	0
Duplicate rows (%)	0.0%
Total size in memory	517.6 KiB
Average record size in memory	240.1 B

Variable types

Numeric	16
Categorical	14

Overview | Alerts 43 | Reproduction

Alerts

`Z_CostContact` has constant value "3"	Constant
`Z_Revenue` has constant value "11"	Constant
`Dt_Customer` has a high cardinality: 662 distinct values	High cardinality
`Income` is highly correlated with `Kidhome` and 13 other fields	High correlation
`Kidhome` is highly correlated with `Income` and 3 other fields	High correlation
`MntWines` is highly correlated with `Income` and 9 other fields	High correlation
`MntFruits` is highly correlated with `Income` and 5 other fields	High correlation
`MntMeatProducts` is highly correlated with `Income` and 2 other fields	High correlation
`MntFishProducts` is highly correlated with `Income` and 8 other fields	High correlation

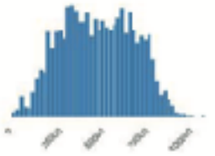
Income

Real number ($\mathbb{R}_{\geq 0}$)

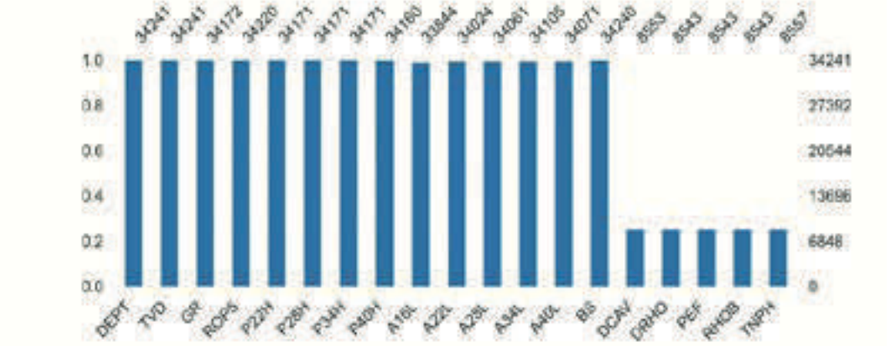
`HIGH CORRELATION`

Distinct	1966
Distinct (%)	89.0%
Missing	0
Missing (%)	0.0%
Infinite	0
Infinite (%)	0.0%
Mean	51633.63813

Minimum	1730
Maximum	113734
Zeros	0
Zeros (%)	0.0%
Negative	0
Negative (%)	0.0%
Memory size	17.4 KiB



Count | Matrix | Heatmap | Dendrogram



Home Work: Exploratory Data Analysis

Collect Data

Collect data about Lots of Public Procurement from https://mtender.gov.md/

Generate Descriptive Statistics

Calculate key statistical measures such as mean, median, mode, standard deviation, and quartiles to gain a quantitative understanding of dataset's characteristics.

Identify Outliers

Identify records with outliers in “Days from tender close to award decision” and “Length of tender period (days)”

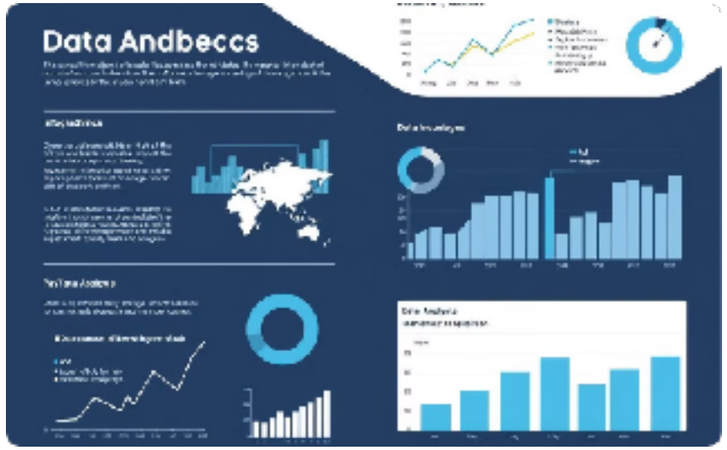
Summarize Findings

Compile a comprehensive report detailing key observations and insights derived from your EDA, honing your ability to communicate complex data findings effectively.

Deliverables and Assessment

Python Notebooks

Submit comprehensive notebooks containing well-documented code for data collection, cleaning, and EDA processes, demonstrating your practical skills and understanding.



Analysis Report

Prepare a concise yet thorough report summarizing your EDA findings, showcasing your ability to interpret data and communicate insights effectively.

Assessment Criteria

Your work will be evaluated based on the accuracy of data collection methods, effectiveness of data cleaning, quality of EDA visualizations, and the depth of analysis in your summary report.