Vietnam National University Ho Chi Minh City
Ho Chi Minh City University of Technology
Faculty of Computer Science and Engineering

ASSIGNMENT

PROBABILITY AND STATISTICS

TOPIC: CLASSIFICATION AND PREDICTION OF INTERNET ADVERTISEMENTS USING LOGISTIC REGRESSION MODELS

GROUP: 13
Supervisor: Nguyễn Tiến Dũng

Student Name	Student ID
Nguyễn Tiến Minh	2352755
Tán Khánh Phong	2352911
Thái Đức Khang	2352500
Nguyễn Hữu Cầu	2352129
Nguyễn Bùi Huỳnh Khương	2352637

May, 2025

Team Task Assignment

Task	**Assigned Members**
Linear Regression Theory	Nguyễn Tiến Minh, Nguyễn Bùi Huỳnh Khương
Literature Search	All members
Description of the Data	All members
Data Cleaning	All members
Descriptive Statistics: histogram, boxplot, scatter plot; mean, median	Tán Khánh Phong, Nguyễn Bùi Huỳnh Khương, Nguyễn Hữu Cầu
Simple Linear Regression	Tán Khánh Phong, Nguyễn Bùi Huỳnh Khương
Multiple Linear Regression	Nguyễn Tiến Minh, Thái Đức Khang, Nguyen Hữu Cầu
R Code Implementation	Nguyễn Tiến Minh, Tán Khánh Phong, Nguyễn Bùi Huỳnh Khương
Summary	All members
Report Writing	All members

Contents

1 Introduction

In the age of digital media, distinguishing between web content and advertisements is an essential task in areas such as online user experience optimization, web scraping, and ad-blocking technologies. Web advertisements are often embedded as images with distinctive properties that can be statistically analyzed.

This project aims to explore and analyze a dataset containing image-based features in order to classify whether an image is an advertisement or not. The dataset includes variables such as image `height`, `width`, the `height-to-width ratio`, and a `locality` score that may reflect content placement or layout structure.

The goal of this report is to build statistical models, particularly using logistic regression techniques, to predict the binary target variable—advertisement (1) or non-advertisement (0). The methodology involves data preprocessing, exploratory data analysis, statistical modeling, and performance evaluation using real-world data. By applying logistic regression, a method not covered in the classroom, this project also aims to demonstrate the practical application of advanced statistical techniques to solve a classification problem in a web-related context.

1.1 Data Introduction

- **Data Source:** add.csv
- **Acknowledgements:** The dataset is hosted by the UCI Machine Learning Repository, provided by M. Lichman (2013). Source: `http://archive.ics.uci.edu/ml`.
- **Contents:** This dataset contains numerical attributes describing image features, aimed at predicting whether an image is an advertisement ("ad") or not ("nonad").
- **Population:** Collection of images labeled from various internet sources.
- **Number of Variables:** 1,559
- **Number of Observations:** 3,278

The main variables in the analysis include:

- **Height (Numerical):** Height of the image, with approximately 28% missing data.
- **Width (Numerical):** Width of the image, also with around 28% missing data.
- **Aspect_Ratio (Numerical):** Aspect ratio, calculated from height and width, with 28% missing values.
- **URL_Term Features (Numerical):** Columns 3 to 1557 contain numerical attributes derived from URL terms and image features, with column 3 (named *Local*) being a typical example.
- **Label (Categorical):** Classification label with two categories: "ad" (advertisement) and "nonad" (non-advertisement).

2 Theoretical Background of Logistic Regression

2.1 Definition and Purpose

Logistic regression is a statistical method used for solving binary classification problems, where the outcome variable y can take only two values (e.g., 0 or 1, Yes or No, Ad or Non-Ad). The goal is to model the probability that $y = 1$ given a set of input features $\boldsymbol{x}$.

Unlike linear regression, which predicts continuous outcomes, logistic regression models probabilities constrained to the interval [0, 1] by applying a nonlinear sigmoid transformation to a linear combination of input variables.

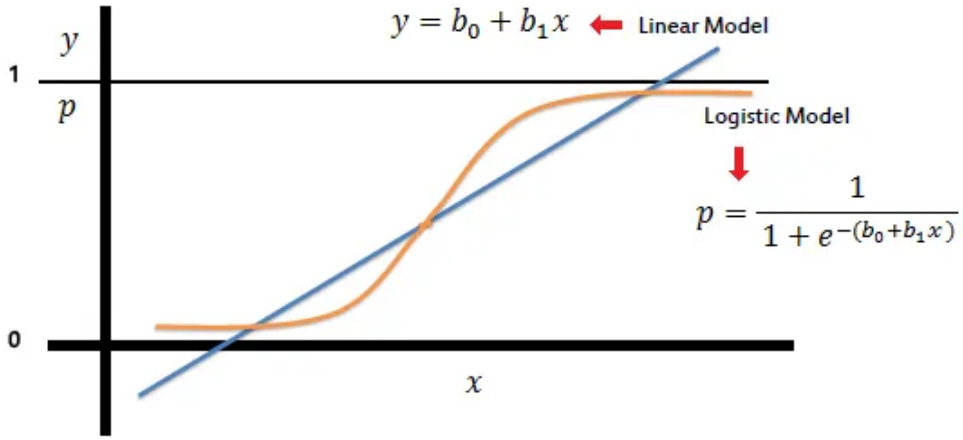

Figure 2.1.1: Logistic regression uses a nonlinear sigmoid function to map inputs to probabilities.

2.2 The Sigmoid Function and Probability Model

The sigmoid (logistic) function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

where $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$ is the linear combination of features. This transformation ensures the predicted value lies between 0 and 1, suitable for interpreting as a probability.

Hence, the logistic model estimates:

$$P(y = 1 \mid \boldsymbol{x}) = \sigma(\boldsymbol{x}^\top \boldsymbol{\beta})$$

2.3 Parameter Estimation via Maximum Likelihood

Logistic regression parameters $\boldsymbol{\beta}$ are typically estimated using **Maximum Likelihood Estimation (MLE)**. The log-likelihood function for a dataset of n samples is:

$$\mathcal{L}(\boldsymbol{\beta}) = \sum_{i=1}^{n} [y_i \log(p_i) + (1 - y_i) \log(1 - p_i)]$$

where $p_i = \sigma(\boldsymbol{x}_i^\top \boldsymbol{\beta})$ is the predicted probability. Since this function has no closed-form solution, optimization techniques are used:

- **Gradient Descent (GD)** – iterative updates using the gradient of the loss.

- **Newton-Raphson / IRLS (Iteratively Reweighted Least Squares)** – uses second-order derivatives for faster convergence.

In R, the `glm()` function performs logistic regression using IRLS by default.

2.4 Loss Function: Cross-Entropy

Training the logistic regression model involves minimizing the **cross-entropy loss**, which quantifies the distance between the actual and predicted probability distributions:

$$L = -\frac{1}{n}\sum_{i=1}^{n}[y_i \log(p_i) + (1 - y_i)\log(1 - p_i)]$$

Where:

- y_i: true label (0 or 1)
- p_i: predicted probability
- n: number of samples

The loss is minimized when the model predicts values close to the true labels.

2.5 Model Evaluation Metrics

Logistic regression models are evaluated using various metrics:

1. Confusion Matrix

Actual \ Predicted	1 (Ad)	0 (Non-Ad)
1 (Ad)	True Positive (TP)	False Negative (FN)
0 (Non-Ad)	False Positive (FP)	True Negative (TN)

Accuracy is computed as:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

While accuracy is easy to compute, it can be misleading on imbalanced datasets.

2. ROC Curve and AUC

The **ROC Curve (Receiver Operating Characteristic)** plots the trade-off between: - **True Positive Rate (TPR)**:

$$\text{TPR} = \frac{TP}{TP + FN}$$

- **False Positive Rate (FPR)**:

$$\text{FPR} = \frac{FP}{FP + TN}$$

The curve starts from (0, 0) and ideally moves toward (0, 1). A random classifier would produce a diagonal line from (0, 0) to (1, 1).

Area Under the Curve (AUC) is a scalar value between 0 and 1 summarizing the ROC curve:

- AUC = 1.0: Perfect classifier

- AUC = 0.5: Random guess
- AUC > 0.9: Outstanding discrimination.
- 0.8 ≤ AUC < 0.9: Excellent discrimination.
- 0.7 ≤ AUC < 0.8: Acceptable discrimination.

AUC is threshold-independent and robust for imbalanced datasets. It is often preferred over accuracy when evaluating classifiers.

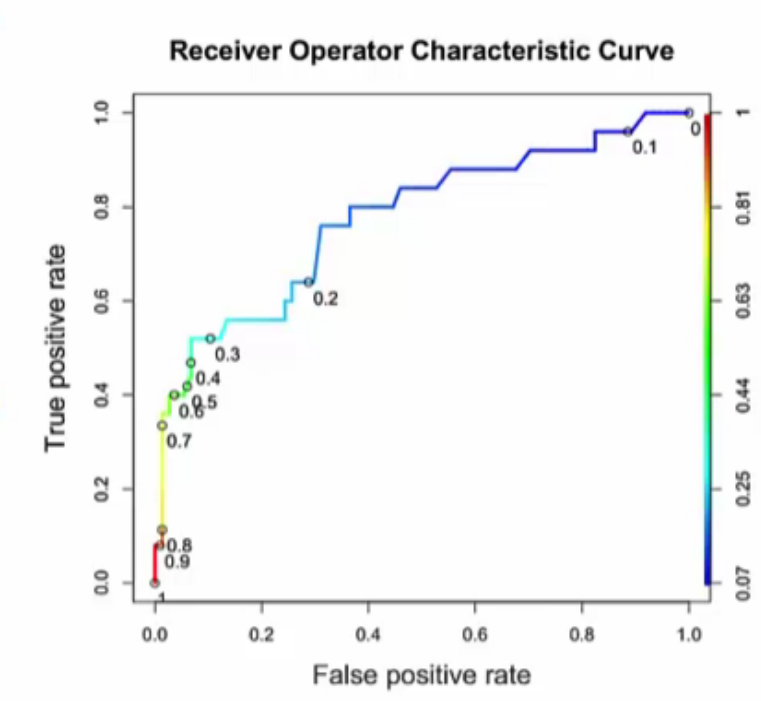

Figure 2.5.1: Example ROC Curve: AUC represents the area under the curve.

3 Data Preprocessing

3.1 Context

The task is to predict whether an image is an advertisement ("ad") or not ("nonad").

3.2 Description of the Data

The dataset used in this project originates from the UCI Machine Learning Repository and is publicly available on Kaggle. Each record in the dataset corresponds to a single image, labeled either as an advertisement ("ad") or a non-advertisement ("nonad").

The complete dataset includes 1,559 variables, most of which are numerical attributes extracted from HTML and image metadata. For the purpose of this project, we selected only 4 key numeric variables:

- Column 0 **(height)**: the pixel height of the image
- Column 1 **(width)**: the pixel width of the image
- Column 2 **(ratio)**: calculated as height divided by width
- Columns 3 to 1557: Represent URL and term-related data (with **"local (column 3)"** used as a representative column for the URL that may relate to content layout or rendering locality).

The target variable is a binary indicator which is represented by the last column in the dataset and contains two possible values:

- 1 = Advertisement image (`ad`)
- 0 = Non-advertisement image (`nonad`)

	0	1	2	3	4	5	6	7	8	9	...	1549	1550	1551	1552	1553	1554	1555	1556	1557	1558
0	125	125	1	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ad.
1	57	468	8.2105	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ad.
2	33	230	6.9696	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ad.
3	60	468	7.8	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ad.
4	60	468	7.8	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	ad.
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3274	170	94	0.5529	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	nonad.
3275	101	140	1.3861	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	nonad.
3276	23	120	5.2173	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	nonad.
3277	?	?	?	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	nonad.
3278	40	40	1	1	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	nonad.

Figure 3.2.1: Initial view of the dataset before preprocessing

0 height of picture	1 width of picture	2 aspect ratio	3 3 to 1557 colums are u.r.l+term(n)	4
? 28% 60 9% Other (2093) 64%	? 27% 468 6% Other (2192) 67%	? 28% 1 9% Other (2068) 63%	1 76% 0 23% Other (15) 0%	0 1
125	125	1	1	0
57	468	8.2105	1	0
33	230	6.9696	1	0
60	468	7.8	1	0
60	468	7.8	1	0
60	468	7.8	1	0
59	460	7.7966	1	0

Figure 3.2.2: Preview of raw dataset structure and distribution summary

3.3 Data Reading

Before working with the provided dataset, we need a quick overview to get familiar with its structure for better understanding and analysis. We use the `read.csv()` and `head()` commands to read the dataset from the CSV file and then print out the first few rows. The dataset summary is shown in Figure 3.3.1.

The dataset includes image attributes (height, width, aspect ratio) and a binary number of URL and term-related features from column 3 to 1557. As the number of columns is too large to be fully examined and managed, we cannot use all these features directly. Therefore, we select a representative and easily interpretable feature, the *local* attribute, which indicates whether an image is hosted locally or remotely. This helps reduce dimensionality while retaining meaningful information.

```
> head(add_data,10)
   height width  ratio local target_raw
1     125   125      1     1         ad.
2      57   468 8.2105     1         ad.
3      33   230 6.9696     1         ad.
4      60   468    7.8     1         ad.
5      60   468    7.8     1         ad.
6      60   468    7.8     1         ad.
7      59   460 7.7966     1         ad.
8      60   234    3.9     1         ad.
9      60   468    7.8     1         ad.
10     60   468    7.8     1         ad.
>
```

Figure 3.3.1: First 10 rows of the dataset before preprocessing

3.4 Clean the data

The dataset initially consists of columns with data types such as *string* and *integer*. To prepare the data for analysis, we used the `as.numeric()` function to convert these columns into *numeric* format. Specifically, the *height*, *width*, *ratio*, and *local* columns, which initially contained string representations of numbers, were all converted to numeric values.

The initial inspection of the dataset shows that missing values are represented by the symbol **"?"** instead of standard `NA` markers. Specifically, missing data appear in the continuous attributes such as *height*, *width*, and *ratio*. Before applying any preprocessing techniques, we first converted all "?" symbols into proper `NA` values so that they can be handled correctly in the subsequent steps.

As shown in **Figure 3.4.2**, the *ratio* column contains 28% missing values, while *height* and *width* contain 28% and 27% missing values, respectively. If these missing values are not properly addressed, machine learning models and statistical algorithms may either fail to execute or produce unreliable results. In particular, **logistic regression models cannot handle missing values** and will return errors during training if the dataset contains any `NA` values. Therefore, handling missing data is a critical preprocessing step.

```
> summary(add_data)
     height           width            ratio            local
 Min.   :  1.00   Min.   :  1.0   Min.   : 0.0015   Min.   :0.0000
 1st Qu.: 25.00   1st Qu.: 80.0   1st Qu.: 1.0357   1st Qu.:1.0000
 Median : 51.00   Median :110.0   Median : 2.1020   Median :1.0000
 Mean   : 64.02   Mean   :155.3   Mean   : 3.9120   Mean   :0.7672
 3rd Qu.: 85.25   3rd Qu.:184.0   3rd Qu.: 5.3333   3rd Qu.:1.0000
 Max.   :640.00   Max.   :640.0   Max.   :60.0000   Max.   :1.0000
 NA's   :903      NA's   :901     NA's   :910       NA's   :15
                                 missing  %
                      ratio          910 28
                      height         903 28
                      width          901 27
                      local           15  0
                      target_raw       0  0
```

Figure 3.4.1: Percentage of missing data before preprocessing

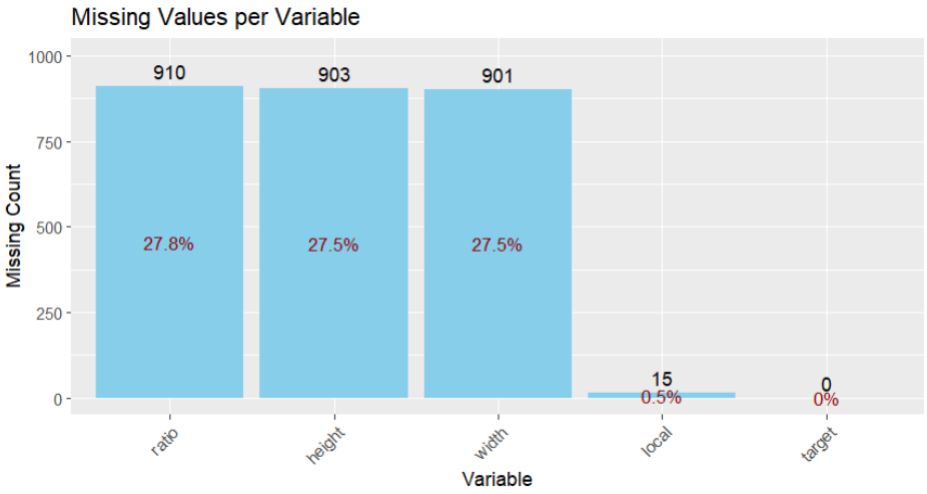

Figure 3.4.2: Percentage of missing data before preprocessing

In this study, we replaced missing values by imputing the height and width columns with the median of each corresponding column. For the ratio column, which depends on the ratio between width and height, missing values were imputed based on this relationship. This approach ensures that the imputation for the ratio column remains consistent with its underlying dependency on the other two columns. Additionally, we converted the target variable into a binary format, where records labeled as 'ad.' were mapped to 1 and others to 0. This transformation ensures that the target variable is suitable for machine learning models such as logistic regression. Using the median for imputing height and width helps stabilize the dataset, making it more robust and reliable for machine learning models like logistic regression, as the median is less sensitive to extreme values (see **Figure 3.4.3**).

	height	width	ratio	local	target
229	60	468	7.8000	1	1
230	90	120	1.3333	*NA*	1
231	90	120	1.3333	*NA*	1
232	90	120	1.3333	*NA*	1
233	90	120	1.3333	*NA*	1
234	60	468	7.8000	1	1
235	*NA*	*NA*	*NA*	1	1
236	*NA*	*NA*	*NA*	1	1
237	*NA*	*NA*	*NA*	1	1
238	*NA*	*NA*	*NA*	1	1
239	93	261	2.8064	1	1

⇒

	height	width	ratio	local	target
228	60	88	1.4666000	1	1
229	60	468	7.8000000	1	1
230	90	120	1.3333000	*NA*	1
231	90	120	1.3333000	*NA*	1
232	90	120	1.3333000	*NA*	1
233	90	120	1.3333000	*NA*	1
234	60	468	7.8000000	1	1
235	51	110	0.4636364	1	1
236	51	110	0.4636364	1	1
237	51	110	0.4636364	1	1

Figure 3.4.3: Column data before (left) and after (right) median imputation for missing values.

Moreover, using the median reduces bias in cases where the data distribution is skewed or non-normal. Consequently, models trained on the imputed data are expected to perform better and are less likely to be negatively impacted by inconsistencies in the dataset.

However, for the **local** column, the number of missing values is relatively small — only **15 records** (accounting for less than **0.5%** of the dataset). Additionally, since the local column contains **binary data** (0 and 1) with **1 being the majority class**, imputing missing values using the median would default to 1, potentially introducing bias and distorting the true distribution.

Therefore, in this case, **removing the incomplete records is a safer and more reliable choice**, as it avoids imputation bias while having minimal impact on the overall dataset (see **Figure 3.4.4**).

	height	width	ratio	local	target
228	60	88	1.4666000	1	1
229	60	468	7.8000000	1	1
230	90	120	1.3333000	NA	1
231	90	120	1.3333000	NA	1
232	90	120	1.3333000	NA	1
233	90	120	1.3333000	NA	1
234	60	468	7.8000000	1	1
235	51	110	0.4636364	1	1
236	51	110	0.4636364	1	1
237	51	110	0.4636364	1	1

⇒

	height	width	ratio	local	target
224	60	88	1.4666000	1	1
225	51	110	0.4636364	1	1
226	60	88	1.4666000	1	1
227	60	468	7.8000000	1	1
228	60	88	1.4666000	1	1
229	60	468	7.8000000	1	1
234	60	468	7.8000000	1	1
235	51	110	0.4636364	1	1
236	51	110	0.4636364	1	1
237	51	110	0.4636364	1	1
238	51	110	0.4636364	1	1

Figure 3.4.4: Column data before (left) and after (right) removing rows with missing *local* values

```
> freq.na(add_data)
        missing %
local        15 0
height        0 0
width         0 0
ratio         0 0
target        0 0
> add_data <- na.omit(add_data)
```

Figure 3.4.5: Table of missing values

In particular, after preprocessing, the only remaining missing values are found in the **local** column. Thus, we applied the **`na.omit()`** function in R to remove all rows containing missing values. This approach ensures that the dataset remains clean, consistent, and free from the risk of introducing artificial information during analysis.

```
# check missing variable
        library(questionr)
        freq.na(add_data)
        add_data <- na.omit(add_data)
```

The dataset is now complete and free of missing values. To confirm this, we used the **`freq.na()`** function from the `questionr` package in RStudio. This function provides a frequency table of missing values, allowing us to verify that all missing values have been properly handled and imputed. By running this check, we ensure that the dataset is ready for further analysis and modeling without any remaining issues related to missing data.

```
# check missing variable
        library(questionr)
        freq.na(add_data)
```

As a result:

```
> freq.na(add_data)
        missing %
height        0 0
width         0 0
ratio         0 0
local         0 0
target        0 0
>
```

Figure 3.4.6: Table of missing values

4 Descriptive statistics

After the cleaning data step, we now have a summary of new data file by using summary() in R. Then, we have the summary of the data in **Figure 4.0.1**.

```
> summary(add_data)
     height           width            ratio              local            target        
 Min.   :  1.00   Min.   :  1.0   Min.   : 0.0015   Min.   :0.0000   Min.   :0.0000  
 1st Qu.: 32.00   1st Qu.: 90.0   1st Qu.: 0.4636   1st Qu.:1.0000   1st Qu.:0.0000  
 Median : 51.00   Median :110.0   Median : 1.1904   Median :1.0000   Median :0.0000  
 Mean   : 60.45   Mean   :142.9   Mean   : 2.9584   Mean   :0.7672   Mean   :0.1391  
 3rd Qu.: 61.00   3rd Qu.:144.0   3rd Qu.: 3.9000   3rd Qu.:1.0000   3rd Qu.:0.0000  
 Max.   :640.00   Max.   :640.0   Max.   :60.0000   Max.   :1.0000   Max.   :1.0000  
>
```

Figure 4.0.1: Summary of Data After Handling Missing Values

4.1 Analyzing the Distribution of Variables by Using Histograms

At the beginning of this part, we will show the distribution of width by using a histogram to see whether it follows the Normal Distribution or not.

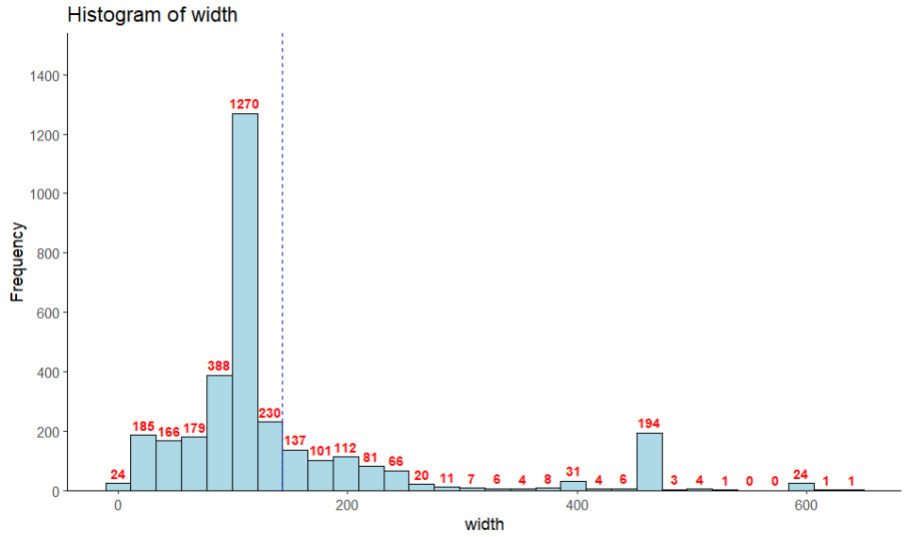

Figure 4.1.1: The original Width Histogram

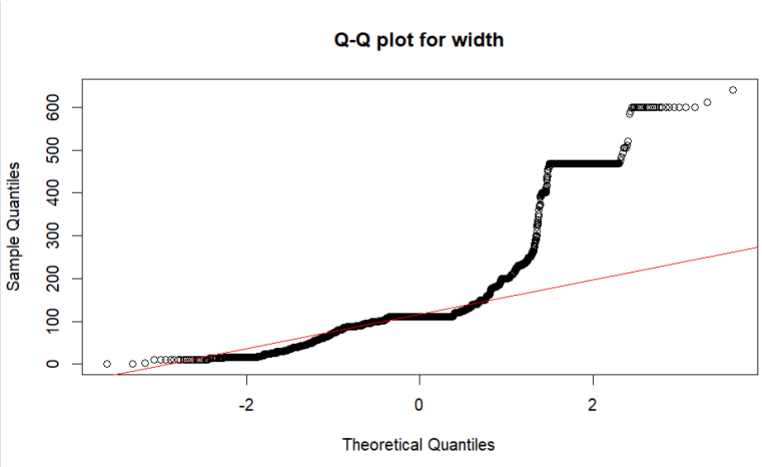

Figure 4.1.2: The original Width QQ-plot

Comments on the Q-Q plot:

- In **Figure 4.1.2**, the red straight line represents the normal distribution (if the data follow a normal distribution, the points should lie close to this line).
- **Observed data:**
  - The lower end (on the left, negative quantiles) deviates significantly downward.
  - The upper end (on the right, positive quantiles) shows a sharp upward deviation.
  - In the middle, the points form step-like patterns rather than a smooth curve. This suggests the presence of repeated values or strong outliers in the data.
- **Conclusion:** This is a typical pattern of data that is *right-skewed* and contains strong *outliers* at both tails.

Therefore, we applied outlier treatment methods by identifying extreme values beyond the quantile range and replacing them with missing values (NA). This approach aimed to mitigate the influence of abnormal values on the analysis results.

```
rm.out <- function(x, na.rm = TRUE) {
  qnt <- quantile(x, probs = c(0.25, 0.75), na.rm = na.rm)  # Q1 and Q3
  H <- 1.5 * IQR(x, na.rm = na.rm)                          # 1.5 * IQR

  y <- x  # create a copy
  # eliminate outlier < Q1 - 1.5*IQR
  y[x < (qnt[1] - H)] <- NA
  #eliminate outlier with > Q3 + 1.5*IQR
  y[x > (qnt[2] + H)] <- NA

  return(y)
}
new_data<-add_data
new_data$height = rm.out(new_data$height)
new_data$width = rm.out(new_data$width)
```

As a result (can be seen in **Figure 4.1.3**) The distribution is more centered around a typical value (around 110).

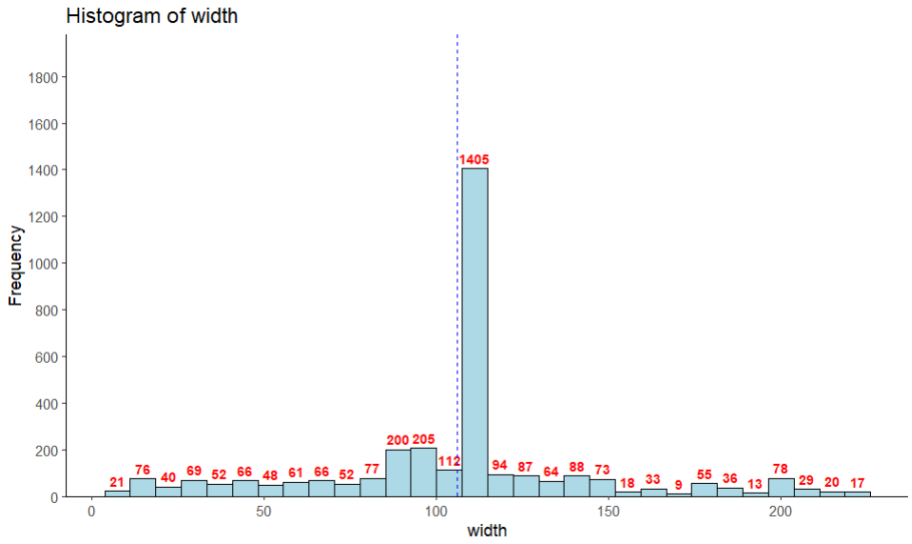

Figure 4.1.3: The Width Histogram after replacing by using median

Next, we present five histograms that display three datasets that before and after applying outlier treatment methods by identifying extreme values beyond the quantile range.

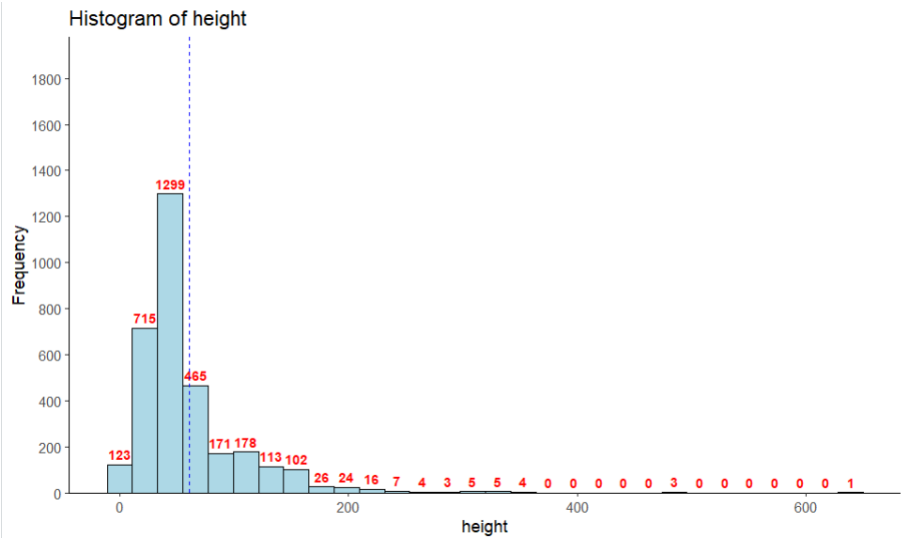

Figure 4.1.4: The original Height Histogram

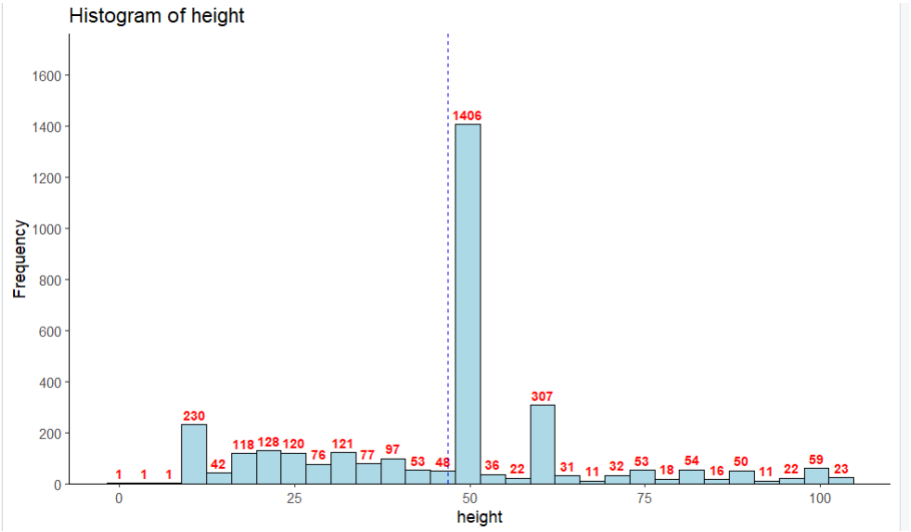

Figure 4.1.5: The Height Histogram after replacing by using median

Figures 5.1.4 and 5.1.5 illustrate the height distributions before and after applying median replacement, respectively. In Figure 5.1.4, the original histogram displays a strongly right-skewed distribution, with the majority of data points concentrated in the lower range (below 100). However, it also includes several extreme outliers extending beyond 400, which significantly affect the overall distribution shape.

After replacing abnormal values with the median, as shown in Figure 5.1.5, the distribution becomes more centralized around the median value (approximately 50), with the highest frequency recorded at this point (1406). The right-skewness is notably reduced, and the histogram appears more symmetric. Additionally, the frequency of extreme values—both low and high—has decreased substantially.

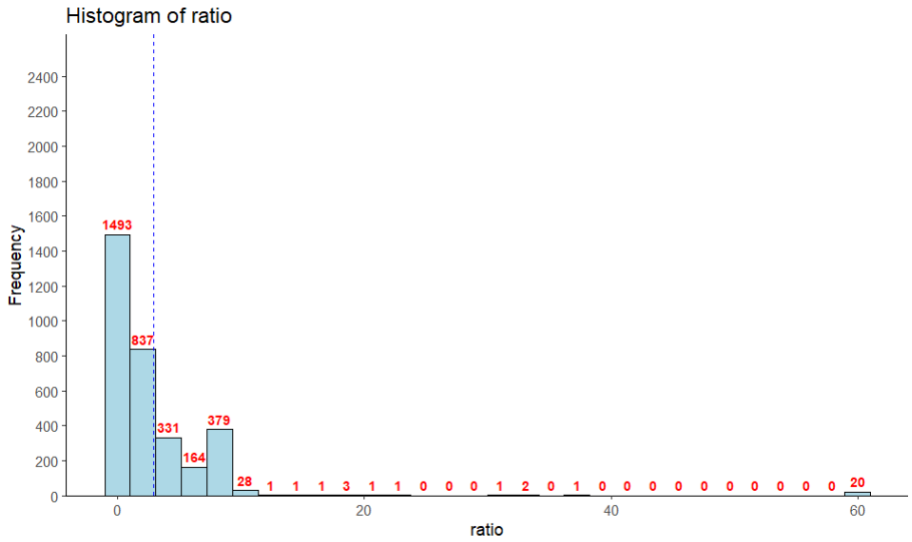

Figure 4.1.6: The original Ratio Histogram

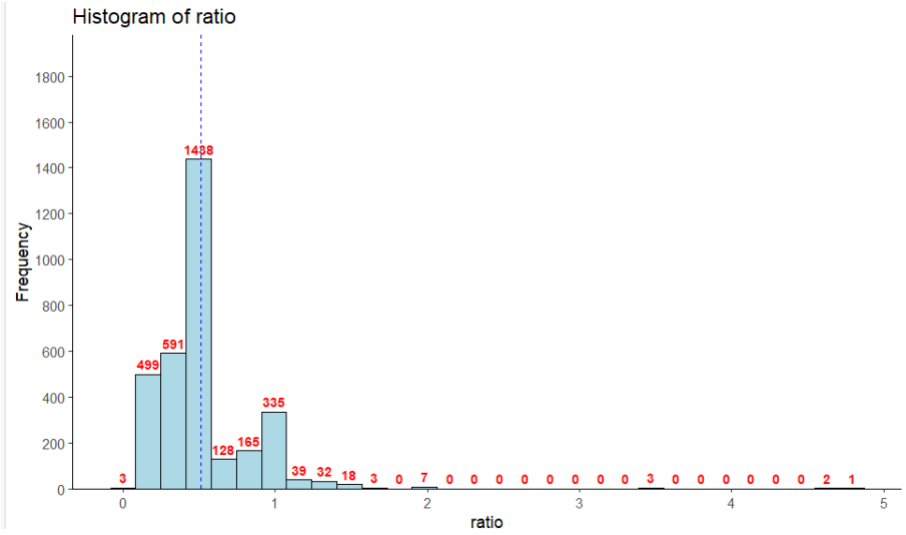

Figure 4.1.7: The Ratio Histogram after replacing by using median

Figures 5.1.6 and 5.1.7 show the ratio distributions before and after applying median replacement. In the original histogram (Figure 5.1.6), the data is heavily right-skewed, with a sharp concentration in the lower bins and several extreme outliers, including values exceeding 60. This long tail suggests the presence of anomalies that may distort statistical analysis.

After median replacement (Figure 5.1.7), the distribution becomes more compact and symmetric. The majority of values are now clustered within a narrow range below 2, and the extreme outliers have been effectively removed. This transformation results in a cleaner distribution that better represents the typical behavior of the data.

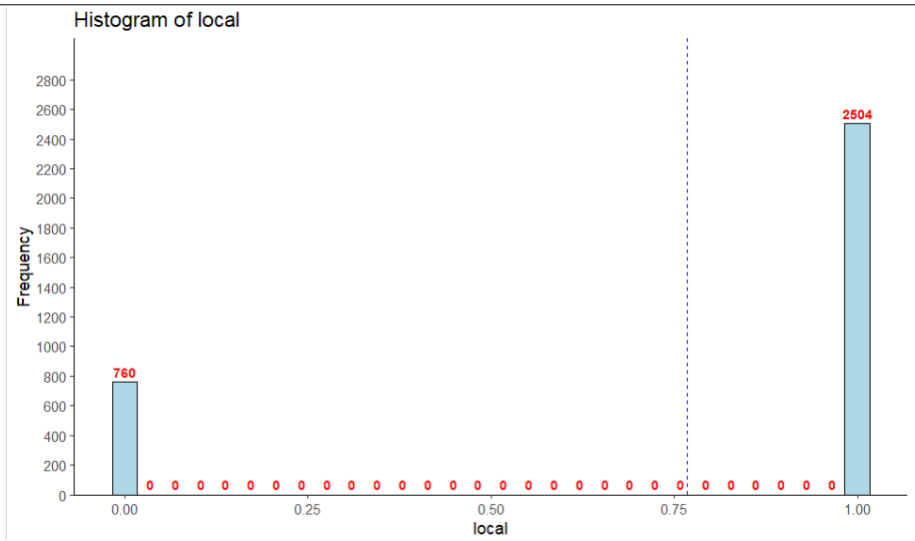

Figure 4.1.8: The Ratio Histogram after replacing by using median

And the final histogram, **Figure 4.1.8** displays the histogram of the *local* variable, which is strictly binary, taking only values 0 and 1. As such, the distribution is inherently discrete, with no intermediate or continuous values. Due to its binary nature, the variable does not contain any outliers, and no further transformation or cleaning is required. The observed counts -760 for the value 0 and 2504 for the value 1 - indicate a notable imbalance between the two categories.

However, based on these histograms (particularly histograms after replacement with the median) and the corresponding statistical values, it became clear that the majority of data points are heavily concentrated around the median. Therefore, further removal or replacement of outliers would risk disturbing the natural balance of the distribution, leading to analysis outcomes that no longer accurately reflect the true nature of the original dataset.

Therefore, we decided to retain the original values and continue using the initial histograms in subsequent analyses to ensure the findings remain consistent with the true characteristics of the dataset.

4.2 Analyzing the distribution and relationships of features by target category

The dataset is divided into two classes:

- **Non-Ad (0)**: Samples that are not advertisements.
- **Ad (1)**: Samples that are advertisements.

The boxplot illustrates key summary statistics including the median, interquartile range (IQR), and outliers.

4.2.1 Analysis boxplot of Height by Target Class

Figure 4.2.1 illustrates the distribution of the *Height* variable across two target classes: **Non-Ad (0)** and **Ad (1)**. This visualization is useful for comparing how height values differ between advertisements and non-advertisement samples.

General distribution

It can be observed that most of the data points lie between approximately 20 and 100 units of height. The overall distribution for the Non-Ad group appears wider, indicating greater variability. In contrast, the Ad group shows a more compact distribution, suggesting that advertisement content typically has smaller and more consistent heights.

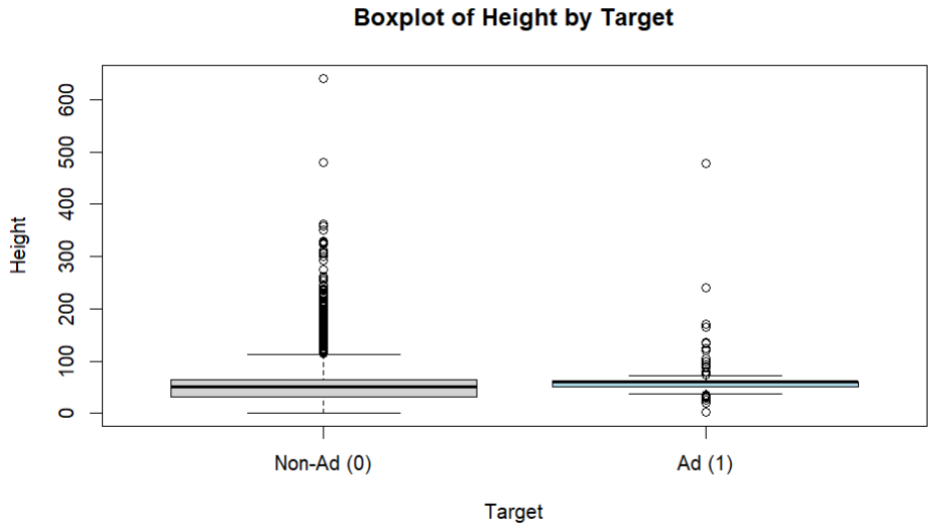

Figure 4.2.1: Boxplot of Height by Target Class

Median and Quartiles

- **Median:** The horizontal line inside each box represents the median height. For the Non-Ad class, the median is around 60 units, while for the Ad class it is approximately 40 units. This implies that advertisement samples generally have lower heights.
- **Quartiles:** The bottom and top edges of the box represent the first quartile (Q1) and the third quartile (Q3), respectively. This indicates that 50% of the height values fall within this interquartile range (IQR). The Non-Ad group has a wider IQR compared to the Ad group.

Outliers

The whiskers extending from the boxes show the range of the data excluding outliers. Any data points beyond the whiskers are considered outliers and are displayed as individual dots. Both groups contain outliers, but the Non-Ad group has more extreme values, with some height measurements exceeding 600 units.

4.2.2 Analysis boxplot of Width by Target Class

Figure 4.2.2 shows the distribution of the *Width* variable for two target classes: **Non-Ad (0)** and **Ad (1)**. This plot helps us understand how the width values differ between advertisement and non-advertisement content.

General distribution

The width of Non-Ad samples is generally lower and more tightly clustered compared to Ad samples. The Ad group has a much wider spread, indicating a broader range of width values. While Non-Ad widths are concentrated between approximately 50 and 200 units, Ad widths range from near 0 to over 600 units.

Median and Quartiles

- **Median:** The thick horizontal line inside each box represents the median width. The median width of the Ad group is noticeably higher than that of the Non-Ad group, indicating that advertisements tend to be wider on average.
- **Quartiles:** The box spans from the first quartile (Q1) to the third quartile (Q3), covering the middle 50% of the data. The interquartile range (IQR) of the Ad group is considerably larger than that of the Non-Ad group, reflecting a much higher variability in advertisement widths.

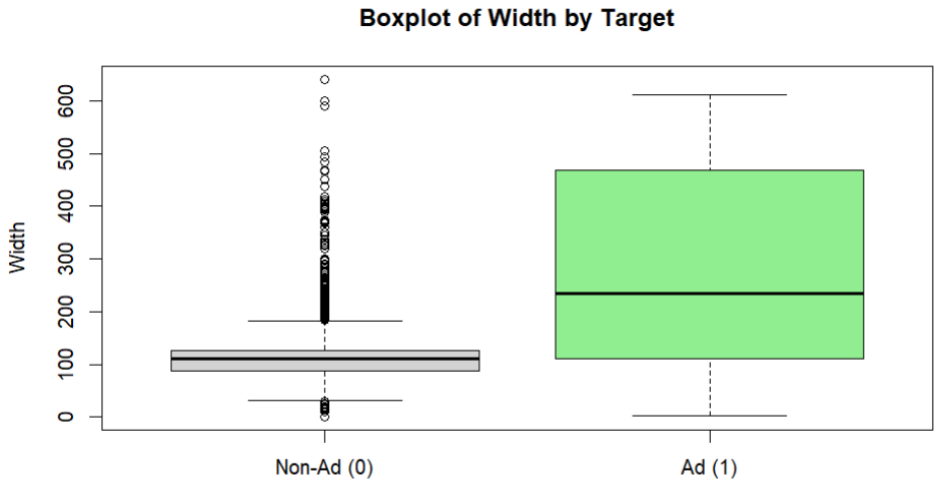

Figure 4.2.2: Boxplot of Width by Target Class

Outliers

In the Non-Ad group, many outliers appear above the upper whisker, with widths exceeding 500 units. These outliers are likely unusual or extreme non-ad content. In contrast, the Ad group has no visible outliers, implying that its width values are more uniformly distributed within the whisker range.

4.2.3 Analysis boxplot of Ratio by Target Class

Figure 4.2.3 illustrates the distribution of the *Ratio* variable for two target classes: **Non-Ad (0)** and **Ad (1)**. This ratio may represent a shape-related measure such as width-to-height or area-to-size proportion, which can help differentiate the structural patterns between advertisements and non-advertisements.

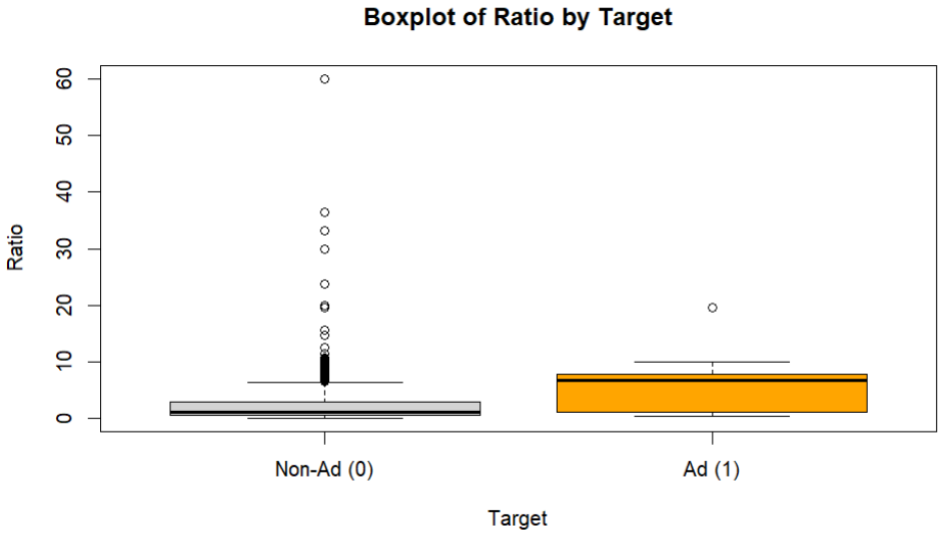

Figure 4.2.3: Boxplot of Ratio by Target Class

General distribution

Overall, the Ad group shows a higher central tendency and greater variability compared to the Non-Ad group. The Non-Ad data are more concentrated around lower values (close to 1), while the Ad data spread across a broader range and display a higher median value. **Median and Quartiles**

- **Median:** The thick horizontal line inside each box represents the median ratio. The Ad group has a noticeably higher median, suggesting that advertisements tend to have higher shape ratios.
- **Quartiles:** The box represents the interquartile range (IQR), from the first quartile (Q1) to the third quartile (Q3), containing the middle 50% of data. The Ad group's IQR is wider, reflecting greater variation in aspect ratios across different ads.

Outliers

The Non-Ad group shows numerous high outliers, with several extreme values exceeding 60. These indicate a few non-ad samples with unusually high ratios, possibly due to formatting artifacts or exceptional values. In contrast, the Ad group has relatively fewer outliers, suggesting a more uniform structure among advertisements.

4.2.4 Correlation Diagram

The correlation matrix was constructed using the continuous numerical variables *height*, *width*, and *ratio*. These variables are suitable for Pearson correlation analysis, which measures the strength and direction of linear relationships between pairs of continuous variables.

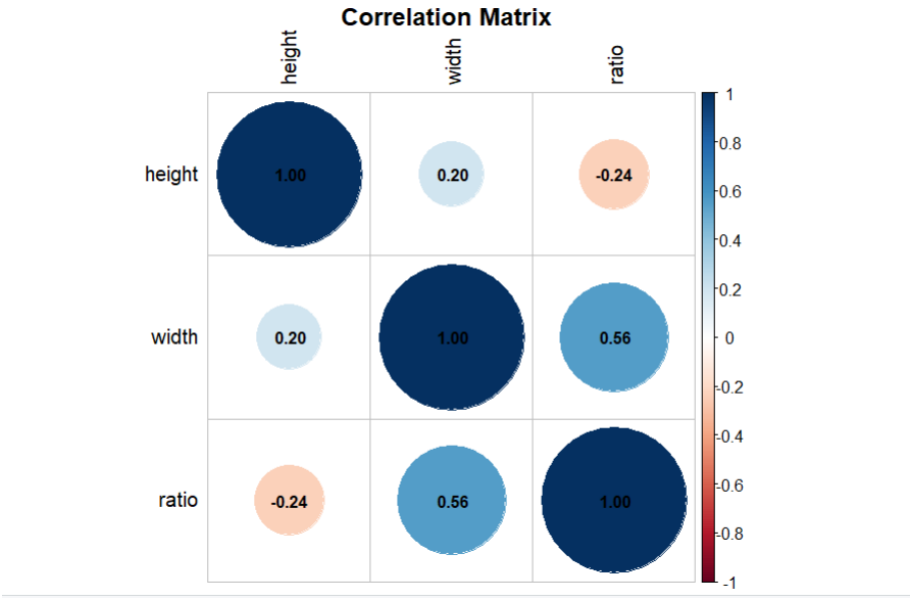

Figure 4.2.4: Correlation matrix visualized using circular markers and color gradients.

The circular markers in the diagram above represent the Pearson correlation coefficients, with their size and color intensity reflecting the magnitude and direction of each relationship:

- **Positive correlation (blue):** As one variable increases, the other tends to increase. A value close to +1 indicates a strong positive relationship.
- **Negative correlation (red):** As one variable increases, the other tends to decrease. A value close to -1 indicates a strong negative relationship.
- **Near-zero correlation:** Indicates little to no linear relationship between variables.

Interpretation of Correlations:

- **Width vs. Ratio:** A moderate positive correlation of **0.56**, suggesting that wider values are generally associated with higher ratios.

- **Height vs. Width:** A weak positive correlation of **0.20**, indicating only a slight upward trend—height and width do not strongly vary together.
- **Height vs. Ratio:** A weak negative correlation of **-0.24**, meaning that as height increases, the ratio slightly decreases, although the relationship is not strong.

Note: The variable *local* was excluded from the correlation matrix because it is binary in nature (e.g., 0 or 1), and not a continuous numerical variable. Including such categorical data in a Pearson correlation matrix may lead to misleading or statistically invalid interpretations, as the Pearson method assumes interval or ratio-scale data with a normal distribution.

5 Logistic Regression Analysis

5.1 Objective and Methodology

This section presents the results of applying logistic regression to classify web images as advertisements or non-advertisements. The classification is based on four numerical features: `height`, `width`, `ratio`, and `local`.

The dataset was randomly split into a training set (70%) and a test set (30%) to evaluate model generalization. Each model was trained using the `glm()` function with the `binomial` family, and evaluated with performance metrics including accuracy, AIC, and AUC. ROC curves were plotted for visual comparison of classification ability.

5.2 Univariate Logistic Regression Models

Four univariate models were trained using each predictor separately:

- Model 1: `target` ~ `width`
- Model 2: `target` ~ `ratio`
- Model 3: `target` ~ `height`
- Model 4: `target` ~ `local`

Model 1: Logistic Regression with Width

```
Call:
glm(formula = target ~ width, family = binomial, data = train_data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.6754192  0.1272510  -28.88   <2e-16 ***
width        0.0100708  0.0005304   18.99   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1795.5  on 2283  degrees of freedom
Residual deviance: 1336.7  on 2282  degrees of freedom
AIC: 1340.7

Number of Fisher Scoring iterations: 5
```

Figure 5.2.1: Model summary output for logistic regression using `width`

The estimated coefficient for `width` was 0.01007 with a standard error of 0.0005 and a z-value of 18.99. The associated p-value was less than 2e-16, indicating that this variable is highly statistically significant. The positive sign of the coefficient suggests that an increase in image width is associated with a higher likelihood of the image being classified as an advertisement.

The **null deviance** of 1795.5 represents the deviance of a model with only the intercept, serving as a baseline. The **residual deviance** of 1336.7 indicates the deviance of the fitted model, which includes the `width` predictor. The notable reduction in deviance (from 1795.5 to 1336.7) demonstrates that the model with `width` fits the data significantly better than the intercept-only model.

The **Akaike Information Criterion (AIC)** for the model was 1340.7. A lower AIC value reflects a better balance between goodness of fit and model complexity. Compared to other univariate models, this AIC value confirms that the `width` model has strong explanatory power with a relatively simple structure.

In summary, the logistic regression model using `width` as a single predictor is statistically significant and shows strong model performance based on deviance reduction and AIC.

Figure 5.2.2: Logistic regression curve: `width` vs `target`

Model 2: Logistic Regression with Ratio

```
Call:
glm(formula = target ~ ratio, family = binomial, data = train_data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.016147   0.069259  -29.11  < 2e-16 ***
ratio        0.042470   0.008361    5.08 3.78e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1795.5  on 2283  degrees of freedom
Residual deviance: 1770.3  on 2282  degrees of freedom
AIC: 1774.3

Number of Fisher Scoring iterations: 4
```

Figure 5.2.3: Model summary output for logistic regression using `ratio`

The estimated coefficient for `ratio` is 0.0425 with a standard error of 0.0084 and a z-value of 5.08. The p-value is 3.78×10^{-7}, indicating the relationship is highly statistically significant. The positive coefficient suggests that images with higher aspect ratios are more likely to be advertisements.

The **null deviance** is 1795.5, and the **residual deviance** is 1770.3, showing a small but notable improvement in model fit. The **AIC** value of 1774.3 is lower than many other univariate models, suggesting that `ratio` has moderate predictive value when used alone.

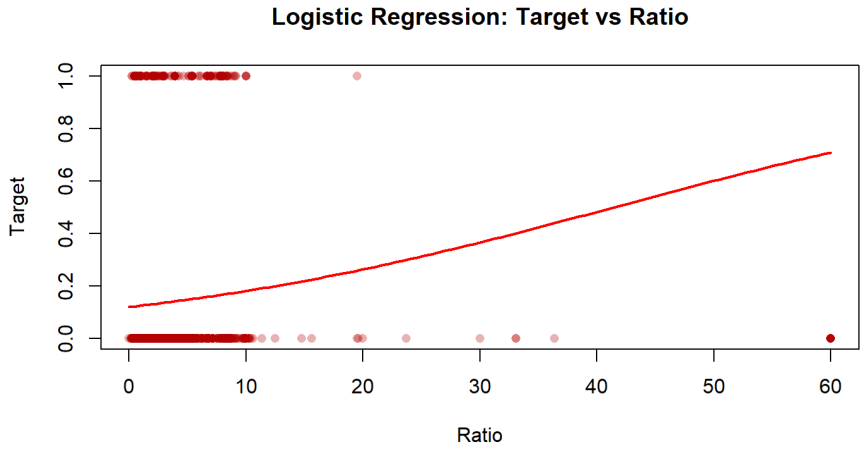

Figure 5.2.4: Logistic regression curve: `ratio` vs `target`

Model 3: Logistic Regression with Height

```
Call:
glm(formula = target ~ height, family = binomial, data = train_data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.023881   0.096184 -21.042   <2e-16 ***
height       0.002479   0.001152   2.153   0.0314 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1795.5  on 2283  degrees of freedom
Residual deviance: 1791.2  on 2282  degrees of freedom
AIC: 1795.2

Number of Fisher Scoring iterations: 4
```

Figure 5.2.5: Model summary output for logistic regression using `height`

The estimated coefficient for `height` is 0.0025 with a standard error of 0.0012, and the corresponding p-value is 0.0314. This result is statistically significant at the 5% level, but the effect is weak due to the small magnitude of the coefficient.

The **null deviance** is 1795.5 and the **residual deviance** is 1791.2, suggesting only a slight improvement. The **AIC** value of 1795.2 is relatively high, confirming that `height` alone does not contribute much to model performance.

Figure 5.2.6: Logistic regression curve for `height` vs `target`

Model 4: Logistic Regression with Local

```
Call:
glm(formula = target ~ local, family = binomial, data = train_data)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.88792    0.12734 -14.825   <2e-16 ***
local        0.02338    0.14544   0.161    0.872
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1795.5  on 2283  degrees of freedom
Residual deviance: 1795.5  on 2282  degrees of freedom
AIC: 1799.5

Number of Fisher Scoring iterations: 4
```

Figure 5.2.7: Model summary output for logistic regression using `local`

The coefficient for `local` is 0.0234 with a standard error of 0.1454 and a p-value of 0.872, which is not statistically significant. This implies that `local` has no meaningful relationship with the target outcome.

There is no reduction in deviance (both **null deviance** and **residual deviance** are 1795.5), and the **AIC** remains high at 1799.5. These results indicate that `local` is not a useful predictor when used alone.

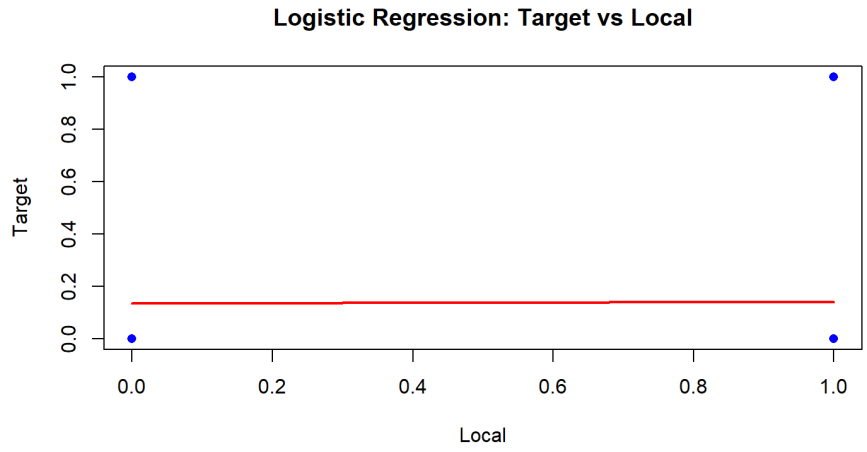

Figure 5.2.8: Logistic regression curve: `local` vs `target`

5.3 Summary Evaluation of Univariate Models

To compare the four univariate logistic regression models systematically, we evaluated each one on three fronts: classification performance, discriminative power (AUC), and model fit (AIC). The following figures and tables summarize these findings.

```
> prob_height <- evaluate_model(model_height, test_data, "Model 1: height")
Model 1: height:
Accuracy:  0.848
Confusion Matrix:
          Actual
Predicted   0   1
        0 831 149

> prob_width  <- evaluate_model(model_width,  test_data, "Model 2: width")
Model 2: width:
Accuracy:  0.9
Confusion Matrix:
          Actual
Predicted   0   1
        0 809  76
        1  22  73

> prob_ratio  <- evaluate_model(model_ratio,  test_data, "Model 3: ratio")
Model 3: ratio:
Accuracy:  0.8429
Confusion Matrix:
          Actual
Predicted   0   1
        0 826 149
        1   5   0

> prob_local  <- evaluate_model(model_local,  test_data, "Model 4: local")
Model 4: local:
Accuracy:  0.848
Confusion Matrix:
          Actual
Predicted   0   1
        0 831 149
```

Figure 5.3.1: Confusion matrices and accuracy for univariate models

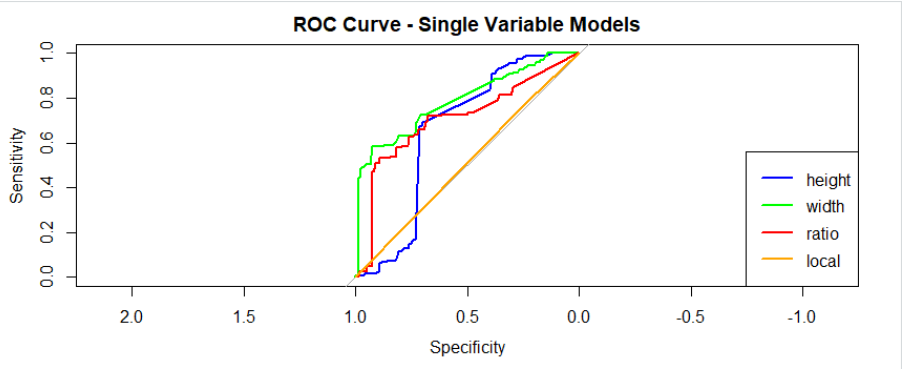

Figure 5.3.2: ROC Curve Comparison for Univariate Models

```
> cat("AUC - Model 1 (height):", auc(roc_height), "\n")
AUC - Model 1 (height): 0.6544714
> cat("AUC - Model 2 (width):",  auc(roc_width),  "\n")
AUC - Model 2 (width): 0.7906703
> cat("AUC - Model 3 (ratio):",  auc(roc_ratio),  "\n")
AUC - Model 3 (ratio): 0.7060144
> cat("AUC - Model 4 (local):",  auc(roc_local),  "\n")
AUC - Model 4 (local): 0.5096916
```

Figure 5.3.3: AUC Scores for Univariate Models

```
> # Tính và in AIC
> cat("AIC - Model 1 (height):", AIC(model_height), "\n")
AIC - Model 1 (height): 1795.185
> cat("AIC - Model 2 (width):",  AIC(model_width),  "\n")
AIC - Model 2 (width): 1340.653
> cat("AIC - Model 3 (ratio):",  AIC(model_ratio),  "\n")
AIC - Model 3 (ratio): 1774.259
> cat("AIC - Model 4 (local):",  AIC(model_local),  "\n")
AIC - Model 4 (local): 1799.457
```

Figure 5.3.4: AIC Values for Univariate Models

5.4 Multivariate Logistic Regression Models

Model Selection Strategy

The full multivariate logistic regression **Model 5** incorporates all four available predictors: `height`, `width`, `ratio`, and `local`. This model serves as a comprehensive baseline to examine the joint predictive capacity of all features.

However, it is important to recognize that the variable `ratio` is mathematically defined as `height/width`. Consequently, using `height`, `width`, and `ratio` together in the same model introduces multicollinearity—a condition where predictors are linearly dependent. Multicollinearity can inflate the standard errors of the estimated coefficients and reduce the model's stability and interpretability.

To address this issue and to isolate the predictive effects of correlated features, we constructed three alternative models for comparison:

- **Model 6**: Includes `height`, `width`, and `local`, excluding `ratio`.
- **Model 7**: Includes `ratio` and `local`, excluding `height` and `width`.
- **Model 8**: Includes `height`, `width`, and `ratio`, excluding `local`.

These reduced models are not inherently better or worse than the full model. Instead, they offer alternative perspectives by simplifying the model structure and removing potentially redundant or less

informative variables. This design strategy enables a clearer analysis of each feature's individual and combined contribution to classification performance, while mitigating the risks of over

Model 5: Logistic Regression with All Predictors (Full Model)

This model includes all available predictors: `height`, `width`, `ratio`, and `local`. It is used as the baseline multivariate model for performance comparison.

```
Call:
glm(formula = target ~ height + width + ratio + local, family = binomial,
    data = train_data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.617013   0.240116 -15.064  < 2e-16 ***
height      -0.009127   0.002202  -4.144 3.41e-05 ***
width        0.015836   0.001010  15.684  < 2e-16 ***
ratio       -0.182047   0.035144  -5.180 2.22e-07 ***
local        0.179869   0.185428   0.970    0.332
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1795.5  on 2283  degrees of freedom
Residual deviance: 1223.7  on 2279  degrees of freedom
AIC: 1233.7

Number of Fisher Scoring iterations: 6
```

Figure 5.4.1: Model summary output for multivariate logistic regression

The **null deviance** was 1795.5 and the **residual deviance** dropped to 1223.7, indicating a substantial improvement in model fit compared to the null model. The **AIC** value of 1233.7 is among the lowest across all models, suggesting that the full model provides a good balance between fit and complexity.

However, this model includes both `height`, `width`, and their ratio `ratio`, which introduces multicollinearity due to their mathematical relationship. This can affect the reliability of individual coefficient estimates.

While the full model performs well overall, it was primarily constructed as a benchmark for comparison. Further analysis using reduced models is necessary to disentangle the contributions of correlated variables and avoid redundancy.

Model 6: Logistic Regression without Ratio

This reduced multivariate model includes the predictors `height`, `width`, and `local`, explicitly excluding `ratio` to mitigate multicollinearity issues. The aim is to assess whether using the raw dimensions `height` and `width` independently—without their ratio—can retain strong predictive performance.

```
> model_hwloc <- glm(target ~ height + width + local, data = train_data, family = binomial)
> summary(model_hwloc)

Call:
glm(formula = target ~ height + width + local, family = binomial,
    data = train_data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.7144443  0.2155818 -17.230   <2e-16 ***
height      -0.0003732  0.0015084  -0.247    0.805
width        0.0100900  0.0005336  18.908   <2e-16 ***
local        0.0803315  0.1732294   0.464    0.643
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1795.5  on 2283  degrees of freedom
Residual deviance: 1336.3  on 2280  degrees of freedom
AIC: 1344.3

Number of Fisher Scoring iterations: 5
```

Figure 5.4.2: Model summary output for logistic regression with `height`, `width`, and `local`

The **null deviance** was 1795.5 and the **residual deviance** decreased to 1336.3, indicating substantial improvement over the null model. The **AIC** value of 1344.3 is slightly higher than that of the full model (Model 5), but still relatively low, showing that this simpler model maintains competitive performance.

Notably, both `height` and `width` remain statistically significant predictors, while `local` continues to contribute marginally, though not significantly. By excluding the derived variable `ratio`, this model avoids redundancy and improves the interpretability of coefficients, especially in practical contexts where understanding the effect of raw dimensions is preferred.

Model 6 offers a viable alternative to the full model, striking a balance between simplicity and predictive accuracy while mitigating the effects of multicollinearity.

Model 7: Logistic Regression with Ratio and Local

This compact model focuses on structural and locality features only. It aims to evaluate the predictive value of aspect ratio (`ratio`) and local hosting (`local`) in the absence of the individual dimensions `height` and `width`.

```
> model_rloc <- glm(target ~ ratio + local, data = train_data, family = binomial)
> summary(model_rloc)

Call:
glm(formula = target ~ ratio + local, family = binomial, data = train_data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.984950   0.129050 -15.381  < 2e-16 ***
ratio        0.042699   0.008403   5.082 3.74e-07 ***
local       -0.041834   0.146608  -0.285    0.775
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1795.5  on 2283  degrees of freedom
Residual deviance: 1770.2  on 2281  degrees of freedom
AIC: 1776.2

Number of Fisher Scoring iterations: 4
```

Figure 5.4.3: Model summary output for logistic regression with `ratio` and `local`

The **null deviance** remained at 1795.5, while the **residual deviance** dropped to 1770.2, yielding a modest improvement in fit. The **AIC** value for this model was 1776.2—higher than both the full model and Model 6—indicating a weaker overall fit.

The coefficient for `ratio` was statistically significant, confirming its value as a structural predictor. However, `local` remained non-significant and had a limited effect. This model confirms that `ratio` alone carries useful information, but also illustrates that discarding `height` and `width` in favor of it alone may lead to a loss in model performance.

Overall, Model 7 offers a simplified alternative but with trade-offs in predictive accuracy and information loss due to omitted variables.

Model 8: Logistic Regression with Height, Width, and Ratio

This variation retains all structural variables—`height`, `width`, and `ratio`—but excludes `local`. The objective is to assess the collective effect of visual geometry alone, without the influence of URL-related factors.

```
> model_hwroc <- glm(target ~ height + width + ratio, data = train_data, family = binomial)
> summary(model_hwroc)

Call:
glm(formula = target ~ height + width + ratio, family = binomial,
    data = train_data)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.458168   0.172996 -19.990  < 2e-16 ***
height      -0.009434   0.002186  -4.315 1.59e-05 ***
width        0.015823   0.001013  15.620  < 2e-16 ***
ratio       -0.182434   0.035504  -5.138 2.77e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1795.5  on 2283  degrees of freedom
Residual deviance: 1224.7  on 2280  degrees of freedom
AIC: 1232.7

Number of Fisher Scoring iterations: 6
```

Figure 5.4.4: Model summary output for logistic regression with `height`, `width`, and `ratio`

The **residual deviance** dropped significantly to 1224.7 from the null deviance of 1795.5, with an **AIC** of 1232.7—very close to the full model (Model 5), suggesting strong predictive value even without `local`.

However, this model includes all three geometrically dependent variables, raising concerns of multicollinearity. While the model fit remains strong, the interpretability of individual coefficients may be compromised due to redundancy among predictors.

Model 8 provides an important contrast to the full model: it confirms that visual dimensions alone can achieve comparable classification performance, though at the potential cost of coefficient reliability.

5.5 Summary Evaluation of Multivariate Models

To compare the four multivariate logistic regression models systematically, we evaluated each one on three fronts: classification performance, discriminative power (AUC), and model fit (AIC). The following figures and table summarize these findings.

```
> # Đánh giá từng mô hình đa biến
> prob_full <- evaluate_model(model_full, test_data, "Model 5: Full (height + width + ratio + local)")
Model 5: Full (height + width + ratio + local):
Accuracy:  0.9051
Confusion Matrix:
          Actual
Predicted   0   1
        0 814  76
        1  17  73

> prob_hwloc <- evaluate_model(model_hwloc, test_data, "Model 6: height + width + local")
Model 6: height + width + local:
Accuracy:  0.901
Confusion Matrix:
          Actual
Predicted   0   1
        0 810  76
        1  21  73

> prob_rloc <- evaluate_model(model_rloc, test_data, "Model 7: ratio + local")
Model 7: ratio + local:
Accuracy:  0.8429
Confusion Matrix:
          Actual
Predicted   0   1
        0 826 149
        1   5   0

> prob_hwroc <- evaluate_model(model_hwroc, test_data, "Model 8: height + width + ratio")
Model 8: height + width + ratio:
Accuracy:  0.9051
Confusion Matrix:
          Actual
Predicted   0   1
        0 814  76
        1  17  73
```

Figure 5.5.1: Confusion matrices and accuracy for multivariate models

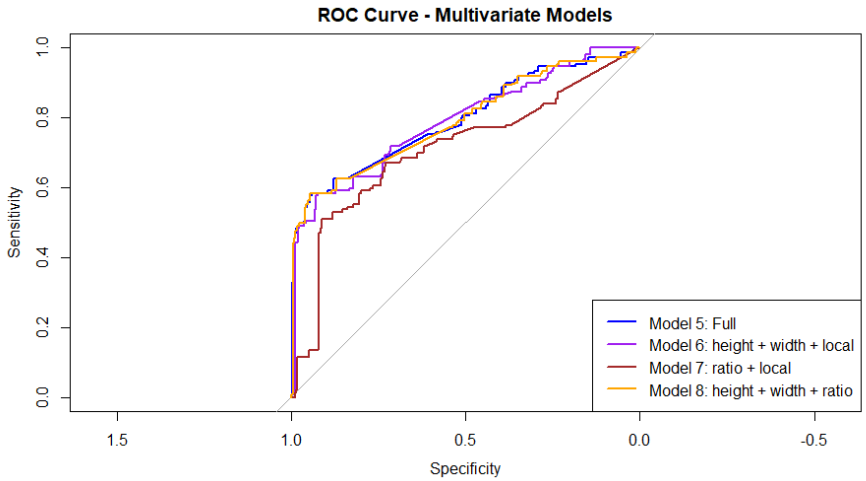

Figure 5.5.2: ROC Curve Comparison for Multivariate Models

```
> # AUC
> cat("AUC - Model 5 (Full):", auc(roc_full), "\n")
AUC - Model 5 (Full): 0.7948013
> cat("AUC - Model 6 (height + width + local):", auc(roc_hwloc), "\n")
AUC - Model 6 (height + width + local): 0.7898424
> cat("AUC - Model 7 (ratio + local):", auc(roc_rloc), "\n")
AUC - Model 7 (ratio + local): 0.7087442
> cat("AUC - Model 8 (height + width + ratio):", auc(roc_hwroc), "\n")
AUC - Model 8 (height + width + ratio): 0.7921684
```

Figure 5.5.3: AUC Scores for Multivariate Models

```
> # Tính và in AIC
> cat("AIC - Model 5 (Full):", AIC(model_full), "\n")
AIC - Model 5 (Full): 1233.731
> cat("AIC - Model 6 (height + width + local):", AIC(model_hwloc), "\n")
AIC - Model 6 (height + width + local): 1344.325
> cat("AIC - Model 7 (ratio + local):", AIC(model_rloc), "\n")
AIC - Model 7 (ratio + local): 1776.178
> cat("AIC - Model 8 (height + width + ratio):", AIC(model_hwroc), "\n")
AIC - Model 8 (height + width + ratio): 1232.69
```

Figure 5.5.4: AIC Values for Multivariate Models

5.6 Overall Evaluation and Comparative Analysis of All Models

Table 1: Summary of All Logistic Regression Models

Model	Type	Accuracy	AUC	AIC
Model 1: height	Univariate	0.8480	0.6545	1795.19
Model 2: width	Univariate	0.9000	0.7907	1340.65
Model 3: ratio	Univariate	0.8429	0.7060	1774.26
Model 4: local	Univariate	0.8480	0.5097	1799.46
Model 5: full (h + w + r + l)	Multivariate	**0.9051**	**0.7948**	1233.73
Model 6: h + w + local	Multivariate	0.9010	0.7898	1344.33
Model 7: ratio + local	Multivariate	0.8429	0.7087	1776.18
Model 8: h + w + ratio	Multivariate	**0.9051**	0.7922	**1232.69**

Metric Interpretation:

- **AUC** measures the model's ability to distinguish between classes (higher AUC → better classification performance).
- **AIC** measures statistical fit, balancing goodness-of-fit with model simplicity (lower AIC → better fit with fewer parameters).

These two metrics do not always align: a model with higher AUC may have higher AIC due to increased complexity or potential overfitting. Hence, model comparison must consider both interpretability and statistical trade-offs.

Univariate Model Evaluation:
Among the single-variable models:

- **Model 2 (width)** achieved the best overall performance: highest AUC (0.7907), second-best AIC (1340.65), and 90% accuracy. It proves that width alone is a highly informative feature.
- **Model 4 (local)** performed the worst: AUC ≈ 0.51, suggesting near-random classification, and the highest AIC (1799.46). It contributes little predictive value on its own.
- **Model 1 (height)** and **Model 3 (ratio)** had moderate accuracy (84.3–84.8%), but high AICs (>1770), indicating poor fit relative to complexity.

Multivariate Model Evaluation:
Within the multivariate group:

- **Model 5** (Full) and **Model 8** (without **local**) performed similarly in terms of accuracy, but **Model 5** achieved a higher AUC, indicating better discriminatory ability. On the other hand, **Model 8** obtained a lower AIC (1232.69 vs 1233.73), reflecting its simpler structure and better balance between model fit and complexity. Both AUC and AIC are important criteria, and the choice between the two models depends on the specific goal: if better classification performance is desired, **Model 5** is preferable; if model simplicity is prioritized, **Model 8** may be more appropriate.
- **Model 6 (without ratio)** also performed well, with accuracy = 0.901 and AUC = 0.7898, but had a notably higher AIC (1344.33), indicating a weaker fit.
- **Model 7 (ratio + local)** underperformed significantly: lowest accuracy (84.3%), high AIC (1776.18), and lowest AUC among multivariate models (0.7087). Excluding `height` and `width` harms predictive power.

Final Comparison: Univariate vs. Multivariate:

- All multivariate models (except Model 7) outperformed univariate models in both AUC and AIC, showing the advantage of combining features.
- **Model 8** is arguably the best trade-off: highest classification accuracy (0.9051), excellent AUC (0.7922), and the lowest AIC (1232.69). It achieves full performance without using the `local` variable.
- **Model 2 (width)** is the best univariate baseline, showing that even a simple model can perform competitively when strong features are used.

These results confirm that combining structural image features improves classification, and careful feature selection (e.g., omitting `local`) can reduce model complexity without compromising accuracy.

6 Discussion and Expansion

This study demonstrates the effectiveness of logistic regression in classifying web images as ads or non-ads based on visual features. Among single predictors, **width** performed best, with the highest accuracy and lowest AIC, highlighting the importance of visual dimensions.

The multivariate model (Model 5), combining *height*, *width*, *ratio*, and *local*, outperformed all others with the highest AUC and accuracy and the lowest AIC. This shows that combining features captures complex patterns better than individual variables.

These results are valuable for applications like ad detection, content filtering, and digital media analysis, offering insights for tools relying on visual data.

6.1 Limitations

While the logistic regression framework proved effective, several limitations remain:

- **Feature Simplicity:** The study only used four manually selected visual features, excluding hundreds of available URL-term-based variables that may offer additional predictive power.
- **Linear Model Constraints:** Logistic regression assumes a linear relationship between features and the log-odds of the outcome, potentially missing complex interactions or nonlinear patterns.
- **Data Scope:** The dataset used is limited to static image metadata and does not incorporate real-time factors such as user engagement or loading context.
- **Class Balance and Label Noise:** The binary labels may not fully capture nuanced distinctions in ad types or mislabeling from automated data collection.

6.2 Future Directions

To address these limitations and build upon this work, future research could explore the following:

1. **Adopting Non-Linear Models:** Utilizing tree-based algorithms or neural networks to model non-linear interactions and complex dependencies among features.
2. **Expanding Feature Space:** Including additional metadata, text-based URL terms, or image content embeddings using computer vision could improve classification performance.
3. **Dataset Enrichment:** Applying the model on more recent and diverse datasets with varying ad formats and layouts to enhance generalizability.
4. **Scenario-Based Evaluation:** Testing the model in real-world use cases, such as browser extensions or mobile applications, could provide insights into its practical utility.

Ultimately, this research demonstrates how statistical modeling can provide valuable insights in classifying online media and paves the way for more advanced machine learning applications in content filtering and web automation.

7 Data and Code Sources

- **Dataset:** The study uses the "Internet Advertisements Data Set" from Kaggle. It can be accessed at:
  `https://www.kaggle.com/datasets/uciml/internet-advertisements-data-set/data`
- **Code:** All analyses were performed in R, including data preprocessing, logistic regression modeling, evaluation, and visualization. The full R code is available at:
  `https://drive.google.com/drive/folders/15igMnu7hROAjlHoMqs4OObbQClRuS-V0?usp=drive_link`

8 References

1. UCI Machine Learning Repository: Internet Advertisements Data Set. Available at: `https://www.kaggle.com/datasets/uciml/internet-advertisements-data-set/data`
2. Dilaratop. Data Processing Notebook. Available at: `https://www.kaggle.com/code/dilaratop/data-processing`
3. Luận Văn Việt. Mô hình hồi quy logistic là gì?
   `https://luanvanviet.com/mo-hinh-hoi-quy-logistic/`
4. Trí Tuệ Nhân Tạo. Bài 6 – Hồi quy Logistic (Logistic Regression).
   `https://trituenhantao.io/machine-learning-co-ban/bai-6-logistic-regression-hoi-quy-logistic/`
5. UCLA Institute for Digital Research and Education. Logit Regression | R Data Analysis Examples. Available at: `https://stats.oarc.ucla.edu/r/dae/logit-regression/`
6. DataCamp. Logistic Regression in R Tutorial. Available at: `https://www.datacamp.com/tutorial/logistic-regression-R`
7. GeeksforGeeks. Logistic Regression in R Programming. Available at: `https://www.geeksforgeeks.org/logistic-regression-in-r-programming/`