END CAPSTONE-1: RESUME PARSER

Table of Contents:

Introduction:

In today's fast-paced job market, companies receive a large number of resumes for every open position, making it time-consuming and labor-intensive for HR professionals to manually review and assess each resume. Extracting key information such as candidate details, skills, and experience from diverse resume formats can be cumbersome and prone to errors.

To streamline this process, the Resume Parser project aims to automate resume data extraction, processing, and analysis. By utilizing technologies such as Python for data extraction, SQL Server for structured data storage, and Power BI for visual insights, this system reduces the manual workload involved in resume handling and improves the accuracy and efficiency of candidate assessment.

This automation not only saves time but also helps organizations make data-driven decisions while shortlisting and interviewing candidates.

Objective of the Project:

To develop an automated system for extracting, cleaning, storing, and analyzing resume data using Python, SQL, machine learning, and data visualization tools. Ensure efficient data handling, robust analysis, and insightful visualizations to streamline the management of resume information and enhance data-driven decision-making.

Scope of the Project:

The scope of the Resume Parser Project encompasses the following key areas:

1. Data Extraction:

- The system will parse resumes in formats such as PDF and, Word.

- Key information including the candidate’s name, contact details, city, state, education, skills, experience, and job roles will be automatically extracted from the resume files.

2. Data Processing and Storage:

- Extracted resume data is first stored in Excel then cleaned, structured, and stored in a SQL Server database for easy access and future retrieval.
- A staging table will be used for initial data loading, followed by automated procedures to move data into the main tables.
- Any changes made to the resume data in the Excel interface will be automatically synchronized with the SQL database.

3. Automation:

- A VBA script is used to trigger the data extraction and insertion process whenever a new resume is added or an existing one is updated.
- Stored procedures and triggers in SQL Server will ensure that data is processed, validated, and logged in an efficient manner.
- The final stored data is then used for visualization in Power BI, in which the dashboard gets updated automatically with the database update.

4. Machine Learning Model Integration:

- The project integrates a machine learning model that predicts the candidate's primary skill based on the resume content.
- This prediction will assist HR professionals in quickly identifying the suitable job role or field for the candidate.

5. Data Analysis and Visualization:

- Power BI is integrated to provide graphical insights and reports, such as skill distribution by job role, experience levels across candidates, etc.

6. User Interaction:

- HR professionals will be able to view the parsed data, analyze candidate profiles, and generate reports through Power BI dashboards.
- The Excel interface will allow users to manage resumes while automating backend processes through VBA and SQL triggers.

7. Limitations:

- The system focuses on extracting data from structured resumes and may face challenges when dealing with unstructured or visually complex resume formats.
- The project will handle resumes in English only, without support for multilingual parsing at this stage

Executive Summary:

The Resume Parser Project aims to develop a comprehensive system that automates the extraction, processing, storage, and analysis of resume data. By leveraging Python, SQL, machine learning, and data visualization tools, the project focuses on efficiently handling resumes in various formats (PDF, Word, text files) and transforming them into structured data.

The system will automatically extract relevant information such as candidate details, skills, education, and experience. This data will then be cleaned, pre-processed, and stored in a SQL Server database for seamless access and retrieval. Additionally, the project integrates machine learning models to predict a candidate's primary skill based on the resume content, further enhancing data-driven decision-making for HR professionals.

Data visualization tools such as Seaborn and Power BI will provide advanced analysis and insights, including skill frequency, experience distribution, and job role segmentation. The system also includes automation features via VBA scripts and SQL triggers, ensuring that changes in the data are automatically synchronized and updated in the database. Overall, the project provides an end-to-end solution for efficient resume management and analysis.

Research Methodology:

The project followed an agile approach to development.

- Initial research focused on comparing various Python libraries for resume parsing, including PyPDF2, pdfplumber for PDF extraction, docx for Word documents, and spaCy for text processing. After evaluating the capabilities of each library, we implemented a hybrid parsing solution to handle various resume formats and selected pdfplumber for pdf format.
- Then searched various keywords and patterns of presenting information like location, experience, skills, email, primary skill, and education in different resume formats to extract them properly.
- Further research was done related to the machine learning model for enhancing the performance of the XGBoost model, for this first the correct sampling technique was found to address the imbalanced class data, and then for tuning hyperparameters, we gathered information about various techniques and finally implemented Grid Search CV technique.
- The performance of the ML models is evaluated using a classification report and confusion matrix and accordingly suitable model is selected for primary skill prediction based on resume content.

Code Explanation:

- We initiated the project by writing a Python script to extract data from a folder containing multiple resumes in PDF and DOC formats. This script reads each file, extracts relevant information (such as name, email, phone number, email, city, state, education, skills, experience, and primary skill), and saves it into an Excel sheet for further processing.

```python
# importing libraries
import os
import re
import docx
import pdfplumber  # instead of PyPDF2 for better PDF text extraction
import pandas as pd

# Helper function to extract text from docx
def extract_text_from_docx(file_path):
    doc = docx.Document(file_path)
    return '\n'.join([para.text for para in doc.paragraphs])

# Helper function to extract text from pdf
def extract_text_from_pdf(file_path):
    text = ''
    with pdfplumber.open(file_path) as pdf:
        for page in pdf.pages:
            text += page.extract_text()
    return text

# Function to normalize text
def normalize_text(text):
    return text.replace('\n', '').replace('\r', '').strip()

# Function to extract name (assumed to be the first string of characters in the resume)
def extract_name(text):
    name = text.split('\n')[0].strip()
    return name
```

```python
# Function to extract mobile number
def extract_mobile_number(text):
    phone_patterns = [
        re.compile(r'\+?\d{1,4}[\s.-]?\(?\d{1,4}\)?[\s.-]?\d{1,4}[\s.-]?\d{1,9}', re.IGNORECASE),
        re.compile(r'\(?\d{3}\)?[\s.-]?\d{3}[\s.-]?\d{4}', re.IGNORECASE),
        re.compile(r'\d{10}', re.IGNORECASE)
    ]
    for pattern in phone_patterns:
        match = pattern.search(text)
        if match:
            return match.group().strip()
    return None

# Fuction to extract email
def extract_email(text):
    text = normalize_text(text)
    email_regex = re.compile(r'[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}', re.IGNORECASE)
    matches = email_regex.findall(text)
    return matches[0].strip() if matches else None

# Function to extract city and state
def extract_location(text):
    location_patterns = [
        re.compile(r'([A-Za-z\s]+),\s?([A-Za-z\s]+)', re.IGNORECASE),
        re.compile(r'([A-Za-z\s]+)\s*-\s*([A-Za-z\s]+)', re.IGNORECASE),
        re.compile(r'([A-Za-z\s]+)\s+([A-Za-z\s]+)', re.IGNORECASE)
    ]
    for pattern in location_patterns:
        match = pattern.search(text)
        if match:
            city, state = match.groups()
            return city.strip(), state.strip()
    return None, None
```

```python
# Function to extract education details
def extract_education(text):
    education_keywords = ['Bachelor', 'Master', 'PhD', 'B.Sc', 'M.Sc', 'B.A', 'M.A', 'degree', 'university', 'college']
    education_section = ''
    for line in text.split('\n'):
        if any(keyword.lower() in line.lower() for keyword in education_keywords):
            education_section += line.strip() + '; '
    return education_section.strip() if education_section else None

# Function to extract skills
def extract_skills(text):
    skills_keywords = ['Skills', 'Technical Skills', 'Core Competencies', 'Proficiencies', 'Expertise']
    skill_text = ''

    for keyword in skills_keywords:
        keyword_regex = re.compile(f'{keyword}.*?:\n?(.+)', re.IGNORECASE | re.DOTALL)
        match = keyword_regex.search(text)
        if match:
            skill_text = match.group(1).strip().replace('\n', ', ')
            break

    if skill_text:
        unwanted_phrases = ['good at', 'familiar with', 'proficient in', 'knowledge of']
        for phrase in unwanted_phrases:
            skill_text = re.sub(phrase, '', skill_text, flags=re.IGNORECASE)

        skill_text = ', '.join([skill.strip() for skill in skill_text.split(',') if skill.strip()])

    return skill_text if skill_text else None
```

```python
# Function to extract experience and 0 if no experience found
def extract_experience(text):
    experience_patterns = [
        re.compile(r'(\d+)\s+years? of experience', re.IGNORECASE),
        re.compile(r'experience of\s?(\d+)\s+years?', re.IGNORECASE),
        re.compile(r'(\d+)\s+years? experience', re.IGNORECASE),
        re.compile(r'experience\s*:\s*(\d+)\s+years?', re.IGNORECASE),
        re.compile(r'(\d+)[+\s]*years experience', re.IGNORECASE)
    ]

    for pattern in experience_patterns:
        match = pattern.search(text)
        if match:
            return match.group(1)

    return '0'

# Function to extract primary skill
def extract_primary_skill(text):
    primary_skill = {
        'Data Analyst': ['data analyst'],
        'Data Scientist': ['data scientist'],
        'Data Engineer': ['data engineer'],
        'Cloud Engineer': ['cloud engineer'],
        'Web Developer': ['web developer', 'frontend', 'backend'],
        'Digital Marketing': ['digital marketing', 'seo', 'social media'],
        'Software Developer': ['software developer', 'programmer'],
        'Software Engineer': ['software engineer']
    }

    for primary_skill, keywords in primary_skill.items():
        for keyword in keywords:
            if re.search(keyword, text, re.IGNORECASE):
                return primary_skill

    return 'Unknown'

# Function to clean text and remove illegal characters
def clean_text(text):
    if text is None:
        return ''
    return re.sub(r'[^\x20-\x7E]', '', text)  # Remove non-printable characters

# Function to process each resume
def process_resume(file_path):
    ext = os.path.splitext(file_path)[1].lower()
    if ext == '.docx':
        text = extract_text_from_docx(file_path)
    elif ext == '.pdf':
        text = extract_text_from_pdf(file_path)
    else:
        return None

    name = clean_text(extract_name(text))
    mobile = clean_text(extract_mobile_number(text))
    email = clean_text(extract_email(text))
    city, state = (clean_text(loc) for loc in extract_location(text))
    education = clean_text(extract_education(text))
    skills = clean_text(extract_skills(text))
    experience = clean_text(extract_experience(text))
    primary_skill = extract_primary_skill(text)

    return {
        'Name': name,
```

```
    return {
        'Name': name,
        'Mobile': mobile,
        'Email': email,
        'City': city,
        'State': state,
        'Education': education,
        'Skills': skills,
        'Experience (Years)': experience,
        'Primary_Skill': primary_skill
    }

# Function to process all resumes in the folder
def process_all_resumes(folder_path):
    resume_data = []
    for file_name in os.listdir(folder_path):
        file_path = os.path.join(folder_path, file_name)
        if file_path.endswith(('.docx', '.pdf')):
            data = process_resume(file_path)
            if data:
                resume_data.append(data)

    return resume_data

# Function to save extracted resume data to an Excel file
def save_to_excel(resume_data, output_file):
    df = pd.DataFrame(resume_data)
    df.to_excel(output_file, index=False)
    print(f"Data saved to {output_file}")

folder_path = r'C:\Users\DELL\Documents\RESUMES'  # Folder containing the resumes
output_file = r'C:\Users\DELL\Documents\Resume_data.xlsx' # excel filr to save extracted data
```

```
# Process resumes and save to Excel
resume_data = process_all_resumes(folder_path)
save_to_excel(resume_data, output_file)
```

```
Data saved to C:\Users\DELL\Documents\Resume_Data.xlsx
```

- Extracted data in Excel sheet-

	A	B	C	D	E	F	G	H	I	J	K
1	Name	Mobile	Email	City	State	Education	Skills	Experience (Years)	Primary_Skill		
14	HARSHAD OBEROI	9992232045	harshad.oberoi04@gm	HARSHAD C	Maharashtra	Bachelor of Science in M	Machine Learning, S	2	Data Analyst		
15	KRITIKA YADAV	9992200654	kritikaverma109@gma	KRITIKA YA	Chhattisgarh	Bachelor of Engineering	Machine Learning, S	4	Data Analyst		
16	MOHIT SEN	7000739721	mohitsen@gmail.com(	MOHIT SEN	Karnataka	Bachelor of Engineering	Python, Machine Lea	0	Data Analyst		
17	NANDINI MURTHY	7000711001	nandinimurty@gmail.c	NANDINI M	Karnataka	Bachelor of Engineering	Machine Learning, S	8	Data Analyst		
18	Neha Rajput	7000794996	neha.rajput08@gmail.(	Hyderabad	Telangana	Co Bachelor of Engineering	Java Script, C++, Rea	0	Software Developer		
19	Nikhil Shrivastava	8803609313	nikhil.shrivastava12@g	Shimla	Himachal Prade	Bachelor of Engineering	Technical Skills, Java	0	Software Engineer		
20	Nikhil Shrivastava	8803609313	nikhil.shrivastava12@g	Shimla	Himachal Prade	Bachelor of Engineering	Technical Skills, Java	0	Software Engineer		
21	Payal Shrivastava	8823093975	payalshrivastava013@g	Guwahati	Assam	Contact Bachelor of Engineering	Technical Skills, Pyth	4	Software Engineer		
22	PRAKHAR SHARMA	7000777621	prakharsharma@gmail	PRAKHAR S	Maharashtra	Bachelor of Science	Machine Learning, S	4	Data Scientist		
23	PRANAV GUPTA	9901238722	pranavgupta11@gmail.	Indore	Madhya Pradesh	Bachelor of Engineering	Java Script, C++, Rea	1	Web Developer		
24	PRASHANT SHARM	9901238766	prashantsharma101@g	Raipur	Chhattisgarh	(Bachelor of Commerce	Machine Learning, S	2	Data Analyst		
25	RAKESH KUMAR DA	7000725867	rakesh.dansena@gmai	Hyderabad	Telangana	Co Bachelor of Engineering	Technical Skills, Clou	6	Cloud Engineer		
26	RAYNA SHARMA	8871580999	raynasharma101@gma	Bhilai	Chhattisgarh	Bachelor of Engineering	Technical Skills, Java	3	Web Developer		
27	RISHAB SINGH	8837054320	rishabsingh@gmail.cor	RISHAB SIN	Punjab	Bachelor of Engineering	Machine Learning, S	1	Data Analyst		
28	RUPALI JAIN	9584238766	rupali.jain@gmail.com	Pune	Maharashtra	(Bachelor of Engineering	Machine Learning, S	3	Data Analyst		
29	Shiva Choudhary	9523093988	shivachoudhary013@g	Guwahati	Assam	Contact Bachelor of Engineering	Technical Skills, Pyth	4	Software Engineer		
30	Sneha Mishra	7000548796	sneha08@gmail.comPl	Hyderabad	Telangana	Co Bachelor of Engineering	Java Script, C++, Rea	0	Software Developer		
31	SUNIL MANJHI	7000725555	manjhi.sunil05@gmail.	Udupi	Karnataka	Cont Master of Science in Ele	Technical Skills, Clou	0	Cloud Engineer		
32	VANDANA SHARM	9901771237	vandanasharma27@gm	Bhilai	Chhattisgarh	Bachelor of Engineering	Java Script, C++, Rea	0	Web Developer		

- A VBA script is employed in Excel to clean the extracted data. This script removes duplicate records based on name, email, and mobile number. It also checks for

correct formats for emails and mobile numbers and highlights any missing information.

```
Sub CleanResumeData()

    Dim ws As Worksheet
    Set ws = ThisWorkbook.Sheets(1) ' data is on the sheet1

    Dim lastRow As Long
    lastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).row '  last row with data

    ' Track rows that will be removed due to duplication
    Dim removedRows As Dictionary
    Set removedRows = New Dictionary
    Dim currentData As Range
    Dim nameCol, emailCol, phoneCol As String
    nameCol = "A" '  names are in column A
    emailCol = "C" '  emails are in column C
    phoneCol = "B" '  phone numbers are in column B

    ' Loop through data to find potential duplicate rows
    For i = 2 To lastRow
        Dim nameVal, emailVal, phoneVal As String
        nameVal = ws.Cells(i, nameCol).Value
        emailVal = ws.Cells(i, emailCol).Value
        phoneVal = ws.Cells(i, phoneCol).Value

        ' Create a unique key for each row based on name, email, and phone
        Dim key As String
        key = nameVal & "|" & emailVal & "|" & phoneVal

        ' If the key already exists, mark the row as removed
        If removedRows.Exists(key) Then
            removedRows(key).Add i
        Else
            removedRows.Add key, New Collection
        End If
    Next i
```

```
    ' Remove duplicates based on Name (A), Email (C), and Mobile (B) columns
    ws.Range("A1:H" & lastRow).RemoveDuplicates Columns:=Array(1, 2, 3), Header:=xlYes

    ' Update last row after removing duplicates
    lastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).row

    ' Trim leading/trailing whitespace for all text fields
    Dim cell As Range
    For Each cell In ws.Range("A2:H" & lastRow) ' Adjust range as per data columns
        If IsText(cell.Value) Then
            cell.Value = Trim(cell.Value)
        End If
    Next cell

    ' Standardize phone numbers to format: XXX-XXX-XXXX
    Dim phoneRange As Range
    Set phoneRange = ws.Range("B2:B" & lastRow) ' phone numbers are in column B

    Dim regex As Object
    Set regex = CreateObject("VBScript.RegExp")
    regex.Pattern = "\D" ' Non-numeric characters pattern

    Dim phoneCell As Range
    For Each phoneCell In phoneRange
        If Not IsEmpty(phoneCell.Value) Then
            Dim phone As String
            phone = regex.Replace(phoneCell.Value, "") ' Remove all non-numeric characters

            If Len(phone) = 10 Then
                ' Standardizing phone format
                phoneCell.Value = Mid(phone, 1, 3) & "-" & Mid(phone, 4, 3) & "-" & Mid(phone, 7, 4)
            End If
        End If
    Next phoneCell
```

```
    ' Standardizing email addresses by removing everything after ".com"
    Dim emailRange As Range
    Set emailRange = ws.Range("C2:C" & lastRow) ' emails are in column C

    Dim emailCell As Range
    Dim dotComPos As Long
    For Each emailCell In emailRange
        If Not IsEmpty(emailCell.Value) Then
            ' Find the position of ".com" in the email address
            dotComPos = InStr(LCase(Trim(emailCell.Value)), ".com")
            If dotComPos > 0 Then
                ' Remove everything after ".com"
                emailCell.Value = Left(emailCell.Value, dotComPos + 3) ' Keep ".com" part and remove the rest
            End If
        End If
    Next emailCell

    ' Highlight missing values (in yellow) excluding rows that were removed due to duplication
    Dim dataRange As Range
    Set dataRange = ws.Range("A2:H" & lastRow) ' Adjust range as per data columns

    For Each cell In dataRange
        ' Check if the row has been removed due to duplication before highlighting
        If IsEmpty(cell.Value) And Not IsRowInRemovedCollection(removedRows, cell.row) Then
            cell.Interior.Color = RGB(255, 255, 0) ' Highlight missing values in yellow
        End If
    Next cell

    MsgBox "Data cleaning complete!"

End Sub
```

```
' Helper function to check if a value is text
Function IsText(val As Variant) As Boolean
    IsText = VarType(val) = vbString
End Function
```

```
' Helper function to check if a row was removed due to duplication
Function IsRowInRemovedCollection(removedRows As Dictionary, rowNum As Long) As Boolean
    Dim col As Collection
    Dim found As Boolean
    found = False
    Dim key As Variant
    For Each key In removedRows.Keys
        Set col = removedRows(key)
        On Error Resume Next
        If col.Exists(rowNum) Then
            found = True
        End If
        On Error GoTo 0
        If found Then Exit Function
    Next key
    IsRowInRemovedCollection = found
End Function
```

- Data after applying VBA pre-processing code. We can see the number of records is reduced as duplicate records has been removed, the mobile number gets standardized, and if any text got extracted along with email that has been removed too.

	Name	Mobile	Email	City	State	Education	Skills	Experience (Yea	Primary_Ski
11	MOHIT SEN	700-073-9721	mohitsen@gmail.com	MOHIT SEN	Karnataka	Bachelor of Engineering	Python, Machine Learning, Statis	0	Data Analyst
12	NANDINI MURTH	700-071-1001	nandinimurty@gmail.com	NANDINI MURTHY	Karnataka	Bachelor of Engineering	Machine Learning, Statistics, Pyth	8	Data Analyst
13	Neha Rajput	700-079-4996	neha.rajput08@gmail.com	Hyderabad	Telangana Cor	Bachelor of Engineering	Java Script, C++, React, Python, H	0	Software Developer
14	Nikhil Shrivastav	880-360-9313	nikhil.shrivastava12@gmail.com	Shimla	Himachal Prades	Bachelor of Engineering	Technical Skills, Java Script, C++,	0	Software Engineer
15	Payal Shrivastav	882-309-3975	payalshrivastava013@gmail.com	Guwahati	Assam Contact	Bachelor of Engineering	Technical Skills, Python, Java Scri	4	Software Engineer
16	PRAKHAR SHARM	700-077-7621	prakharsharma@gmail.com	PRAKHAR SHARM	Maharashtra	Bachelor of Science	Machine Learning, Statistics, Pyth	4	Data Scientist
17	PRANAV GUPTA	990-123-8722	pranavgupta11@gmail.com	Indore	Madhya Pradesh	Bachelor of Engineering	Java Script, C++, React, HTML, SQ	1	Web Developer
18	PRASHANT SHAR	990-123-8766	prashantsharma101@gmail.com	Raipur	Chhattisgarh C	Bachelor of Commerce	Machine Learning, Statistics, Pyth	2	Data Analyst
19	RAKESH KUMAR I	700-072-5867	rakesh.dansena@gmail.com	Hyderabad	Telangana Cor	Bachelor of Engineering	Technical Skills, Cloud Platforms	6	Cloud Engineer
20	RAYNA SHARMA	887-158-0999	raynasharma101@gmail.com	Bhilai	Chhattisgarh (	Bachelor of Engineering	Technical Skills, Java Script, C++,	3	Web Developer
21	RISHAB SINGH	883-705-4320	rishabsingh@gmail.com	RISHAB SINGH	Punjab	Bachelor of Engineering	Machine Learning, Statistics, Pyth	1	Data Analyst
22	RUPALI JAIN	958-423-8766	rupali.jain@gmail.com	Pune	Maharashtra C	Bachelor of Engineering	Machine Learning, Statistics, Rea	3	Data Analyst
23	Shiva Choudhary	952-309-3988	shivachoudhary013@gmail.com	Guwahati	Assam Contact	Bachelor of Engineering	Technical Skills, Python, Java Scri	4	Software Engineer
24	Sneha Mishra	700-054-8796	sneha08@gmail.com	Hyderabad	Telangana Cor	Bachelor of Engineering	Java Script, C++, React, Python, H	0	Software Developer
25	SUNIL MANJHI	700-072-5555	manjhi.sunil05@gmail.com	Udupi	Karnataka Conta	Master of Science in Electror	Technical Skills, Cloud Platforms	0	Cloud Engineer
26	VANDANA SHAR	990-177-1237	vandanasharma27@gmail.com	Bhilai	Chhattisgarh (	Bachelor of Engineering	Java Script, C++, React, HTML, SQ	0	Web Developer
27	Vishal Rajput	700-079-4217	vishalrajput@gmail.com	Indore	Madhya Pradesh	Bachelor of Engineering	Java Script, C++, React, HTML, SQ	0	Web Developer
28									

- The pre-processed data is now loaded into Power Query for further transformations necessary for analysis. This step involves adjusting data types, further cleaning of data by using split function, and replace value functions in the power query.
- Finally cleaned resume data is –

	A	B	C	D	E	F	G	H	I
1	Name	Mobile	Email	City	State	Education	Skills	Experience (	Primary_Skill
2	ALIYA SAXSENA	877-045-6789	aliya.saxsena14@gmail.com	Hyderabad	Telangana	Bachelor of Engineering	Python, SQL Server, Power BI, Advar	0	Data Analyst
3	ANUBHA SHARMA	700-075-4321	anubhasharma@gmail.com	Mumbai	Maharashtra	Bachelor of Engineering	Machine Learning, Statistics, Pytho	4	Data Analyst
4	ANUSHKA SEN	700-075-5521	anu.sen03@gmail.com	Mumbai	Maharashtra	Bachelor of Science	Python, SQL Server, Power BI, Advar	0	Data Analyst
5	ATHARAV MALHOTRA	958-421-1991	malhotra.atharav@gmail.com	Bengaluru	Karnataka	Bachelor of Engineering	Machine Learning, Statistics, Python	0	Data Analyst
6	BHAGYASHREE VERMA	882-116-7312	bhagyashreeverma@gmail.com	Vadodara	Gujarat	Master of Science	Cloud Platforms AWS, Docker, Kube	0	Cloud Engineer
7	Diksha Kapoor	882-309-3123	kapoor.diksha12@gmail.com	Amritsar	Punjab	Bachelor of Engineering	Python, Java Script, C++, SQL, Scrum	2	Software Engineer
8	Dimple Ahuja	882-309-3412	dimple.ahuja@gmail.com	Amritsar	Punjab	Master of Science	Cloud Platforms AWS, Docker, Kube	2	Cloud Engineer
9	HARSHAD OBEROI	999-223-2045	harshad.oberoi04@gmail.com	Pune	Maharashtra	Bachelor of Science	Machine Learning, Statistics, Python	2	Data Analyst
10	KRITIKA YADAV	999-220-0654	kritikaverma109@gmail.com	Bhilai	Chhattisgarh	Bachelor of Engineering	Machine Learning, Statistics, Python	4	Data Analyst
11	MOHIT SEN	700-073-9721	mohitsen@gmail.com	Bengaluru	Karnataka	Bachelor of Engineering	Python, Machine Learning, Statistics	0	Data Analyst
12	NANDINI MURTHY	700-071-1001	nandinimurty@gmail.com	Mangaluru	Karnataka	Bachelor of Engineering	Machine Learning, Statistics, Python	8	Data Analyst
13	Neha Rajput	700-079-4996	neha.rajput08@gmail.com	Hyderabad	Telangana	Bachelor of Engineering	Java Script, C++, React, Python, HTMI	0	Software Developer
14	Nikhil Shrivastava	880-360-9313	nikhil.shrivastava12@gmail.com	Shimla	Himachal Pradesh	Bachelor of Engineering	Java Script, C++, SQL, Scrum, Waterf	0	Software Engineer
15	Payal Shrivastava	882-309-3975	payalshrivastava013@gmail.com	Guwahati	Assam	Bachelor of Engineering	Python, Java Script, C++, SQL, Scrum	4	Software Engineer
16	PRAKHAR SHARMA	700-077-7621	prakharsharma@gmail.com	Mumbai	Maharashtra	Bachelor of Science	Machine Learning, Statistics, Python	4	Data Scientist
17	PRANAV GUPTA	990-123-8722	pranavgupta11@gmail.com	Indore	Madhya Pradesh	Bachelor of Engineering	Java Script, C++, React, HTML, SQL, Pl	1	Web Developer
18	PRASHANT SHARMA	990-123-8766	prashantsharma101@gmail.com	Raipur	Chhattisgarh	Bachelor of Commerce	Machine Learning, Statistics, Python	2	Data Analyst
19	RAKESH KUMAR DANSENA	700-072-5867	rakesh.dansena@gmail.com	Hyderabad	Telangana	Bachelor of Engineering	Cloud Platforms AWS, Microsoft Az	6	Cloud Engineer
20	RAYNA SHARMA	887-158-0999	raynasharma101@gmail.com	Bhilai	Chhattisgarh	Bachelor of Engineering	Java Script, C++, React, HTML, SQL, P	3	Web Developer

- Now, to store the cleaned resume data in a SQL Server database (RESUME), a VBA code is employed to facilitate a connection between Excel and SQL Server, ensuring that any new record added in the Excel sheet is automatically reflected in the SQL database.

```
Sub UpdateSQLServer()
    Dim conn As Object
    Dim connStr As String
    Dim ws As Worksheet
    Dim lastRow As Long
    Dim resumeData As Range
    Dim row As Range
    Dim query As String
    Dim name As String, mobile As String, Email As String, City As String, State As String
    Dim Education As String, skills As String, experience_years As Double, Primary_Skill As String

    ' Set the worksheet to Cleaned_resume_data
    Set ws = ThisWorkbook.Sheets("Cleaned_resume_data")

    ' Find the last row with data
    lastRow = ws.Cells(ws.Rows.Count, "A").End(xlUp).row

    ' Set the range for the data
    Set resumeData = ws.Range("A2:I" & lastRow) ' As data starts from A2 and goes till column I

    ' Connection string to SQL Server
    connStr = "Provider=SQLOLEDB;Data Source=DESKTOP-THH90DU;Initial Catalog=RESUME;Integrated Security=SSPI;"

    Set conn = CreateObject("ADODB.Connection")

    On Error GoTo ConnectionError
    conn.Open connStr
```

```
    ' Iterating through the resume data and insert it into the temp table
    For Each row In resumeData.Rows
        name = row.Cells(1, 1).Value
        mobile = row.Cells(1, 2).Value
        Email = row.Cells(1, 3).Value
        City = row.Cells(1, 4).Value
        State = row.Cells(1, 5).Value
        Education = row.Cells(1, 6).Value
        skills = row.Cells(1, 7).Value
        experience_years = row.Cells(1, 8).Value
        Primary_Skill = row.Cells(1, 9).Value

        ' SQL query to insert data into the temporary table (Staging_Resume_table table)
        query = "INSERT INTO Staging_Resume_data (Name, Mobile, Email, City, State, Education, Skills, Experience_Years, Primary_Skill)" & _
                "VALUES ('" & name & "', '" & mobile & "', '" & Email & "', '" & City & "', '" & State & "', '" & Education & "', '" & _
                skills & "', " & experience_years & ", '" & Primary_Skill & "');"

        conn.Execute query  ' Executing the query
    Next row
    ' After inserting data, executing the stored procedure to move data from staging to the main table
    query = "DECLARE @ResumeData ResumeDataType; " & _
            "INSERT INTO @ResumeData " & _
            "SELECT Name, Mobile, Email, City, State, Education, Skills, Experience_Years, Primary_Skill " & _
            "FROM Staging_Resume_data; " & _
            "EXEC Insert_Batch_Resumes @ResumeData;"

    conn.Execute query

    conn.Close
    Set conn = Nothing
    Exit Sub

ConnectionError:
    MsgBox "Connection failed: " & Err.Description
    If Not conn Is Nothing Then conn.Close
End Sub
```

- The records are initially stored in a temporary table (Staging_Resume_data) in the SQL Server database. A stored procedure is then executed to move the data from the temporary table to the main table (Resume_data).

```sql
CREATE DATABASE RESUME;

USE RESUME;

SELECT * FROM Staging_Resume_data; -- to see the resume data stored temporarily in Staging_Resume_data table.

/* Creating Final table to store resume data  after execution of stored procedure along with assigning resumeid. Logid will be assigned
on execution of trigger. This final table will be then used for further analysis. */
CREATE TABLE Resume_data (
    ResumeID INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(100),
    Mobile NVARCHAR(20),
    Email NVARCHAR(100),
    City NVARCHAR(50),
    State NVARCHAR(50),
    Education NVARCHAR(100),
    Skills NVARCHAR(MAX),
    Experience_Years NVARCHAR(10),
    Primary_Skill NVARCHAR(50)
);


SELECT * FROM Resume_data;
```

```sql
CREATE TYPE ResumeDataType AS TABLE (
    ResumeID INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(100),
    Mobile NVARCHAR(20),
    Email NVARCHAR(100),
    City NVARCHAR(50),
    State NVARCHAR(50),
    Education NVARCHAR(100),
    Skills NVARCHAR(MAX),
    Experience_Years NVARCHAR(10),
    Primary_Skill NVARCHAR(50)
);


-- Stored Procedure for inserting batch of resumes.
GO
CREATE PROCEDURE Insert_Batch_Resumes
    @ResumeData ResumeDataType READONLY -- Table-valued parameter
AS
BEGIN
    BEGIN TRANSACTION;
    BEGIN TRY
        INSERT INTO Resume_data(Name, Mobile, Email, City, State, Education, Skills, Experience_Years, Primary_Skill)
        SELECT Name, Mobile, Email, City, State, Education, Skills, Experience_Years, Primary_Skill
        FROM @ResumeData;
        COMMIT TRANSACTION;
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION;
    END CATCH;
END;
```

```sql
-- execution

DECLARE @ResumeData ResumeDataType;

INSERT INTO @ResumeData
SELECT  Name, Mobile, Email, City, State, Education, Skills, Experience_Years, Primary_Skill
FROM Staging_Resume_data;

EXEC Insert_Batch_Resumes @ResumeData;

SELECT * FROM Resume_Insertion_Log -- to check insertion log
```

- Also given a stored procedure for insertion of single resume data instead of batch of resumes.

```sql
-- Stored Procedure for single resume insertion
CREATE PROCEDURE Insert_Single_Resume
   @Name NVARCHAR(100),
   @Mobile NVARCHAR(20),
   @Email NVARCHAR(100),
   @City NVARCHAR(50),
   @State NVARCHAR(50),
   @Education NVARCHAR(100),
   @Skills NVARCHAR(MAX),
   @Experience_Years NVARCHAR(10),
   @Primary_Skill NVARCHAR(50)
AS
BEGIN
    BEGIN TRANSACTION;
    BEGIN TRY
        INSERT INTO Resume_data(Name, Mobile, Email, City, State, Education, Skills, Experience_Years, Primary_Skill)
        VALUES (@Name,@Mobile, @Email, @City, @State, @Education, @Skills, @Experience_Years, @Primary_Skill);
        COMMIT TRANSACTION;
    END TRY
    BEGIN CATCH
        ROLLBACK TRANSACTION;
    END CATCH;
END;
```

- Output on executing stored procedure: All the data get stored in the main table('Resume_data') along with the assigned ResumeID which we can see by selecting the execution of the main table.

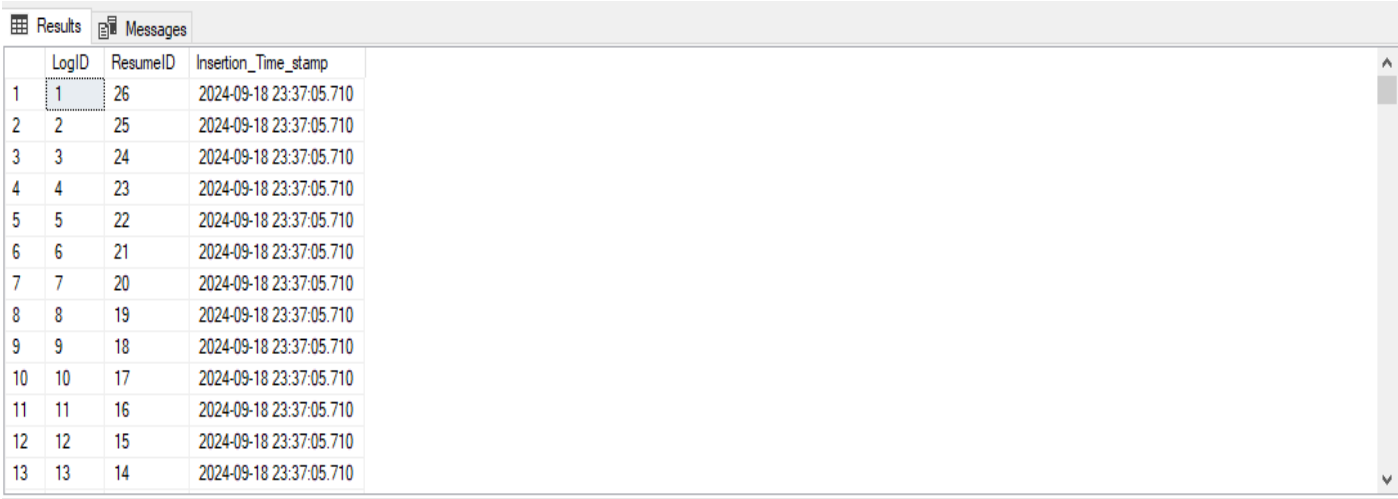
Results | Messages

	ResumeID	Name	Mobile	Email	City	State	Education	Skills	Experience_Years	Primary_Skill
1	1	ALIYA SAXSENA	877-045-6789	aliya.saxsena14@gmail.com	Hyderabad	Telangana	Bachelor of Engineering	Python, SQL Serv...	0	Data Analyst
2	2	ANUBHA SHARMA	700-075-4321	anubhasharma@gmail.com	Mumbai	Maharashtra	Bachelor of Engineering	Machine Learning...	4	Data Analyst
3	3	ANUSHKA SEN	700-075-5521	anu.sen03@gmail.com	Mumbai	Maharashtra	Bachelor of Science	Python, SQL Serv...	0	Data Analyst
4	4	ATHARAV MALHOTRA	958-421-1991	malhotra.atharav@gmail.com	Bengaluru	Karnataka	Bachelor of Engineering	Machine Learning...	0	Data Analyst
5	5	BHAGYASHREE VERMA	882-116-7312	bhagyashreeverma@gmail....	Vadodara	Gujarat	Master of Science	Cloud Platforms ...	0	Cloud Engi...
6	6	Diksha Kapoor	882-309-3123	kapoor.diksha12@gmail.com	Amritsar	Punjab	Bachelor of Engineeri...	Python, Java Scri...	2	Software E...
7	7	Dimple Ahuja	882-309-3412	dimple.ahuja@gmail.com	Amritsar	Punjab	Master of Science	Cloud Platforms ...	2	Cloud Engi...
8	8	HARSHAD OBEROI	999-223-2045	harshad.oberoi04@gmail.c...	Pune	Maharashtra	Bachelor of Science	Machine Learning...	2	Data Analyst
9	9	KRITIKA YADAV	999-220-0654	kritikaverma109@gmail.com	Bhilai	Chhattisgarh	Bachelor of Engineering	Machine Learning...	4	Data Analyst
10	10	MOHIT SEN	700-073-9721	mohitsen@gmail.com	Bengaluru	Karnataka	Bachelor of Engineering	Python, Machine ...	0	Data Analyst
11	11	NANDINI MURTHY	700-071-1001	nandinimurty@gmail.com	Mangaluru	Karnataka	Bachelor of Engineering	Machine Learning...	8	Data Analyst
12	12	Neha Rajput	700-079-4996	neha.rajput08@gmail.com	Hyderabad	Telangana	Bachelor of Engineeri...	Java Script, C++, ...	0	Software ...
13	13	Nikhil Shrivastava	880-360-9313	nikhil.shrivastava12@gmail...	Shimla	Himachal ...	Bachelor of Engineeri...	Java Script, C++, ...	0	Software E...
14	14	Payal Shrivastava	882-309-3975	payalshrivastava013@gma...	Guwahati	Assam	Bachelor of Engineeri...	Python, Java Scri...	4	Software E...
15	15	PRAKHAR SHARMA	700-077-7621	prakharsharma@gmail.com	Mumbai	Maharashtra	Bachelor of Science	Machine Learning...	4	Data Scie...
16	16	PRANAV GUPTA	990-123-8722	pranavgupta11@gmail.com	Indore	Madhya Pr...	Bachelor of Engineeri...	Java Script, C++, ...	1	Web Deve...
17	17	PRASHANT SHARMA	990-123-8766	prashantsharma101@gmail...	Raipur	Chhattisgarh	Bachelor of Commerce	Machine Learning...	2	Data Analyst
18	18	RAKESH KUMAR DANSENA	700-072-5867	rakesh.dansena@gmail.com	Hyderabad	Telangana	Bachelor of Engineeri...	Cloud Platforms ...	6	Cloud Engi...

- Additionally, a trigger is created to log the entry of each resume, generating a log ID and automatically creating a resume ID upon insertion into the main table.

```sql
-- creating resume insertion log table
CREATE TABLE Resume_Insertion_Log (
    LogID INT IDENTITY(1,1) PRIMARY KEY,
    ResumeID INT,
    Insertion_Time_stamp DATETIME DEFAULT GETDATE()
);

SELECT * FROM Resume_Insertion_Log

-- Creating Trigger
GO
CREATE TRIGGER Log_Resume_Insertion
ON Resume_data
AFTER INSERT
AS
BEGIN
    INSERT INTO Resume_Insertion_Log (ResumeID)
    SELECT ResumeID
    FROM inserted;
END;
```

- Output of resume insertion logging can be seen by selecting execution of 'Resume_Insertion_Log' table:



Results | Messages

	LogID	ResumeID	Insertion_Time_stamp
1	1	26	2024-09-18 23:37:05.710
2	2	25	2024-09-18 23:37:05.710
3	3	24	2024-09-18 23:37:05.710
4	4	23	2024-09-18 23:37:05.710
5	5	22	2024-09-18 23:37:05.710
6	6	21	2024-09-18 23:37:05.710
7	7	20	2024-09-18 23:37:05.710
8	8	19	2024-09-18 23:37:05.710
9	9	18	2024-09-18 23:37:05.710
10	10	17	2024-09-18 23:37:05.710
11	11	16	2024-09-18 23:37:05.710
12	12	15	2024-09-18 23:37:05.710
13	13	14	2024-09-18 23:37:05.710

So, the resumes were inserted into the database, on what date and time they were inserted was logged, and a unique log ID was also assigned to it.

- Then we developed an XGBoost Machine Learning model to predict the primary skill of the candidate based on the cleaned resume data, which we loaded from Excel to train the model and then tested with the new data.
- The model correctly predicts the primary skill for 86% of resumes in the test data

```python
# Importing libraries.
import pandas as pd
import xgboost as xgb
from sklearn.model_selection import train_test_split, GridSearchCV, cross_val_score
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
from sklearn.utils import resample
import matplotlib.pyplot as plt
import seaborn as sns

class ResumeSkillPredictor:  # creating class for prediction.
    def __init__(self, filepath, sheet_name):
        self.filepath = filepath
        self.sheet_name = sheet_name
        self.model = None
        self.label_encoder = LabelEncoder()

    def load_and_preprocess_data(self): # function to load and pre-process data.
        resume_data = pd.read_excel(self.filepath, self.sheet_name)

        # Separating Data Analyst data from other classes
        data_analyst_data = resume_data[resume_data['Primary_Skill'] == 'Data Analyst']
        other_data = resume_data[resume_data['Primary_Skill'] != 'Data Analyst']

        # Upsampling other classes to match the size of the Data Analyst class
        upsampled_data = pd.DataFrame()
        for skill in other_data['Primary_Skill'].unique():
            skill_data = other_data[other_data['Primary_Skill'] == skill]
            upsampled_skill_data = resample(
                skill_data,
                replace=True,
                n_samples=len(data_analyst_data),
                random_state=0
            )
            upsampled_data = pd.concat([upsampled_data, upsampled_skill_data])
    # Combining upsampled data with Data Analyst data
    combined_data = pd.concat([data_analyst_data, upsampled_data])

    # Shuffling the dataset
    combined_data = combined_data.sample(frac=1, random_state=0).reset_index(drop=True)

    # separating features and target into X and y.
    X = combined_data.drop(['Primary_Skill', 'Name', 'Mobile', 'Email', 'City', 'State', 'Education'], axis=1, errors='ignore')
    y = combined_data['Primary_Skill']

    # One-hot encoding the skills in 'Skills' column
    X = pd.get_dummies(X, columns=['Skills'], drop_first=True)

    # Label encoding for the target column (Primary_Skill)
    y = self.label_encoder.fit_transform(y)

    # Splitting the data into training and testing sets.
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0, stratify=y)
    return X_train, X_test, y_train, y_test

def train_model(self, X_train, y_train):  # function to train the model using GridSearchCV
    xgb_model = xgb.XGBClassifier(eval_metric='mlogloss')  # defining XGBoost model

    # Defining parameter grid for GridSearchCV
    param_grid = {
        'n_estimators': [100, 200, 300],
        'max_depth': [3, 4, 5],
        'learning_rate': [0.01, 0.1, 0.2],
        'subsample': [0.8, 1.0],
        'colsample_bytree': [0.8, 1.0]
    }
```

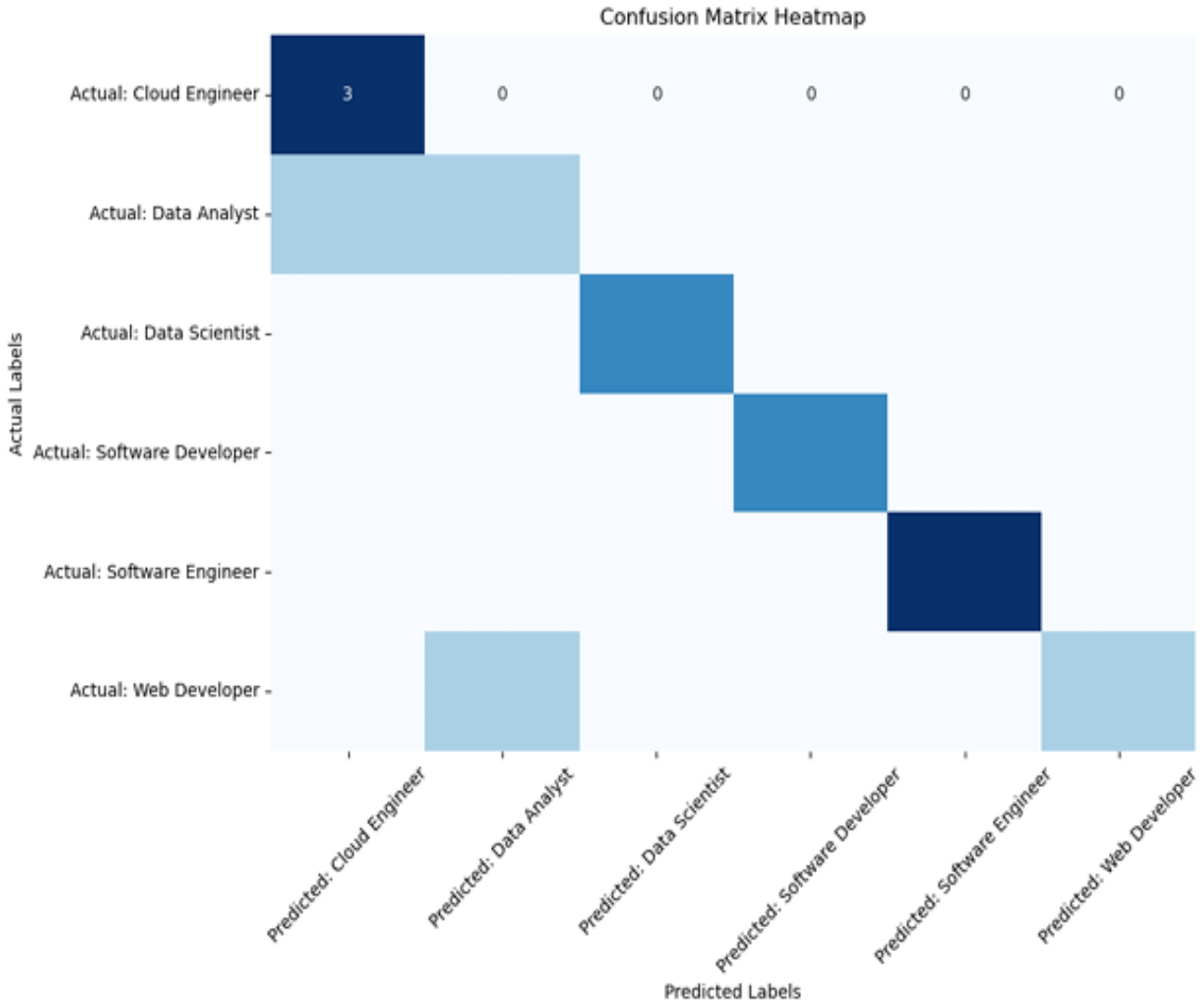
```
    # Applying GridSearchCV with 5-fold cross-validation
    grid_search = GridSearchCV(estimator=xgb_model, param_grid=param_grid, scoring='accuracy', cv=5, verbose=1, n_jobs=-1)
    grid_search.fit(X_train, y_train)

    # Getting the best model
    self.model = grid_search.best_estimator_

    # Evaluating the model using cross-validation
    cv_scores = cross_val_score(self.model, X_train, y_train, cv=5)
    print(f"Cross-validation scores: {cv_scores}")
    print(f"Mean CV score: {cv_scores.mean():.2f}")

def evaluate_model(self, X_test, y_test):    # function to evaluate the model
    y_pred = self.model.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)

    # Generating a classification report
    report = classification_report(
        y_test, y_pred,
        labels=list(range(len(self.label_encoder.classes_))),
        target_names=self.label_encoder.classes_,
        zero_division=0
    )

    print(f"Model Accuracy: {accuracy:.2f}")
    print("\nClassification Report:\n", report)

    # Generating Confusion Matrix table with all labels
    conf_matrix = confusion_matrix(y_test, y_pred)
    conf_matrix_df = pd.DataFrame(
        conf_matrix,
        index=[f"Actual: {label}" for label in self.label_encoder.classes_],
        columns=[f"Predicted: {label}" for label in self.label_encoder.classes_]
```

```
        )
        print("\nConfusion Matrix with Labels:\n", conf_matrix_df)

        # Confusion Matrix Visualization (Heatmap)
        plt.figure(figsize=(10, 7))
        sns.heatmap(conf_matrix_df, annot=True, fmt='d', cmap='Blues', cbar=False)
        plt.title('Confusion Matrix Heatmap')
        plt.xlabel('Predicted Labels')
        plt.ylabel('Actual Labels')

        plt.xticks(rotation=45)
        plt.yticks(rotation=0)

        plt.show()

    def predict_primary_skill(self, new_resume_data):    # function to make prediction for new resume data
        new_resume_data = pd.get_dummies(new_resume_data, columns=['Skills'], drop_first=True)
        return self.model.predict(new_resume_data)

# path of the cleaned extracted resume data.
filepath = r'C:\Users\DELL\Documents\Resume_data.xlsm'
sheet_name = 'Cleaned_resume_data'

# Output Generation by calling class and different functions.
skill_predictor = ResumeSkillPredictor(filepath, sheet_name)

X_train, X_test, y_train, y_test = skill_predictor.load_and_preprocess_data()

skill_predictor.train_model(X_train, y_train)

skill_predictor.evaluate_model(X_test, y_test)
```

- The output of the code gives an evaluation of the performance of the model including accuracy, classification report, and confusion matrix. The model's accuracy is 86% which is achieved by tuning the hyperparameter using the Grid Search CV technique, before tuning it was 77%.
- By Classification report and Confusion matrix we understand and evaluate the model’s performance for each class (Primary Skill), like which primary skill is being predicted precisely and for which more improvement is needed or are any skills getting confused with others.

```
Fitting 5 folds for each of 108 candidates, totalling 540 fits
Cross-validation scores: [0.81818182 0.63636364 0.7        0.9        0.8
]
Mean CV score: 0.77
Model Accuracy: 0.86

Classification Report:
                    precision    recall  f1-score   support

    Cloud Engineer       0.75      1.00      0.86         3
      Data Analyst       0.50      0.50      0.50         2
    Data Scientist       1.00      1.00      1.00         2
Software Developer       1.00      1.00      1.00         2
 Software Engineer       1.00      1.00      1.00         3
     Web Developer       1.00      0.50      0.67         2


          accuracy                           0.86        14
         macro avg       0.88      0.83      0.84        14
      weighted avg       0.88      0.86      0.85        14


Confusion Matrix with Labels:
                                Predicted: Cloud Engineer  \
Actual: Cloud Engineer                                  3
Actual: Data Analyst                                    1
Actual: Data Scientist                                  0
Actual: Software Developer                              0
Actual: Software Engineer                               0
Actual: Web Developer                                   0

                                Predicted: Data Analyst  \
Actual: Cloud Engineer                                0
Actual: Data Analyst                                  1
Actual: Data Scientist                                0
Actual: Software Developer                            0
Actual: Software Engineer                             0
Actual: Web Developer                                 1

                               Predicted: Data Scientist  \
Actual: Cloud Engineer                                 0
Actual: Data Analyst                                   0
Actual: Data Scientist                                 2
Actual: Software Developer                             0
Actual: Software Engineer                              0
Actual: Web Developer                                  0

                               Predicted: Software Developer  \
Actual: Cloud Engineer                                     0
Actual: Data Analyst                                       0
Actual: Data Scientist                                     0
Actual: Software Developer                                 2
Actual: Software Engineer                                  0
Actual: Web Developer                                      0

                               Predicted: Software Engineer  \
Actual: Cloud Engineer                                    0
Actual: Data Analyst                                      0
Actual: Data Scientist                                    0
Actual: Software Developer                                0
Actual: Software Engineer                                 3
Actual: Web Developer                                     0
```

```
                                  Predicted: Web Developer

Actual: Cloud Engineer                                   0
Actual: Data Analyst                                     0
Actual: Data Scientist                                   0
Actual: Software Developer                               0
Actual: Software Engineer                                0
Actual: Web Developer                                    1
```



- Finally, a dashboard is created in Power BI by connecting the live data from the SQL server database via Direct Query mode, so that any update in the database gets automatically reflected in the dashboards.
- SQL queries used for data visualization in Power BI are –

```sql
-- View for list of all the skills in the resume and the count applicants having those skills.

CREATE VIEW Skills_list AS
WITH Skills_CTE AS (
    SELECT       -- Cleaning the skills extracted for analysis
        ResumeID,
        LTRIM(RTRIM(REPLACE(Skill.value, 'non-', ''))) AS Raw_Skill
    FROM
        Resume_data
    CROSS APPLY STRING_SPLIT(Skills, ',') AS Skill
),
Split_Skills_CTE AS (
    SELECT
        ResumeID,
        CASE
            WHEN Raw_Skill LIKE '%collaboration and communication%' THEN 'Team Work'
            WHEN Raw_Skill LIKE '%time management and multitasking%' THEN 'Time Management'
            WHEN Raw_Skill LIKE '%Creativity and design sense%' THEN 'Creativity'
            ELSE Raw_Skill
        END AS Final_Skill_1,
        CASE
            WHEN Raw_Skill LIKE '%collaboration and communication%' THEN 'Communication'
            WHEN Raw_Skill LIKE '%time management and multitasking%' THEN 'Multitasking'
            WHEN Raw_Skill LIKE '%Creativity and design sense%' THEN 'Design Sense'
            ELSE NULL
        END AS Final_Skill_2
    FROM
        Skills_CTE
)
```

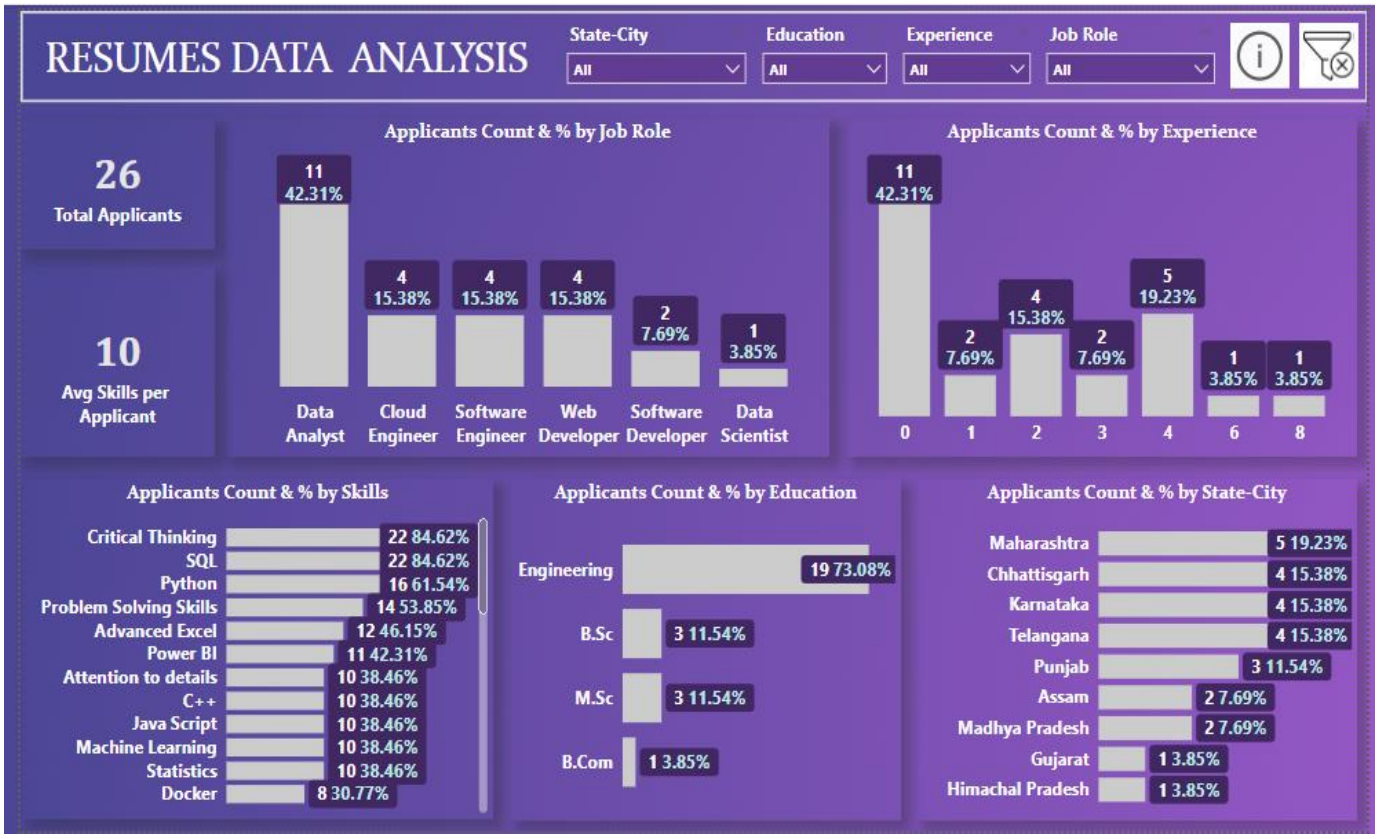
```sql
-- Combining results and counting resumes with each skill
SELECT
    Final_Skill AS Skill,
    COUNT(DISTINCT ResumeID) AS Skill_Count
FROM (
    SELECT
        ResumeID,
        CASE
            WHEN Final_Skill_1 LIKE 'attention to detail%' THEN 'Attention to details'
            WHEN Final_Skill_1 LIKE 'problem solving%' THEN 'Problem Solving Skills'
            WHEN Final_Skill_1 = 'SQL Server' THEN 'SQL'
            ELSE Final_Skill_1
        END AS Final_Skill
    FROM Split_Skills_CTE
    WHERE Final_Skill_1 IS NOT NULL

    UNION ALL

    SELECT
        ResumeID,
        CASE
            WHEN Final_Skill_2 LIKE 'attention to detail%' THEN 'Attention to details'
            WHEN Final_Skill_2 LIKE 'problem solving%' THEN 'Problem Solving Skills'
            WHEN Final_Skill_2 = 'SQL Server' THEN 'SQL'
            ELSE Final_Skill_2
        END AS Final_Skill
    FROM Split_Skills_CTE
    WHERE Final_Skill_2 IS NOT NULL
) AS Combined_Skills
WHERE LTRIM(RTRIM(Final_Skill)) <> ''
GROUP BY Final_Skill;

SELECT * FROM Skills_list order by Skill_Count desc
```

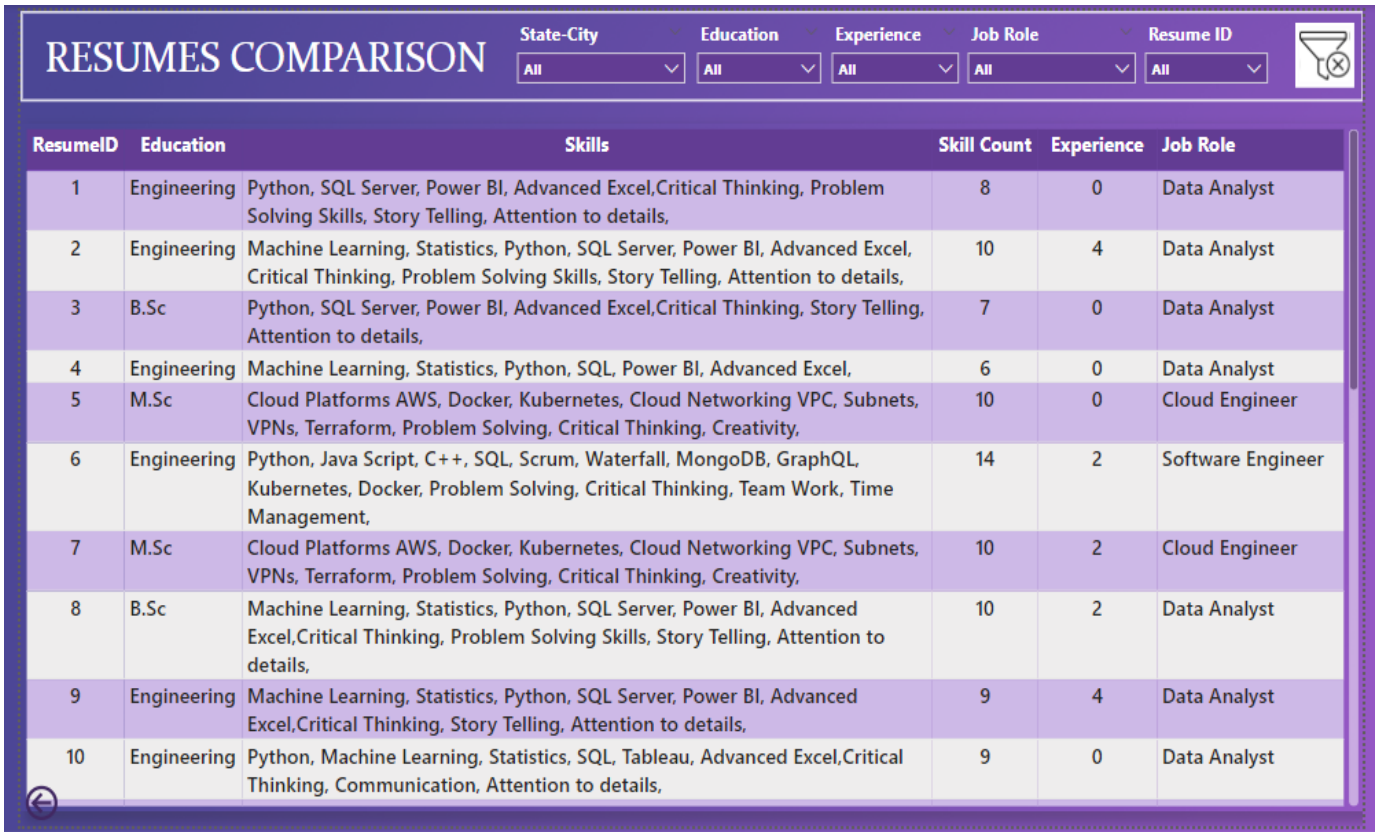
```sql
-- View to count the number of skills for each ResumeID

CREATE VIEW skills_count AS
WITH Cleaned_Skills_CTE AS (
    SELECT    -- Cleaning of extracted skills for analysis
        ResumeID,
        LTRIM(RTRIM(REPLACE(Skill.value, 'non', ''))) AS Cleaned_Skill
    FROM
        Resume_data
    CROSS APPLY STRING_SPLIT(Skills, ',') AS Skill
),
Processed_Skills_CTE AS (
    SELECT
        ResumeID,
        CASE
            WHEN Cleaned_Skill LIKE 'attention to detail%' THEN 'Attention to details'
            WHEN Cleaned_Skill LIKE 'problem solving%' THEN 'Problem Solving Skills'
            WHEN Cleaned_Skill = 'SQL Server' THEN 'SQL'
            ELSE Cleaned_Skill
        END AS Final_Skill
    FROM
        Cleaned_Skills_CTE
    WHERE LTRIM(RTRIM(Cleaned_Skill)) <> ''
)
-- Count the number of skills for each ResumeID
SELECT
    ResumeID,
    COUNT(DISTINCT Final_Skill) AS Skill_Count
FROM
    Processed_Skills_CTE
GROUP BY
    ResumeID;
```

```sql
-- view to abbrevate the education degrees

CREATE VIEW Abbreviated_Education AS
SELECT
    ResumeID,
    CASE
        WHEN Education LIKE '%Bachelor of Engineering%' THEN 'Engineering'
        WHEN Education LIKE '%Bachelor of  Engineering%' THEN 'Engineering'
        WHEN Education LIKE '%Bachelor of Engineering and Technology%' THEN 'Engineering'
        WHEN Education LIKE '%Bachelor of Science%' THEN 'B.Sc'
        WHEN Education LIKE '%Master of Science%' THEN 'M.Sc'
        WHEN Education LIKE '%Master of  Science%' THEN 'M.Sc'
        WHEN Education LIKE '%Master of Business Administration%' THEN 'MBA'
        WHEN Education LIKE '%Doctor of Philosophy%' THEN 'Ph.D'
        WHEN Education LIKE '%Bachelor of Commerce%' THEN 'B.Com'
        ELSE Education
    END AS Abbreviated_Education
FROM
    Resume_data;

SELECT * FROM  Abbreviated_Education
```

- Also, various measures are developed in Power BI using DAX formulas like COUNT, COUNTA, AVERAGEA, AVERAGE, CALCULATE, ALL, DIVIDE, and SUM.

- Dashboard Showcase





ResumeID	Education	Skills	Skill Count	Experience	Job Role
1	Engineering	Python, SQL Server, Power BI, Advanced Excel,Critical Thinking, Problem Solving Skills, Story Telling, Attention to details,	8	0	Data Analyst
2	Engineering	Machine Learning, Statistics, Python, SQL Server, Power BI, Advanced Excel, Critical Thinking, Problem Solving Skills, Story Telling, Attention to details,	10	4	Data Analyst
3	B.Sc	Python, SQL Server, Power BI, Advanced Excel,Critical Thinking, Story Telling, Attention to details,	7	0	Data Analyst
4	Engineering	Machine Learning, Statistics, Python, SQL, Power BI, Advanced Excel,	6	0	Data Analyst
5	M.Sc	Cloud Platforms AWS, Docker, Kubernetes, Cloud Networking VPC, Subnets, VPNs, Terraform, Problem Solving, Critical Thinking, Creativity,	10	0	Cloud Engineer
6	Engineering	Python, Java Script, C++, SQL, Scrum, Waterfall, MongoDB, GraphQL, Kubernetes, Docker, Problem Solving, Critical Thinking, Team Work, Time Management,	14	2	Software Engineer
7	M.Sc	Cloud Platforms AWS, Docker, Kubernetes, Cloud Networking VPC, Subnets, VPNs, Terraform, Problem Solving, Critical Thinking, Creativity,	10	2	Cloud Engineer
8	B.Sc	Machine Learning, Statistics, Python, SQL Server, Power BI, Advanced Excel,Critical Thinking, Problem Solving Skills, Story Telling, Attention to details,	10	2	Data Analyst
9	Engineering	Machine Learning, Statistics, Python, SQL Server, Power BI, Advanced Excel,Critical Thinking, Story Telling, Attention to details,	9	4	Data Analyst
10	Engineering	Python, Machine Learning, Statistics, SQL, Tableau, Advanced Excel,Critical Thinking, Communication, Attention to details,	9	0	Data Analyst

Conclusion:

The Resume Parser project successfully automates the process of extracting key information from resumes then storing it in an SQL Server database and updating the dashboard accordingly for better decision-making. The system reduces the manual effort involved in processing resumes and provides valuable insights through data visualization. Future enhancements could include integrating real-time resume submission through web forms and refining the machine learning models for better skill prediction.