Data Collection using Web Scraping & Visualization for Job Market Analysis

# A Case Study on Data Scientist Jobs in India from Naukri.com

Author: Muhammad Kamran Syed

Course: Foundation of Information

Project: Project 1, Job Market Data Extraction & Analysis

Date: Tuesday, August 20, 2024

Table of Contents

[A Case Study on Data Scientist Jobs in India from Naukri.com 1](#_Toc175093376)

[Table of Contents 2](#_Toc175093377)

[Introduction 3](#_Toc175093378)

[Overview of Web Scraping Techniques 3](#_Toc175093379)

[Tools, Technologies and Data-Source 3](#_Toc175093380)

[Methodology 4](#_Toc175093381)

[Data Collection Process 4](#_Toc175093382)

[Challenges and Solutions 5](#_Toc175093383)

[Analysis and Interpretation of Data 6](#_Toc175093384)

[Your Ideal Job: 9](#_Toc175093385)

[Conclusion 9](#_Toc175093386)

[References 9](#_Toc175093387)

# Introduction

The job market is a dynamic environment where trends change rapidly, making it crucial for professionals and organizations to stay updated. One old yet workable solution to gather up-to-date job market data is through web scraping. This report outlines the process of using web scraping techniques using Selenium and Beautiful Soup libraries with Python, to collect and analyze data related to Data Scientist job postings in India from Naukri.com. Then utilization of this data to provide insights.

# Overview of Web Scraping Techniques

Web scraping is the process of automatically extracting publicly listed data from websites. Nowadays, due to their wide popularity, use cases, ease of use, speed, security, and effectiveness in separating the data from the presentation logic, web APIs are the preferred way to request data between 2 systems. However, there are still use cases when we are required to fall back to Web Scraping techniques, thus making it an essential tool for businesses and individuals seeking to gather large amounts of data quickly and efficiently. A few of such use cases are:

1. Job Market Analysis
2. E-Commerce Price Monitoring
3. Market Research
4. Sentiment Analysis
5. News Aggregation

However, web scraping also involves understanding and obliging the legal and ethical considerations that comes with extracting data from public websites.

# Tools, Technologies and Data-Source

Naurki.com: I choose to collect data from Naukri.com, because it doesn’t require you to log in, plus most of my required info is available on the Job Listing page, hence I don’t have to follow a drill-down approach. Plus it lets you navigate its pages using a simple https://www.{base-url}/{job-title}-jobs-in-{location}-{page-no} scheme.

Python: A general-purpose high-level programming language that is widely used for data collection and analysis, especially within the Data Science community.

Selenium: A powerful tool/library for automating web browsers. Selenium implements the W3C WebDriver protocol and allows users to navigate web pages, interact with elements, and scrape data from dynamic websites.

Beautiful Soup: A Python library for parsing HTML and XML documents.

lxml: A Python library that works with Beautiful Soup to parse HTML/XML documents.

Matplotlib.Pyplot: Another Python library for data visualization.

Seaborn: A Python library built on top of the Matplotlib library for better-looking visualization.

Pandas: A Python library for data access, storage, transformation, and manipulation.

re: A Python library for Regular Expressions.

# Methodology

Most websites reject simple web scraping attempts such as using Python's built-in ‘requests’ library. Initially, I attempted to scrape it using the request and beautiful soup combo by setting the user-agent header. However, it was not allowing me to scrape more than the 1st page with 20 records only. This data was not enough to come up with any significant analysis, hence, I decided to use Selenium. This however involves several steps, beginning with Downloading WebDriver, Setup and Configuring, Navigating Naukri.Com, Data Extraction, and Storage:

# Data Collection Process

The data collection process involved the following detailed steps:

1. Downloading a suitable Selenium WebDriver: All the popular Web Browsers, supports the W3C WebDriver protocol. For this project, I decided to use the Selenium Chrome WebDriver from Google. Firstly, we need to check the installed Chrome Browser Version and your OS platform, then download the same suitable version of Selenium WebDriver for your browser and platform. In my case the Web Driver version was 127.0.6533.120, running on the Windows X64 platform. Hence, I downloaded the WebDriver for Chrome for the same version and saved it on my local drive in the Drivers folder of my project home directory so that I could load it using a relative path from within Python code.

|  |  |
| --- | --- |
|  |  |

Initial Setup and Configuration: After that, I wrote the code for Setting up Python, Selenium, Beautiful Soup plus other required packages, and configured the Selenium WebDriver. Selenium WebDriver was configured to automate the browsing process. This included loading & setting up the ChromeDriver for Google Chrome and defining the target URL (Naukri.com).

|  |  |
| --- | --- |
|  |  |

1. Navigating the Website: Using Selenium to navigate Naukri.com, enter the search team for "Data Scientist" jobs in “India”, and interact with dynamic elements such as pagination. Selenium was passed a dynamically constructed URL by a Python function to navigate to the next page (if available). The function can either parse a fixed number of pages or could use the information available on Naukri.Com's currently scraped page to deduce if the next page is available.
2. Data Extraction: Using Beautiful Soup and lxml parser to parse the HTML content and extract relevant job details, such as titles, company, company ratings, locations, experience, ratings, salary range, job description, posted date, and required skill set. I, decided to extract 50 pages (950+) of Job listings from Naukri.com and saved them in a CSV file. Since collecting all the info would take a significant amount of time.

|  |  |
| --- | --- |
|  |  |

1. Data Storage: After extraction, I Stored the extracted data in a CSV file format to avoid scraping it every time program is executed and performing further analysis. For this project, I scraped 900+ records of posted jobs from Naukri.com and saved them in a CSV file on the local disk.

|  |  |
| --- | --- |
|  |  |
|  | |

# Challenges and Solutions

While scraping data from Naukri.com, several challenges were encountered:  
  
1. Dynamic Content: Some page content was loaded dynamically. Selenium's ability to interact with JavaScript allowed for proper handling of such content.

2. Use of Cookies and Headers: Websites like Naukri.Com uses Cookies and Headers to deduce the request origin from a browser, a simple Request and Beautiful Soap-based scraping would not work and we have to use a browser emulator environment like Selenium Web Driver.

3. Timeouts and Delays: Managing timeouts and implementing delays between actions helped avoid being blocked by the website.

4. Broken Pages: During the execution of the program if there is latency or the Page is not loaded correctly, the program's attempts to scrape rendered HTML gets failed. We need to add appropriate delays after the call to load the URL using WebDriver to make sure the page is loaded fully before we attempt to parse the HTML.

5. Hidden Job Listings: The total number of Jobs count does not match the actual scraped Jobs. This is due to hidden job listings which is not listed on the Job Page. Hence the total number of jobs details parsed are not exactly a match with the Job listing Page \* Per Page Job Count. We ignored such discrepancies.

7. Missing / Inconsistent Info: Not all the listed jobs contain all the required info. For example, most of the jobs are without any Salary-Range, single job posting contains multiple locations data, Required Experience format is not consistent, almost all the data is categorical in nature instead of numerical such as Experience, Salary, etc. I used data transformation to partly overcome such issues in the analysis phase.

|  |  |
| --- | --- |
|  |  |
|  |  |

# Analysis and Interpretation of Data

The collected data was analyzed to identify trends in the Data Scientist job market in India. Preliminary analysis revealed key insights such as the top job titles, top 10 companies, top job locations, top 15 required skills, salary ranges, and required experience for Data Scientists jobs. Below are some of the details graphs:

|  |
| --- |
| Most of the job listings are for Data Scientist and Senior Data Scientist roles |
| There is a high demand for Data scientist roles in Bengaluru, Hyderabad, Pune, Chennai, and Mumbai areas. |
| IBM, Diverse Lynx, Response Informatics, Ericsson, Walmart, Xoom, Paypal, Winning Edge, Fusion Plus, and Global Pharma Tek are the top 10 employers for Data Scientists. |
| ML, Data Science, Python, Data Analysis and Analytics, SQL, NLP, Computer Science, Data Mining, AI, Predictive Modeling, and Data Visualization are top-in-demand skills for a Career in Data Science. |
| Most Jobs demand an experience of 7 to 10 years. |
| Average Salary per Annum is around 2.2 million Indian Rupees per Annum. Salary ranges are normally from 1.4 million to 3 million INR per year. However, this data can’t be trusted as there are too few Job Postings with a salary range. |
|  |
| It’s not possible to deduce a relationship between, years of experience & salary and location & salary due to insufficient salary data. However, we can assume that a higher-demand location would yield better earning opportunities. Your Ideal Job: After looking at different visualizations, we come up with the following parameters for our ideal job:  Top Job Title: Data Scientist.  Required Skills: ML, DS, Python, Data Analysis, SQL, NLP, Data Mining, AI and Deep Learning.  Location to target in India: Bengaluru  Companies to target: IBM, Diverse Lynx, Response Informatics, Ericsson & Walmart  Expected Salary Range: 2.2 million INR per annum (not conclusive due to insufficient data) |

# Conclusion

Web scraping is a powerful tool for real-time job market analysis. By automating the data collection process, it is possible to gather large datasets that provide valuable insights into market trends. This study demonstrated the effectiveness of using Selenium, Python, and Beautiful Soup to collect job market data, with a focus on Data Scientist positions in India.

# References

1. Selenium Documentation: <https://www.selenium.dev/documentation/webdriver/>   
2. Beautiful Soup Documentation: <https://www.crummy.com/software/BeautifulSoup/bs4/doc/>

3. Real Python: <https://realpython.com/beautiful-soup-web-scraper-python/>

4. Real Python Seaborn: <https://realpython.com/python-seaborn/>

4. Chrome Driver: <https://developer.chrome.com/docs/chromedriver/downloads>

5. W3C Pandas Tutorial: <https://www.w3schools.com/python/pandas/default.asp>