# Bahria University,

## Karachi Campus



LAB EXPERIMENT NO.

**\_02\_**

LIST OF TASKS

|  |  |
| --- | --- |
| **TASK NO** | **OBJECTIVE** |
| 01 | Develop a Python application to generate data visualizations Scenario: You are a data analyst working with a large dataset containing various types of data. Your task is to create a Python application that uses the Pandas, Matplotlib, and Seaborn libraries to perform exploratory data analysis and generate interactive visualizations. The application should allow users to load their dataset, explore the data, and create suitable charts and plots for visual analysis. |
| 02 | Implement a text summarization model using Transformers Scenario: As a natural language processing (NLP) researcher, you have been tasked with developing a text summarization model that can generate concise summaries of long text documents. Your task is to utilize the Transformers library in Python to build and train a summarization model. The model should be able to take a long text document as input and generate a concise summary that captures the key information and main ideas. |
| 03 | Convert images to sketches using OpenCV Scenario: You are a computer vision enthusiast working on a project to develop a photo editing application. Your task is to create a Python script that uses the OpenCV library to convert regular images into sketches. The script should allow users to select an image file, apply appropriate filters and transformations to convert it into a sketch-like image, and save the resulting image to disk. |
| 04 | Build a web scraper using Beautiful Soup Scenario: You are a data engineer working for Pakveels. Your task is to develop a Python script that uses the Beautiful Soup library to scrape product information from competitor websites. The script should be able to extract data such as product names, descriptions, prices, and images from the target websites and store the data in a structured format (e.g., CSV or JSON) for further analysis. |
| 05 | Automate WhatsApp messaging using PyWhatKit Scenario: You are a software developer working on a project to automate communication for a small business. Your task is to create a Python script that uses the PyWhatKit library to automate the sending of messages and images through WhatsApp. The script should allow users to schedule the sending of messages or images to one or more contacts at specific times or intervals. |

Submitted On:

Date: 20/02/2024

**Task No 01:** Develop a Python application to generate data visualizations Scenario: You are a data analyst working with a large dataset containing various types of data. Your task is to create a Python application that uses the Pandas, Matplotlib, and Seaborn libraries to perform exp

loratory data analysis and generate interactive visualizations. The application should allow users to load their dataset, explore the data, and create suitable charts and plots for visual analysis.

**Solution:**

import pandas as pd

first\_names=['Shoaib','Abdul','Sarah']

last\_names=['Akhter','Ahad','Connor']

df = pd.DataFrame({

    'first\_name':first\_names,

    'last\_name':last\_names

})

df

import altair as alt

chart = alt.Chart(df).mark\_arc().encode(

    theta=alt.Theta(field = "first\_name", type="nominal"),

    color=alt.Color(field="first\_name",type="nominal"),

).properties(

    width=300,

    height=200,

)

Chart

import seaborn as sns

import matplotlib.pyplot as plt

from vega\_datasets import data

df=data.cars()

sns.scatterplot(x="Horsepower",y="Miles\_per\_Gallon",data=df)

plt.title=("Scatter Plot of Horsepower vs. Miles per Gallon")

plt.xlabel("Horsepower")

plt.ylabel("Miles per Gallon")

plt.show()

sns.histplot(df['Acceleration'],bins=20,kde=True)

plt.title('Histogram of Acceleration')

plt.xlabel('Acceleration')

plt.ylabel('Frequency')

plt.show()

plt.bar(df['Origin'].unique(),df['Origin'].value\_counts())

plt.title("Bar Chart of Car Origins")

plt.xlabel('Origin')

plt.ylabel('Count')

plt.show()

plt.boxplot(df['weight\_in\_lbs'])

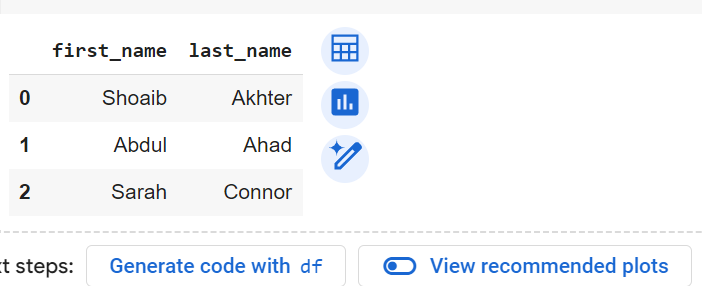
plt.title("Box Plot of Car Weights")

plt.ylabel("Weight in lbs")

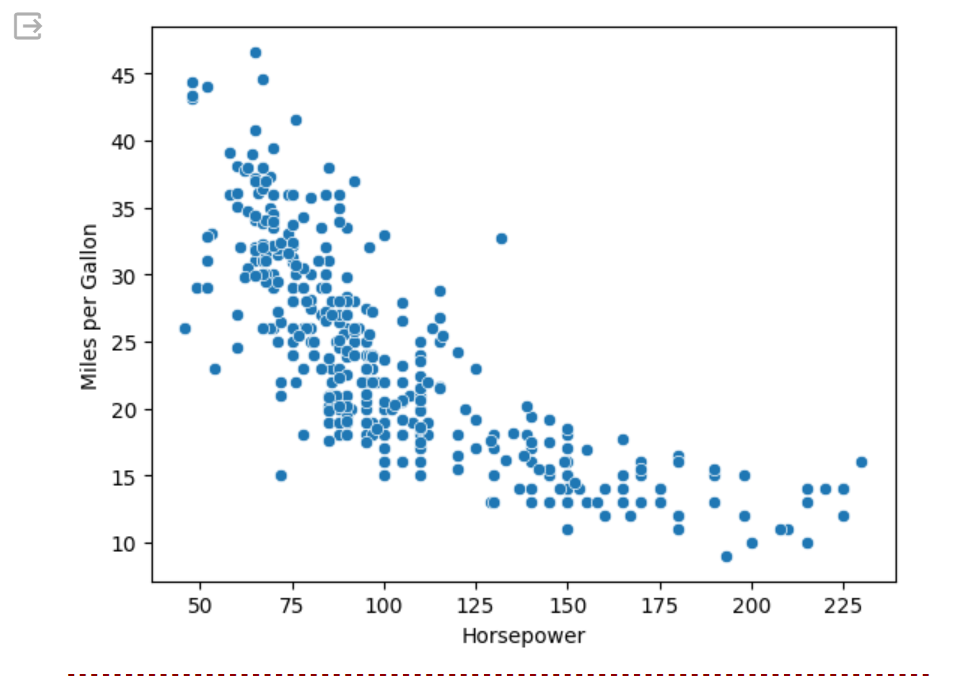
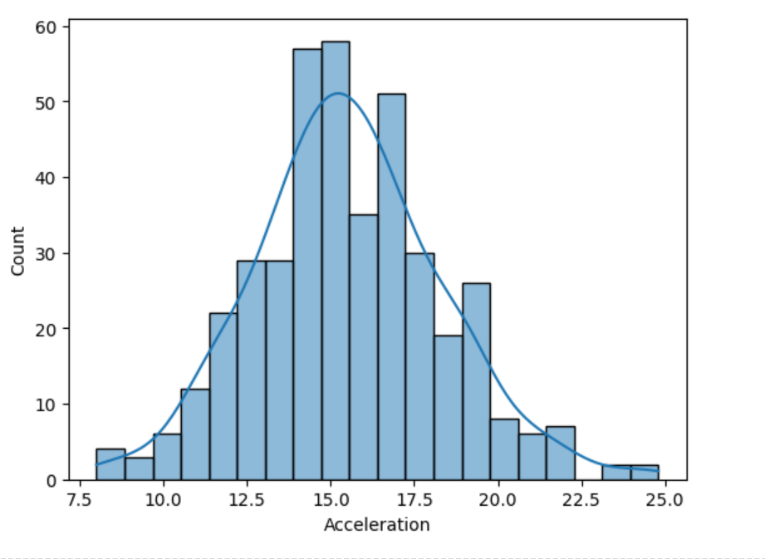
plt.show()

**Output:**

**A pie chart with text and numbers

Description automatically generated with medium confidence** **A screenshot of a graph

Description automatically generated**

****

**Task No 02:** Implement a text summarization model using Transformers Scenario: As a natural language processing (NLP) researcher, you have been tasked with developing a text summarization model that can generate concise summaries of long text documents. Your task is to utilize the Transformers library in Python to build and train a summarization model. The model should be able to take a long text document as input and generate a concise summary that captures the key information and main ideas.

**Solution:**

!pip install transformers

import torch

from transformers import pipeline

summarizer = pipeline("summarization")

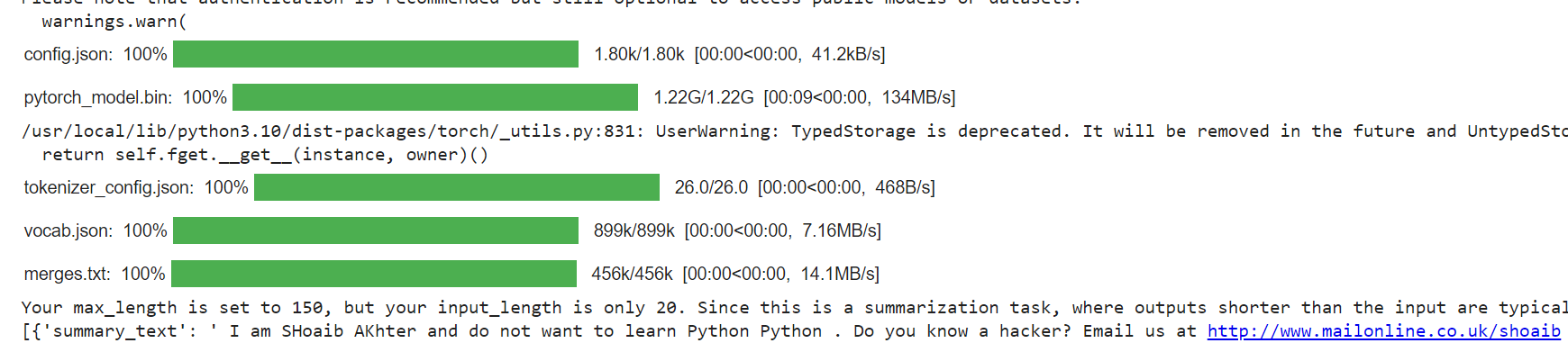
text = """I am SHoaib AKhter and do not want to learn Python"""

summary = summarizer(text, max\_length=150, min\_length=30, do\_sample=False)

print(summary)

**Output:**

****

****

**Task No 03:** Convert images to sketches using OpenCV Scenario: You are a computer vision enthusiast working on a project to develop a photo editing application. Your task is to create a Python script that uses the OpenCV library to convert regular images into sketches. The script should allow users to select an image file, apply appropriate filters and transformations to convert it into a sketch-like image, and save the resulting image to disk.

**Solution:**

!pip install opencv-python

import cv2

def convert\_to\_sketch(image\_path, output\_path):

    image = cv2.imread(image\_path)

    grayscale\_image = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

    blurred\_image = cv2.GaussianBlur(grayscale\_image, (5, 5), 0)

    edges = cv2.Laplacian(blurred\_image, cv2.CV\_64F, ksize=5)

    inverted\_edges = 255 - edges.astype('uint8')

    sketch = cv2.addWeighted(grayscale\_image, 0.5, inverted\_edges, 0.5, 0)

    cv2.imwrite(output\_path, sketch)

input\_image\_path = 'myInput.jpg'

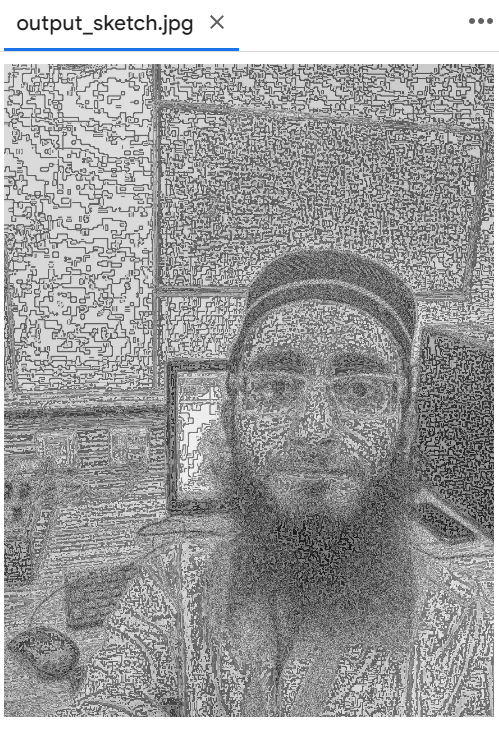
output\_sketch\_path = 'output\_sketch.jpg'

convert\_to\_sketch(input\_image\_path, output\_sketch\_path)

print("Sketch saved successfully at:", output\_sketch\_path)

**Output:**

**A person with a beard and glasses

Description automatically generated**

**Task No 04:** Build a web scraper using Beautiful Soup Scenario: You are a data engineer working for Pakveels. Your task is to develop a Python script that uses the Beautiful Soup library to scrape product information from competitor websites. The script should be able to extract data such as product names, descriptions, prices, and images from the target websites and store the data in a structured format (e.g., CSV or JSON) for further analysis.

**Solution:**

!pip install beautifulsoup4

from bs4 import BeautifulSoup

import requests

import csv

url = 'https://daraz.pk'

response = requests.get(url)

html\_content = response.text

soup = BeautifulSoup(html\_content, 'html.parser')

products = []

product\_elements = soup.find\_all('div', class\_='product')

for product\_element in product\_elements:

    name = product\_element.find('h2', class\_='product-name').text.strip()

    description = product\_element.find('p', class\_='product-description').text.strip()

    price = product\_element.find('span', class\_='product-price').text.strip()

    image\_url = product\_element.find('img')['src']

    product = {

        'name': name,

        'description': description,

        'price': price,

        'image\_url': image\_url

    }

    products.append(product)

csv\_file = 'products.csv'

fieldnames = ['name', 'description', 'price', 'image\_url']

with open(csv\_file, 'w', newline='', encoding='utf-8') as file:

    writer = csv.DictWriter(file, fieldnames=fieldnames)

    writer.writeheader()

    writer.writerows(products)

print('Scraping completed. Data saved to', csv\_file)

**Output:**

****

**A screen shot of a computer

Description automatically generated**

**Task No 05:** Automate WhatsApp messaging using PyWhatKit Scenario: You are a software developer working on a project to automate communication for a small business. Your task is to create a Python script that uses the PyWhatKit library to automate the sending of messages and images through WhatsApp. The script should allow users to schedule the sending of messages or images to one or more contacts at specific times or intervals.

**Solution:**

!pip install pywhatkit

import pywhatkit as kit

import datetime

def send\_message(phone\_number, message, time):

    hour, minute = map(int, time.split(':'))

    scheduled\_time = datetime.datetime.now().replace(hour=hour, minute=minute, second=0, microsecond=0)

    kit.sendwhatmsg(phone\_number, message, scheduled\_time.hour, scheduled\_time.minute)

def send\_image(phone\_number, image\_path, caption, time):

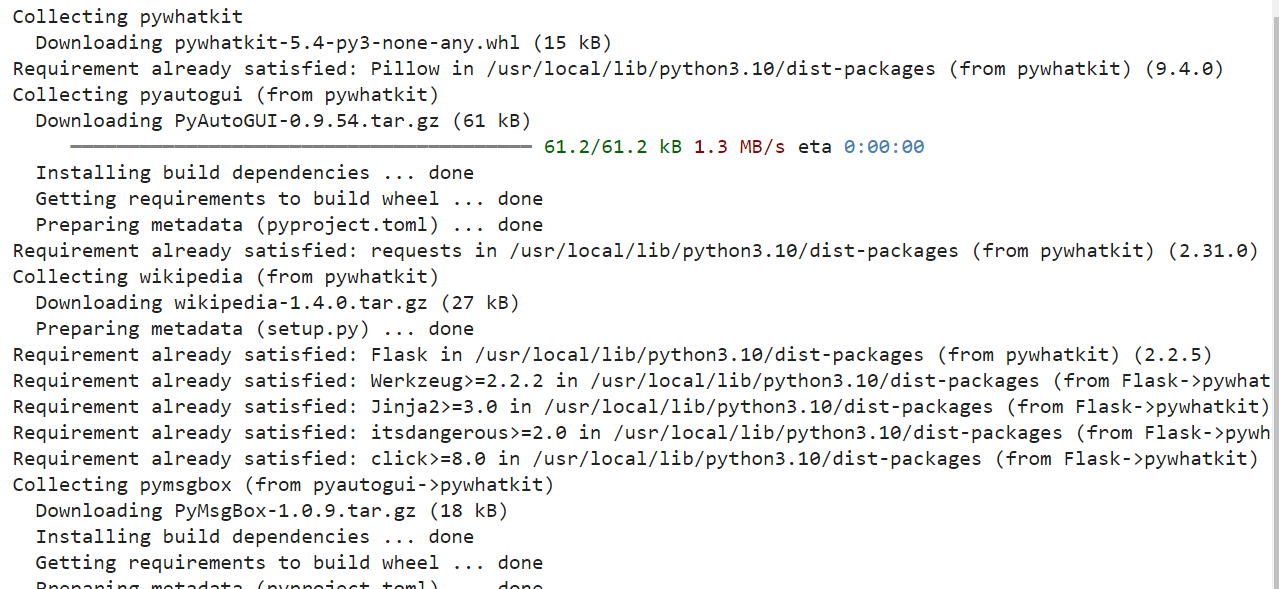
    hour, minute = map(int, time.split(':'))

    scheduled\_time = datetime.datetime.now().replace(hour=hour, minute=minute, second=0, microsecond=0)

    kit.sendwhats\_image(phone\_number, image\_path, caption, scheduled\_time.hour, scheduled\_time.minute)

send\_message("+923495004807", "Hello, this is an automated message!", "13:30")

**Output:**

****

**Task No 06:** Develop a text-to-speech application using pyttsx3

**Scenario**: You are a developer working on an accessibility project to help visually impaired users interact with digital content. Your task is to create a Python application that uses the pyttsx3 library to convert text into spoken words. The application should allow users to input text, select voice settings (e.g., language, gender, rate), and generate audio output that can be played or saved to a file.

**Solution:**

!pip install pyttsx3

import pyttsx3

engine = pyttsx3.init()

def set\_voice\_rate(rate):

    engine.setProperty('rate', rate)

def set\_voice\_volume(volume):

    engine.setProperty('volume', volume)

def set\_voice\_voice(voice\_id):

    voices = engine.getProperty('voices')

    engine.setProperty('voice', voices[voice\_id].id)

def text\_to\_speech(text):

    engine.say(text)

    engine.runAndWait()

text = input("Enter the text you want to convert to speech: ")

rate = int(input("Enter the speaking rate (words per minute): "))

set\_voice\_rate(rate)

volume = float(input("Enter the volume (between 0 and 1): "))

set\_voice\_volume(volume)

print("Select a voice:")

voices = engine.getProperty('voices')

for i, voice in enumerate(voices):

    print(f"{i}. {voice.name}")

voice\_id = int(input("Enter the voice ID: "))

set\_voice\_voice(voice\_id)

text\_to\_speech(text)