HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Capstone project report

Laptop Specs Insight & Prediction

Submitted by

Nguyen Nhat Minh - 20225510

Ngo Duy Dat - 20225480

Vu Tuan Truong - 20225535

Nguyen Hoang Son Tung - 20225536

Doi Sy Thang - 20225528

Guided by

Assoc.Prof. Than Quang Khoat

Hanoi, December 2024

Abstract

Laptops play a crucial role in modern life, serving a wide range of purposes, from casual browsing to high-end computational tasks. To provide a valuable insight of the laptop market, this project focuses on exploring the complex relationships between various laptop specifications, such as CPU, GPU, RAM, storage, and display, as well as their influence on price. By analyzing a massive manually collected dataset, we uncover how different components interact and contribute to overall performance and pricing. The study includes detailed information on the correlations among key features, trends in price segments, and the identification of the most impactful specifications. Furthermore, multiple machine learning models are developed to predict laptop prices and performance scores based on their specifications. The findings are presented through intuitive visualizations, enabling manufacturers and consumers to make data-driven decisions regarding design, purchasing, and marketing strategies.

Contents

Part I
Introduction

1 Motivation

Laptops have become irreplaceable tools in today's world, with infinite kinds of laptops designed differently to specialize in diverse tasks from office works, gaming to high-end computational tasks. With the rapid evolution of technology, laptop specifications have become increasingly complex, offering consumers a variety of features to consider, such as CPU performance, GPU capabilities, RAM size, storage type, display quality, and build design. These components influence not only the functionality of the laptop but also its price, making it challenging for consumers to make smart purchasing decisions and for manufacturers to optimize product offerings.

This project aims to address these challenges by carefully exploring the relationships between various laptop specifications and their impact on performance and pricing. The study uses data visualization to provide an intuitive picture of the laptop market. For example, which brand dominates which price segments, how gaming laptops and laptops for work differ in price. Furthermore, we investigate how specific configurations contribute to price and laptop performance, such as which factor has the most noticeable impacts display performance of a laptop.

Based on those derived insights, the project employs machine learning to build predictive models such as KNN, Random Forest, Gradient Boosting for estimating laptop prices and performance based on specifications. This model not only serves practical applications for consumers and retailers but also provides a better understanding about which components have the most significant effect on price. By combining data analysis and predictive modeling, this study offers a comprehensive view of the laptop market

2 Data

2.1 Data scraping& scrawling

The first objective of the project is to collect comprehensive data on laptops, specifically their configuration and market price, to enable comparisons and insights between models. For the benefit of the final product, the data should be new and updated, contains the most recent laptop models. As such, two sources is involved in our scraping process:

- **LaptopMedia.com**: This site provided detailed information on laptop models, including CPU, GPU, display, storage, RAM, and design specifications. The website also includes the market price (if available) of the models, together with the community-rated score for each laptop in various aspects in daily use, like portability, display, work and gaming.

- **TechPowerUp.com**: Known for its extensive CPU and GPU databases, TechPowerUp was used to obtain additional details on the performance and architecture of various processors and graphics cards.

The scraping process was implemented using Python script. Multiple libraries was used to help with the scraping process: **BeautifulSoup**: For parsing HTML documents, and to locate the info hidden in the parsed document for saving. **Selenium**: used to access and extract data from the websites. Selenium is specifically used, not Scrapy, because both websites relies heavily on JavaScript rendering. Beside solving the lazy-load problem, Selenium also provides tools for multithreading and handle rate-limited by the websites.

By using **BeautifulSoup**, we were able to locate and extract useful data about the laptops and CPUs/GPUs on the website. The scraping process is then validated again using the same tools, to ensure no data loss could happen. The scraping process was done through multiple days, using multiple PCs running overnight. The process was very tedious, with all of the problems occurs because of the inconsistent format and typo of LaptopMedia.com, and the heavy rate-limit to prevent scraping from TechPowerUp.com. Despite their hassle, we have completed scraping the data in about 15 days.

2.2 Data description

At the end of the process, we collected 5 data tables scraped from the two sources:

- **Laptop**: We collected 289,000 laptop entries from the sources website. Even though the number of data rows is high, the number of cells with missing data are also high. This make preprocessing data become more problematic, but also very necessary. Some of the important features of each entries are:
  - **CPU & GPU Name**: These are the names of the CPU and GPU used inside the laptop. These are two important feature. Though they do not provide any quantitative meanings to the laptop, they will be the key to integrate more features to the laptop later.
  - **RAM Memory**: This is the number of memory used in the laptop. Ram is an important feature in the laptop, and serve as an important quantitative feature for later use.
  - **Body Material**: These are the materials that are used to create the laptop. This has a big impact on the comfort of the laptop.
  - **Scores**: These are the community-rated, combined with comprehensive reviews from trust-worthy reviewers. These serves as our main way to evaluate the performance of an laptop.
  - **Price**: Straightforward, this is the price of the laptop on the market. It, of course, is very important to our analysis process.
- **CPU**: As the CPUs are collected from both sources, we obtained two different tables for CPU data(cpu.csv from LaptopMedia.com, this file focuses on CPU specifications. and best_cpu.csv from TechPowerUp.com, this file supplements the CPU data with additional details). Many CPU entries appear in both tables, some exist in only one, and others are missing entirely. This variability increases the richness of the dataset but also complicates the integration process. Some of the important features of each CPU entry are:
  - **Codename**: The internal or architecture codename of the CPU, which identifies the generation and family of processors.

  - **Core/Thread Count**: Specifies the number of cores and threads, essential indicators of CPU performance and multitasking capabilities.
  - **Frequency**: The base and boost clock speeds of the CPU, indicating processing power and performance potential under different loads.
  - **LL Cache**: The amount of cache memory, which affects data retrieval speeds and overall CPU efficiency.

- **GPU**: GPU data was collected in the same way as CPU data, resulting in two distinct tables(gpu.csv and best_gpu.csv, similarly provides expanded information on GPUs). This provides variety but also challenges for integration. Some of the important features of each GPU entry are:
  - **Architecture (Codename)**: The architecture codename of the GPU, which helps identify the GPU family and generation.
  - **Core Frequency**: The base clock frequency of the GPU, which influences processing speeds for graphics rendering.
  - **Memory Size**: The amount of VRAM, which is critical for handling high-resolution textures and supporting demanding graphics applications.

2.3 Data cleaning & integration

Each of these files underwent cleaning, leveraging Python's pandas library along with OpenRefine (formerly GoogleRefine). The following section outlines the steps taken in data cleaning.

Data Cleaning

Laptop.csv

This file predominantly consists of string-based information, directly scraped from the web. Since some entries may not follow a consistent format, our goal is to selectively extract relevant information while handling inconsistencies. Below is a description of the methods we used for data cleaning:

- **Standardization to Lowercase:** Name, OS, Body Material, CPU (with manufacturer removed), GPU (with manufacturer removed).
- **Regex Extraction:** Display (extracted attributes: size, resolution, refresh rate, panel type), Dimensions (depth), M.2 Slot, USB Type-C, USB Type-A, HDMI, Bluetooth, Ethernet LAN, Cost.
- **Manual Cleaning in OpenRefine Followed by Regex:** RAM, HDD/SSD.
- **One-Hot Encoding/Splitting:** Body material.
- **Boolean Conversion:** M.2 Slot, Security Lock Slot, Fingerprint Reader, Backlit Keyboard, Wi-Fi, Card Reader.

Cpu.csv

Similar to `Laptop.csv`, this file predominantly contains string-based information. However, unlike `Laptop.csv`, the data strictly adheres to a consistent format, likely because of the small size. This makes it easier to clean using regex:

- **Standardization to Lowercase:** Name (with manufacturer removed), Core/Architecture.
- **Regex Extraction:** Ranking, Base/Max CPU Frequency (split into separate columns), Cores/Threads (split), TDP, Max Operating Temperature, Released (converted to quarter-based float, e.g., Q3 2024 to 2024.75), Lithography, Max GPU Frequency, Base GPU Frequency, Memory Type, Max Memory, LL Cache.
- **URL Cleaning:** Official Website (keep only the middle part).
- **Dropping Columns:** cpu_link, NPU AI Performance, Total AI Performance.

Best_cpu.csv

Like `Cpu.csv`, this file also contains primarily string-based information that follows a consistent format:

- **Standardization to Lowercase:** Socket, Foundry, Core/Architecture, Generation, Memory Support, Package, Codename, Name (with manufacturer removed).
- **Regex Extraction:** Process Size, Transistors, Die Size, tJMax, Release Date, Frequency, Turbo Clock, Multiplier, TDP, Cache L1, Cache L2, Cache L3, Rated Speed, Memory Bus, Bus Interface.
- **Boolean Conversion:** Production Status, ECC Memory, Multiplier Unlocked.

Gpu.csv

This file is cleaned similarly to `Cpu.csv`, as follows:

- **Standardization to Lowercase:** Name (with manufacturer removed), Architecture, Memory Type, Drivers.
- **Regex Extraction:** Ranking, Manufacturing Process, Base Frequency, Maximum Frequency, Memory Frequency, Released, Power Consumption, Cores, Memory Capacity, Memory Bus.
- **Dropping Columns:** gpu_link, AI Cores, AI Performance.

Best_gpu.csv

Cleaned similarly to `Gpu.csv`, with:

- **Standardization to Lowercase:** Name, GPU Name, Architecture, Foundry, Generation, Bus Interface.

- **Regex Extraction:** Process Size, Transistors, Die Size, Release Date, Base Clock, Boost Clock, Memory Clock, Memory Size, Memory Bus, Bandwidth, Pixel Rate, Texture Rate, FP32, FP64.
- **Dropping Columns:** SMM Count, SM Count, Tensor Cores, RT Cores.

Data Integration

Integrate the CPU and GPU Files

Merge the Names

To integrate the datasets effectively, we created two mapping files, `cpu_key.csv` and `gpu_key.csv`, to link the names of CPUs and GPUs across the datasets. This process involved:

- Automatic matching of approximately 70% of entries in `cpu_key.csv` and 40% in `gpu_key.csv`.
- Manual review for unmatched entries, ensuring high accuracy in dataset integration.

Merge the Columns

We standardized and prioritized data based on reliability:

- For CPUs: Selected columns include name, architecture, max operating temperature, number of cores/threads, base/max frequency, release quarter, LL cache, and lithography.
- For GPUs: Selected columns include name, lithography, architecture, memory type, memory frequency, memory capacity, memory bus, base/max frequency, release quarter, and power consumption.

Integrate the CPU and GPU Data into the Laptop File

Unmatched entries in `laptop.csv` retained their original names, with unmatched data columns filled with `NaN`.

Part II
Insight

3 Basic visualization of data

To better understand the dataset and identify patterns relevance before analysis and regression part, a series of visual graph and statistics were implemented, highlighting trends and relationships among features.

Missing data analysis The graph 23 reveals that the dataset has relatively few missing values overall. However, a significant issue lies with the target variables: over 70% of the cost values are missing, and approximately 20% of all scores are also absent.

Furthermore, it is noticable that Ethernet LAN also missing a significant amount of data, nearly 30%, which make us to drop this column in the future regression part, as it is not particularly important, and we decided to remove it rather than trying to make our biased assumption to fill in the missing values.

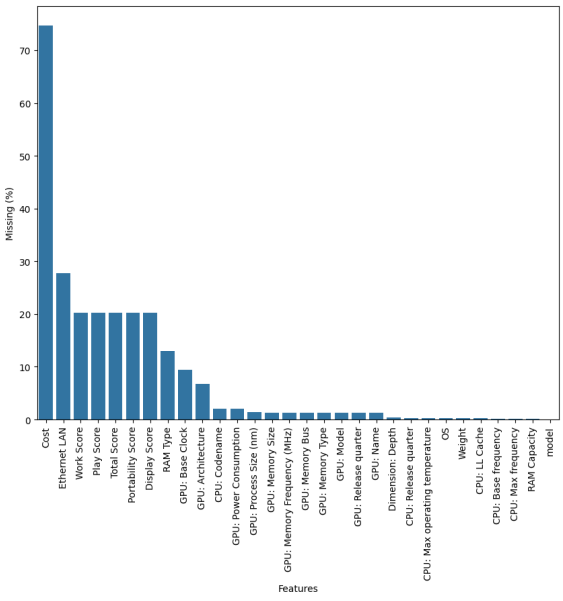

Figure 1

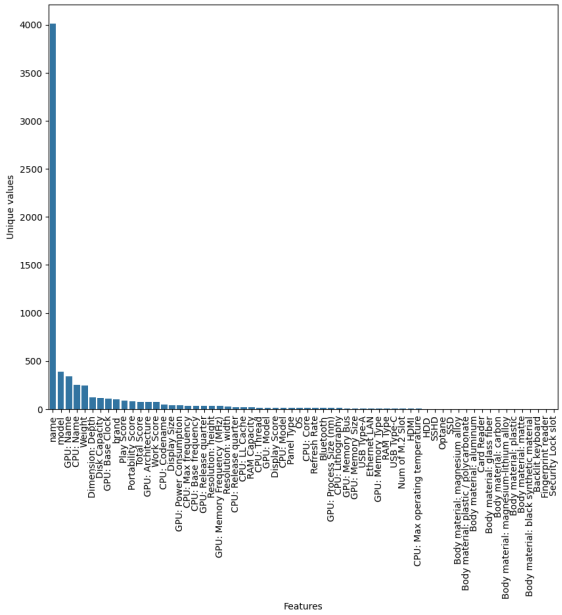

Figure 2

Unique value distribution

The diversity of variables was evaluated by visualizing the count of unique values per column in Figure 24. From the graph, we can see that except for the cost and names, each column only has around 50 to 100 unique values.

Brand and model analysis

To explore the hierarchical structure of laptop brands and their respective models, various sunburst charts were created.

Figure 3 . Pie chart of laptop brand

The graph 3 indicate that the laptop market is primarily dominated by six major brands: HP, Lenovo, ASUS, Dell, Acer, and MSI. In the low-range segment, Chromebooks, along with affordable work laptops like Lenovo's IdeaPad series and Dell's 1x generations, hold a significant share. In contrast, the high-range segment is led by flagship gaming laptops and premium creative work models from these brands. One notable thing is that each of these six major brands have laptop models across all price ranges, so we can guess that models, not brands, could have a strong correlation to cost.

Distribution analysis of the cost and scores

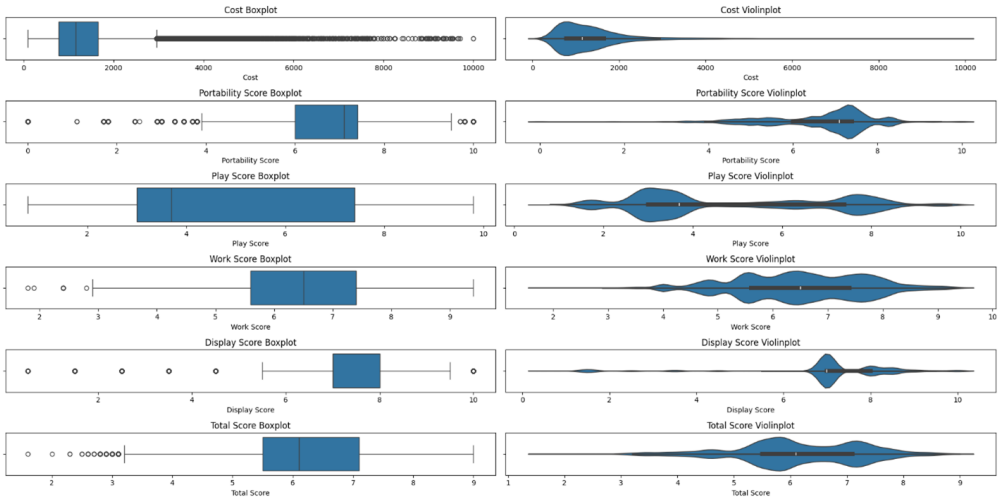

Figure 4 . Pie chart of laptop brand

The visualizations 9 highlight several important insights into the distribution of laptop features and performance. The majority of laptops are priced in the lower range, with only a small fraction occupying the premium segment, as shown by the right-skewed cost distribution and the presence of high-cost outliers. Portability scores reveal that most laptops are moderately portable, with a dense concentration around a score of 7. Work scores suggest that laptops are reasonably optimized for professional tasks, with a median score of 6.5 and a relatively even spread between 5.5 and 8. Gaming performance, however, shows a stark divide, with one peak at a score of 3 and another at 8, indicating that laptops are often either highly suitable for gaming or poorly equipped for it. Display quality tends to be above average, as most laptops cluster around a score of 7, reflecting strong screen performance. Finally, total scores indicate that most laptops fall within the mid-performance range, striking a balance between affordability and well-rounded features suitable for general use.

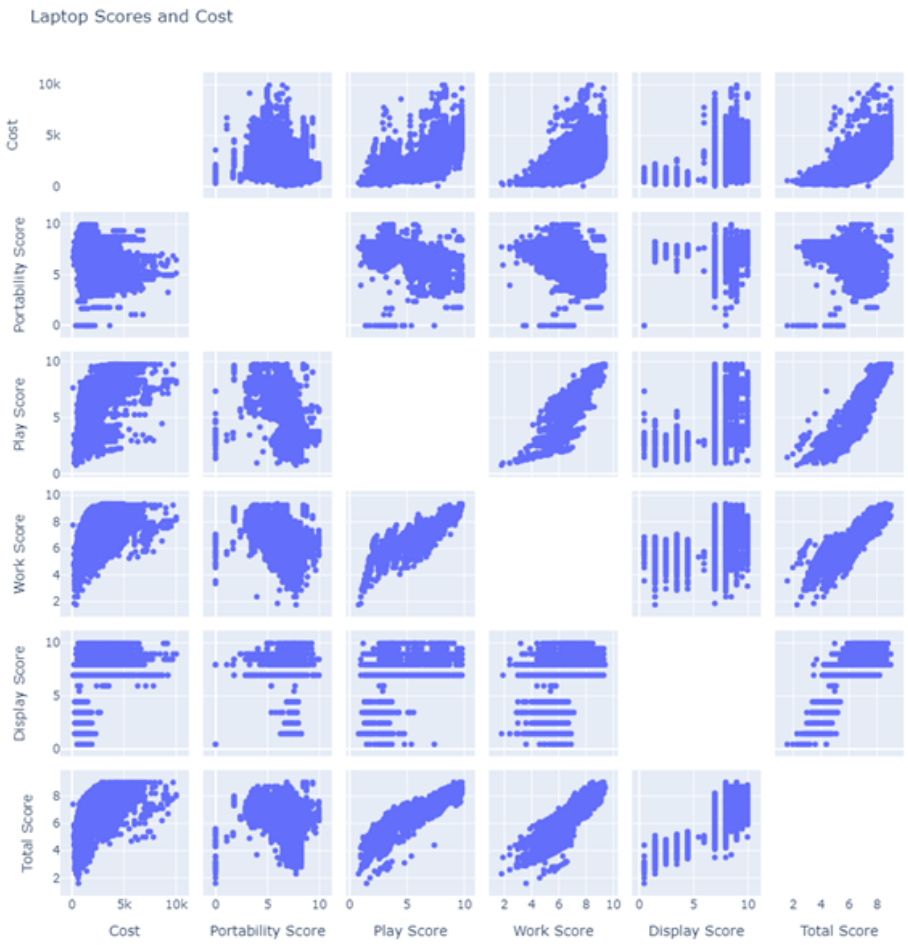

Figure 5 . Pie chart of laptop brand

The scatterplot matrix reveals key insights into the relationships between laptop features and their costs. Cost is most strongly influenced by the total score, which is very reasonable. Gaming and work performance also shows a relatively strong correlation with cost, which is consistent with the previous observation that the high-cost range is dominated by gaming and premium creative work laptop. On the other hand, portability scores are relatively independent of cost, with laptops across all price ranges exhibiting similar levels of portability. Additionally, there is a noticeable overlap between gaming and work performance, suggesting that many high-end laptops are designed to serve dual purposes. Display quality shows a modest impact on both total score and cost, reflecting its role as a contributing but not dominant factor in overall performance.

Other observations

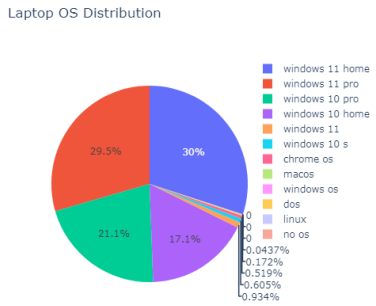

Figure 6 . Pie chart of laptop OS distribution

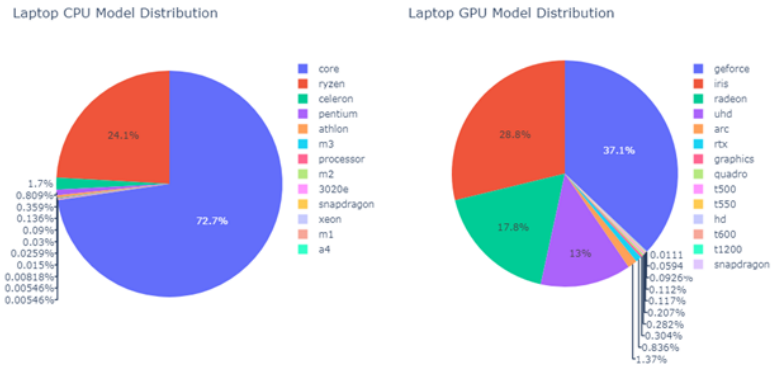

Figure 7 . Pie chart of CPU and GPU distribution

Pie chart6 shows over 60% of laptop comes preinstalled with windows 11, 28% with windows 10 and the remaining 2% for other Operating Systems.7 indicates over 70% of laptops are powered by Intel Core processors, while 24% utilize AMD Ryzen chips. In contrast, GPU distribution is more balanced, with NVIDIA GeForce, Intel Iris, AMD Radeon, and Intel UHD chips sharing a relatively even market share.

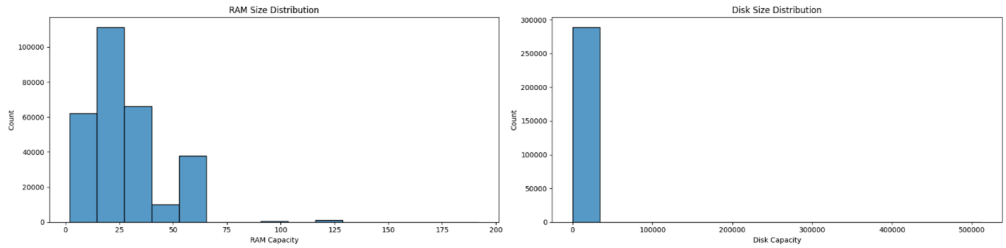

Figure 8 . RAM and Disk distribution graph

The display size follows a roughly normal distribution, with the average size around 16 inches. In contrast, RAM size exhibits a right skew, primarily driven by outliers and server laptops with massive amounts of RAM, while disk size is heavily right-skewed due to extreme outliers in the upper range.

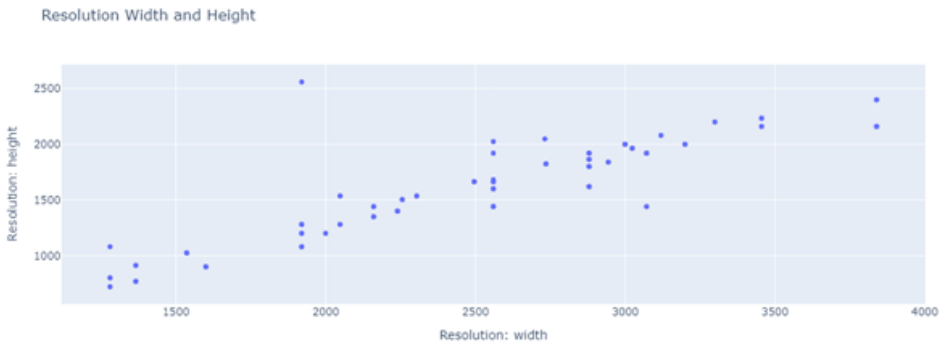

Figure 9 . Resolution width and height relationship

The resolution width and height generally maintain a consistent 16:9 ratio, with two notable exceptions: ultrawide and ultrahigh screens.

4 Exploratory data analysis

In this project, we aim to help buyer select the best laptop based on their budget X and the task Y, particularly, portability score, work score, play score, display score, total score. To achieve this, we will begin with Exploratory Data Analysis (EDA), which will involve visualizing the correlation of laptop prices and performance scores for various tasks. Additionally, we will investigate how specific laptop features—such as GPU for gaming or RAM for multitasking—affect performance scores(Regression Analysis part). Ultimately, through a combination of visualizations, statistical analyses, we will generate actionable recommendations, helping users make informed decisions and select the best laptop within their budget for their needs. To address this issue, we will systematically pose hypothesis-driven questions and then explain them based on data (visualization), followed by reasoning derived from the data. To address this issue, we will systematically pose hypothesis-driven questions and then explain them based on data (visualization), followed by reasoning derived from the data.

4.1 The problem

Restate the problem that we use EDA and hypotheses to resolve:

- **Budget**: How much they are prepared to put in for the laptop?
- **Preference**: Do they have any personal preference about the laptop? Do they specificly want a laptop of some brand, a CPU from some brand, or something else?
- **Use case**: What are they expect to use this laptop for?

As a computer market analysis, we will propose suggestion to customer based on our insights into laptop, we will analyze score and cost of laptop based on graph data visualization. After some effort in analyzing and from our experiences in laptop field, some questions raised as sub-problem we need to answer to get the final decision, which are:

- In the same price segment, which laptop brand has the largest overall performance in a specific task?
- Given a target performance/price in a specified task, which brand has the largest number of products in that segment? This will help users focus on brands that have a niche market tailored to that customer segment.
- Given the specific budget and task, what level of performance (specifically the score) can they expect? Additionally, how much more would they need to spend in order to achieve a higher performance?
- What is the trends of performance increase of CPU and GPU? How much is the increase in performance, with the same budget? Is it worth it to wait for the next generation to buy or to buy now?

4.2 Question 1: In the same price segment, which laptop brand has the largest overall performance in a specific task?

In this section, we analyze the overall performance of each brand as represented in the dataset. Specifically, we evaluate the performance of laptops across five key categories(as 5 score in our dataset): gaming task, work task, display quality, portability task, and overall performance over 5 price segment(which is cut based on 5 equally frequent bins). Particularly, we do the steps;

- Segment the dataset by price ranges to ensure fair comparisons within the same price segment(based on frequent bins).
- Calculate the average scores for each brand in the five performance categories within each price segment.
- Visualize the data using bar plots to identify the brand with the highest performance in each category for a given price range.

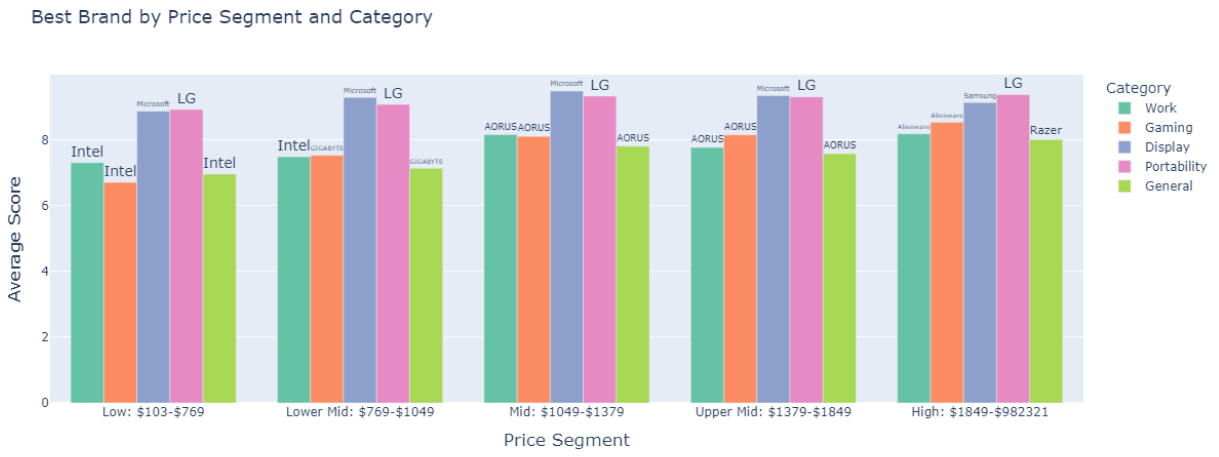

Figure 10 . Overall performance of specific task over price segment

Figure10 show that the best brands for display and portability are Microsoft and LG in almost price segment(except Samsung in High segment for Display score). This is easily proven in real scenario, with their wide lineups of lightweight with beautiful display and high battery, where they are mostly used for light task like Office and coding. As for working, gaming and overall performance, the winner are distributed proportionally. The best brands in low budget are Samsung and MSI, together with GIGABYTES/AORUS in mid and Razer/Alienware in high.

These visualizations will allow us to identify which brand provides the best overall performance in specific tasks, helping guide customer recommendations based on their budget and use case. Sellers and stores can also leverage these insights to stock and promote laptops from brands that dominate specific performance categories, aligning inventory with customer preferences. This also empowers individual customers and enhances the ability of laptop stores to meet diverse customer needs effectively, fostering greater customer satisfaction and loyalty.

Take the example of a freshman from Hanoi University of Science and Technology who wants to buy a laptop for university to play games and code. This student has a budget of around $1000, equivalent to about 25 million VND. We would recommend that he purchase a laptop from the brand AORUS, or GIGABYTE for a more budget-friendly option.

4.3 Question 2: Given a target performance/price in a specified task, which brand has the largest number of products in that segment?

This question is motivated by our belief that each brand has its own strengths and weaknesses; based on their strengths, they tend to produce a lot of laptops that exploit it. Brands that produce a wide range of devices for a specific task are likely to invest heavily in that area and offer various incentives when those products are in demand (for example, gaming laptops are often released in the time when new university students are enrolling, then many promotional policies and discounts are proposed in some gaming laptop brand). Therefore, choosing a brand that produces many laptops for a particular purpose, rather than solely relying on the score, is a reasonable option.

Moreover, when we have a deep insight into a psychology of consumers, an assumption can be raised that force us to answer question 2 to choose suitable brand for specific task: Relying solely on the score is a hasty assumption and lacks careful consideration, because we are buying one laptop, not type of multiple products from the same brand. What we are truly concerned with is the individual machine, not the average score. Moreover, scores can be heavily influenced by user psychology and their expectations of the brand.

For example, ASUS brand can be expected to always provide good gaming laptops, but unfortunately, some types of laptops have not met those expectations from the community, leading to anger for who buy it, cause lower scores. However, another brand that buyers do not have high expectations for in terms of gaming performance might be rated quite good (possibly even lower than the worst ASUS laptop), but users find it satisfying compared with their expectation and rate it highly. Therefore, these scores are sometimes not fair across brands. Particularly, a 'high score' can mean that it has surpassed its own standards, but still may not be truly good, while a 'low score' may reflect that it hasn't exceeded its own limitations but is still good.

In this section, we analyze the number of choices each brand offers in each price segment as represented in the dataset. As the above section, we also evaluate the number of laptops across six key categories: price, gaming performance, productivity performance, display quality, portability, and overall performance. To achieve this, we will:

- Segment the dataset by price ranges to ensure fair comparisons within the same price segment.
- Count the number of laptops for each brand in the five performance categories within each price segment.
- Visualize the data using bar plots to identify the brand with the highest number of laptops in each category for a given price range.

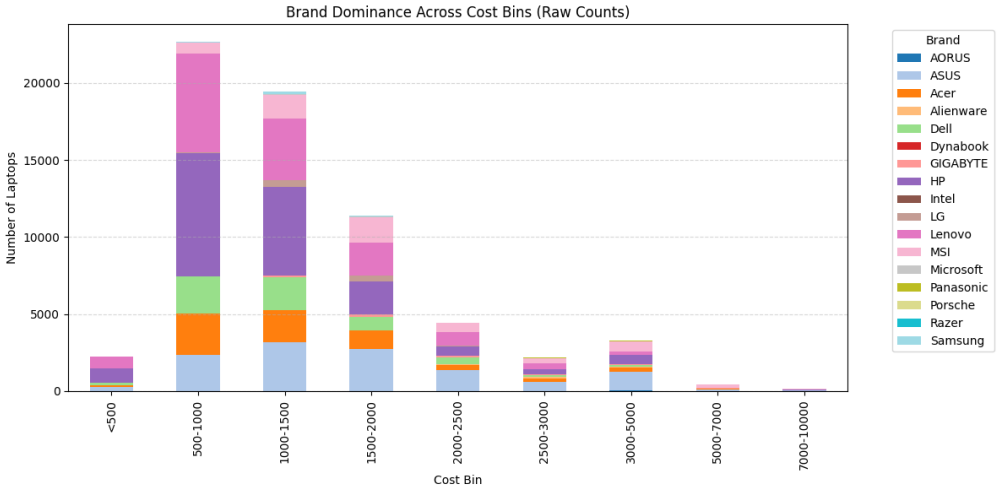

Figure 11 . Number of laptop over price segment for all task

Figure 11 indicate the king of low-cost laptops are HP and Levono, where they occupy over 70% of the market in their price segment. As the price segment goes up, many gaming/workstation laptop brands like ASUS and MSI occupy more of the market. There are also some brands like Acer and Dell, who generally have the same market percentage over all price segment.

As for the other categories:

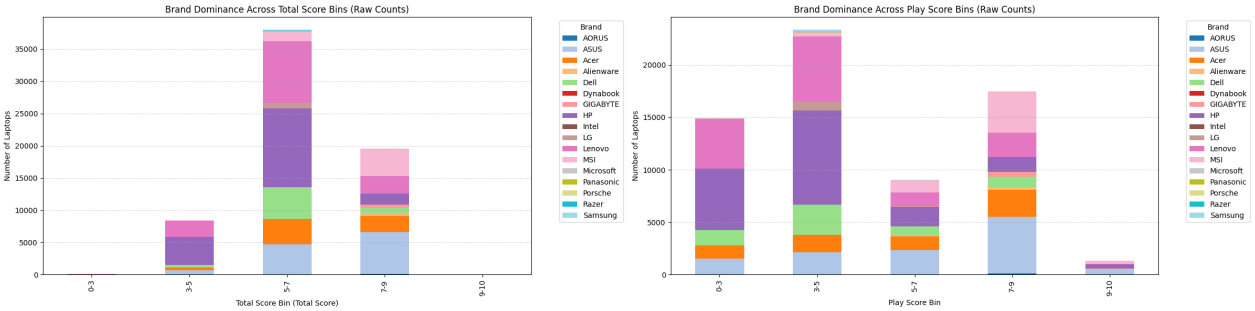

Figure 12 . No. of laptop over total score segment

Figure 13 . No. of laptop over play score segment

In the total score categories,lightweight laptop companies like HP and Levono are brands that contains mostly middle performance market. As for the high performance, gaming laptop companies like MSI and ASUS completely dominate it. At the same time, the play score categories typically follows the same trends as the total score. The major difference is that the gaming laptop market is typically very big compared to other fields, and so that multiple brands don't focus on just a single performance type, and so they target multiple part of the market. The market looks very balanced, with big brands spread out on every point segments, while small brands get it's own part in the market.

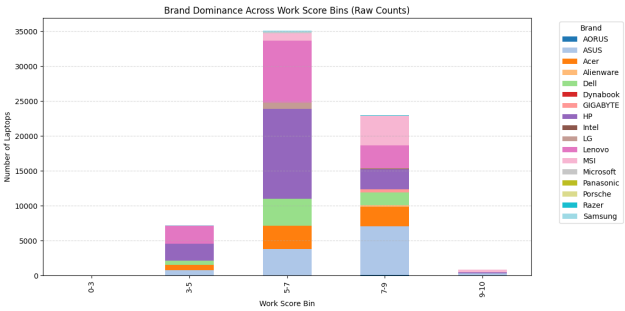

Figure 14 . No. of laptop over work score segment

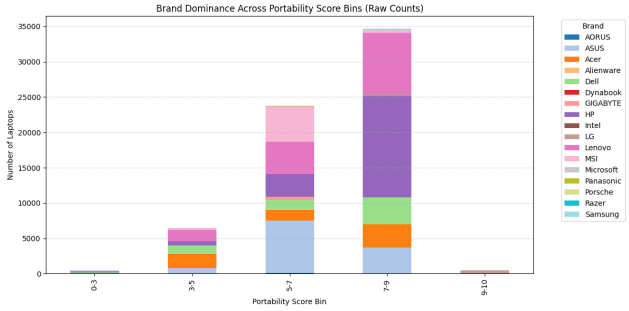

Figure 15 . No. of laptop over portability score segment

The work score, again, show the big domination of HP and Lenovo in the mid score segment. This is because while gaming laptop have good performance for working, the working score also consider multiple other aspect, like battery life,etc... These aspects make multiple gaming laptop companies spread out in the low-mid/mid-high score segment. At the same time, most gaming laptop brands have mid-high scores. This shows that at least gaming laptops can also serve as a good laptop for working heavy task like image edits or rendering. As for the portability score, most of the gaming laptop brands reside at the mid score segment, while companies that focus on work laptops like LG, HP or Lenovo reside mostly on the mid-high and high segment. Especially with LG, where over 75% of their laptops reside at the high score segment. This is proved in real scenario, as LG is very famous for it's lightweight laptop with very low weight and almost perfect portability ability.

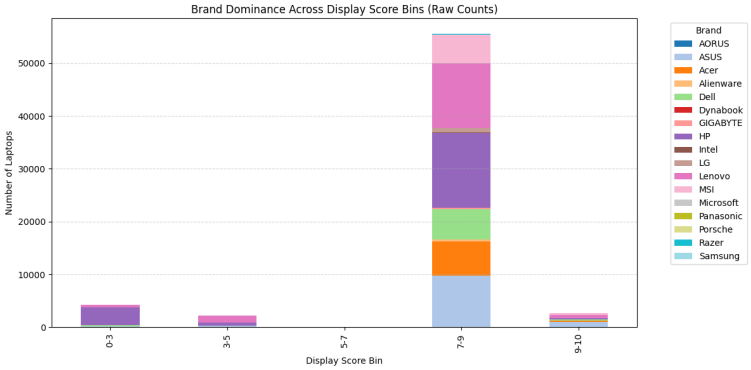

Figure 16 . No. of laptop over display score segment

Finally, for the display score, it is not surprising to see all of the laptops sitting at the mid-high score segment. Over the years, the FHD screen resolution, with 60Hz base refresh rate has become a normal in the laptop industry. While there are some old laptops with old displays, which explain some low score laptops, most of them now follow this industry formula, which make up most of the laptops. Some brands, however, can produce very good display screen (QHD+ or higher) with gaming-competitive refresh rate as high as 300Hz. This make them the exception that can produce very good laptops screens.

These visualizations will allow us to identify which brand has the largest number of products in the customer's budget segment. This insight is particularly valuable, as companies often cater to

specific market segments with a diverse range of offerings. Brands with a larger number of products in a specific segment are more likely to provide options tailored to different preferences and use cases, increasing the likelihood of meeting customer needs. Furthermore, identifying brands with strong representation in a given segment helps highlight their market strategy, such as focusing on affordability, premium features, or specific performance metrics. They also tends to have better customer services toward these market segment. This information can guide customers toward brands that are more likely to have a suitable product, simplifying their decision-making process.

4.4 Question3: Given the specific budget and task, what level of performance (specifically the score) can they expect? Additionally, how much more would they need to spend in order to achieve a higher performance?

One of the common sales techniques is that sellers often entice buyers: 'Just add a little more money and get a product that is better several times than the once you wanna buy in the first time.' This can sometimes be true (a strategy to encourage customers to purchase a brand's main products), but at other times, it can be just a trick from sellers, which can confuse consumers. Therefore, in this section, we will address two sub-questions: With X amount of money for Y purpose, how much score can buyer get when buying a device? How much more does consumer need to spend to achieve a higher score, and is that spending extra money worth?

To answer these questions,we sketch bar graph, which are the cost to increase the performance of the laptop for each task. To achieve this, we will:

- Segment the dataset by score ranges in each bin.
- Calculate the average cost in each score ranges.
- Visualize the data using bar plots to identify the average cost of general score ranges to show their differences.

As the above section, we evaluate the number of laptops across three key categories: gaming performance, productivity performance and overall performance.

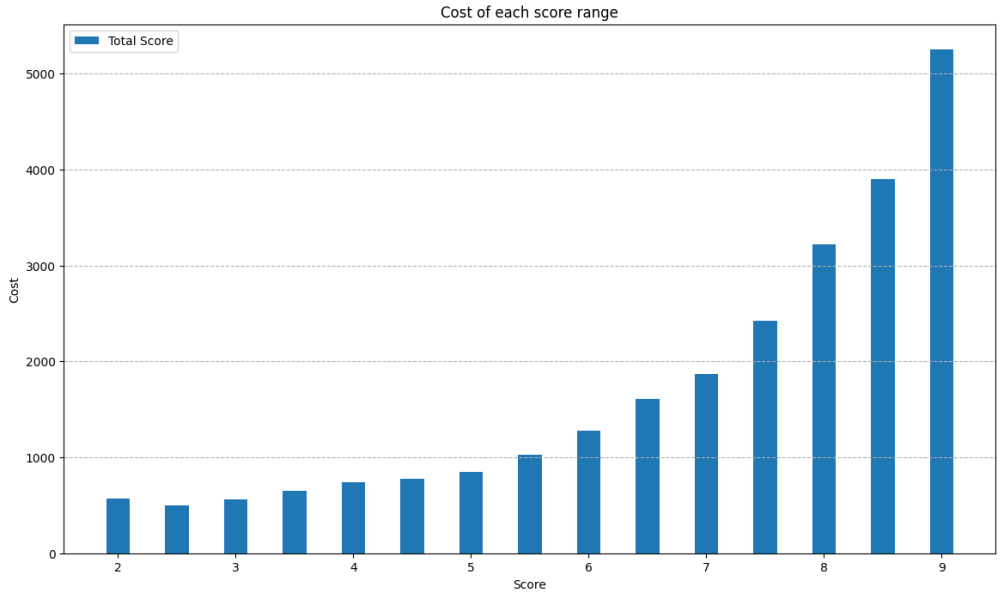

Figure 17 . Percentage of laptop number over price segment for all task

As you can see in the Figure 17, the general plot follows an exponential relationship between price and general score. This is easily proven in real scenario. The important information, in this case, is the difference between bars. This is the general cost that the user will pay to increase the performance of the laptop by some percent. As the score become higher, the cost needed to raise by the same amount of score increase exponentially.

These visualizations will allow us to identify the relationship between cost and performance, helping customers understand what level of performance they can expect within their budget. This is particularly valuable for setting realistic expectations and making informed decisions. Additionally, the insights from this analysis can highlight the diminishing returns in performance improvements as the price increases. For instance, customers may find that spending significantly more may not yield proportional performance gains, especially in high-performance segments. Conversely, in lower budget ranges, smaller investments may lead to more substantial improvements. By quantifying the cost-to-performance trade-offs, this analysis enables better alignment of customer expectations with their financial constraints and desired performance outcomes. This information can also guide targeted recommendations, ensuring customers achieve the best possible value within their budget.

4.5 Question 4: What is the trends of performance increase of CPU and GPU? How much is the increase in performance, with the same budget? Is it worth it to wait for the next generation to buy or to buy now?

In this section, we analyze the trends in performance improvement across CPU and GPU generations, specifically focusing on whether the increase in performance justifies waiting for the next generation or purchasing a current model. This analysis is based on the dataset, segmented by CPU and GPU models across different generations and price ranges.

To achieve this, we will:

- Segment the dataset by generation/codename for both CPUs and GPUs.

- Compare the performance scores of each generation/codename within the same price range to measure the generational improvement.
- Calculate the average performance gain per unit of cost across generations.
- Visualize the data using scatter plots to show performance, cost and release year of each CPU/GPU generation pairs.

Here's an example of the result visualizations:

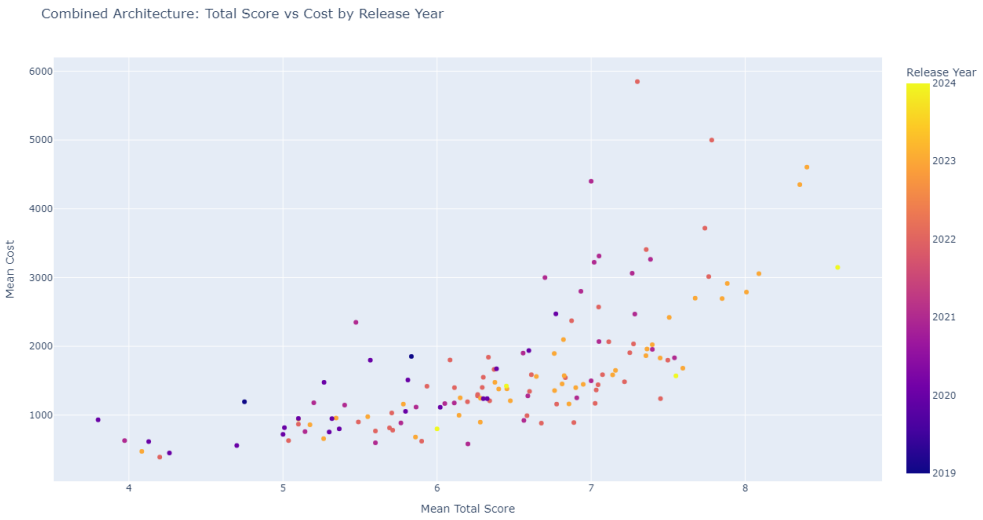

Figure 18 . Scatter plot of architecture pairs and their mean total scores

As you can see in the image, in every generation of the same year, the total performance/cost always have diminishing return: high increase in performance at low cost, but low increase at high cost. This diminishing effect is weaker in older generations, where the top performance is smaller than in newer generation. At the same time, we can see that the newer generations always have an increase of 0.3-0.5 points than in older generations, at the same price segment.

The above statement can also be said for working/playing performance:

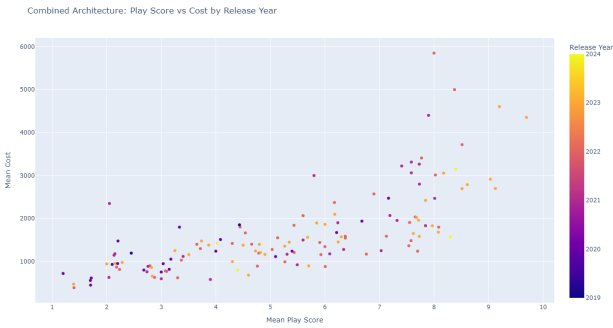

Figure 19 . Play score segment

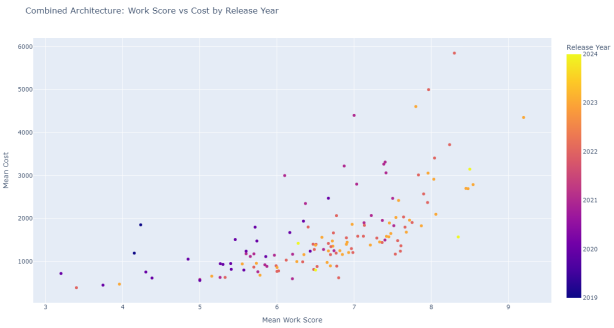

Figure 20 . Work score segment

These visualizations will allow us to identify the performance trends across CPU and GPU generations, helping customers evaluate the value of waiting for a new generation versus purchasing a current model. Specifically, this analysis reveals:

- The diminishing returns in performance gains for higher-cost models within the same generation, which is particularly helpful for budget-conscious buyers.
- The consistent performance improvements in newer generations, typically ranging from 0.3 to 0.5 points in the same price segment, showcasing the technological advancements over time.
- The trade-offs between waiting for newer generations versus opting for an older generation, especially for customers looking for value in mid-range or budget segments.

This information is invaluable for guiding purchase decisions. If a customer prioritizes having the latest technology or seeks significant performance improvements, waiting for the next generation may be worthwhile. However, for customers who need a laptop immediately or do not require cutting-edge performance, choosing from the current generation could provide better value and immediate utility. Additionally, understanding the diminishing returns at higher price points helps customers optimize their investment within their budget constraints.

4.6 Example

To illustrate how the proposed methodology works in practice, we will consider a hypothetical example. Suppose a customer approaches a laptop store with the following requirements:

- **Budget:** $1700
- **Preference:** Prefers a laptop with a dedicated GPU for gaming purposes.
- **Use case:** Primarily for gaming, with occasional productivity tasks such as video editing.

Using the steps outlined in the previous sections, the following process is applied:

1. **Identify suitable brands:** This step focuses on the hypothesis remarked in Question 1. Using the same process as proposed above, we get the bar plot that showcase the rankings of the brands based on gaming performance, in the budget range of $1500-$2000:

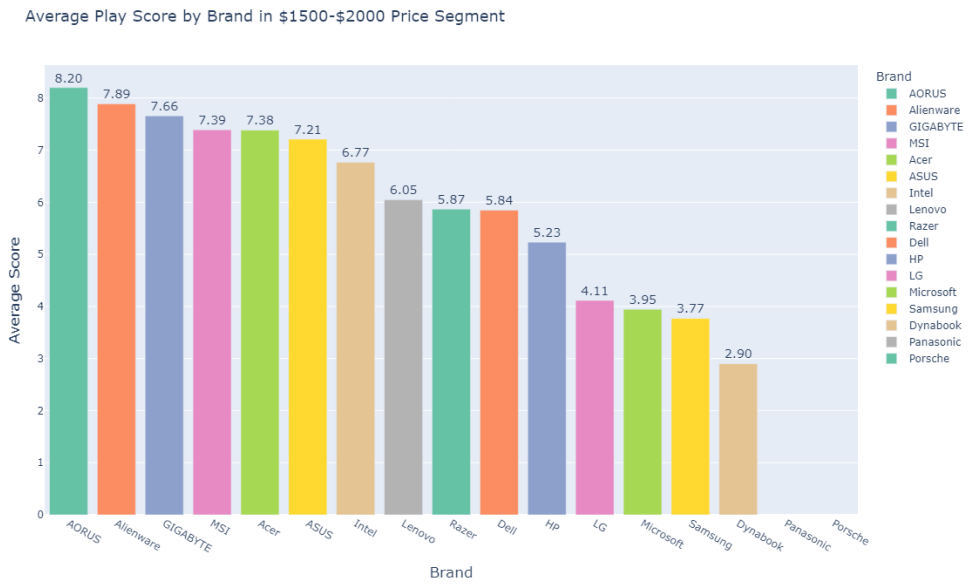

Figure 21 . Average play score of brands in $1500-$2000 price segment

As the image has shown, the three major brands about gaming laptop at the middle-high budget range, are AORUS, Alienware and GIGABYTES. At the same time, these brands are also famous gaming laptop brands in the world, which are also recommended by multiple stores.

At the same time, we also get various deep insights into the brands themselves. By doing the same time with their product line, we can evalute the best product line of each brands to recommend to the user:

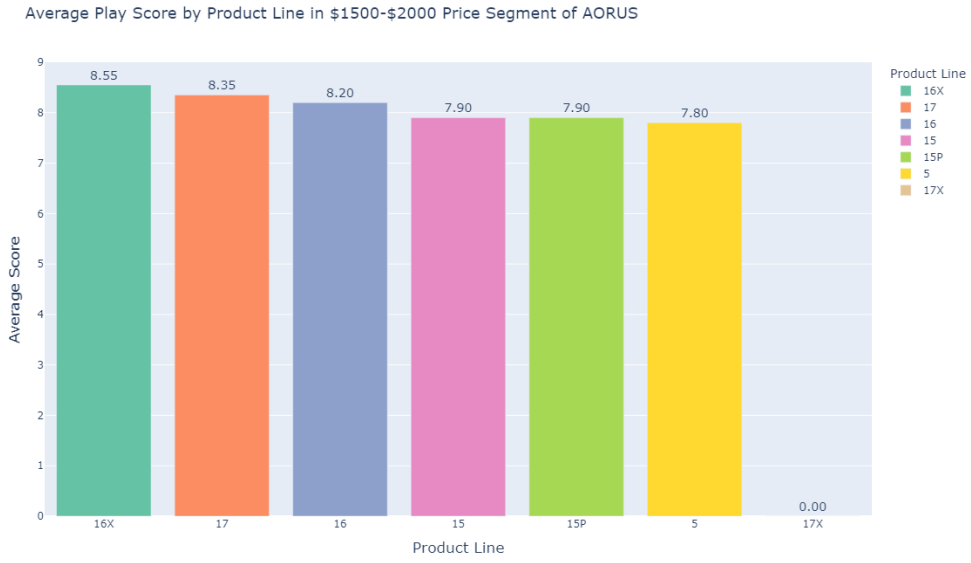

Figure 22 . Average play score of product lines of AORUS in $1500-$2000 price segment

For example, with the above images, we can clearly see that the two best gaming line in AORUS is 16X and 17. These will be the two models that we will try to focus on.

2. **Evaluate brand options:** Using the hypothesis remarked in Question 2, we found that in fact, the above three brands are small brands and only produce a handful of laptops. This not only reduce the number of choices that the consumer can choose, but also shows the difference of customer service level and stability that these brands would have difficulty at. As such, laptops from the big brands will also be recommended over the smaller ones. Back to question 1, for that reason, we will also include ASUS and Acer in our choice. Even though they have smaller performance that the top 3, the difference in performance is not that bad, compare to the number of options in their lineup. The difference in performance, not only can be explained by that the big number of lineups can create bad outliers and thus reduce the overall performance of the brands, but also are covered by their enourmous number of choices and, as a result, better customer service and stability. Thus, we will revisit Question 1 to get access to the model lineup options for each of the new brands:

Figure 23 . Avg play score of ASUS

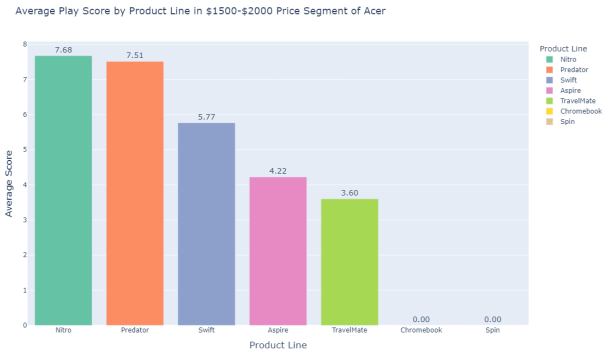

Figure 24 . Avg play score of Acer

For example, with the above images, we can clearly see that the two best gaming line in Acer is Nitro and Predator, while the two best of ASUS is TUF and ROG.

From the two above questions, the most viable options for the customer are an Acer Nitro or an ASUS TUF. Both are very famous gaming laptop lines that are recommended by multiple retailer over the world, and are still a solid options nowadays. At the same time, if the customer favor performance over everything else, we could also recommend an AORUS 16x.

3. **Assess cost-to-performance trade-offs:** This step focus on cost-to-performance trade-offs, which is one major reasons for the hesitation of the buyer when buying the laptop. This problem is analyzed thoroughly in Question 3. Using those insight, we see that a laptop made for gaming specifically, in a budget of around $1500-$2000 can be as high as about 7 points, whereas a laptop made for both gaming and working jobs will have lower performance score to it. At the same time, it will cost around $250 to increase the score by 0.5. This is a reasonable cost to increase the playing the performance of the user, so we should give broader range to the customer.

4. **Factor in generational trends:** Using insights from subsection 5, newer GPU models in this price range, such as NVIDIA's Ada Lovelace generations, offer 10% higher performance than older generations for gaming tasks. As for upcoming generation, the RTX 5000s series will take a long time to be served, so it is best to just buy a laptop for now.

Recommendation: Based on the analysis, the store recommends two products: an Acer Nitro V 5 with Intel Core i9-13900H, NVIDIA GeForce RTX 4060 with 32 GB of RAM at the price of $1849, and an ASUS TUF Gaming F16 FX607 with Intel Core i7-13650HX, NVIDIA GeForce RTX 4060 with 32 GB of RAM at the price of $1825. Both of this model balances the customer's budget, gaming performance requirements, and long-term value with the latest GPU generation. Otherwise, if the customer favors performance over stability, we could also recommend an AORUS 16X with Intel Core i7-13650HX, NVIDIA GeForce RTX 4070 with 32 GB or RAM at the price of $1699.

The customer appreciates the clarity and detail of the recommendation, feeling confident in their decision. The store benefits from utilizing data-driven insights to enhance customer satisfaction and drive sales.

4.7 Conclusion

This project showcases the potential of leveraging data science to solve real-world problems in a laptop retail context. By systematically analyzing the data and addressing customer needs through tailored insights, the methodology provides clear and actionable recommendations. Key outcomes include:

- Identifying the best-performing brands in specific price and performance segments to match customer budgets and use cases.
- Highlighting brands with the largest product offerings in various segments, simplifying decision-making for customers.
- Analyzing cost-to-performance trade-offs, enabling customers to optimize their investment.
- Providing insights into generational trends in CPU and GPU performance to guide purchase timing decisions.

By integrating these analyses, both customers and laptop stores benefit. Customers receive well-informed recommendations aligned with their preferences, while stores can refine their inventory strategies and enhance customer satisfaction. The example provided demonstrates the practical application of the methodology, emphasizing its effectiveness and value. Ultimately, this project highlights how data-driven approaches can bridge the gap between complex datasets and meaningful customer solutions, fostering better outcomes for all stakeholders involved.

Part III
Regression analysis

5 Data preprocessing

5.1 Categorical Encoding & Normalization

Preprocessing categorical variables is a critical step before taking it as inputs to Machine Learning models. Techniques like one-hot encoding, label encoding, and target encoding are widely used to handle categorical data. This report focuses on target encoding, highlighting its advantages, its suitability for our projects.

Target Encoding[7](aka likelihood encoding) replaces each categorical value with the mean (or another statistic but usually mean, so does our project) of the target variable corresponding to that category. Mathematically, the transformation is as follows:

$$E(x_i) = \frac{\sum_{j \in C_i} y_j}{|C_i|}$$

where $E(x_i)$ is the encoded value for category x_i, C_i is the set of all instances belonging to category x_i, and y_j is the target value for instance j. Compare to other common techniques, such as:

- **One-Hot Encoding**: is very inefficient for high-cardinality features, leading to the ”curse of dimensionality” and may introduce sparsity in the dataset, which can degrade performance for certain algorithms.
- **Label Encoding**: is less effective for non-ordinal data by imposing a false ordinal relationship between categories, which can mislead the model.

Target encoding by incorporating target-related information into encoding process, can learn patterns more effectively, leading to potentially better performance. However, while powerful, it is susceptible to overfitting, particularly when it uses information from entire dataset to compute mean of target values. This occurs because the encoding can ”leak” information from the target variable into the features, allowing the model to inadvertently memorize rather than generalize.
A approach to mitigating this issue is to carefully handle the train-test split during the preprocessing stage.It means that apply target encoding to the training set only, i.e. computing target means using only the training data, then fit to test set, for categories that do not exist in training set, global target mean can be used.

After encoding categorical features, normalization is proposed to the next step to scale features to a standard range to ensure features with different units are treated equally, making the learning process more efficient., z-score normalization(centering it around zero and scaling it based on its standard deviation) is used in our project.

5.2 Feature selection

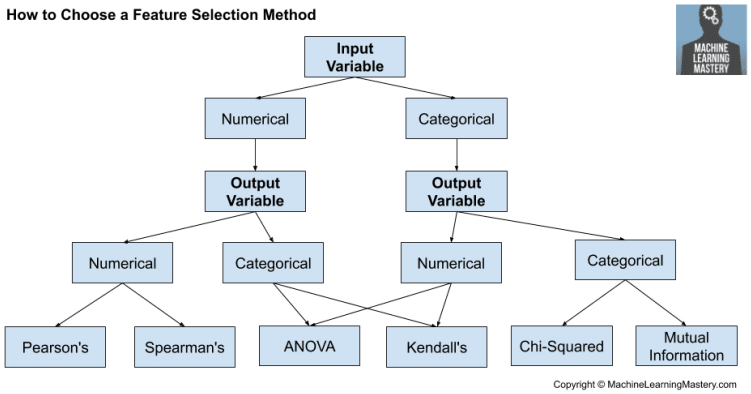

Figure 25 . Feature Selection Scheme(from https://machinelearningmastery.com)

Feature selection [1] is very important step in machine learning pipelines aiming to identify the most relevant features that help reduce dimensionality and noise before putting it into models. The selection technique depends on the type of input, output and the relationship between them. For numerical input variables and numerical outputs, correlation-based methods like Pearson's and Spearman's rank are often employed. Pearson's is always used to capture the linear relationship between input fearues and target variables, whereas, Spearman's rank is used in case of non-linear dependency. On the other hand, Mutual Information is widely used when the relationship between features and the target variable may be non-linear or complex, especially for mixed types of input and output variables. In our project, Spearman's rank correlation and Mutual Information Regression are employed for numerical and categorical input variable respectively.

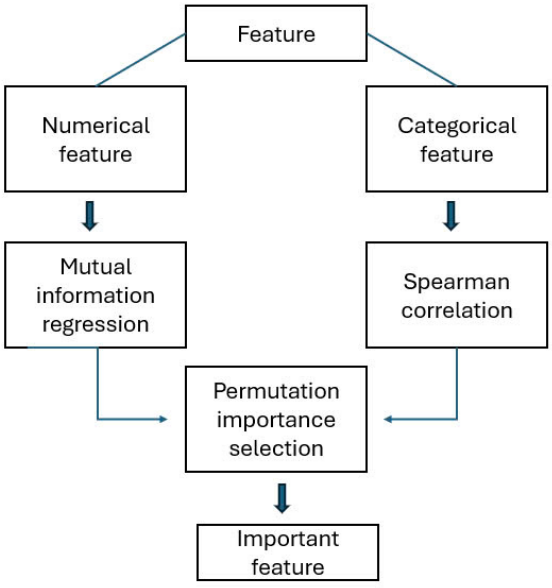

Figure 26 . Our feature selection strategy

Spearman's rank correlation

Spearman's rank correlation coefficient (ρ) is a non-parametric measure that assesses the monotonic relationship between two variables. Unlike Pearson's correlation, which captures linear relationships, Spearman's correlation evaluates whether the relationship is consistently increasing or decreasing, even if not linear. It is robust to outliers and particularly effective for data where monotonic relationships are expected but not necessarily linear.

The formula for Spearman's rank correlation is:

$$\rho = 1 - \frac{6 \sum_{i=1}^{n} d_i^2}{n(n^2 - 1)}$$

where:

- n is the number of observations,
- d_i is the difference between the ranks of the corresponding values of the two variables.

A high absolute value of ρ (close to ± 1) indicates a strong monotonic relationship, while values near zero suggest a weak or no monotonic relationship. This method is particularly useful in regression problems with monotonic but non-linear relationships between features and target variables.

Mutual information regression

Mutual information(MI) rooted in information theory and data analysis, embodies a profound mathematical concept to quantify the amount of information obtained about one random variable through the observation of another random variable. This concept is raised from the notion of entropy, which measures the uncertainty associated with random variable. The mutual information $MI(X, Y)$ between X and Y captures how much knowing one variable reduces the uncertainty about the other.MI can be used for both linear or non-linear relationship which makes it a versatile method for identifying complex relationships in regression tasks. For a categorical input X and numerical output Y, MI is calculated as:

$$I(X;Y) = \int_X \int_Y p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right) dx\, dy$$

where:

- $p(x, y)$ is the joint probability density function of X and Y,
- $p(x)$ and $p(y)$ are the marginal probability density functions of X and Y, respectively.

The mutual information $I(X;Y)$ quantifies the reduction in uncertainty about Y given knowledge of X. It is evident that Mutual information can also be expressed as the Kullback-Leibler (KL) divergence between the joint distribution p(x,y) and the product of the marginal distributions p(x)p(y).When X and Y are independent, their joint distribution factorizes into the product of their marginals, resulting in zero MI.

Comparison with other techniques

- **ANOVA:** statistical test used for comparing the means of different groups, assuming normal distribution, and variances of different groups are equal (homoscedasticity). ANOVA measures only linear dependencies between the input and the output, making it unsuitable for capturing non-linear relationships.

- **Kendall's Tau:** is a rank-based non-parametric measure of correlation that evaluates the strength and direction of a monotonic relationship between two variables. While Kendall's Tau works well for small datasets with monotonic relationships, it has notable limitations:cannot detect non-linear dependencies, its computation becomes inefficient for large datasets due to pairwise comparisons.

Permutation importance Permutation importance [1] is a feature evaluation technique. It quantifies the importance of each feature in a model by measuring the decrease in model performance when the feature values are randomly shuffled. This distorts the relationship between the feature and the target, which tells how much model performance depends on a feature. The greater the decrease in model performance, the more important the shuffled features is. The process of permutation importance is as follow:

1. Train a model on the original dataset and compute the baseline performance of the model using a scoring metric (e.g. accuracy, R2).

2. Shuffle the values of a single feature while keeping others constant.

3. Use the model to predict distorted data and measure the new score of model's performance

4. The drop in performance indicates the importance of the shuffled feature.

5. Repeat the process multiple times and average the results.

The advantage of using permutation importance in feature selection is that it is not tied to any model since this method depends on the relationship between features and target in the data, not the internal model architecture. However, the disadvantage is that with large datasets, this technique is resource consuming, so it is important for us to manually select features by statistical coefficient first before applying this method.

Another view Although the characteristics of each score's feature selection are as described, in practice and within our project, it is challenging to determine which features are truly suitable to data. Therefore, alongside the techniques we used above, we also experimented with other techniques in parallel to observe their results. It is noticeable that some changes in the ranking of feature importance for a few features (though not significant). In general,all techniques to extract top 5 important feature produced similar results during experimentation. The techniques we employed are popular methods widely used and favored in data feature engineering.

Reasons for our feature selection strategy: The feature selection strategy we used is first to considers the features (whether numeric or categorical) and then applies scored-based technique to extract the important features within these two groups. Afterward, we apply permutation feature importance to both the categorical and numerical groups to identify the important in total. The question raised here: why not directly use permutation feature importance? or why not use dimensionality reduction techniques like SVD or PCA directly for faster results instead of going through this more complex process?

The answer for the first question is that we consider the permutation importance technique which permute in each feature to re-evaluate the model will be very excessively computational for high-dimensional data(particularly in our data with about 50 features). Furthermore the second reason

and also the main reason lies on when there are many features, the likelihood of some being correlated (or redundant) increases then permutation importance tends to underestimate the importance of correlated features.

For the second question, the solution is that PCA and SVD are unsupervised methods, meaning they focus solely on the variance of the data without considering the relationship between features and the target variable. The main reason also relies on the interpretability. Since we are doing a data science project, instead of simply aiming for the best regression results, we also strive to understand which features are most correlated with each score and the characteristics of the laptop. Our aim is not just to minimize the loss as much as possible, but also to get deep insights into the relevance of individual features.

5.3 Outlier removal

Outlier removal is essential in preprocessing data to ensure the reliability of downstream analysis and model performance. The challenge in detecting outliers is not simply identifying the "outliers," but rather distinguishing whether they represent true anomalies that worsen performance or are rare but valid values that could reappear in future data (such as test data), making their identification and interpretation a critical aspect of data analysis. This project focuses on robust techniques for detecting and managing outliers in both numerical and categorical data.

5.3.1 Numerical features

Interquartile Range (IQR)

For features with symmetric distributions, the Interquartile Range (IQR) is employed. The IQR represents the spread of the middle 50% of the data and is robust to extreme values. It is mathematically defined as:

$$IQR = Q_3 - Q1$$

where Q_1 and Q_3denote the first and third quartiles, respectively. A data point x is flagged as an outlier if it lies outside the following range:

$$x < Q_1 - 1.5 \times \text{IQR} \quad \text{or} \quad x > Q_3 + 1.5 \times \text{IQR}.$$

The IQR method assumes symmetry in the data distribution, making it ideal for detecting outliers in features where extreme deviations are evenly distributed on both ends.

Median Absolute Deviation (MAD)

For features with skewed distributions, the Median Absolute Deviation (MAD) is preferred due to its robustness to skewness and resilience against extreme values. The MAD is calculated as:

$$\text{MAD} = \text{median}\left(|x_i - \text{median}(x)|\right),$$

where x_i are the data points and median(x) is the median of the dataset. To identify outliers, the following condition is applied:

$$|x - \text{median}(x)| > k \times \text{MAD}.$$

Here, k is a scaling constant, often set to 1.482 to make MAD consistent with the standard deviation under a normal distribution.

The IQR method can fail to capture the true spread of skewed data since the quartiles Q1 and Q3 may not align well with the tails of a non-symmetric distribution. By contrast, MAD centers on the median, which is less influenced by extreme values, making it better suited for skewed distributions. Mathematically, MAD minimizes the influence of extreme values as it measures the central tendency and variability based on ranks rather than magnitude.

5.3.2 Categorical features

Frequency-Based Outlier Detection For categorical data, Frequency-based outlier detection is used. This approach identifies rare categories with very low frequencies as potential outliers. The frequency of each category c is calculated, and a category is flagged as an outlier if:

$$P(c) < \tau,$$

where P(c) represents the percentile of the frequency of category c in the dataset, τ is the predefined threshold percentile(always used 1% in our project).

Some insights: That being said, however, determining how define an outlier is actually difficult and not straightforward. The techniques we use are just applied to certain features, while most features we consider do not have . Each feature has a different distribution, making it challenging to identify outliers, we only remove those that are clearly outliers and where the number of outliers is small enough to not significantly affect the model.

For example, in real data, in countries with a younger population, very few people live beyond 100 years, so we would consider someone over 100 as an outlier (they may indeed exist, but they are rare and could influence the regression model). However, in countries with an older population like Japan, the number of people over 100 may be large enough to be considered a valid age value in the model.

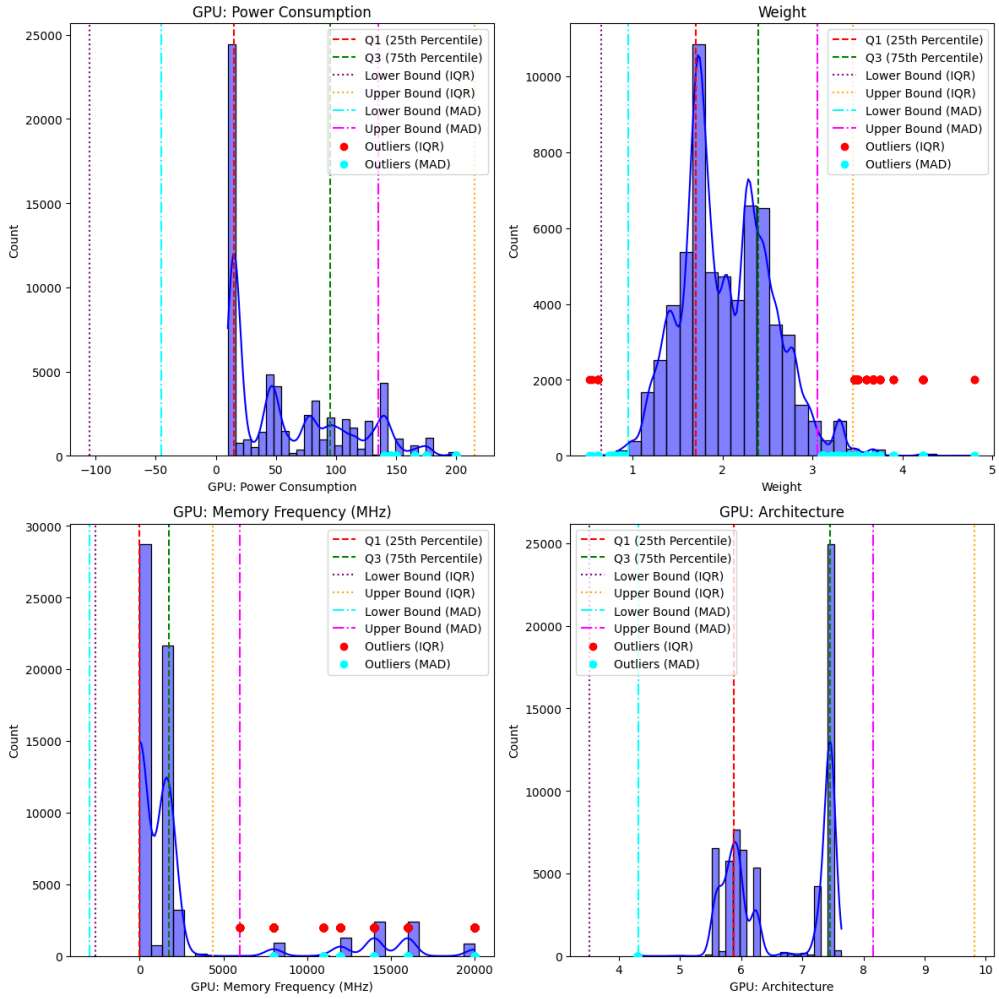

Figure 27 . Distribution of some features

For example in portability step, after feature selection,we can get numerical features like in the Figure27, GPU: Power Consumption, GPU memory frequency when apply IQR outlier removal(red dot) can be numerous due to improper technique to this distribution(The number of rare value can be consider as large enough to maintain). For the weight, we also modify the range in IQR to maintain more value [Q-2 IQR; Q+2IQR](since the outlier removal only implemented in training dataset, not the whole dataset then predict in test set, then removal of a lot of data in training set can worsen performance in test set).

6 Machine learning models

6.1 Ridge Regression

Ridge regression is a method for regularization to estimate the coefficient of a linear regression model where independent variables are highly correlated. Different from the OLS estimator method which just focuses on optimizing the loss function, ridge regression minimizes an objective function by combining loss and the L2 norm of the coefficients vector. The inclusion of constraints on

the coefficient magnitude support reduces the variance of estimates and stabilizes the process of prediction, thereby avoiding overfitting.

Overall ridge regression problem :

$$W^* = argmin_w(RSS(W) + \lambda\|W\|_2^2)$$

Where $0 \leq \lambda$ is a regularization parameter that controls the complexity of the model and can avoid overfitting.
We consider that λ is a parameter in the objective function and we must find an optimal λ so that the model can perform the best prediction. By derivative the objective function and solving the gradient of the objective function equal to zero, we can easily find the solution of the model.

$$W^*_{ridge} = (X^T X + \lambda I)^{-1} X^T y$$

The efficiency of the model is apart depends on the parameter λ. If we choose the large λ then the magnitude of the vector coefficient will decrease and tend to zero. In this case, the variance of coefficient distribution will be minimal, and the model does not pay more attention to any feature then it does not capacity to generalization from the data that are reason lead to under-fitting. On the other hand, if λ is small leads to the size of the vector coefficient increasing, so the variance of distribution is very large then the model can be overfitting. This is the reason why parameter λ is crucial in training ridge regression models.

6.2 KNN Regression

The K-Nearest Neighbor(KNN) regression [3]is non-parametric supervised learning method that predicts based on the labels of its nearest neighbors in the feature space.

The predicted target value $\hat{y}_{\text{test}}$ is computed as:

- **Unweighted Mean:**

$$\hat{y}_{\text{test}} = \frac{1}{k} \sum_{i \in N_k} y_i$$

- **Weighted Mean (distance-based weights):**

$$\hat{y}_{\text{test}} = \frac{\sum_{i \in N_k} w_i y_i}{\sum_{i \in N_k} w_i}, \quad \text{where } w_i = \frac{1}{d(\mathbf{x}_i, \mathbf{x}_{\text{test}}) + \epsilon}.$$

Here, ϵ is a small constant to avoid division by zero. To predict the target value for test point, algorithm computes distances between this point to all points in training dataset, common distance metrics include Euclidean, Manhattan, choosen based on the data characteristics

$$d(\mathbf{x}_i, \mathbf{x}_{\text{test}}) = \sqrt{\sum_{j=1}^{d} (x_{i,j} - x_{\text{test},j})^2}$$

KNN regression relies heavily on three key factors: the number of neighbors (k), the distance metric, and the weighting scheme. A smaller k captures local variations but can be noisy, while a larger k smoothens predictions but may overlook finer details. Similarly, feature scaling is crucial as the algorithm is sensitive to the relative magnitudes of features; normalization or standardization ensures fair contributions during distance calculations.

6.3 Decision Tree Regression

Decision Tree Regression [5] is a supervised learning algorithm used for predicting continuous target variables. It models data by recursively partitioning the feature space into distinct regions and fitting a constant value in each region. The dataset is represented as $D = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in R^d, y_i \in R, i = 1, 2, \ldots, n\}$, where $\mathbf{x}_i$ is a d-dimensional feature vector, and y_i is the corresponding target value. The algorithm begins at the root node, which contains all samples, and splits the dataset into child nodes by selecting the feature j and threshold t that minimize a cost function. For regression tasks, the common cost function is the Mean Squared Error (MSE), defined as

$$\text{MSE} = \frac{1}{|D_{\text{node}}|} \sum_{(\mathbf{x}_i, y_i) \in D_{\text{node}}} (y_i - \bar{y}_{\text{node}})^2$$

where $\bar{y}_{\text{node}}$ is the mean target value of the samples in the node.

The algorithm evaluates all possible splits, where a split partitions the data into

$$D_{\text{left}} = \{(\mathbf{x}_i, y_i) \mid x_{i,j} \leq t\}$$

and

$$D_{\text{right}} = \{(\mathbf{x}_i, y_i) \mid x_{i,j} > t\}$$

and selects the one that minimizes the weighted impurity of the resulting subsets. This process is repeated recursively for each child node until a stopping criterion is met, such as reaching a maximum tree depth, having fewer than a minimum number of samples in a node, or when further splits fail to significantly reduce the impurity.

Once the tree is fully constructed, predictions for a new data point $\mathbf{x}_{\text{test}}$ are made by traversing the tree from the root node to a leaf node based on the feature values of $\mathbf{x}_{\text{test}}$. The predicted value is the mean target value of the samples in the corresponding leaf node, given by

$$\hat{y}_{\text{test}} = \frac{1}{|D_{\text{leaf}}|} \sum_{(\mathbf{x}_i, y_i) \in D_{\text{leaf}}} y_i$$

Decision Tree Regression produces a piecewise constant approximation of the target variable and is particularly effective at capturing non-linear relationships, though its performance heavily depends on proper parameter tuning to avoid overfitting or underfitting.

Random Forest Regression

Random Forest Regression [4]is an ensemble learning method that improves the predictive performance and robustness of Decision Tree Regression by combining the outputs of multiple decision trees. It mitigates the limitations of individual trees, such as overfitting, by aggregating predictions across a diverse set of trees trained on different subsets of the data. The algorithm creates B decision trees, each trained on a bootstrap sample of the original data. Bootstrap sampling randomly selects n samples with replacement, ensuring that each tree is trained on a unique subset of the data, while some samples are excluded, forming an out-of-bag (OOB) set.

During tree construction, Random Forest introduces additional randomness by selecting the best split from a random subset of features rather than evaluating all features. For a given node, if m features are selected randomly from the total d features, the split is chosen to minimize the Mean Squared Error (MSE), as in Decision Tree Regression. This random feature selection decorrelates

the trees, enhancing ensemble diversity and reducing overfitting. After constructing the forest, predictions for a test point $\mathbf{x}_{\text{test}}$ are obtained by averaging the predictions from all individual trees. Let $\hat{y}_b$ be the prediction from the b-th tree; the final prediction is computed as:

$$\hat{y}_{\text{test}} = \frac{1}{B} \sum_{b=1}^{B} \hat{y}_b.$$

An important feature of Random Forest is the use of OOB samples to estimate model performance without requiring additional validation data. The algorithm also provides feature importance scores by measuring the reduction in impurity (e.g., MSE) attributable to each feature across all trees. Random Forest Regression effectively balances bias and variance, making it robust to overfitting and suitable for capturing complex, non-linear relationships in the data, though its computational complexity increases with the number of trees and features.

6.4 Support vector regression

Support vector machines (SVM) [2] have become a popular classification and linear regression method. However, the original SVM focused on classification problems by learning a hyperplane to classify labels. Instead of learning to restrict error classification, on regression problems, it tries to learn a function that has a margin with minimized error prediction. In this section, we discuss how to adapt SVM for regression problems.

Support vector regression does not minimize the number of missing predictions. It evaluates the error prediction of error by an ϵ parameter, and two softs margin values ζ and $\zeta^{'}$.

$$y_i - (W^T x_i + b) \leq \epsilon + \zeta_i, \ with \ \zeta_i \geq 0 \tag{1}$$

$$(W^T x_i + b) - y_i \leq \epsilon + \zeta_i^{'}, \ with \ \zeta_i^{'} \geq 0 \tag{2}$$

Similar to SVM for classification problems, the objective function of SVR is the maximization of the margin. By maximizing the margin, it is proved based on VC dimension theory that with maximal margin the SVM model has the least test error.

$$\min_{W, \zeta_1, \zeta_2, \ldots, \zeta_n} \frac{1}{2}||W||_2^2 + C \sum_{i=1}^{n} (\zeta_i + \zeta_i^{'})$$

subject to, equation (1) and (2) , and $\zeta_i^{'}, \zeta_i \geq 0$

Here is an optimization problem with two constraints (1) and (2). By solving the problem, we convert it to a duality problem. The formula of language function is

$$\begin{aligned} L(W, b, \alpha, \alpha^{'}, \lambda, \lambda^{'}) = \frac{1}{2}||W||_2^2 + C \sum_{i=1}^{n} (\zeta_i + \zeta_i^{'}) + \sum_{i=1}^{n} \alpha_i (y_i - W^T x_i - b - \epsilon - \zeta_i) \\ + \sum_{i=1}^{n} \alpha_i^{'} (W^T x_i + b - y_i - \epsilon - \zeta_i^{'}) - \sum_{i=1}^{n} (\lambda_i \zeta_i + \lambda_i^{'} \zeta_i^{'}) \end{aligned}$$

The duality problem:

$$g(\alpha, \alpha^{'}) = \max_{\alpha,\alpha^{'}} \sum_{i=1}^{n}(\alpha_i - \alpha_i^{'})y_i - \sum_{i=1}^{n}(\alpha_i + \alpha_i^{'})\epsilon - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}(\alpha_i - \alpha_i^{'})(\alpha_j - \alpha_j^{'}) < x_i, x_j > (**)$$

It is proved that the duality function is a convex function, then it has a strong duality that is similar to the optimal solution of the duality problem, and the primary problem is the same. The objective function for SVR (**) is solved by quadratic programming. In addition, we can modify the objective function to deal with gradient descent. The solution $\alpha, \alpha^{'}$ often focuses weight on some special data points called support vectors. Support vector regression also applies to the kernel method on high-dimension data to learn a complex decision boundary.

6.5 Gradient Boosting

The previous section discussed some classical machine-learning algorithms that use a single learner. A problem in the machine learning model is to trade off bias-variance, with a simple learner having difficulty representing complex data. Otherwise, a complex learner tends to be biased with training data. In the following section, we introduce the Gradient Boosting algorithm in the class algorithm of ensemble learning that boosts the balance between bias and variance.

Gradient Boosting [6] is a popular boosting algorithm used for classification and regression tasks. It combines several weak learners to perform a robust model. Like other boosting algorithms, gradient boosting training is sequential. Each new model tries to correct the error (residual) made by the previous model. This sequential approach creates a model that can capture complex patterns over iteration.

$$F(x) = \sum_{m=1}^{M} \gamma_m h_m(x) + b$$
$$F^*(x) = \arg\min_{F} E_{x,y}[L(y, F(x))] \tag{3}$$

Here is the formula predictor of gradient boosting to minimize the MSE loss function, where $h_m(x)$ is a weak learner. Learning in Gradient Boosting is sequential, meaning sequential updates residual error from an older model to a new model. The formula update of gradient boosting is shown below:

$$F_m(x) = F_{m-1} + \gamma_m h_m(x)$$
$$h_m = \arg\min_{h_m} \sum_{(x_i,y_i)} [L(y_i, F_{m-1}(x_i) + h_m(x_i))]$$

In this formula, the error of the previous learner will train to fix by the new learner. Up to now, the Optimization problem in gradient boosting includes estimated parameter of each sub-model and the weight between each sub-model. The pseudo algorithm of gradient boosting below will overall detail in process learning of GDB.

Algorithm 1 Gradient Boosting (GDB)

Input: Training set $\{x_i, y_i\}_{i=1}^{n}$.
Number M iteration.

Main:

1. Initialization model with a constant value.
$$F_0(x) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma)$$
2. Loop m = 1 to M.
2.1 Training current learner.
$$\text{Residual Gradient Descent: } \nabla W_m = -\frac{\partial\big[\sum_{i=1}^{n} L(y_i, F_{m-1} + \gamma_m h_m(x_i))\big]}{\partial W_m}$$
$$\text{Update weak learner model: } W_t = W_t - \epsilon \nabla W_m$$
2.2 Find multiplier γ parameter by solve one-dimension optimization.
$$\gamma_m = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma_m h_m(x_i))$$
2.3 Update model
$$F_m(x) = F_{m-1} + \gamma h_m(x)$$
3. Return F_M

Gradient boosting is typically used with a decision tree of a fixed size as base learners. In practice, the size of a tree is often between 4 and 8 (the size mentioned here is many terminal nodes), which works well for boosting. Gradient boosting is. With efficient gradient boosting, many boosting algorithms developed from it like XGBoost, LightGBM, and Catboost.

7 Evaluation

Evaluation plays a crucial role in model prediction as it measures the model's performance and guides the fine-tuning of hyperparameters. Evaluation can be divided into two parts: selection metrics for evaluation and the use of train, validation, and test sets in the hyperparameter fine-tuning process.

In our approach, we use grid search to fine-tune hyperparameters. For each candidate combination in the grid search, we apply k-fold cross-validation in the train set to evaluate the model and use RMSE and MAE for evaluation. After determining the hyperparameters with the highest score in k-fold validation, we use them to make predictions on the test set to assess the final performance of the model.

8 Applications & Empirical Results

In this section, we will perform five score regression problems using the same preprocessing technique and models approach to keep things simple and easy to follow. Specifically, we will prepro-

cess the data using various techniques and use it as input for the machine learning models mentioned earlier in the data preprocessing and machine learning model section. We will provide a detailed explanation of the steps for the first score (display score), while the other scores will follow a similar process, with only notable results or differences in feature selection being highlighted.

8.1 Display Score

Display score of laptop typically refer to evaluate the quality and performance of a laptop's display, prominent features that affected this score can include display size, resolution, panel type in this dataset.

Initially, we began by addressing missing values and redundancy in dataset. The column representing Ethernet LAN was dropped due to high proportion of missing values(we also find it quite difficult to equally resolve missing values of this feature without any human bias), which rendered it unsuitable for analysis. Moreover, duplicate entries were removed to eliminate redundancy, as similar laptop configurations could belong to the same computer types, potentially introducing the noise into the model.

To prepare the categorical features for features engeering and machine learning model, we utilized a target encoder. Howerver, before applying the target encoding, we performed a train test split using hold-out approach with 0.2 test size(this split also used for normalization and machine model in future). This ensured that the encoder's training was confined to training set, thereby mitigating the risk of overfitting by preventing information leakage from test set and from label into the training process,which target encoder can encounter.

Feature Engineering

To enhance the predictive capability of out dataset, several derived features were created based on domain knowledge:

- **Aspect Ratio**: Calculated as the ratio of screen width to height(width/height), representing the screen's shape, which may influence portability.
- **Pixel Density**: Derived as $\sqrt{width^2 + height^2}$/ display size, representing the clarity and sharpness of display, which directly affects the display score.
- **GPU Performance**: GPU plays crucial role in helping handle graphics-related work like graphics, effects, and videos. Then the feature formulated by memory size x memory frequency is proposed to capture the computational power of the GPU.

After that, taking into the feature selection step. It was conducted in two phases: one for categorical features and another for numerical features.Before applying feature selection techniques, all numerical features were normalized using the z-score formula. Importantly, normalization was performed exclusively like in encoder on the training dataset, and the resulting parameters were used to transform the test set, maintaining the assumption that no information about X test and y test is available during training. For categorical feature, we used mutual information regression to evaluate the relevance of each categorical feature with respect to the target variable. On the other hand, the Pearson correlation coefficient was calculated to identify numerical features strongly correlated with the portability score. Finally, permutation feature importance was applied to both subsets (categorical and numerical) to quantify the contribution of each feature to the predictive performance of the model.

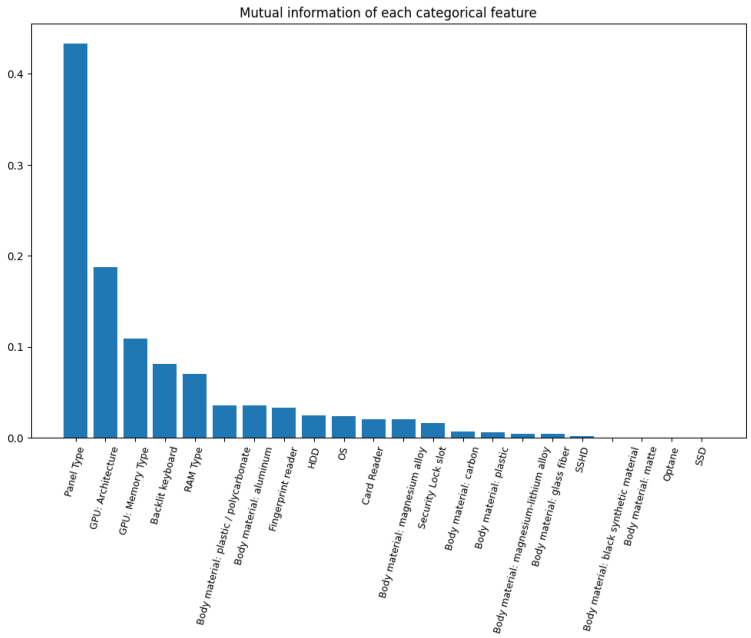

Figure 28 . Mutual info for categorical features

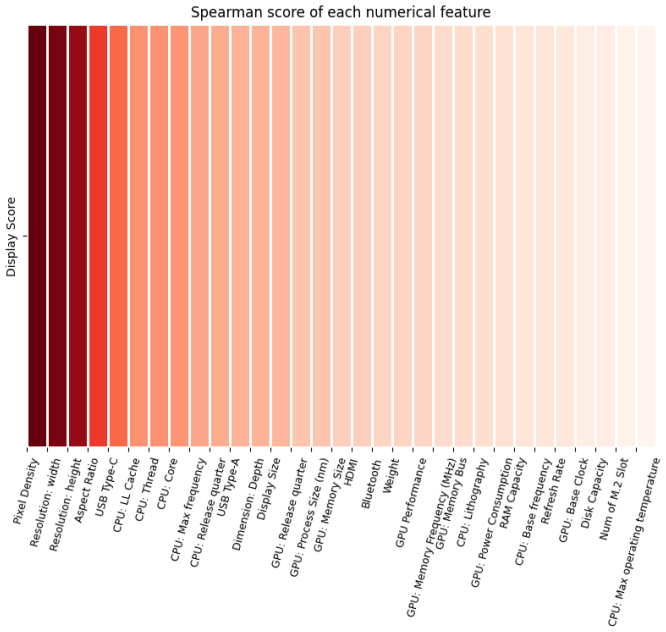

Figure 29 . Spearman score for numerical features

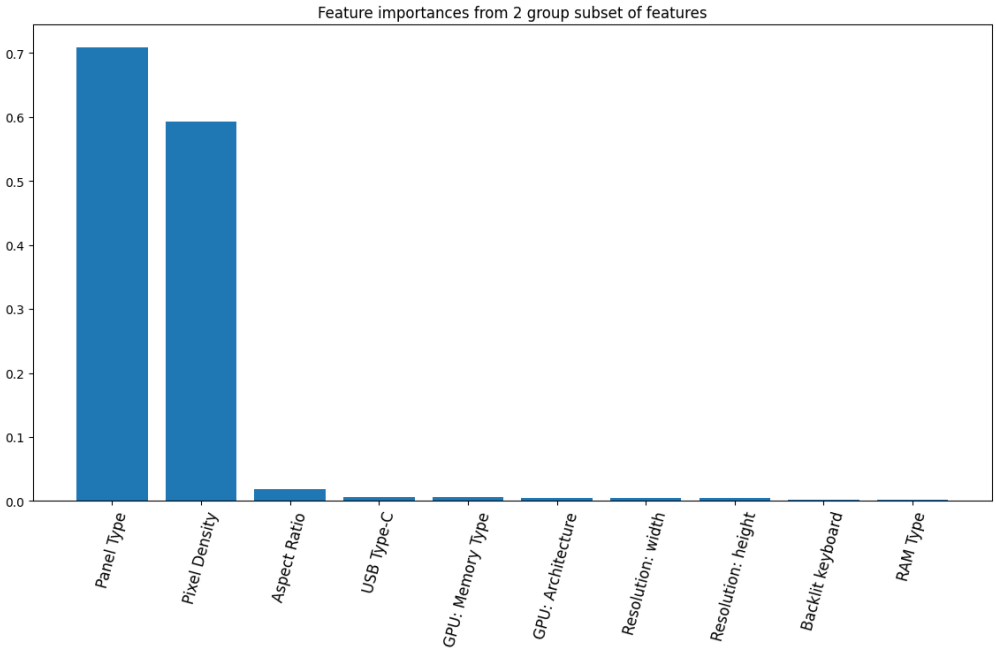

Figure 30 . Permutation feature importance

Based on Figure 30, it is noticeable that Panel Type and Pixel Density feature have remarkable score compared with the others, then only 2 features is selected for further regression work.

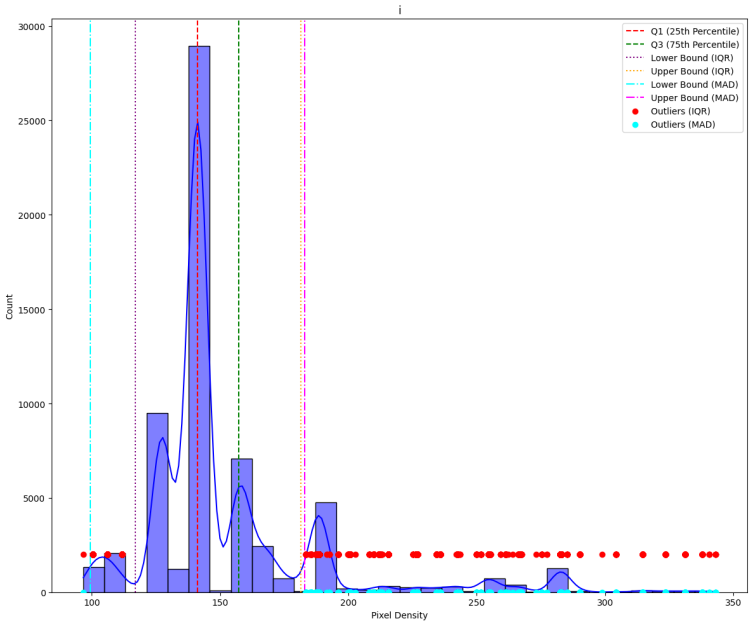

Figure 31 . Permutation feature importance

Then outlier removal was applied in the unnormalized dataset, however in the figure 31 shows the distribution and simulative outliers removal í also shown if MAD or IQR were applied, the distribution appears highly skewed with a long and relatively heavy tail. Therefore, removing these outliers is not reasonable, and we decided not to exclude outliers for Pixel Density feature. We only remove Panel Type feature(categorical feature) by set threshold of 1%(removal 408 out of 61 thousands data points).

The refined dataset was fed into several predictive models, and their performance was compared using standard metrics such as Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE). The same hold-out split (20% test size) used during the encoding and normalization steps was maintained for consistency and fairness. This ensured that no information about X test and y test was inadvertently revealed during training, thereby preventing overfitting and preserving the integrity of model evaluation.

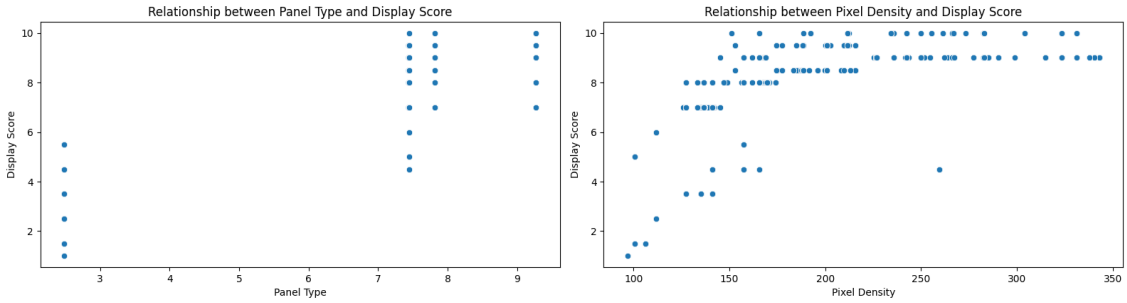

Figure 32 . Permutation feature importance

This systematic approach combines robust preprocessing, feature engineering, and selection with rigorous model evaluation to predict the portability score effectively. Each step is carefully designed to mitigate potential biases and enhance the predictive capabilities of the model.

Empirical result

Table 1 . Display score results

Models	RMSE	MAE
Ridge Regression	0.5342	0.3725
KNN-regression	0.3121	0.0749
Gradient Boosting	0.3069	0.1188
SVM-regression	0.3374	0.1608
Random Forest Regression	0.3069	0.1188

8.2 Portability Score

The portability score of laptop in the dataset evaluates how easily the laptop can be carried or used on the go. It is also a measure of the capability of laptop for many tasks in long duration(battery life) or connectivity(ports, wireless connectivity) and convenience of laptop(backlit keyboard,etc).

Feature Engineering

- **Connectivity Score**: is the sum of the connection gate availability in laptop(including USB Type-C, USB Type-A, HDMI, Card Reader)
- **Lightweight build**: is binary variable that combine feature of weight(as a threshold) and lightweight materials of laptop(magnesium alloy, carbon, plastic),
- **Portability feature count**: is the sum of convenience of laptop(Fingerprint reader, Blacklit keyboard, Security Lock slot) Then we also preprocess like in Display score, here is the result of feature selection:

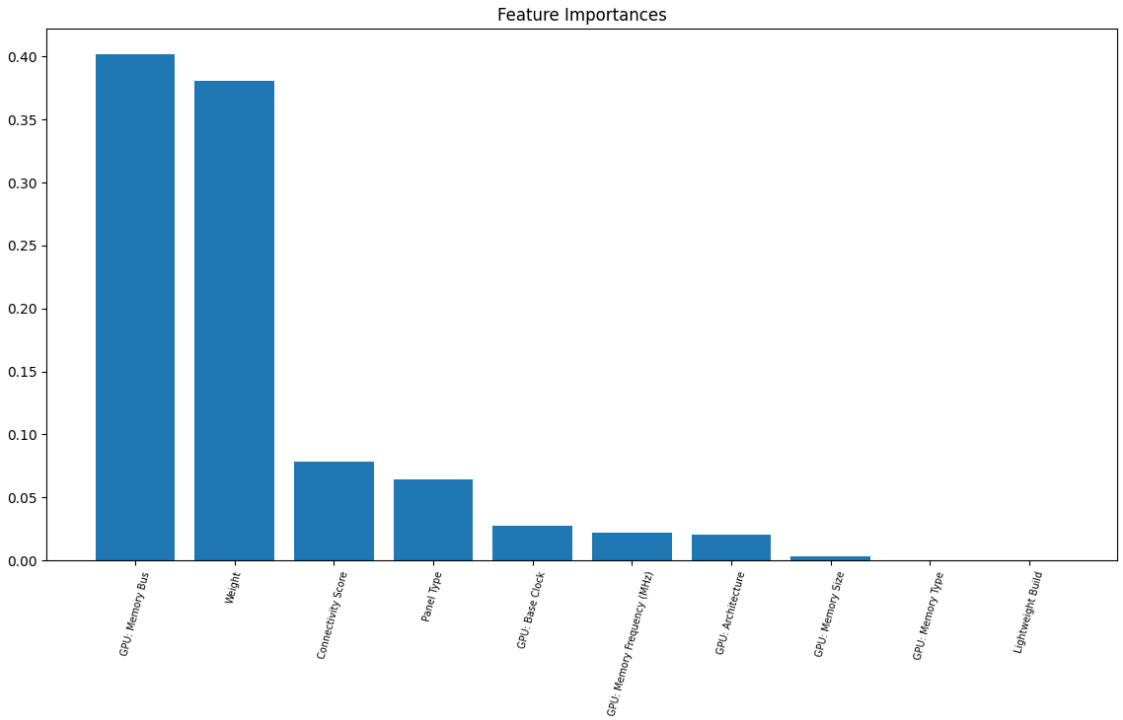

Figure 33 . Permutation feature importance

Based on Figure 33, we choose GPU: Memory Bus, Weight, Connectivity Score, Panel Type, GPU: Base Clock, GPU: Memory Frequency (MHz), GPU: Architecture features for regression model.

Empirical result

Table 2

Models	**RMSE**	**MAE**
Ridge Regression	0.8527	0.6141
KNN-regression	0.4596	0.0952
Gradient Boosting	0.4958	0.1386
SVM-regression	0.9026	0.4320
Random Forest Regression	0.4376	0.0897

8.3 Work Score

Work score is the evaluation of laptop performance regarding work-related jobs, this metrics emphasizes the performance of CPU with prominent features that affected this score are features related to CPU architecture in this dataset. The process of predicting the work score is the same as that of the prediction of the previous scores.

Feature Engineering

For feature engineering, several derived features were created based on domain knowledge:

- **CPU Processing Power**: measured in GHz, calculated as the product of number of CPU cores to CPU base frequency (the speed where each core operates (GHz)). This feature tells the total number of cycles the CPU can operate in a second.
- **GPU Band Width**: measured in bits/s, calculated as the product of GPU memory frequency and GPU memory bus. With GPU memory frequency is the clock speed of GPU memory (MHz) and GPU memory bus is the number of bits the memory can transfer per clock cycle (bits/cycle), this feature represent how fast data can move between the GPU memory and the GPU's cores for processing (bits/s).
- **GPU Efficiency**: measured in cycles/J, calculated as the ratio of GPU base clock (clock speed of GPU core (GHz) and GPU power consumption (the power GPU uses during operation (W)). The feature represent how much work GPU can do per unit of energy consumption (cycles/J).

This is the result of the feature selection phase:

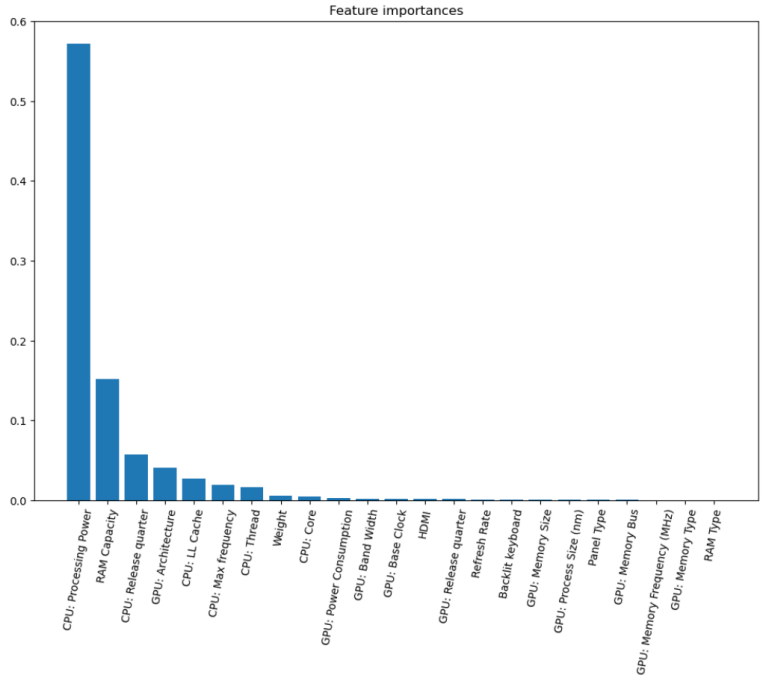

Figure 34 . Permutation feature importance

After feature selection, 10 features are chosen with the most important features being CPU: Processing Power and Ram Capacity.

Empirical result

Table 3 . Work score results

Models	RMSE	MAE
Ridge Regression	0.3420	0.2726
KNN-regression	0.1601	0.1018
Gradient Boosting	0.1501	0.1067
SVM-regression	0.1933	0.1451
Random Forest Regression	0.1462	0.0936

As can be derived from the table, Random Forest Regression is the model producing best result. Overall, as the target variable is in range [0,10], RMSE ranging from 0.14 to 0.34 and MAE ranging from 0.0936 to 0.27 are acceptable.

8.4 Play Score

Play score is the evaluation of laptop performance in gaming which requires a good GPU for rendering graphics, hence the features that have significant impact here should be features related to GPU architecture in this dataset. The process of prediction of the play score is the same as that of the prediction of the work scores. The derived features that have been created in work score prediction are used here as well.

After feature selection, 10 features are chosen with the most important features being GPU: Architecture, GPU: Memory Type and CPU: Max Frequency; which matches our expectation of gaming performance relying heavily on GPU strength and CPU rate.

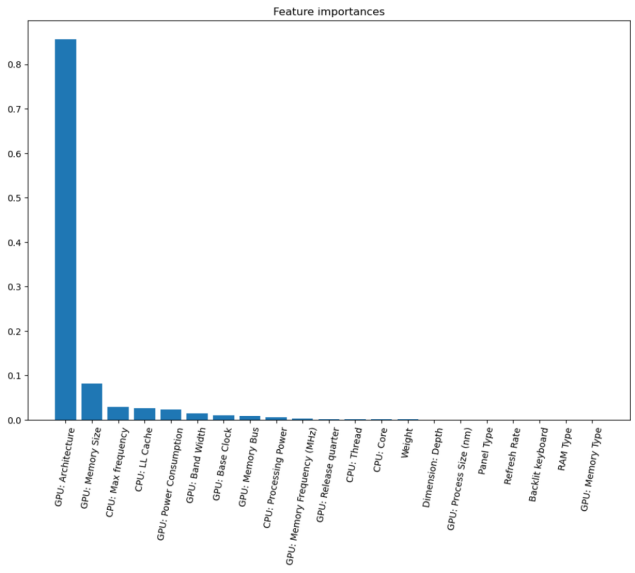

Figure 35 . Permutation feature importance

Empirical result

Table 4 . Play score results

Models	RMSE	MAE
Ridge Regression	0.3072	0.2201
KNN-regression	0.1114	0.0786
Gradient Boosting	0.1096	0.0809
SVM-regression	0.1919	0.1254
Random Forest Regression	0.1045	0.0745

As can be seen from the table, Random Forest Regression is the model producing best result. Overall, as the target variable is in range [0,10], RMSE ranging from 0.1 to 0.3 and MAE ranging from 0.0745 to 0.22 are acceptable.

8.5 Price

Price is the main objective in laptop problem prediction. The data is collected from a website course with 289,000 laptop types. By analyzing all the data, we consider that some potential features of CPU and GPU have a greater effect on laptop price. Following this section, we show the analysis data to predict price.

Laptop price is usually difficult to predict because it is based on many aspects of the computer such as outside and inside. In addition, the main element of laptop price depends on the company's policy that produced the computer. From the laptop price data, we start processing data with nan and encode categorical features by target encoding. As mentioned before, the features of the laptop focus on aspects of CPU, GPU, and memory. Therefore, creating new features for the CPU and GPU is crucial. Based on our knowledge of computers, some new features created from GPU and memory are shown below:

- **CPU Processing Power** (GHz)= CPU: Core x CPU: Base Frequency
- **GPU Band Width** (bits/s) = GPU: Memory Frequency × GPU: Memory Bus
- **GPU Efficiency** (cycles/J) = GPU: Base Clock / GPU: Power Consumption
- **Outside Memory** = sum (Disk Capacity, SSD, HDD, SSHD)

Features Selection is the next step in the training model. By using measure-based and model-based, features with high priority and cost will be selected. The result is shown in the two figures below with a measure-based.

In these two figures, in addition to considering CPU and GPU features, the cost has a slight impact on scores such as play score and work score, which are evaluated by experts in computers.

Empirical result

Table 5

Models	RMSE	MAE
Ridge Regression	480.825	304.562
KNN-regression	294.784	152.412
Gradient Boosting	271.164	153.174
SVM-regression	394.633	216.498
Random Forest Regression	**270.39**1	**145.253**

8.6 Recommendation Algorithm

The recommendation algorithm is to recommend to users laptops that are similar to the one they are interested in. Since there is no user ratings in our dataset, the approach we follow is content-based filtering. For this problem, we will use KNN with a weighted euclidean distance to find products that are most similar to a given laptop.

The first step is feature selection. We want to recommend based on criteria of price and the purpose of the laptop (working, gaming) rather than their specification. That is to say, we focus on laptops in the same price range and serving the same purpose for recommendation. Hence, five features are used in training data: cost, gaming score, working score, display score and portability score. After feature selection, the data is scaled using min-max scaling.

The next step is to define a distance to use in KNN. Weighted euclidean distance is used here with the weights for 5 features: cost, gaming score, working score, display score and portability score being 0.4,0.25,0.25,0.05 and 0.05; respectively. This ensure that laptops in the same price range with the given product are heavily prioritized, followed by that is laptops with the same working and gaming performance.

Part IV
Conclusion

In our analysis of laptop price prediction and insights, we identified key features influencing laptop prices, such as specifications, brand,... while addressing missing data and outliers for improved data quality. At the same time, our EDA results also show great capabilities to support the customer to choose a good laptop for their usage. The predictive model demonstrated promising accuracy, highlighting the potential for data-driven insights in understanding market trends and pricing strategies.

Weakness & Further Work

While the presented work achieved it's primary objective, there are still many limitation that we cannot address. Even though we got a big source of data through crawling, the crawling method is tedious and take too much time. At the same time, all of our data is static, and cannot update to the newest. As hard as it is, the crawled data is still not perfect. The format of the data is very inconsistent, rather random and non-uniform, which multiple missing value. As such, most of the method we used to cleaning the data is rather manual and cannot be automated. This also limits the number of data we can feed into the regression models. This make the final product rather hand-

crafted and require manual changing regularly to be used productively. Even after all of these, many of the data are duplicated, thus reduce the true number of usable data. For the further work, we would make some improvement, including: ensure data consistency, expanding dataset size and sources, and real-time integration.

Another weakness remarkably considered is that the interpretability of preprocessing technique at an extent, not deeply, sometimes, we are confused in selecting the suitable technique based on strong hypothesis or math formula rather solely experiment in the next task(regression) to ”fine-tune” the best strategy for preprocessing phase.

Finally, data is seemingly not enough to have a strong exploratory analysis in Insight part, especially, the EDA sub-part, where we see problem through the sight of a seller to advise laptop buyer to have final decision. For example, not all laptops are released recently, many of them can be released and evaluated at this time, can be outdated to the current, then the score is not actually fair if the website we scraped do not update this score based on the standard of the current technology. Many flaws in our assumption can be found to some extent if we have more data and actually take a deep insight into int. However, with aim to study as much as possible in this project, we try our best with our existing data.

Acknowledgment

We would like to express our sincere gratitude to Professor Than Quang Khoat, our instructor for this final semester project, for giving us invaluable knowledge through Data Science course. We also extend our heartfelt thanks to other lecturers for their unwavering supports and invaluable instructions in class. Besides, giving respect to community research and engineering in field of data science, whose public research material in the conference and article about data science and artificial intelligence. Finally, this report is heard by all members of our team.

References

[1] Christoph Molnar, ”Feature Importance,” *Interpretable Machine Learning*, `https://christophm.github.io/interpretable-ml-book/feature-importance.html`,

[2] Harris Drucker· Chris J.C. Burges” Linda Kaufman” ,Alex Smola·· Vladimir Vapoik + *Support Vector Regression Machines* Journal Advances in Neural Information Processing systems

[3] N.A.A. Jadhav, D.V.Dhamdhere, S.V.Varde *A k-Nearest Neighbor Regression Algorithm for Data Mining and Knowledge Discovery* Proceedings of the International Conference on Information Technology (ICIT), 2011.

[4] Leo Breiman, Jerome H. Friedman, Richard A. Olshen, and Charles J. Stone *Classification and Regression Trees* Wadsworth Brooks/Cole, 1986.

[5] Leo Breiman, Jerome Friedman, Richard Olshen, Charles Stone *Classification and Regression Trees (CART)* Book: Classification and Regression Trees,1984.

[6] Jerome H. Friedman *Greedy Function Approximation: A Gradient Boosting Machine* The Annals of Statistics, 2001.

[7] John L. H. McDonald *A Statistical Approach to Predictive Modeling: Modeling Categorical Features as Targets Proceedings of the Data Mining Workshop (2004).*

[8] Various, including Liaw, A. and Wiener, M. *Random Forests for Feature Selection* Proceedings of the 12th International Conference on Neural Information Processing (ICONIP) (2005).