"Linear Algebra Project Report"

Prepared for

Dr. Mohammad Taheri

TA

Ali Shahsavandi

Prepared by

Ali Maher, Amir Hossein Ezzati

13th of Jul 2024

Introduction

This project involves using linear algebra concepts to analyze and predict product prices based on data extracted from e-commerce websites. By selecting a product category and employing web scraping tools to gather relevant features like price, rating, and reviews, and will transform this data for use in a linear regression model. The model will be trained and evaluated on this data to predict prices.

For the project we choose to build a system for **predicting laptops prices**. In this way we choose the Digikala as our dataset reference as mentioned a good dataset in the project documentation. We faced with lots of things including data scrambling, data cleaning, implementing regression model, creating an AI mode with training, gui for prediction and other processes which explained in detail below.

Let's Get Through

- ## Part 1 (Data Collection & Data Cleaning)

For our linear algebra project, we set out to collect data on laptops to perform a regression analysis on their prices. While ready-to-use datasets were available on various websites, the project guidelines required us to gather the data ourselves. We chose Digikala, a prominent e-commerce platform, as our data source.

Initial Attempts with Beautiful Soup:

Our first approach involved using Beautiful Soup, a popular Python library for web scraping. Initially, we were encouraged by the 200 HTTP status code responses, indicating successful connections. However, we soon realized that Digikala, being a sophisticated and secure website, implemented measures to deter automated scraping. Despite receiving 200 status codes, the HTML content returned was often a decoy, containing no useful data. This revelation meant that our initial code and efforts were futile.

```
<!DOCTYPE html>
<html dir="rtl" lang="fa"><head><meta charset="utf-8"/><meta content="width=device-width" name="viewport"/><meta content="2" name="next-head-count"/><link as="style"
href="/_next/static/css/0bca0666e442d65d.css" rel="preload"/><link data-n-g="" href="/_next/static/css/0bca0666e442d65d.css" rel="stylesheet"/><noscript data-n-css=""></noscript><script
defer="" nomodule="" src="/_next/static/chunks/polyfills-c67a75d1b6f99dc8.js"></script><script defer="" src="/_next/static/chunks/webpack-b7fc7aa6c72ef4cf.js"></script><script defer=""
src="/_next/static/chunks/framework-35e707db5b77429c.js"></script><script defer="" src="/_next/static/chunks/main-0319193acc4fb322.js"></script><script defer=""
src="/_next/static/chunks/pages/_app-cf14a1a560b59b18.js"></script><script defer="" src="/_next/static/chunks/pages/routing-f9896d223dfe6806.js"></script><script defer=""
src="/_next/static/EJdJHhZaOxl7C_qEPDkI2/_buildManifest.js"></script><script defer="" src="/_next/static/EJdJHhZaOxl7C_qEPDkI2/_ssgManifest.js"></script></head><body><img
src="data:image/webp;base64,UklGRrIOAABXRUJQVlA4WAoAAAAQAAAAzwcA5wMAQUxQSGQAAAABBxARERASJP4/bySi/xn/+c9//vOf//znP//5z3/+85///Oc///nPf/7zn//85z//+c9//vOf//znP//5z3/+85///Oc///nPf/7zn//85z//
+c9//vOf//znP//5z3/+8/8FVlA4ICgOAADwqgGdASrQB+gDP/3+/3+/uzayIAgD8D+JaW7hd2EbQAnsA99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe+2TkPfbJyHvtk5D32ych77ZOQ99snIe
+2TkPfbJyHvtk5D31gAP7/hAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA=" style="width: 100%;height: 500px;position: fixed;top: 0;z-index: -100;"/><div
id="__next"></div><script id="__NEXT_DATA__" type="application/json">{"props":{"pageProps":{}},"page":"/routing","query":
{},"buildId":"EJdJHhZaOxl7C_qEPDkI2","nextExport":true,"autoExport":true,"isFallback":false,"scriptLoader":[]}</script></body></html>
```


As you see a fake HTML file is returned.

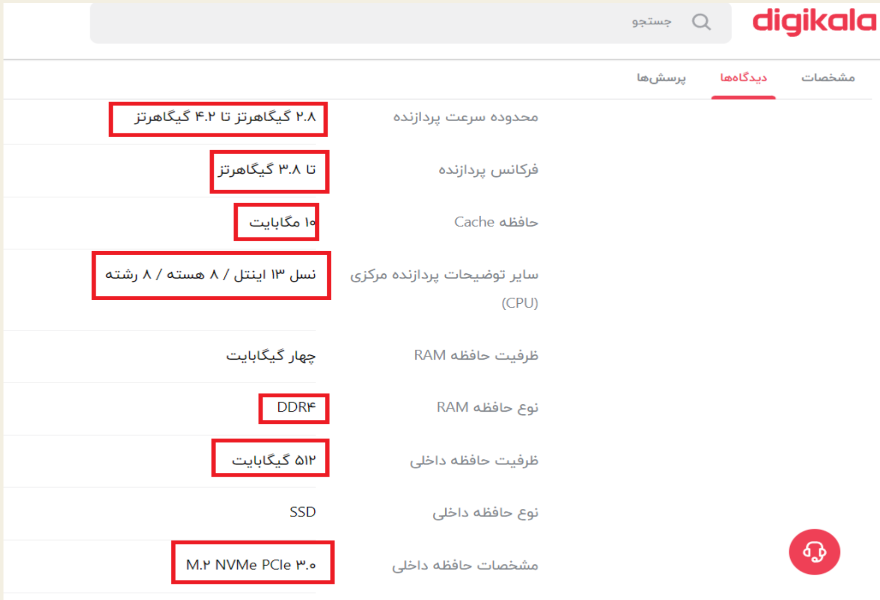
Switching to Selenium:

Given the limitations of Beautiful Soup, we shifted to Selenium, a tool designed for automating web browser interactions. Selenium provided more robust capabilities but introduced its own set of challenges. Each time we loaded a page, an intermediary screen appeared before the actual content was displayed. It took us considerable time to identify and bypass these preliminary screens to access the real HTML content.

Image Suggestion: A sequence of screenshots showing the intermediary screen and the final page content, illustrating the process of bypassing the initial screen.

Handling Diverse Data Formats:

Once we accessed the correct HTML pages, we encountered another layer of complexity. The data was presented in various formats and styles. For instance, numerical values were sometimes embedded in strings, requiring us to parse and extract specific portions. Additionally, the data varied between integers, text, and mixed formats. In some cases, laptop specifications like storage size were written out in Persian, such as "یک ترابایت" (one terabyte), necessitating conversion to numerical values like "1 TB."

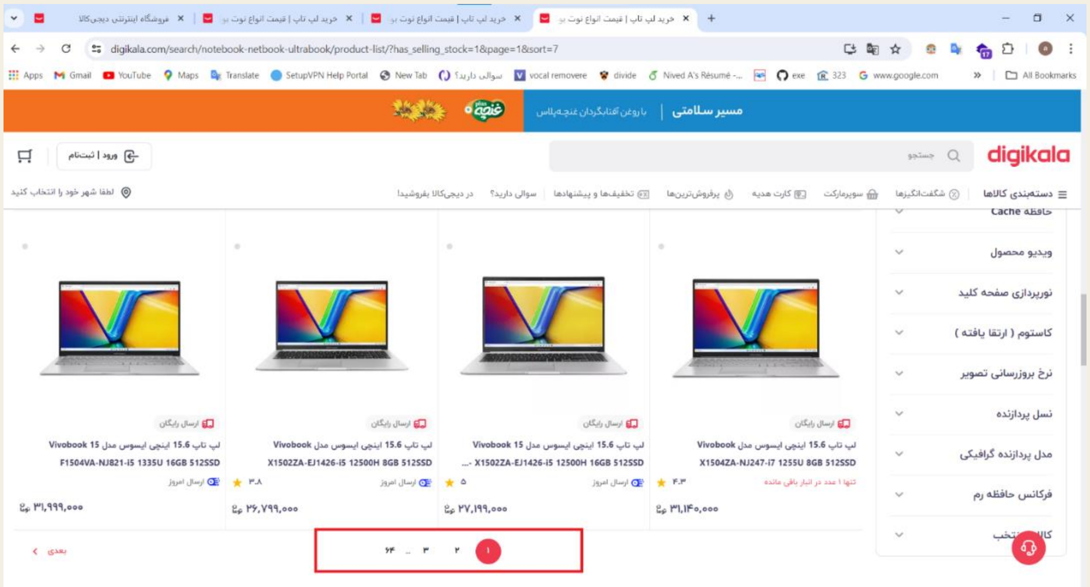


Language and Number Conversion:

Another significant hurdle was the language barrier. All the data, including numbers, was in Persian. We had to convert Persian numerals and words into English equivalents. This included translating Persian words for numbers and units into English, which was a meticulous and time-consuming process.

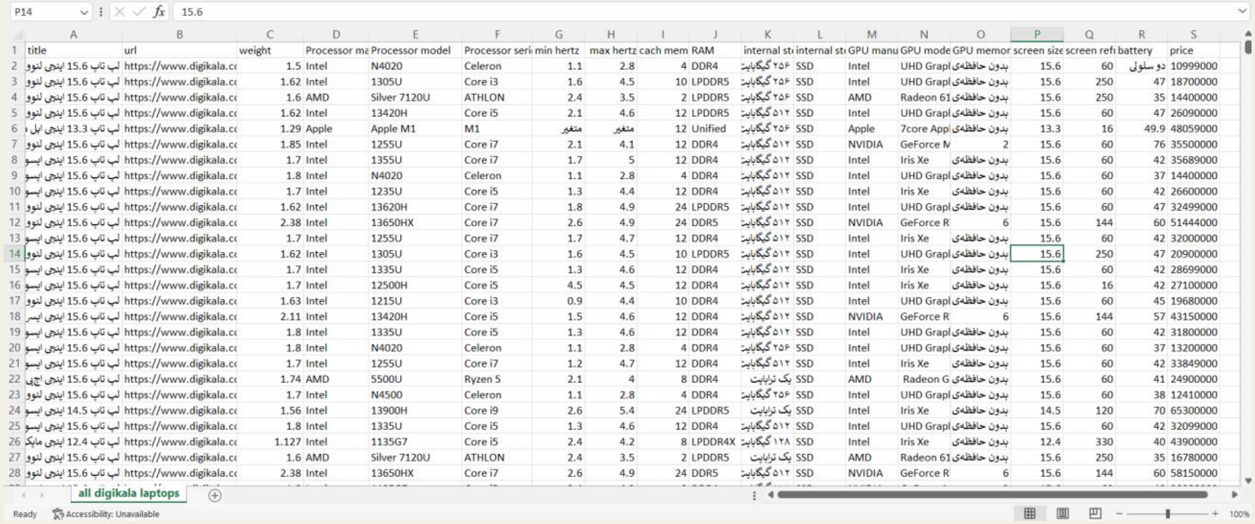
Automating the Process:

To efficiently manage these challenges, we developed a comprehensive script using Selenium. The script was designed to navigate through multiple pages, handle intermediary screens, and extract the relevant laptop information automatically. It also incorporated functions to parse, clean, and convert the data into a consistent format suitable for analysis.



Overcoming Challenges:

Despite the numerous obstacles, including misleading HTTP responses, intermediary page loads, varied data formats, and language translation issues, we successfully collected a robust dataset from Digikala. This dataset included detailed specifications and prices of laptops(1104), which we used for our subsequent regression analysis.



	title	url	weight	Processor ma	Processor model	Processor seri	min hertz	max hertz	cach mem	RAM	internal st	internal st	GPU manu	GPU mode	GPU memor	screen size	screen refi	battery	price
2	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.5	Intel	N4020	Celeron	1.1	2.8	4	DDR4	۲۵۶ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	دو سلولی	10999000
3	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.62	Intel	1305U	Core i3	1.6	4.5	10	LPDDR5	۲۵۶ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	250	47	18700000
4	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.6	AMD	Silver 7120U	ATHLON	2.4	3.5	2	LPDDR5	۲۵۶ گیگابایت	SSD	AMD	Radeon 61	بدون حافظه‌ی	15.6	250	35	14400000
5	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.62	Intel	13420H	Core i5	2.1	4.6	12	LPDDR5	۵۱۲ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	47	26090000
6	لپ تاپ 13.3 اینچی اپل	https://www.digikala.cc	1.29	Apple	Apple M1	M1	متغیر	متغیر	12	Unified	۲۵۶ گیگابایت	SSD	Apple	7core Appl	بدون حافظه‌ی	13.3	16	49.9	48059000
7	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.85	Intel	1255U	Core i7	2.1	4.1	12	DDR4	۵۱۲ گیگابایت	SSD	NVIDIA	GeForce M	2	15.6	60	76	35500000
8	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.7	Intel	1355U	Core i7	1.7	5	12	DDR4	۵۱۲ گیگابایت	SSD	Intel	Iris Xe	بدون حافظه‌ی	15.6	60	42	35689000
9	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.8	Intel	N4020	Celeron	1.1	2.8	4	DDR4	۵۱۲ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	37	14400000
10	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.7	Intel	1235U	Core i5	1.3	4.4	12	DDR4	۵۱۲ گیگابایت	SSD	Intel	Iris Xe	بدون حافظه‌ی	15.6	60	42	26600000
11	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.62	Intel	13620H	Core i7	1.8	4.9	24	LPDDR5	۵۱۲ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	47	32499000
12	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	2.38	Intel	13650HX	Core i7	2.6	4.9	24	DDR5	۵۱۲ گیگابایت	SSD	NVIDIA	GeForce R	6	15.6	144	60	51444000
13	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.7	Intel	1255U	Core i7	1.7	4.7	12	DDR4	۵۱۲ گیگابایت	SSD	Intel	Iris Xe	بدون حافظه‌ی	15.6	60	42	32000000
14	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.62	Intel	1305U	Core i3	1.6	4.5	10	LPDDR5	۵۱۲ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	250	47	20900000
15	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.7	Intel	1335U	Core i5	1.3	4.6	12	DDR4	۵۱۲ گیگابایت	SSD	Intel	Iris Xe	بدون حافظه‌ی	15.6	60	42	28699000
16	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.7	Intel	12500H	Core i5	4.5	4.5	12	DDR4	۵۱۲ گیگابایت	SSD	Intel	Iris Xe	بدون حافظه‌ی	15.6	16	42	27100000
17	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.63	Intel	1215U	Core i3	0.9	4.4	10	DDR4	۵۱۲ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	45	19680000
18	لپ تاپ 15.6 اینچی ایسر	https://www.digikala.cc	2.11	Intel	13420H	Core i5	1.5	4.6	12	DDR4	۵۱۲ گیگابایت	SSD	NVIDIA	GeForce R	6	15.6	144	57	43150000
19	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.8	Intel	1335U	Core i5	1.3	4.6	12	DDR4	۵۱۲ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	42	31800000
20	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.8	Intel	N4020	Celeron	1.1	2.8	4	DDR4	۲۵۶ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	37	13200000
21	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.7	Intel	1255U	Core i7	1.2	4.7	12	DDR4	۵۱۲ گیگابایت	SSD	Intel	Iris Xe	بدون حافظه‌ی	15.6	60	42	33849000
22	لپ تاپ 15.6 اینچی اچ‌پی	https://www.digikala.cc	1.74	AMD	5500U	Ryzen 5	2.1	4	8	DDR4	یک ترابایت	SSD	AMD	Radeon G	بدون حافظه‌ی	15.6	60	41	24900000
23	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.7	Intel	N4500	Celeron	1.1	2.8	4	DDR4	۲۵۶ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	38	12410000
24	لپ تاپ 14.5 اینچی ایسو	https://www.digikala.cc	1.56	Intel	13900H	Core i9	2.6	5.4	24	LPDDR5	یک ترابایت	SSD	Intel	Iris Xe	بدون حافظه‌ی	14.5	120	70	65300000
25	لپ تاپ 15.6 اینچی ایسو	https://www.digikala.cc	1.8	Intel	1335U	Core i5	1.3	4.6	12	DDR4	۵۱۲ گیگابایت	SSD	Intel	UHD Grapl	بدون حافظه‌ی	15.6	60	42	32099000
26	لپ تاپ 12.4 اینچی مایک	https://www.digikala.cc	1.127	Intel	1135G7	Core i5	2.4	4.2	8	LPDDR4X	۱۲۸ گیگابایت	SSD	Intel	Iris Xe	بدون حافظه‌ی	12.4	330	40	43900000
27	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	1.6	AMD	Silver 7120U	ATHLON	2.4	3.5	2	LPDDR5	یک ترابایت	SSD	AMD	Radeon 61	بدون حافظه‌ی	15.6	250	35	16780000
28	لپ تاپ 15.6 اینچی لنوو	https://www.digikala.cc	2.38	Intel	13650HX	Core i7	2.6	4.9	24	DDR5	۵۱۲ گیگابایت	SSD	NVIDIA	GeForce R	6	15.6	144	60	58150000

Still Dealing with data:

As it shown in the above picture there are still some problems with this dataset.

- Column "***min hertz***" and "***max hertz***": In these two columns we have some fields with value "متغیر", which is not a valid data (it should be a numerical value).
  - **Solution:** We surfed the web for this value and find out an acceptable value for both columns.
- "***internal storage***" Column: This Column was a terrible column, because the values were in Persian. So, first of all I cast all of them to English, then another problem with that was that the values showed in the fields were combine of **SSD** and **HDD** values. So, we need to separate them. I defined a new column as "internal_storage_hdd", and with python script, divide them and put each part in the correct filed.

```python
import pandas as pd

df = pd.read_csv('./all digikala laptops.csv')
df1 = pd.read_csv('./all digikala laptops.csv')
# converting persian data to english
df['internal storage'] = df['internal storage'].str.replace('یک ترابایت و ۲۵۶ گیگابایت', '1280')
df['internal storage'] = df['internal storage'].str.replace('۱۲۸ گیگابایت', '128')
df['internal storage'] = df['internal storage'].str.replace('۲۵۶ گیگابایت', '256')
df['internal storage'] = df['internal storage'].str.replace('۵۱۲ گیگابایت', '512')
df['internal storage'] = df['internal storage'].str.replace('یک ترابایت', '1024')
df['internal storage'] = df['internal storage'].str.replace('دو ترابایت', '2048')
df['internal storage'] = df['internal storage'].str.replace('گیگابایت', '')
# convert persian numbers to english
df['internal storage'] = df['internal storage'].str.replace('۰', '0')
df['internal storage'] = df['internal storage'].str.replace('۱', '1')
df['internal storage'] = df['internal storage'].str.replace('۲', '2')
df['internal storage'] = df['internal storage'].str.replace('۳', '3')
df['internal storage'] = df['internal storage'].str.replace('۴', '4')
df['internal storage'] = df['internal storage'].str.replace('۵', '5')
df['internal storage'] = df['internal storage'].str.replace('۶', '6')
df['internal storage'] = df['internal storage'].str.replace('۷', '7')
df['internal storage'] = df['internal storage'].str.replace('۸', '8')
df['internal storage'] = df['internal storage'].str.replace('۹', '9')

# df['internal storage'] = df['internal storage'].str.replace('و', 'and')
print(df['internal storage'])
# print(df)

df.to_csv('allDigikaLalaptops_CLEANED.csv', index=False, encoding='utf-8-sig')
```

- "***GPU memory***" Column: We replace the value "بدون حافظه‌ی مجزا" with zero.
- "***battery***" Column: Replace the value "دو سلولی" with the mean of the all rows in the battery column.
- And lots of the columns were not numerical like: "***processor manufacture***", "***processor mocel***", "***processor serie***", "***RAM***", "***internal storage type***", "***GPU manufacture***", "***GPU model***". We need to somehow convert them into numerical values. As mentioned in the documentation, we used the **one-hot encoding** approach.

- One-hot decoding: One-hot encoding is a technique used to convert categorical data into a numerical format by representing each category as a binary vector with a single high (1) value indicating the presence of that category and all other positions being low (0).

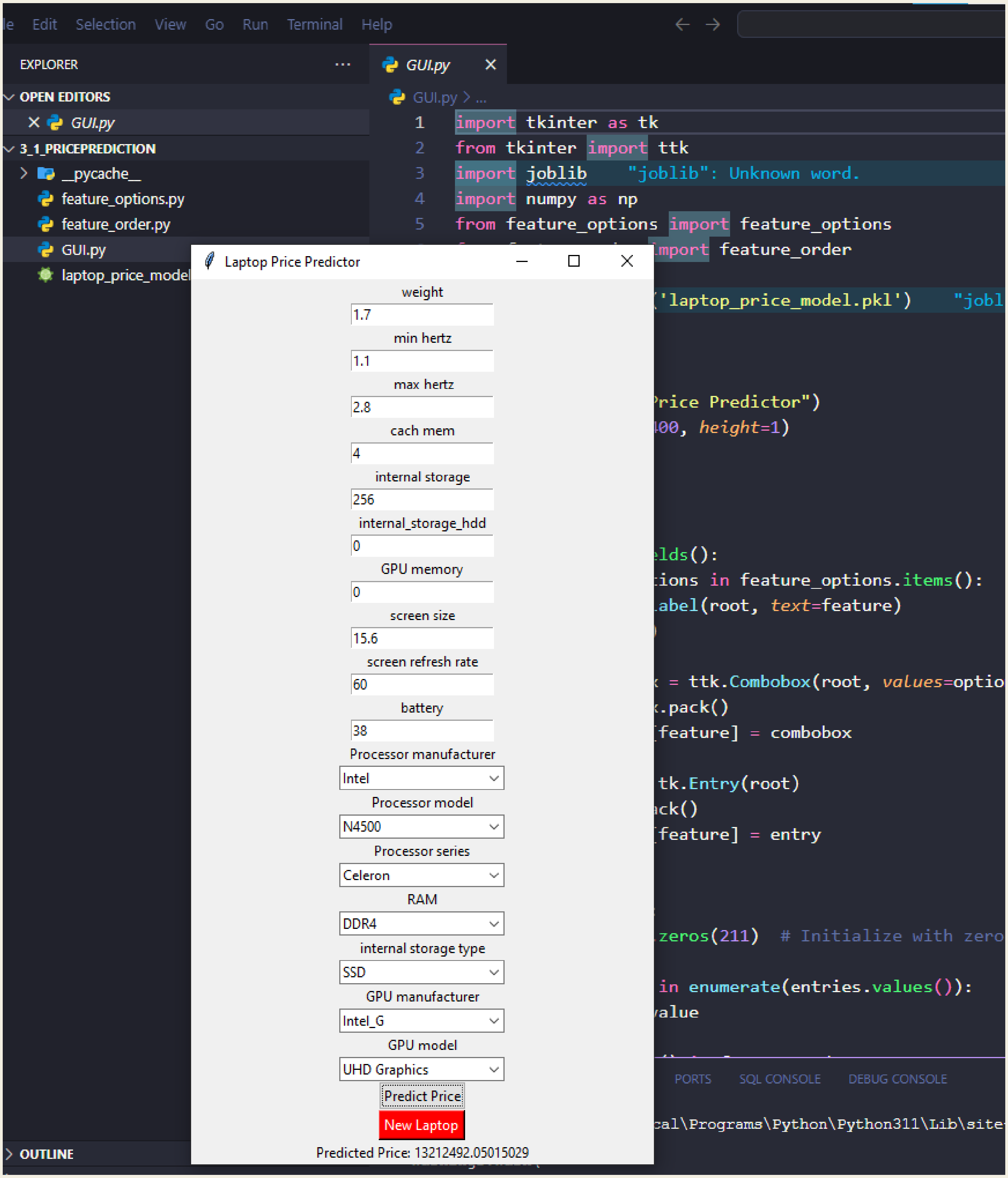
- **Part 2 (Linear Regression Model)**
  - **Data Splitting:** Split the data into training and testing sets. We consider 20 percent of our dataset as test data and 80 percent of that for the model training.
  - **Model Training:** Fit a linear regression model using the training set.
  - **Model Evaluation:** Evaluate the model's performance on the testing set using metrics like
  - Mean Squared Error (MSE) and R-squared score. We achieve about ***94 percent accuracy*** in our model.

We also save the trained model for later price prediction.

```
# Slipt data (80% for train and 20% for test)
X = df_encoded.drop('price', axis=1)
y = df_encoded['price']
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42)

# Train a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Save the model to a file
joblib.dump(model, 'laptop_price_model.pkl')

### Model efficiency calculations ###
# Predict the prices on the test set
y_pred = model.predict(X_test)

# Calculate the Mean Absolute Error (MAE) and R-squared score
mae = mean_absolute_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
```

- ## Part 3 (Price Prediction)

As the last part of the project, we prepare a good GUI for a better user experience with "tkinter".



Challenges: In the process of using tkinter, we faced some bugs. When we want to design drop down menu (as we have now in the gui), we have to give the menu items as a list to them. Here we gave the list as a python file by name "feature_options". And, also when we want to predict the price of the laptop, we need a matrix feature. So, we defined another list in a python file by name "feature_order". And the challenge pops up here. When we want to get the index of the *GPU manufacture*, for example, intel, we get the index of the *cpu manufacture*, with the same name which was intel. In hence, the prediction wan not accurate. As ***Solution***, we change the GPU manufacture names, by appending them "**_G**".

Conclusion

This project demonstrated the application of linear algebra concepts in real-world scenarios through the analysis and prediction of laptop prices based on data scraped from Digikala, an e-commerce platform. We overcame several challenges, including data scraping complexities, data cleaning, and translation of Persian numerical and textual data to English. By using Selenium for web scraping and one-hot encoding for handling categorical data, we successfully prepared a dataset suitable for linear regression analysis.

Our linear regression model, trained on this dataset, achieved a high accuracy of 94%, indicating strong predictive power. Additionally, we developed a user-friendly GUI using Tkinter, despite encountering and resolving several bugs related to menu item indexing.

Overall, this project highlights the importance of data preprocessing, the utility of linear regression in predictive modeling, and the integration of various tools and techniques to achieve a comprehensive solution. The full project files and documentation are available on GitHub for further exploration and validation.

The project files with doc are also available on my GitHub.

Thanks for your time