

VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING



PROBABILITY AND STATISTICS

THE EVOLUTION OF COMPUTER PROCESSORS: A STATISTICS OF COMMON PROPERTIES

Instructor: Dr. Nguyen Thi Mong Ngoc
Students: Chau Dang Minh - 2013748

Ho Chi Minh City, May 2024

Evaluation

| N.O. | Student | ID | Works | Completed |
|------|--------------------|---------|-------------------------------------|-----------|
| 1 | Chau Dang Minh | 2212287 | Dataset overview Preprocessing | 100% |
| 2 | Ha Khoi Nguyen | 2212287 | Descriptive statistics | 100% |
| 3 | Nguyen Thi Mai Anh | 2210103 | Theories Slides and Presentation | |
| 4 | Võ Ninh Giang | 2210834 | Inferential statistics | |
| 5 | Trinh Viet Cuong | 2210447 | Inferential statistics | |

Introduction

Phenomena that are meaningful to humans appear not to be stochastic. In the same sense, datasets produced by humans, or nature in time circulations have insights to be analyzed, which is accounted by Statistics. Thanks to Dr. Nguyen Thi Mong Ngoc's supervision in Probability and Statistics course, we have a chance to study basic statistics within an assignment with a tiny dataset. We organized our report in the following structure

1. Overview of the dataset. In this chapter, we carefully describe in details as much as possible the dataset, specifically the properties of each instance. We also notice which features to be used for later statistical tasks.
2. Preprocessing. We process data cleaning and some computations.
3. Descriptive statistics. We calculate some qualitative features of the dataset.
4. Inferential statistics. Our problems are explicitly stated and solved.

Contents

| | | |
|------------|---|-----------|
| I | Dataset Overview | 4 |
| II | Foundations | 6 |
| 1 | Theoretical Foundation of Sampling | 6 |
| 1.1 | Terminology | 6 |
| 1.2 | Statistical characteristics | 6 |
| 2 | Some General Principles of Statistical Hypothesis Testing | 7 |
| III | Data Preprocessing | 9 |
| 1 | Data Cleaning | 9 |
| 2 | Data Pre-computation | 11 |
| IV | Descriptive Statistics | 14 |
| V | Inferential Statistics | 15 |
| VI | Discussion | 16 |

Chapter I

Dataset Overview

As Computer Science students, we are assigned to analyze a [dataset about computer processors](#), namely CPUs and GPUs. Our dataset is credited to Intel, Game-Debate, and the companies involved in producing the part. Information of CPUs and GPUs are collected separately into two files, namely `Intel_CPUs.csv` and `All_GPUs.csv`. Let us get familiar to some technical features, given in the metadata or acquired over the internet.

Table I.1: Some technical features of CPUs and GPUs

| N.O. | Feature name | Relevant to | Details |
|------|-----------------------|-------------|---|
| 1 | Lithography | CPU, GPU | The semiconductor technology used to manufacture an integrated circuit, and is reported in nanometer |
| 2 | Number of Cores | CPU | A hardware term that describes the number of independent central processing units |
| 3 | Number of Threads | CPU | A Thread, or thread of execution, is a software term for the basic ordered sequence of instructions that can be passed |
| 4 | Base Frequency | CPU | Describes the rate at which the processor's transistors open and close. |
| 5 | Cache | CPU | An area of fast memory located on the processor. |
| 6 | Thermal Design Power | CPU | Represents the average power, in watts, the processor dissipates when operating at Base Frequency with all cores active under an Intel-defined, high-complexity workload. |
| 7 | Embedded Availability | CPU | In essence, an embedded processor is a CPU chip used in a system which is not a general-purpose workstation, laptop or desktop computer. |
| 8 | Embedded Availability | CPU | In essence, an embedded processor is a CPU chip used in a system which is not a general-purpose workstation, laptop or desktop computer. |
| 9 | Memory Types | CPU | Single Channel, Dual Channel, Triple Channel, and Flex Mode. |

Continued on next page

Table I.1: Some technical features of CPUs and GPUs (Continued)

| | | | |
|----|---------------------------------|----------|--|
| 10 | Instruction Set | CPU | |
| 11 | Maximal Temperature | CPU | |
| 12 | Architecture | GPU | |
| 13 | Dedicated and Integrated | GPU | Whether the GPU is solely used or shares memory with a CPU |
| 14 | (Front-side) Bus Speed | CPU, GPU | The speed at which data is transferred between the processors and other components such as the memory, chipset, and peripherals. |
| 15 | No-Execute Bit | CPU | Hardware-based security feature that can reduce exposure to viruses and malicious-code attacks. |
| 16 | Thermal Monitoring Technologies | CPU | Protects the processor package and the system from thermal failure through several thermal management features. |

Chapter II

Foundations

The Statistical Hypothesis is a statement or assumption about the parameters of one or more populations, which may be true or false. Each such truth value can be expressed in the form of a hypothesis, usually denoted as H_0 , called the null hypothesis, and an alternative hypothesis, usually denoted as H_1 or H_a , which is its complement.

1 Theoretical Foundation of Sampling

1.1 Terminology

1. A population is a set $\{x_i\}_{i=1}^N$ of elements belonging to the object under study, which need to be observed, collected, and analyzed based on certain characteristics. A parameter is a numerical description of a population.
2. A sample is a subset $\{x_{i_j}\}_{j=1}^n$ of units selected from the population according to a certain sampling method. A statistic is a numerical description of a sample.
3. Let the population belong to a distribution represented by a random variable X . A random sample is a collection $\{X_i\}_{i=1}^n$ such that X, X_1, \dots, X_n are independent and identically distributed.

1.2 Statistical characteristics

Statistical characteristics are important properties on a population, a sample or a random sample directly related to the content of the study and survey, which need to be collected from the units of the population, generally divided into two groups

1. Central tendency includes mean, median, mode, giving us an idea of the trend of the data points clustering around certain values.
2. Dispersion includes variance, standard deviation, which describe the spread or variability of the data points around the central tendency.

In details, these quantities are formulated as followings.

- Population mean

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i. \quad (2.1.1)$$

- Sample mean

$$\bar{x} = \frac{1}{n} \sum_{j=1}^n x_{i_j}. \quad (2.1.2)$$

- Random sample mean

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i. \quad (2.1.3)$$

- Population variance

$$\sigma^2 = \frac{1}{N} \sum_{n=1}^N (x_n - \bar{x})^2. \quad (2.1.4)$$

- Sample variance

$$s^2 = \frac{1}{n-1} \sum_{j=1}^n (x_{i_j} - \bar{x})^2. \quad (2.1.5)$$

- Random sample variance

$$S^2 = \frac{1}{N-1} \sum_{n=1}^N (X_i - \bar{X})^2. \quad (2.1.6)$$

- Population, sample and random sample derivations are the square roots σ , s and S , respectively.

- Sample ratio

$$f = \frac{n}{N} \quad (2.1.7)$$

- Coefficient of variation

$$CV = \left(\frac{s}{\bar{x}} \right) \cdot 100\% \quad (2.1.8)$$

- Suppose that the sample $\{x_{i_j}\}_{j=1}^n$ is in increasing order. The sample median is calculated as

$$\text{med} = \begin{cases} x_{i_{k+1}}, & \text{if } n = 2k + 1 \\ \frac{1}{2}(x_{i_k} + x_{i_{k+1}}), & \text{if } n = 2k. \end{cases} \quad (2.1.9)$$

- Quartiles include three values Q_1, Q_2 and Q_3 . Where Q_2 is the median, Q_1 and Q_3 is the median of the set of samples less than and more than Q_2 , respectively. The quantity $IQR = Q_3 - Q_1$ is called the interquartile range.
- An outlier is a point outside $(Q_1 - 1.5 \times IQR, Q_3 + 1.5 \times IQR)$.

2 Some General Principles of Statistical Hypothesis Testing

The process of testing a statistical hypothesis is a standardized procedure for making decisions to reject or not reject a hypothesis based on sample data. This process is called hypothesis testing and typically involves four steps:

1. **Setting Hypotheses:** This step involves stating which hypothesis is null and which is alternative. The hypotheses are formulated in a mutually exclusive manner, meaning if one is true, the other must be false.
2. **Constructing an Analysis Plan:** The analysis plan outlines how sample data will be used to evaluate the null hypothesis. Evaluation typically centers around a single test statistic.
3. **Analyzing Sample Data:** Determine the values of the sample statistic (mean, proportion, t-statistic, z-score, etc.) as described in the analysis plan.
4. **Interpreting Results:** Apply decision rules outlined in the analysis plan. If the observed result is inconsistent with the null hypothesis, reject it.

Similar to estimation, statistical hypothesis testing does not provide results with 100% certainty but rather with a certain level of confidence, and errors can occur. Errors can be classified into two types:

- **Type I Error:** Rejecting the null hypothesis H_0 and accepting the alternative hypothesis H_1 whereas H_0 is true.
- **Type II Error:** Failing to reject the null hypothesis H_0 and not accepting the alternative hypothesis H_1 whereas H_1 is true.

Both types of errors can have adverse consequences. Depending on the situation, it's assessed which type of error leads to more serious consequences and should be minimized.

To make decisions about rejecting the null hypothesis, statisticians rely on specific rules. These rules are listed in the analysis plan. Traditionally, statisticians describe these decision rules in two ways: referencing a p-value or referencing a region of acceptance.

- **p-value:** The strength of evidence in favor of a null hypothesis is measured by the p-value. Suppose the test statistic is denoted by S . The p-value is the probability of observing a test statistic as extreme as S , assuming the null hypothesis is true. If the p-value is smaller than the significance level, we reject the null hypothesis.
- **Region of Acceptance:** The region of acceptance is a range of values. If the sample statistic falls within the region of acceptance, the null hypothesis is not rejected. The region of acceptance is set so that the probability of committing a Type I error is equivalent to the significance level.

Values outside the region of acceptance are called the region of rejection. If the sample statistic falls within the region of rejection, the null hypothesis is rejected. In such cases, it's said that the null hypothesis has been rejected at the significance level.

Chapter III

Data Preprocessing

1 Data Cleaning

With RStudio, the working directory is automatically determined. Otherwise, it can be indicated by `here` library.

Listing III.1: Required libraries and working directory setup

```
1 # Libraries and options
2 library(dplyr)
3 library(here)
4 library(knitr)
5 library(kableExtra)
6
7 # Self-defined functions
8 source("utils.R")
9
10 # Working directory
11 setwd(here())
```

Now our working directory have been explicated, we can use relative paths to read the data. With RMarkdown, we can prettify the rendering.

Listing III.2: RStudio data object initialization

```
1 # Read the CSV file into a data frame
2 cpu_data <- read.csv("dataset/Intel_CPUs.csv")
3 gpu_data <- read.csv("dataset/All_GPUs.csv")
4
5 # Inspect the CPU data
6 kable(head(cpu_data), format = "html") %>%
7   kable_styling()
```

Invalid cells may contain NA, an empty string, or other values showing us that this cell's data was not collecting correctly. At the very first step, we want to select only columns such that the percentage of valid cells exceeds our predefined value. Then we filter out all instances with invalid features. Note that careful column selection possibly remains more instances for later tasks. For CPU data, we concern about the price, so let us choose a quota for which the column of prices is maintained.

Listing III.3: Cleaning functions

```
1 # Check if a cell has a valid value
2 is_valid <- function(value) {
3   return(!is.na(value))
}
```

| Product_Collection | Vertical_Segment | Processor_Number | Status | Launch_Date | Lithography | Recommended_Customer_Price | nb_of_Cores | nb_of_Threads | Pr |
|---|------------------|------------------|-------------|-------------|-------------|----------------------------|-------------|---------------|-----|
| 7th Generation Intel® Core™ i7 Processors | Mobile | i7-7Y75 | Launched | Q3'16 | 14 nm | \$393.00 | 2 | 4 | 1.3 |
| 8th Generation Intel® Core™ i5 Processors | Mobile | i5-8250U | Launched | Q3'17 | 14 nm | \$297.00 | 4 | 8 | 1.6 |
| 8th Generation Intel® Core™ i7 Processors | Mobile | i7-8550U | Launched | Q3'17 | 14 nm | \$409.00 | 4 | 8 | 1.8 |
| Intel® Core™ X-series Processors | Desktop | i7-3820 | End of Life | Q1'12 | 32 nm | \$305.00 | 4 | 8 | 3.6 |
| 7th Generation Intel® Core™ i5 Processors | Mobile | i5-7Y57 | Launched | Q1'17 | 14 nm | \$281.00 | 2 | 4 | 1.2 |
| Intel® Celeron® Processor 3000 Series | Mobile | 3205U | Launched | Q1'15 | 14 nm | \$107.00 | 2 | 2 | 1.5 |

Figure 3.1.1: First instances of CPUs data

```

4      & !is.null(value)
5      & !value == ""
6      & !value == "N/A"
7      & !trimws(value) == "-"
8      & !value == "missing"
9      & !value == "unknown")
10     # Add your criteria
11   }
12
13
14   # Select columns with enough valid cells
15   filtered_data <- function(data, valid_percentage=0.5) {
16     selected_columns <- character(0)
17
18     for (col in colnames(data)) {
19       valid_count <- sum(is_valid(data[[col]]))
20       total_instances <- length(data[[col]])
21
22       if ((valid_count / total_instances) >= fill) {
23         selected_columns <- c(selected_columns, col)
24       }
25     }
26
27     return(data[selected_columns])
28   }

```

Listing III.4: Cleaned CPU data

```

1   filtered_cpu_data <- filtered_data(cpu_data, valid_percentage=0.5)
2
3   processed_cpu_data <-
4     filtered_cpu_data[
5       apply(filtered_cpu_data, 1, function(row) all(sapply(row, is_valid))), ]
6
7   selected_cpu_data <- processed_cpu_data[] # Adjust selected columns for your later needs
8   selected_cpu_data <- unique(selected_cpu_data)
9   kable(head(selected_cpu_data), format = "html") %>%
10  kable_styling()

```

Listing III.5: Cleaned GPU data

```

1 filtered_gpu_data <- filtered_data(gpu_data, valid_percentage=0.5)
2
3 processed_gpu_data <-
4   filtered_gpu_data[
5     apply(filtered_gpu_data, 1, function(row) all(sapply(row, is_valid))), ]
6
7 selected_gpu_data <- processed_gpu_data[] # Adjust selected columns for your later needs
8 selected_gpu_data <- unique(selected_gpu_data)
9 kable(head(selected_gpu_data), format = "html") %>%
10  kable_styling()

```

| | Product_Collection | Vertical_Segment | Processor_Number | Status | Launch_Date | Lithography | Recommended_Customer_Price | nb_of_Cores | nb_of_Threads |
|----|---|------------------|------------------|----------|-------------|-------------|----------------------------|-------------|---------------|
| 1 | 7th Generation Intel® Core™ i7 Processors | Mobile | i7-7Y75 | Launched | Q3'16 | 14 nm | \$393.00 | 2 | 4 |
| 2 | 8th Generation Intel® Core™ i5 Processors | Mobile | i5-8250U | Launched | Q3'17 | 14 nm | \$297.00 | 4 | 8 |
| 3 | 8th Generation Intel® Core™ i7 Processors | Mobile | i7-8550U | Launched | Q3'17 | 14 nm | \$409.00 | 4 | 8 |
| 5 | 7th Generation Intel® Core™ i5 Processors | Mobile | i5-7Y57 | Launched | Q1'17 | 14 nm | \$281.00 | 2 | 4 |
| 11 | Intel® Pentium® Processor 2000 Series | Mobile | 2020M | Launched | Q3'12 | 22 nm | \$134.00 | 2 | 2 |
| 14 | Intel® Pentium® Processor 4000 Series | Mobile | 4405U | Launched | Q3'15 | 14 nm | \$161.00 | 2 | 4 |

Figure 3.1.2: First instances of selected CPUs data

It's worth finding the key of the data to know which feature uniquely determine an instance. We prioritize categorical features, specifically pure names.

Listing III.6: Keys of the data

```

1 nrow(selected_cpu_data) - nrow(selected_cpu_data[, c("Product_Collection",
2   "Processor_Number")])
3 nrow(selected_gpu_data) - nrow(selected_gpu_data[, c("Name")])
4 # Outputs: 0

```

2 Data Pre-computation

Some features in our data have values that need to be reformatted for easily later sorting and analyses. Therefore, we need to gain a good understand on the features.

Listing III.7: A processing for selected features

```

1 cpu_columns <- colnames(cpu_data)
2 gpu_columns <- colnames(gpu_data)
3 intersect(cpu_columns, gpu_columns)
4 # Output: character(0)

```

Since the data files have no common features, we took a look at them independently and decided to pre-compute some features.

| | Architecture | Best_Resolution | Core_Speed | DVI_Connection | Dedicated | Direct_X | HDMI_Connection | Integrated | L2_Cache | Manufacturer | Max_Power |
|---|------------------|-----------------|------------|----------------|-----------|----------|-----------------|------------|----------|--------------|-----------|
| 2 | R600 XT | 1366 x 768 | - | 2 | Yes | DX 10 | | 0 No | 0KB | AMD | 215 Watts |
| 3 | R600 PRO | 1366 x 768 | - | 2 | Yes | DX 10 | | 0 No | 0KB | AMD | 200 Watts |
| 5 | RV630 | 1024 x 768 | - | 2 | Yes | DX 10 | | 0 No | 0KB | AMD | 45 Watts |
| 6 | RV630 | 1024 x 768 | - | 2 | Yes | DX 10 | | 0 No | 0KB | AMD | 50 Watts |
| 7 | R700 RV790 XT | 1920 x 1080 | 870 MHz | 1 | Yes | DX 10.1 | | 1 No | 0KB | AMD | 190 Watts |
| 8 | R600 GT | 1024 x 768 | - | 2 | Yes | DX 10 | | 0 No | 0KB | AMD | 150 Watts |

Figure 3.1.3: First instances of selected GPUs data

1. Use the common column name `Release_Date` for both data. Extract `Release_Year` and `Release_Quarter` for each instance.
2. If `Recommended_Customer_Price` is a range, compute the average.

Listing III.8: Extract Year and Quarter from Dates

```

1 month_to_quarter <- function(month) {
2   quarter <- switch(month,
3     "Jan" = "1",
4     "Feb" = "1",
5     "Mar" = "1",
6     "Apr" = "2",
7     "May" = "2",
8     "Jun" = "2",
9     "Jul" = "3",
10    "Aug" = "3",
11    "Sep" = "3",
12    "Oct" = "4",
13    "Nov" = "4",
14    "Dec" = "4",
15    "Unknown")
16   return(quarter)
17 }
18
19 names(selected_cpu_data)[names(selected_cpu_data) == "Launch_Date"] <- "Release_Date"
20 selected_cpu_data$Release_Year <- as.integer(sub("Q[1-4]'(\\d+)", "\\1",
21   gsub("\\s+", "", selected_cpu_data$Release_Date))) +
22   2000
23 selected_cpu_data$Release_Quarter <- as.integer(sub("Q([1-4])'.*", "\\1",
24   gsub("\\s+", "", selected_cpu_data$Release_Date)))
25 selected_gpu_data$Release_Year <-
26   as.integer(sub(".*-(\\d{4}) .*", "\\1", selected_gpu_data$Release_Date))
27 selected_gpu_data$Release_Quarter <-

```

```
28 as.character(sub(".*-(\\w+)-\\d{4}.*", "\\1", selected_gpu_data$Release_Date))
29 selected_gpu_data$Release_Quarter <- sapply(selected_gpu_data$Release_Quarter, month_to_quarter)
```

Chapter IV

Descriptive Statistics

Chapter V

Inferential Statistics

Chapter VI

Discussion