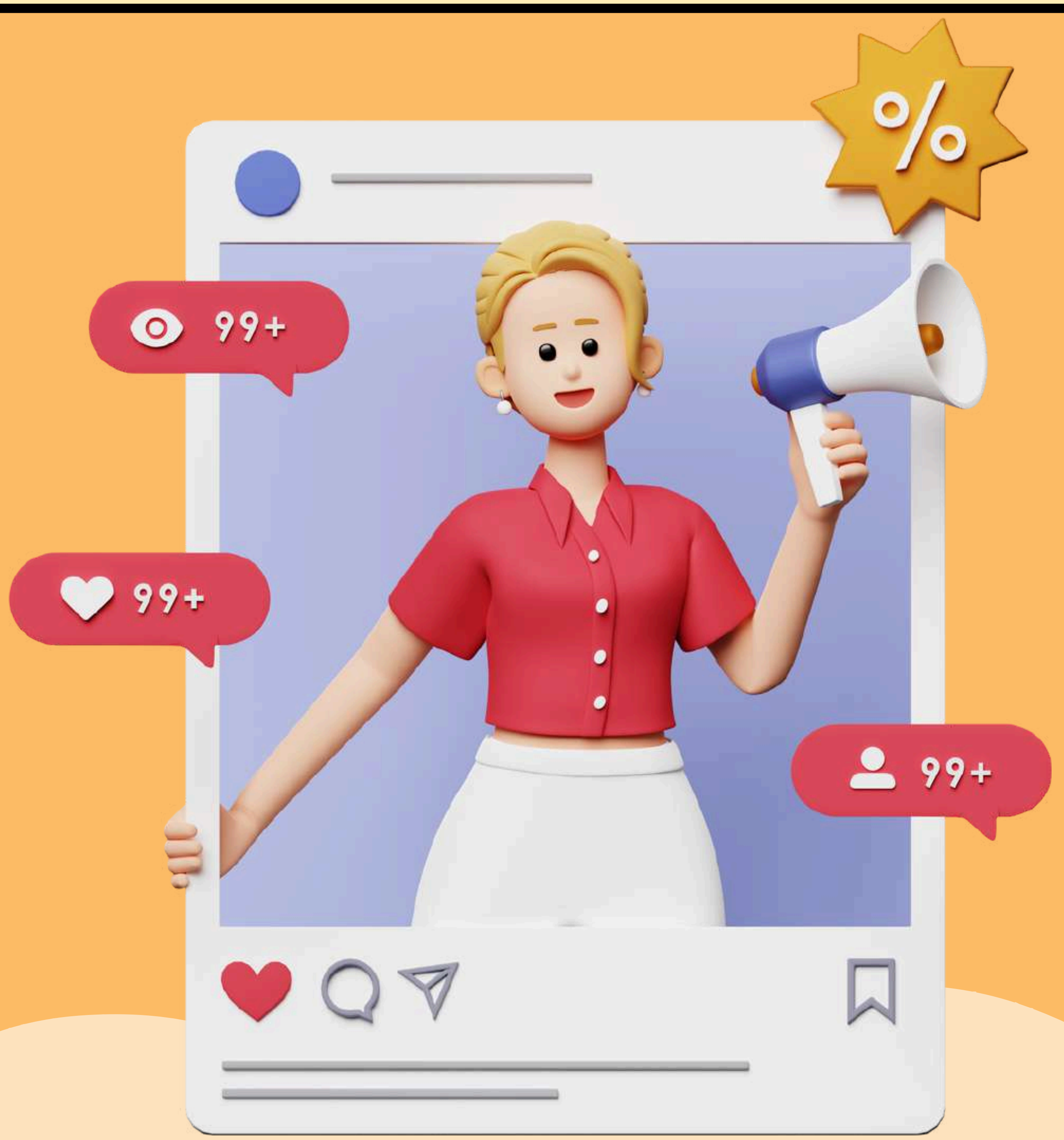
By: Aldi Vibriani
Final project Github: Link Github

Predict Customer Clicked Ads Classification by Using Machine Learning



Background

A a company seeks to evaluate the effectiveness of its advertisements In Indonesia, . Understanding the reach and impact of these ads is crucial for attracting customers and optimizing marketing strategies. By analyzing historical advertisement data to uncover insights and patterns, the company aims to enhance its targeting efforts. This case focuses on developing a machine learning classification model to accurately identify the right customer segments for targeted advertising, thereby improving overall marketing effectiveness.

Exploratory Data Analysis

The clicked on ad distribution tend to balance

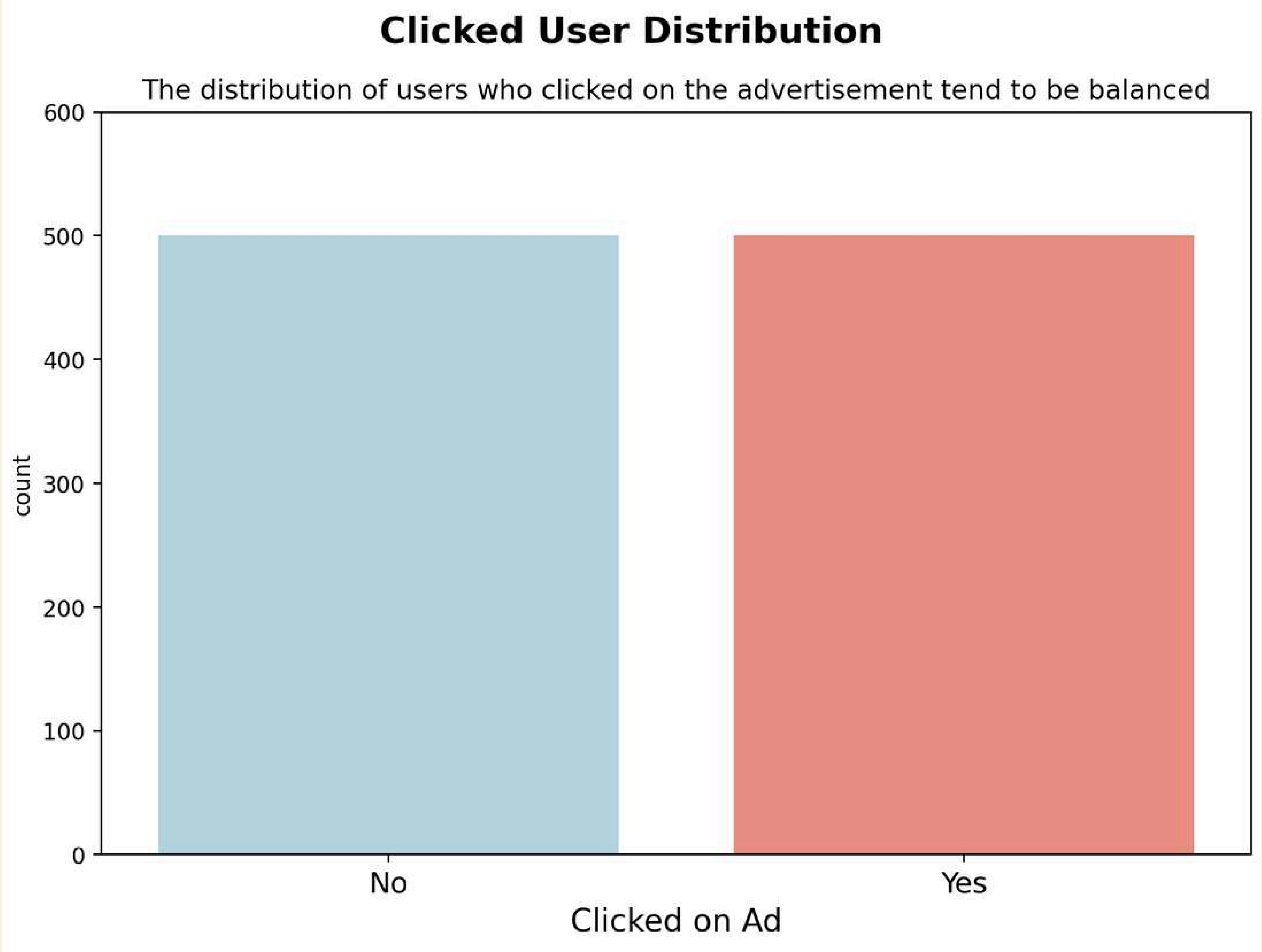


image 1.1 - clicked user distribution

The distribution of users who clicked on the advertisement is relatively balanced, with a close count of those who clicked ("Yes") and those who did not click ("No"). This indicates that the advertisement reached **a diverse audience**, engaging a substantial number of users.

source code EDA: [Link Github]

Exploratory Data Analysis

Older people tend to clicked on ad instead of younger poeple

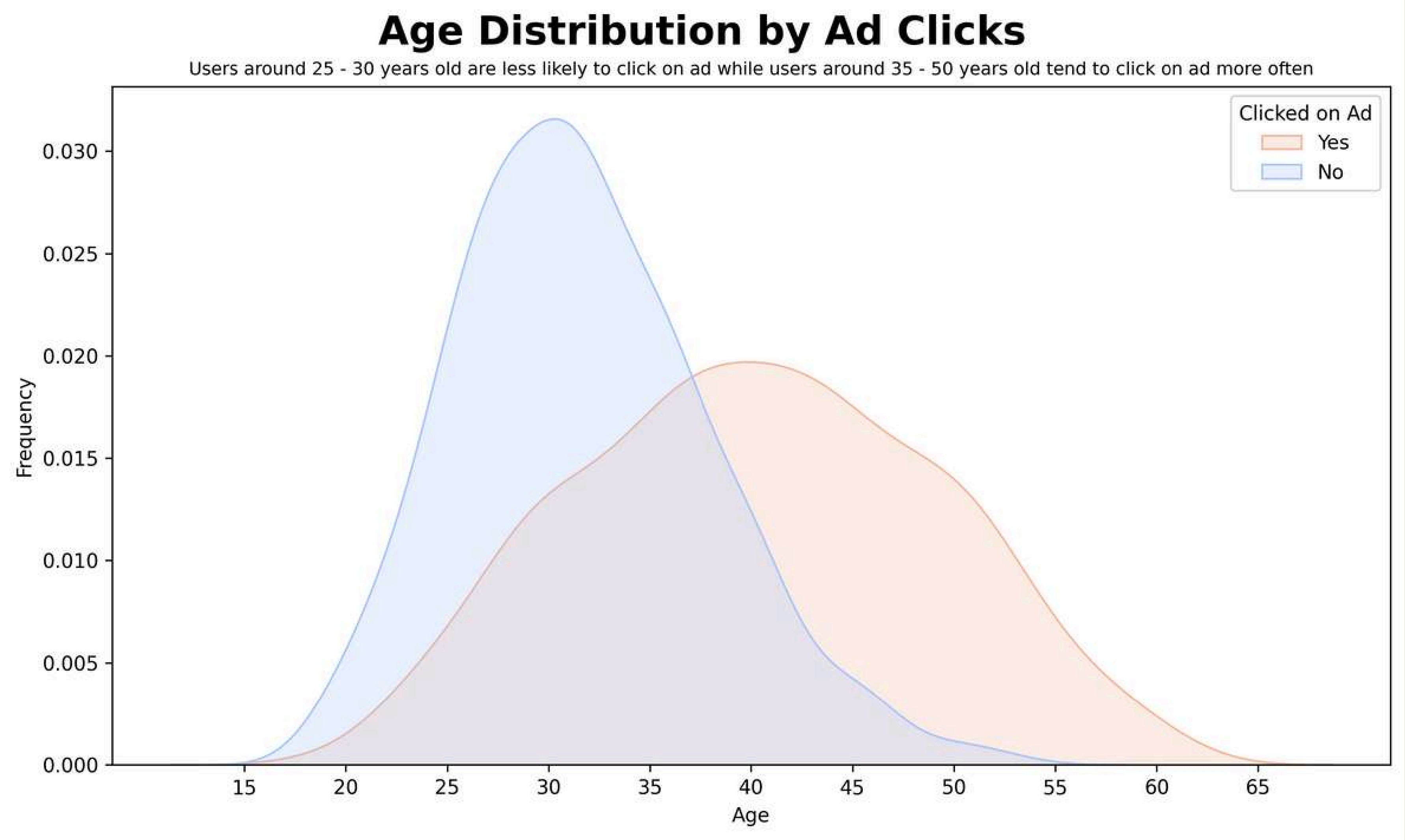


image 1.2 - age distribution

The users around 20 - 30 shows a higher frequency of “No” responses while users around **35 - 50 years** old tend to more clicked on ad, suggesting the ad is more **engaging** to this group of age. Therefore, the ad should be more relevant for older users instead of younger user.

source code EDA: [Link Github]

Exploratory Data Analysis

Female users are a bit higher than male users

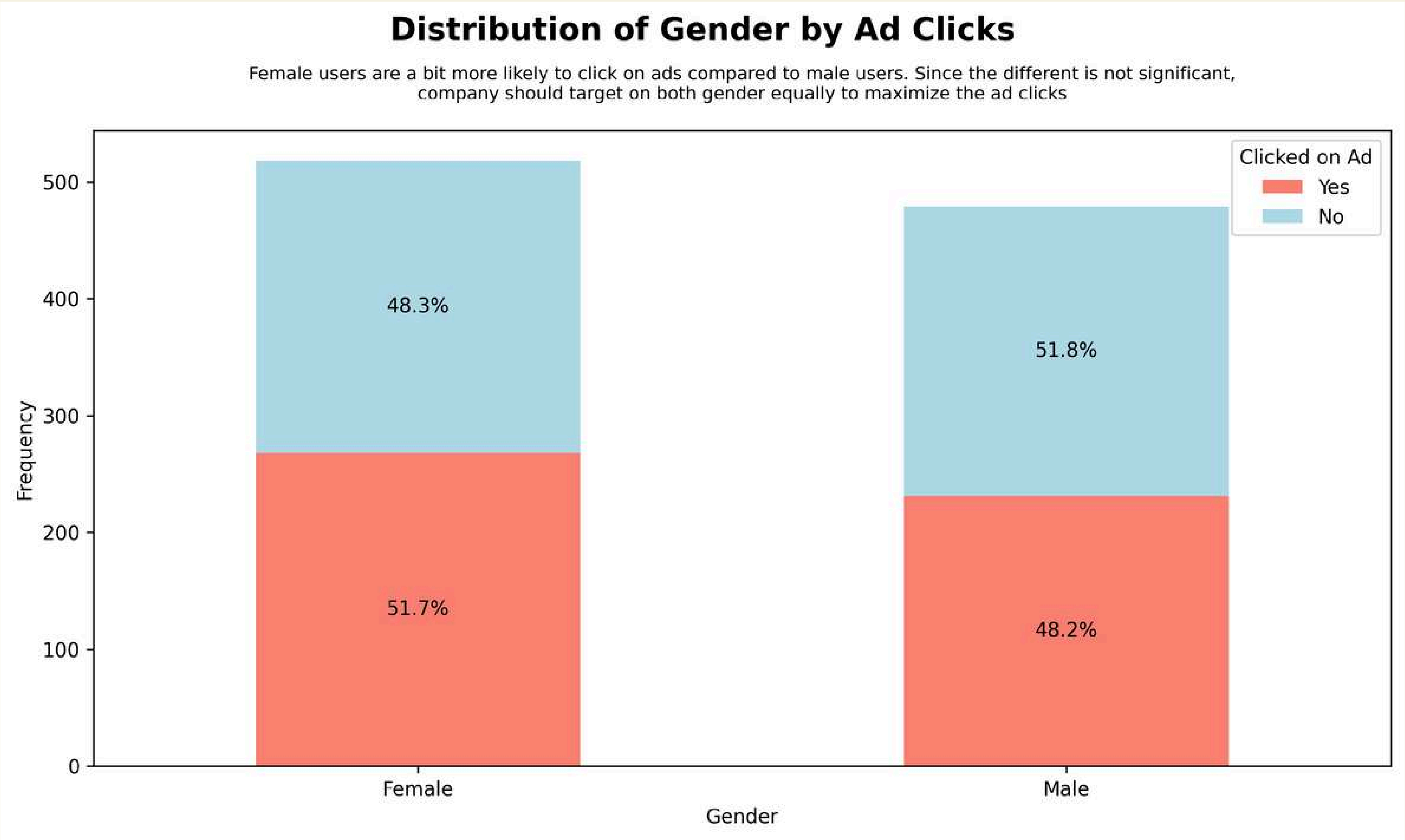


image 1.3 - gender distribution

Female users slightly a bit higher comparing to male users, including the number who clicked on ad. While targeting female users might be beneficial, we should continue to targeting on both.

source code EDA: [Link Github]

Exploratory Data Analysis

The lower users spend on site, the higher likely to engage on ad

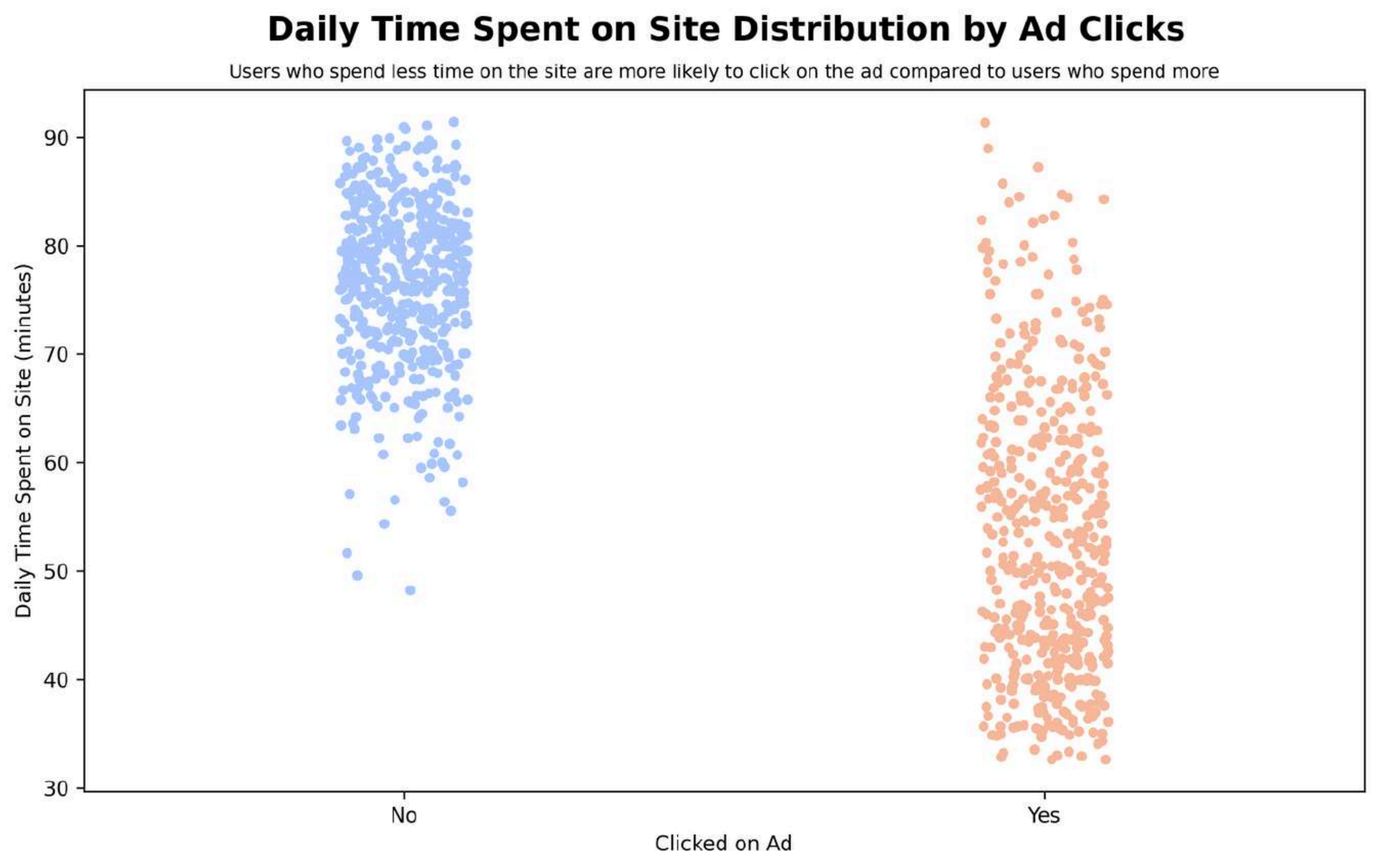


image 1.4 - daily time spent on site distribution

Users who spend on site **over** 65 minutes mostly **not clicked** on ad while users who spend **under** 65 minutes tend to **clicked on ad**. We can assume the users with a high daily time spent are for entertainment or the ad just not match for them.

source code EDA: [Link Github]

Exploratory Data Analysis

User Engagement Peaks During Non-Busy Hours

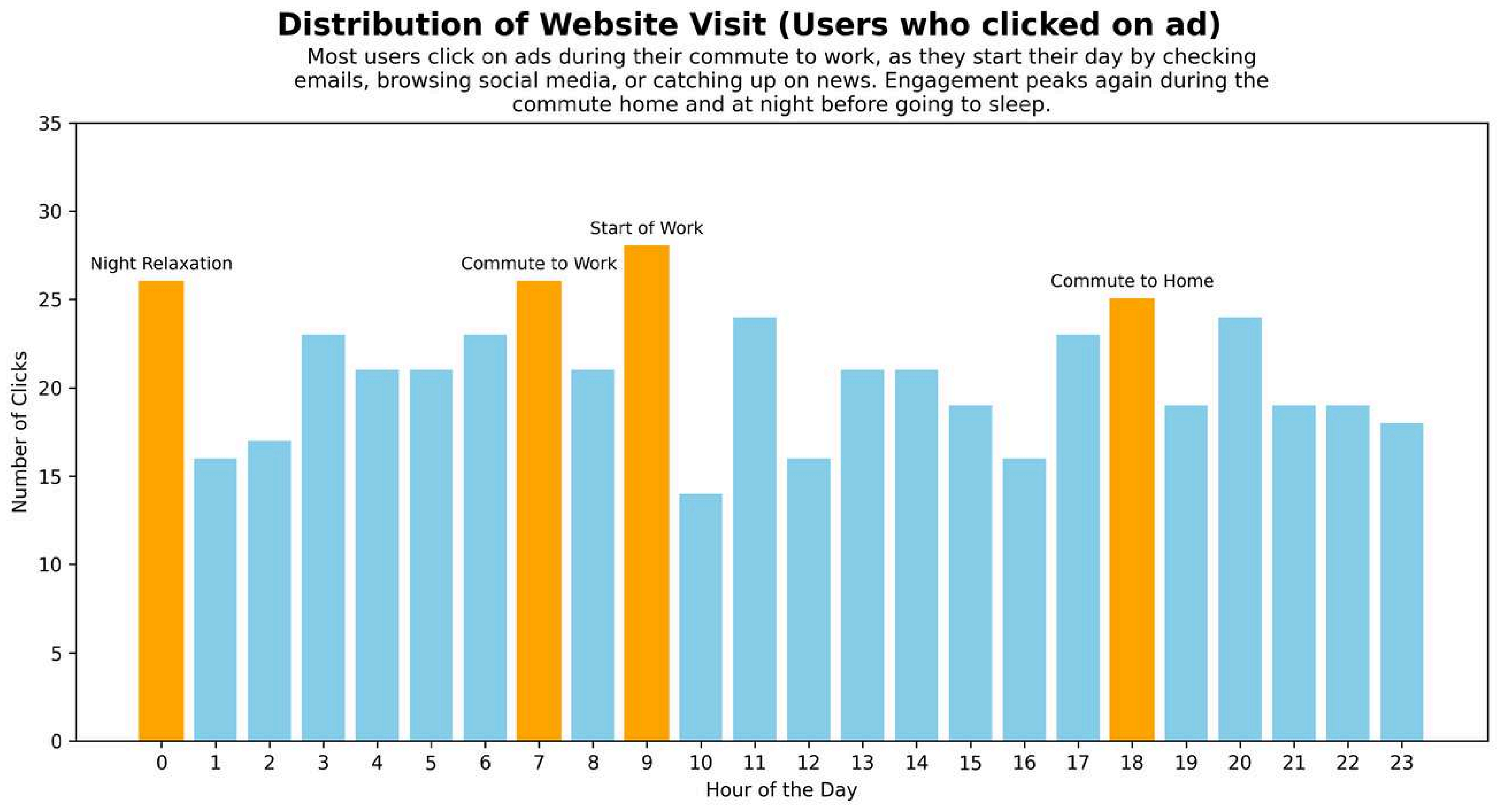


image 1.5 - distribution of website visit (clicked only)

The graphic indicates the users engagement with ads is **highest** during period when they are **less occupied,** specifically during **morning** and **night commute**, as well as at **night**. This suggest that users are more likely to clicked the ad when they have time to browse and engage with the content, rather than during their busy work hours.

source code EDA: Link Github

Exploratory Data Analysis

DKI Jakarta is the highest users but Banten is the better engagement

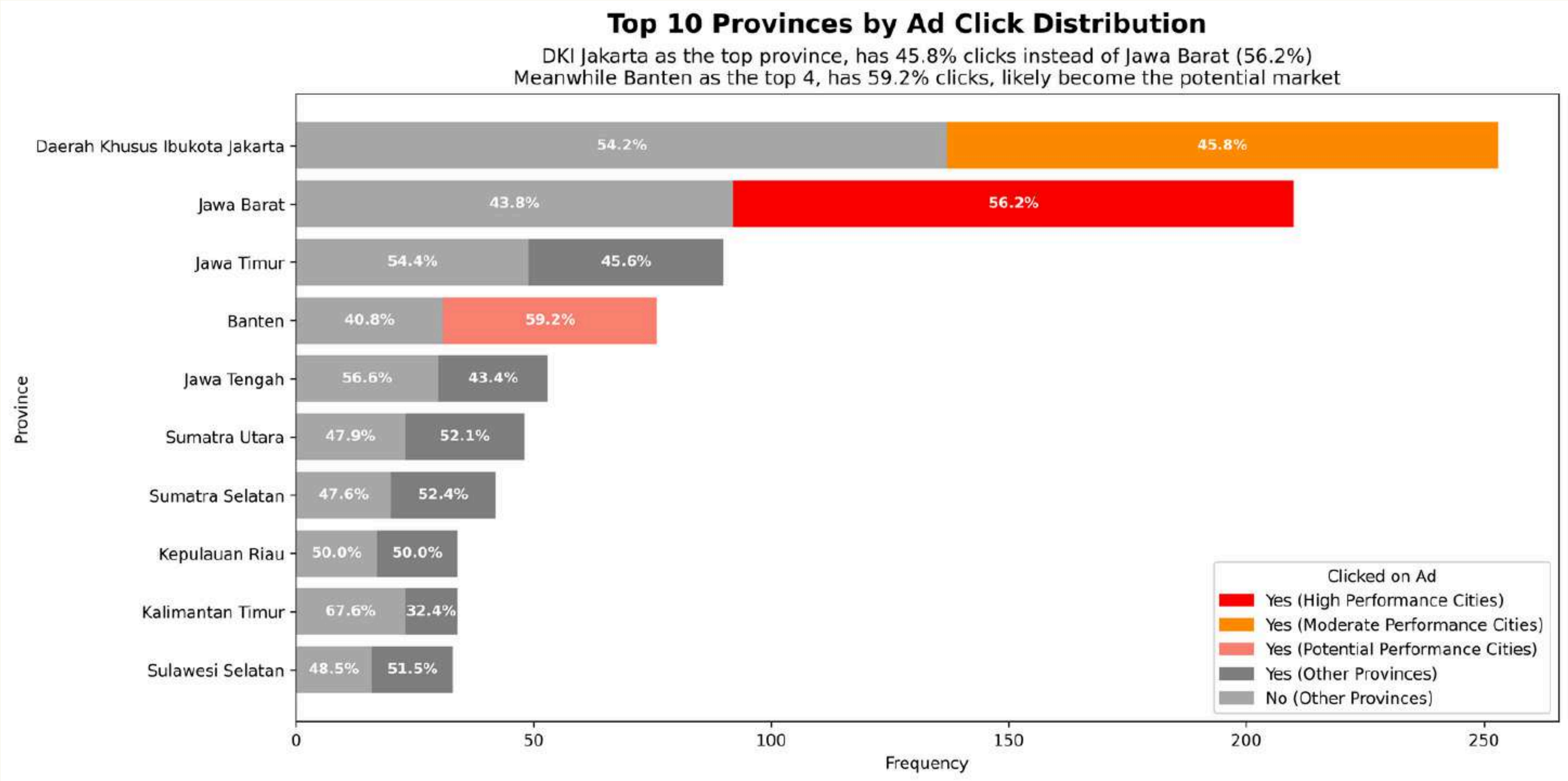


image 1.6 - top 10 provinces

DKI Jakarta as the top users only have 45.8% engagement while **Banten have 59.2% clicked on ad**, suggesting Banten users have **potential** to become a target market. We should targeting users who are from Banten and optimized target users on DKI Jakarta.

source code EDA: [Link Github]

Exploratory Data Analysis

Jawa Barat has Bandung, Jawa Timur has Surabaya, and DKI Jakarta has Jakarta Barat

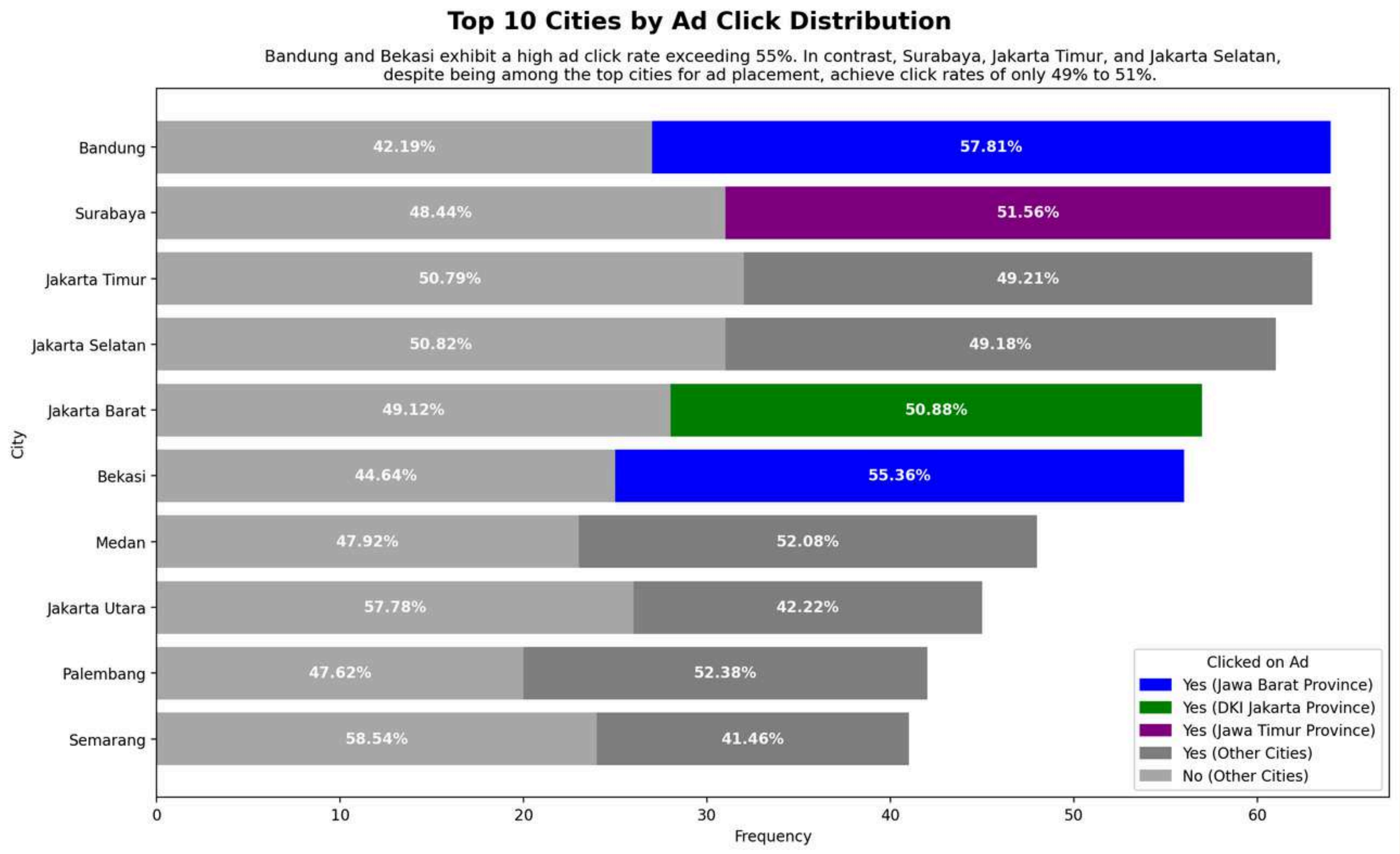


image 1.7 - top 10 cities

Bandung has a high engagement for Jawa Barat, followed up by Bekasi. Meanwhile, Jawa Timur has Surabaya only and DKI Jakarta province has Jakarta Barat with clicked rate 55.36%. This suggest if we **targeting users** in **these cities**, we can increase the clicked rate on ad, leading to efficiency cost.

source code EDA: Link Github

Exploratory Data Analysis

Users under 30 years old have a high spend time but less engagement, vice versa

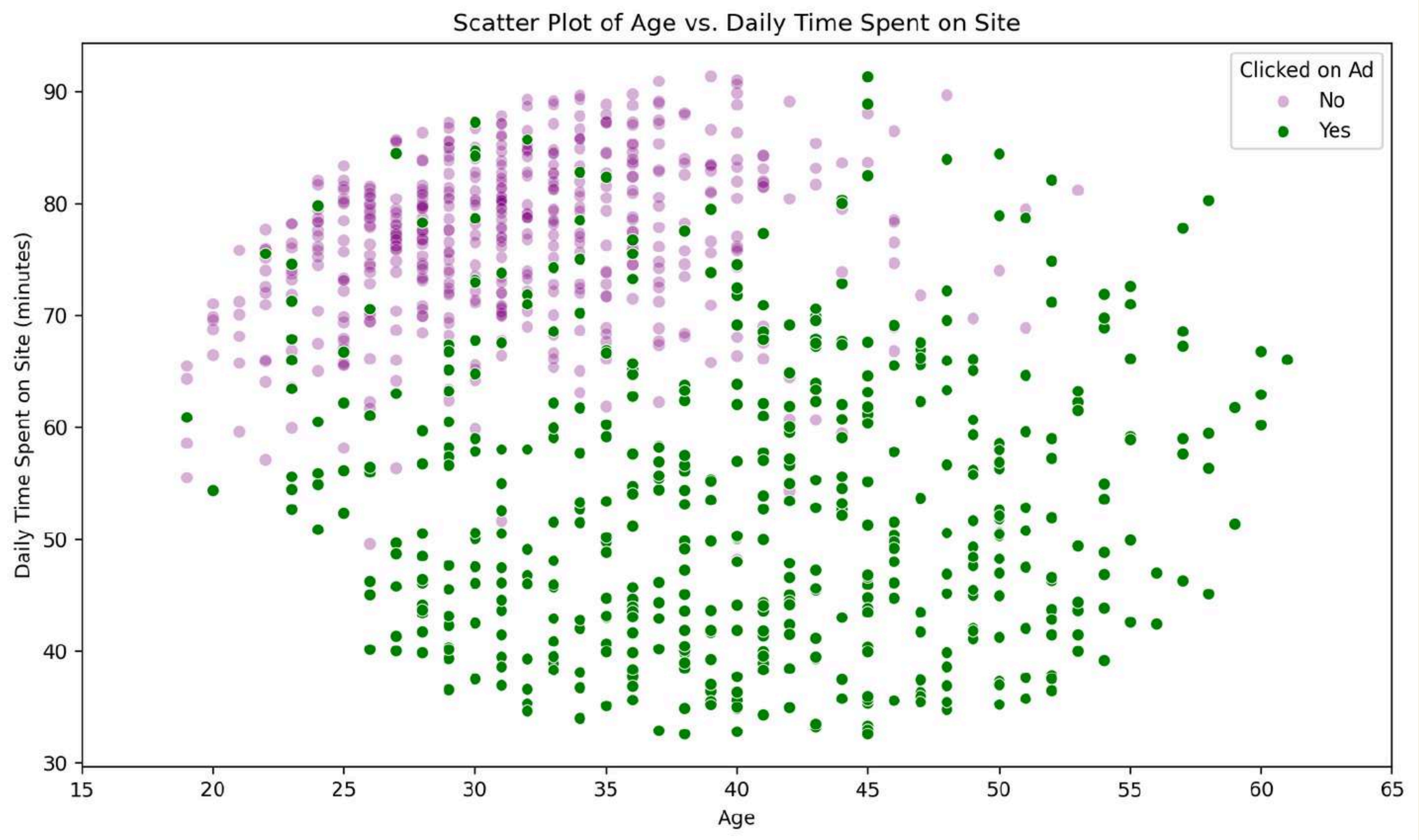


image 1.8 - age vs daily time spent on site

The data shows a high daily spend time (over 65 minutes) most likely users who are not clicked on ad and they are around 30 years old. But, users from 35 - 50 years old with low daily spend time on site (under 60 minutes) shows a high clicks.

source code EDA: Link Github

Exploratory Data Analysis

The younger of user's age, the high daily time spent on site and internet

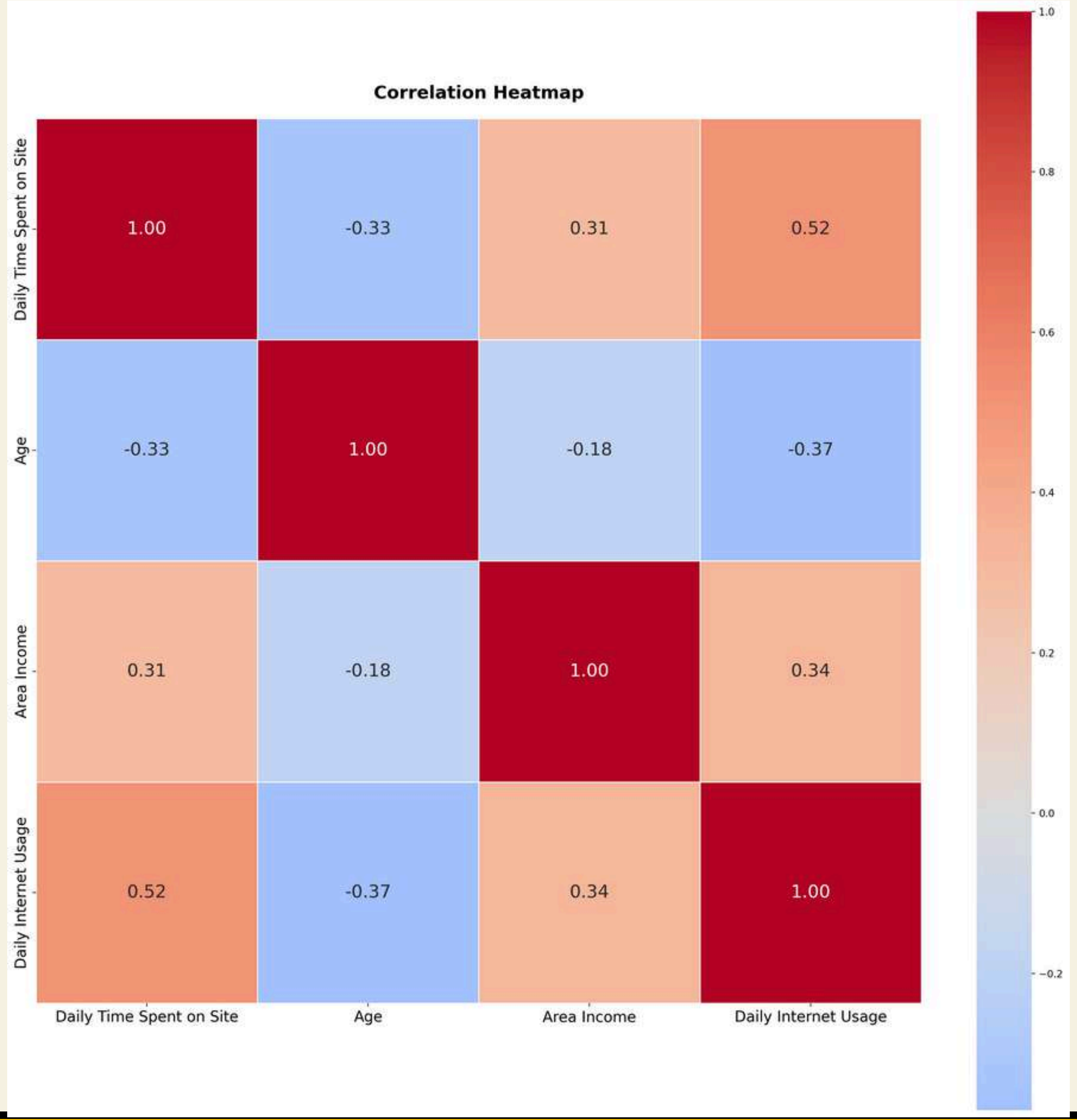


image 1.9 - correlation matrix

- **Age** have a slightly **negative correlation** with **daily time spent on site** (-0.33), **area income** (-0.18), and **daily internet usage** (-0.37).
- **Area Income** have a **positive correlation** with **daily time spent on site** (0.31) and **daily internet usage** (0.34).
- **Daily time spent on site** have a **moderate positive correlation** with **daily internet usage** (0.52)

source code EDA: Link Github

Data Preprocessing

There are 10 features including the label “Clicked on Ad”. In this step, we make 3 preprocessed data such as:

- df_nums: dataframe by excluding categorical features.
- df_OHE: handling categorical features using one hot encoding.
- df_label: handling categorical features using label encoing.

These preprocessed data are used for experiments in machine learning to see which is the best model.

Chi-square Test for Feature Selection on Categorical Features

p-value over 0.05

Feature	chi-square score	**p-value**	DoF
city	34.947	**0.2063**	29
Male	1.091	**0.2961**	15
province	16.0214	**0.3806**	15
category	6.4404	**0.6952**	9

p-value over 0.05 is considered as not associated with be label

Looking at the chi-square test result, it shows every p-value in the categorical features are above 5% (0.05), indicating that **these features is not associated with the label**.

But, even though it does not show a significant association, they **might still contributed** to the model's predictive power when combined with other features. So, we **split them** into **df_nums** (drop categorical features), **df_OHE**, and **df_label**.

source code preprocessed data: [Link Github]()

Missing value and duplicated data

Feature	Total Missing Values	Missing Values Percentage
daily time spent on site	13	1.3%
area income	13	1.3%
daily internet usage	11	0.1%
Male	3	0.3%

Missing values:

- are income, daily time spent, on site and internet usage will be filled with **median** because it robust to outliers.
- Missing value in male feature will be dropped because dropping them will not affect much lost data.

Duplicated Data:

- There are no duplicated data found.

source code preprocessed data: [Link Github]

Outliers

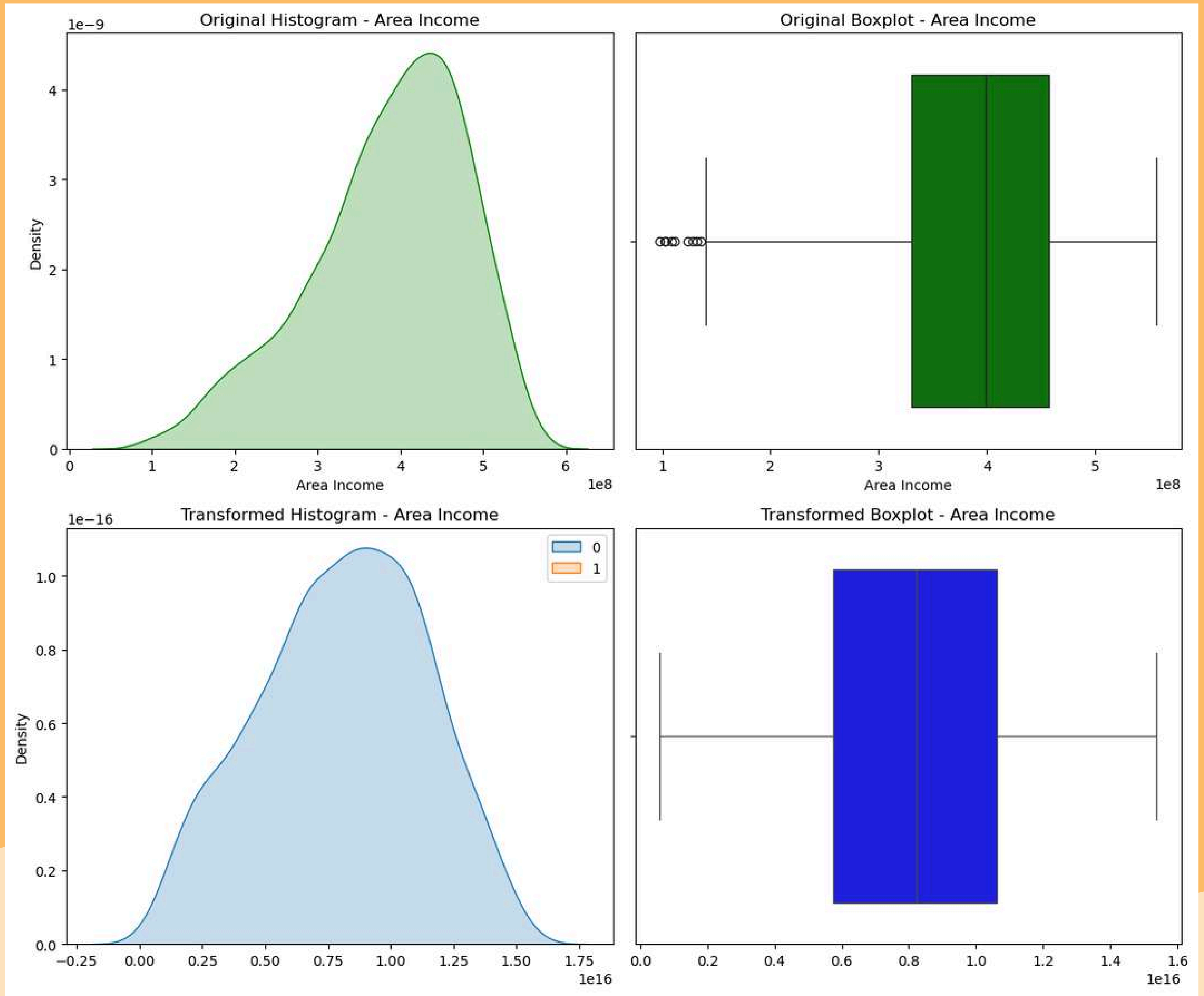


image 2.1 - area income distribution

Outliers found in Area Income feture

Findings:

- There are outliers on the left side of distribution (based on the green boxplot) and tend to left-skewed.
- However, no sign of anomaly, suggesting not to drop these outliers.

Handling Outliers:

- Applying yeo-johnson transformation because it can efectively make the left-skewed data more normal distributed.

source code preprocessed data: Link Github

Handling Categorical Features

Feature	Total Unique Values
city	30
Male	2
province	16
category	10

One Hot Encoding

Label Encoding

source code preprocessed data: [Link Github]

apply OHE

- City:
  - to reduced the number of new features, we create top 3 cities and the rest will be aggregated as “other city”.
- Province:
  - Same as city, province feature is aggregated into “other province” and pick the top three only.
- Category:
  - Grouping the unique values into Tech & Automotive, Home & Living, Health & Wellness, Lifestyle, and Finance & Banking.

One Hot Encoding

Help model to process categorical data and captures relationship between feature more effectively without ordinal relationship.

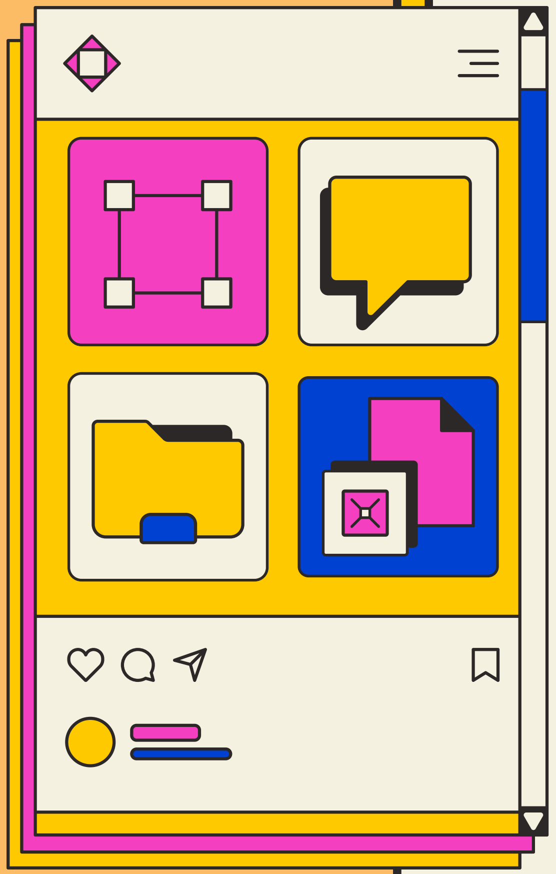
label encoding used on **male feature** and all of them be applied on **df_OHE**

source code preprocessed data: Link Github

Label Encoding

Transform categorical data into numeric without creating new features like one-hot-encoding.

due to their split-based nature, unlike linear model, tree-ensemble **does not require** One-Hot-Encoding



apply label encoding

All of categorical features were applied with label encoding method, including non ordinal features (e.g. city, province, and category).

This features will be applied on **df_label** and the **reason** to use label encoding is because the number of unique values are too many, we want to see if label encoding can give a better performance using tree-based model and ensemble method without worrying of curse of dimensity.

source code preprocessed data: [Link Github]

Datetime feature

Timestamp feature will be extracted into:

- day_of_week
- day_of_month
- month
- hour

and then, we drop the TImestamp feature

```python
def extract_timestamp(data):
    data['Timestamp'] = pd.to_datetime(data['Timestamp'])
    data['day_of_week'] = data['Timestamp'].dt.dayofweek
    data['day_of_month'] = data['Timestamp'].dt.day
    data['month'] = data['Timestamp'].dt.month
    data['hour'] = data['Timestamp'].dt.hour

    data = data.drop(columns=['Timestamp'])

    return data
```

image 2.2 - feature extraction on timestamp

source code preprocessed data: [Link Github]

Modelling

In this section, we will test models using the three preprocessed data. There are some experiments that will be conducted, including no scaling treatment, standardization, and normalization.

The label used is “Clicked on Ad” with distribution 50:50, a balance data. Therefore, the metrics are accuracy, precision, recall, and f1-score

	Model	Accuracy	Precision	Recall	F1 Score	Duration
0	Logistic Regression (No Scaling)	0.500000	0.000000	0.000000	0.000000	0.004000
1	K-Nearest Neighbors (No Scaling)	0.676667	0.685315	0.653333	0.668942	0.009001
2	Naive Bayes (No Scaling)	0.753333	0.811475	0.660000	0.727941	0.001999
3	Decision Tree (No Scaling)	0.950000	0.941176	0.960000	0.950495	0.002998
4	Random Forest (No Scaling)	0.950000	0.947020	0.953333	0.950166	0.096518
5	Gradient Boosting (No Scaling)	0.953333	0.947368	0.960000	0.953642	0.103514
6	AdaBoost (No Scaling)	0.953333	0.959459	0.946667	0.953020	0.049004
7	XGBoost (No Scaling)	0.960000	0.953947	0.966667	0.960265	1.757053
8	Logistic Regression (Standardization)	0.976667	0.993103	0.960000	0.976271	0.002002
9	K-Nearest Neighbors (Standardization)	0.963333	0.986014	0.940000	0.962457	0.008000
10	Naive Bayes (Standardization)	0.960000	0.960000	0.960000	0.960000	0.000000
11	Decision Tree (Standardization)	0.950000	0.941176	0.960000	0.950495	0.002001
12	Random Forest (Standardization)	0.950000	0.947020	0.953333	0.950166	0.094035
13	Gradient Boosting (Standardization)	0.953333	0.947368	0.960000	0.953642	0.101511
14	AdaBoost (Standardization)	0.953333	0.959459	0.946667	0.953020	0.048517
15	XGBoost (Standardization)	0.960000	0.953947	0.966667	0.960265	0.027996
16	Logistic Regression (Normalization)	0.970000	0.993007	0.946667	0.969283	0.002002
17	K-Nearest Neighbors (Normalization)	0.963333	0.992908	0.933333	0.962199	0.008000
18	Naive Bayes (Normalization)	0.960000	0.960000	0.960000	0.960000	0.000997
19	Decision Tree (Normalization)	0.950000	0.941176	0.960000	0.950495	0.002004
20	Random Forest (Normalization)	0.950000	0.947020	0.953333	0.950166	0.094229
21	Gradient Boosting (Normalization)	0.953333	0.947368	0.960000	0.953642	0.105144
22	AdaBoost (Normalization)	0.953333	0.959459	0.946667	0.953020	0.049488
23	XGBoost (Normalization)	0.960000	0.953947	0.966667	0.960265	0.029000

image 3.1 - models test on df_nums

df_nums

the features selection are **numerical and datetime datatype only**

1.

BEST MODELS

Logistic Regression (Standardization) shows the highest result with accuracy score of 0.9767 from 0.5 (without scaling). Precision, recall, and f1 score shows a high result too with running time ≈0.002s

source code Modelling: [Link Github]

	Model	Accuracy	Precision	Recall	F1 Score	Duration
0	Logistic Regression (No Scaling)	0.973333	0.986301	0.960000	0.972973	0.003000
1	K-Nearest Neighbors (No Scaling)	0.866667	0.904412	0.820000	0.860140	0.116525
2	Naive Bayes (No Scaling)	0.953333	0.941558	0.966667	0.953947	0.000000
3	Decision Tree (No Scaling)	0.933333	0.922078	0.946667	0.934211	0.003001
4	Random Forest (No Scaling)	0.956667	0.947712	0.966667	0.957096	0.105525
5	Gradient Boosting (No Scaling)	0.950000	0.941176	0.960000	0.950495	0.121850
6	AdaBoost (No Scaling)	0.953333	0.959459	0.946667	0.953020	0.054007
7	XGBoost (No Scaling)	0.953333	0.947368	0.960000	0.953642	0.033998
8	Logistic Regression (Standardization)	0.973333	0.986301	0.960000	0.972973	0.001999
9	K-Nearest Neighbors (Standardization)	0.866667	0.904412	0.820000	0.860140	0.052005
10	Naive Bayes (Standardization)	0.953333	0.941558	0.966667	0.953947	0.000999
11	Decision Tree (Standardization)	0.933333	0.922078	0.946667	0.934211	0.002000
12	Random Forest (Standardization)	0.956667	0.947712	0.966667	0.957096	0.108191
13	Gradient Boosting (Standardization)	0.950000	0.941176	0.960000	0.950495	0.118893
14	AdaBoost (Standardization)	0.953333	0.959459	0.946667	0.953020	0.054005
15	XGBoost (Standardization)	0.953333	0.947368	0.960000	0.953642	0.027505
16	Logistic Regression (Normalization)	0.973333	0.986301	0.960000	0.972973	0.002002
17	K-Nearest Neighbors (Normalization)	0.866667	0.904412	0.820000	0.860140	0.049000
18	Naive Bayes (Normalization)	0.953333	0.941558	0.966667	0.953947	0.001004
19	Decision Tree (Normalization)	0.933333	0.922078	0.946667	0.934211	0.002001
20	Random Forest (Normalization)	0.956667	0.947712	0.966667	0.957096	0.098511
21	Gradient Boosting (Normalization)	0.950000	0.941176	0.960000	0.950495	0.117022
22	AdaBoost (Normalization)	0.953333	0.959459	0.946667	0.953020	0.052001
23	XGBoost (Normalization)	0.953333	0.947368	0.960000	0.953642	0.029006

image 3.2 - models test on df_OHE

df_OHE

the features selection includes all features with **OHE method** for categorical data.

2.

BEST MODELS

Using OHE method, the score is slightly decreased with logistic regression accuracy of 0.9733 on every scaling treatment. Still, the score is in a high number, indicating new features reducing the model performance.

source code Modelling: [Link Github]

df_label

the features selection includes all features with **label encoding** method for categorical data.

	Model	Accuracy	Precision	Recall	F1 Score	Duration
0	Decision Tree (No Scaling)	0.953333	0.941558	0.966667	0.953947	0.003000
1	Random Forest (No Scaling)	0.956667	0.947712	0.966667	0.957096	0.124514
2	Gradient Boosting (No Scaling)	0.950000	0.941176	0.960000	0.950495	0.118512
3	AdaBoost (No Scaling)	0.953333	0.953333	0.953333	0.953333	0.053511
4	XGBoost (No Scaling)	0.950000	0.947020	0.953333	0.950166	0.023995
5	Decision Tree (Standardization)	0.953333	0.941558	0.966667	0.953947	0.002002
6	Random Forest (Standardization)	0.956667	0.947712	0.966667	0.957096	0.135026
7	Gradient Boosting (Standardization)	0.950000	0.941176	0.960000	0.950495	0.118215
8	AdaBoost (Standardization)	0.953333	0.953333	0.953333	0.953333	0.050558
9	XGBoost (Standardization)	0.950000	0.947020	0.953333	0.950166	0.026509
10	Decision Tree (Normalization)	0.953333	0.941558	0.966667	0.953947	0.002002
11	Random Forest (Normalization)	0.956667	0.947712	0.966667	0.957096	0.136515
12	Gradient Boosting (Normalization)	0.950000	0.941176	0.960000	0.950495	0.121521
13	AdaBoost (Normalization)	0.953333	0.953333	0.953333	0.953333	0.056512
14	XGBoost (Normalization)	0.950000	0.947020	0.953333	0.950166	0.036004

image 3.3 - models test on df_label

3.

BEST MODELS

Random Forest (No Scaling) shows the best result with accuracy of 0.9567, precision of 0.9536, recall 0.9600, f1 score 0.9568, and running time ≈0.1215s

source code Modelling: [Link Github]

Selected Model	Accuracy	Precision	Recall	F1-Score	Duration
Logistic Regression (Standardization) - df_nums	**0.9767**	**0.9931**	**0.9600**	**0.9763**	**0.002002s**
Logistic Regression (Standardization) - df_OHE	0.9733	0.9863	0.9600	0.9729	0.001999s
Random Forest (No Scaling) - df_label	0.9567	0.9536	0.9667	0.9571	0.124514s

Selected Model

Choosing from the top 3 score, **Logistic Regression (Standardization)** on **df_nums** will be chosen as the best model because categorical features does not have a great impact to the model's performance. Therefore, dropping them should reduced the number of features used. Also, the accuracy tend higher with score 0.9767 and precision 0.9931.

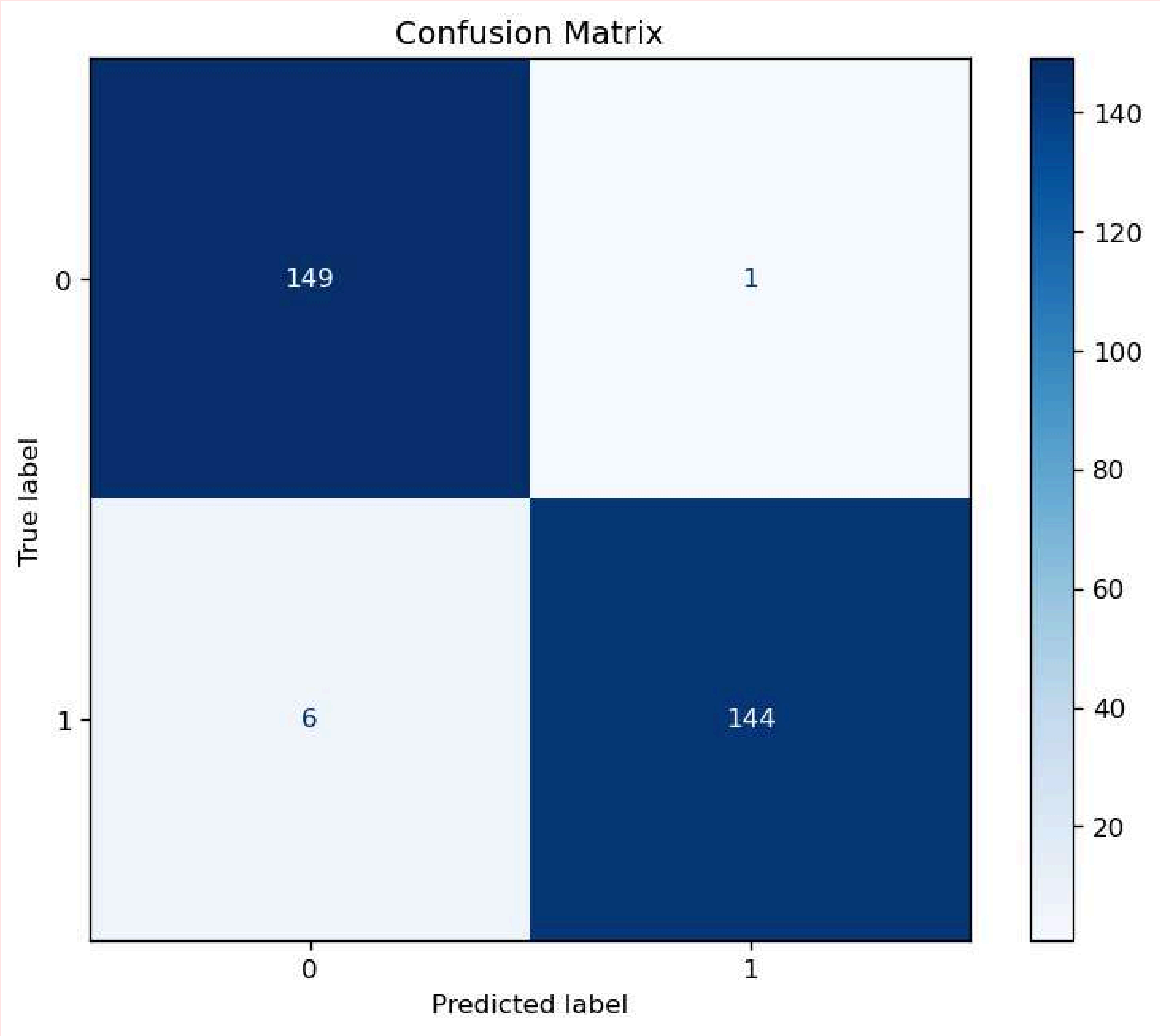


image 3.4 - confusion matrix

The model demonstrates strong performance with high true positive and true negative rates. The low counts of false positives and false negatives indicate that it effectively distinguishes between users who click on ads and those who do not.

source code Modelling: [Link Github]

Feature importance shows:

- **Daily Time Spent on Site** as feature with great impact on users to click the ad.
- Followed up with **daily internet usage** also has an impact in predicting click on ad.
- The top 3 is **area income**, suggesting users who clicked the ad has a common thing in their income.
- **Age** also has an impact, we can make a target market based on their age.
- The rest features have a slightly impact.

Daily Time Spent on Site and Daily Internet Usage are the top 2 features affecting users to click the ad

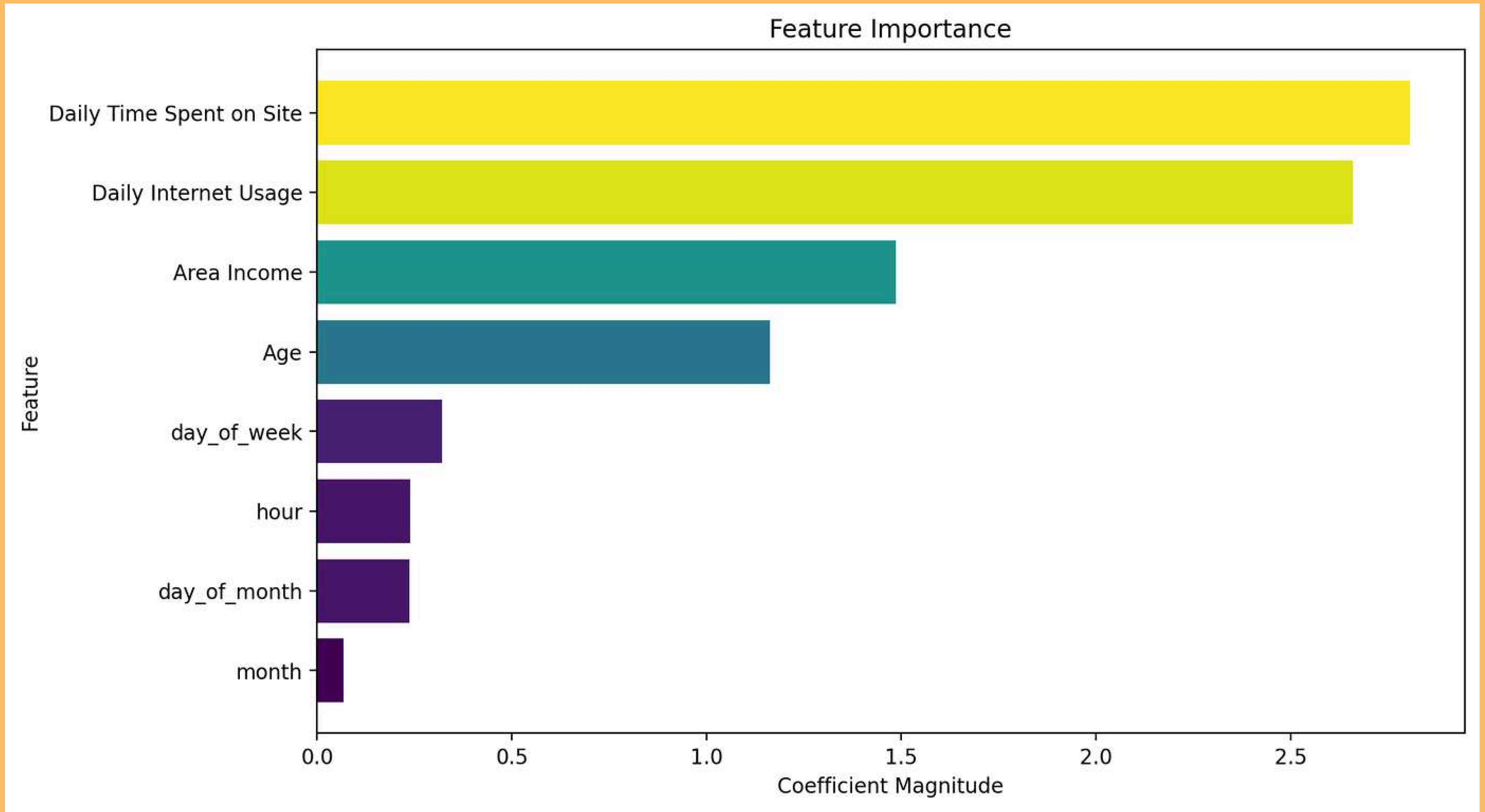


image 3.5 - feature importance

source code Modelling: [Link Github]

Older users with less daily time spent and internet usage with middle to low income tend to clicks on ad

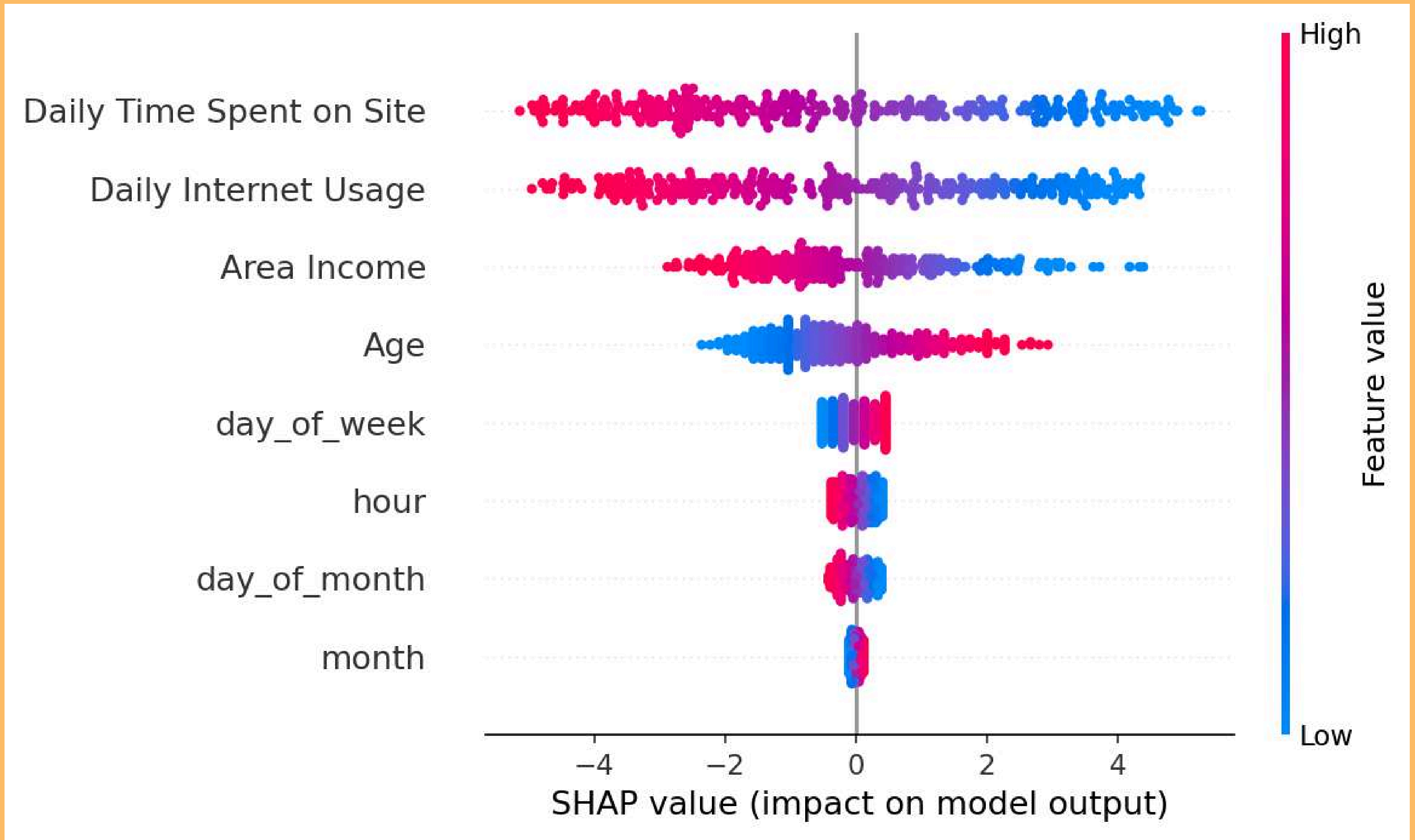


image 3.6 - SHAP value

source code EDA: [Link Github]

FIndings:

- The **lower daily time spent on site**, the higher users likely to click on ad.
- Same as **Daily Internet Usage** have a chance to click on ad with lower usage.
- **Low-middle income** tend to click on ad instead of high income users.
- **Older users (35 - 50 y.o.)** tend to click more on ad instead of younger users.
- users tend to click on ad during **commute to work** and in the **midnight**, where they have time to rest after full day

Business Recommendation

After thoroughly exploring the data and developing a machine learning model, this section will provide actionable business recommendations aimed at optimizing advertising strategies and enhancing user engagement. These insights will help the company make informed decisions to effectively target customers and improve overall ad performance

There are two class of users

Upper-class

This group primarily in 20 - 30 years old with a high income, a high spend time on site, and internet usage.

Lower-Class

While this group consist 35 - 50 years old, typically have lower to middle income, a low spend time on site and internet usage.

Business Recommendation

Recomendation	Insight	Actionable Items
Retargeting Marketing	Users aged 35-50 are more likely to engage with ads, indicating a potential market segment that is currently underserved.	1. **Develop Targeted Campaigns**: Create ad campaigns specifically aimed at the 35-50 age group, focusing on their interests and needs to enhance engagement.
Optimized Ad Time Delivery	Engagement peaks during morning & evening commutes and night time, suggesting optimal times for ad visibility.	2. **Schedule Ads Strategically**: Implement a strategy to display ads prominently during peak engagement times (7-9 AM, 5-7 PM, and 00.00 AM) to maximize visibility and interaction.
Enhance Content Relevance	Users with lower time spent on the site are more likely to click ads, suggesting that ad content needs to be engaging.	3. **Content Customization**: Tailor ad content to align with the interests of users who spend less time on the site, ensuring it captures their attention quickly.

Recomendation	Insight	Actionable Items
Focus on Potential Province	Banten Province has the highest click rate, indicating strong user engagement.	4. **Target Marketing in Banten**: Develop targeted advertising campaigns specifically for Banten to leverage its high engagement.
Leverage DKI Jakarta's Districts	The overall high click rate in DKI Jakarta is driven by districts, with Jakarta Barat having the highest rate.	5. **Strategic Ads in Jakarta Barat**: Focus advertising efforts in Jakarta Barat to maximize engagement and conversions, analyzing what drives clicks in this district.
Capitalize on Surabaya's Performance	Surabaya's high click rate contributes to Jawa Timur being a top province for ad engagement.	6. **Expand Efforts in Jawa Timur:** Increase advertising in Surabaya and other areas of East Java to capture additional market share, utilizing successful strategies from high-performing regions.
Recognize Bandung's User Base	Bandung has a significant user base, surpassing the four districts of Jakarta, making West Java highly competitive.	7. **Engage Users in Bandung**: Develop campaigns that resonate with the user base in Bandung, capitalizing on its strong engagement to drive ad clicks.

Business Simulation

Scenario before implementing machine learning

- targeted users (X_test): 300 users
- clicked users (y_test 50:50): 150 users
- cost per users: Rp 10.000
- revenue: Rp 15.000

- cost ad: 300 x Rp 10.000 = Rp 3.000.000
- converted: 150 x Rp 15.000 = Rp 2.250.000
- **Profit** = **- Rp 750.000**

Before using machine learning, we lost Rp 750.000 for 300 users

Scenario after implementing machine learning

- Predicted users: 145 users
- True Positive label: 144 users
- False Positive label: 1 users
- cost per users: Rp 10.000
- revenue: Rp 15.000

- cost ad: 145 x Rp 10.000 = Rp 1.450.000
- converted: 144 x Rp 15.000 = Rp 2.160.000
- **Profit** = **Rp 710.000**

After using machine learning, with a precise target market, we gain profit of Rp 710.000 with only 145 users

Summary

ML reduced cost and increase profit margin

Machine learning allows for data-driven decision-making, enabling us to analyze user behavior and preferences more accurately. By focusing on high-potential users, we can allocate our ad budget more efficiently, maximizing return on investment. Tailoring ads to the right audience can lead to higher engagement and customer satisfaction, further driving sales.

Thank You

Reach me!

aldivibriani@gmail.com

[Linkedin Profile]

[Github Profile]