Predict Clicked Ads Customer Classification by using Machine Learning

Supported by:
Rakamin Academy
Career Acceleration School
www.rakamin.com

Created by:
Amarindra Ardinova
Amarindra.a@gmail.com
https://www.linkedin.com/in/amarindra-ardinova/

I'm Amarindra Ardinova, and I dedicated over a decade of my career to the aviation industry, starting as an admin staff and later transitioning into a Business Development role. In 2016, I ventured into entrepreneurship, co-founding a company. Unfortunately, in 2021, the global pandemic compelled me to leave the company.

Seeking a new path, I explored remote work opportunities and, in 2023, discovered a keen interest in Data Analysis and Data Science. My passion lies in working with data, coding to unveil insights, and delving into the realms of deep learning and artificial intelligence. I am now excited to embark on this new journey and leverage my skills to contribute to this dynamic and evolving field.

Overview

“Sebuah perusahaan di Indonesia ingin mengetahui efektifitas sebuah iklan yang mereka tayangkan, hal ini penting bagi perusahaan agar dapat mengetahui seberapa besar ketercapainnya iklan yang dipasarkan sehingga dapat menarik customers untuk melihat iklan.

Dengan mengolah data historical advertisement serta menemukan insight serta pola yang terjadi, maka dapat membantu perusahaan dalam menentukan target marketing, fokus case ini adalah membuat model machine learning classification yang berfungsi menentukan target customers yang tepat ”

Customer Type and Behaviour Analysis on Advertisement

- Tulislah proses ***Exploration Data Analysis*** (EDA) yang mencakup ***Statistical analysis*** baik untuk data numerik maupun kategori, Selanjutnya buat visualisasi data untuk ***Univariate*** dan ***Bivariate analysis***, serta ***Multivariate analysis***
- Khusus untuk ***Bivariate analysis***, tunjukan hubungan antara kolom umur, daily internet usage, dan daily time spent on site.
- Tulislah juga **proses korelasi heatmap** untuk mengetahui tingkat korelasi antar kolom
- **Source code** yang sudah kamu buat, dapat ditampilkan dan berikan link untuk mengakses file tersebut. Contohnya seperti di pojok kanan bawah.

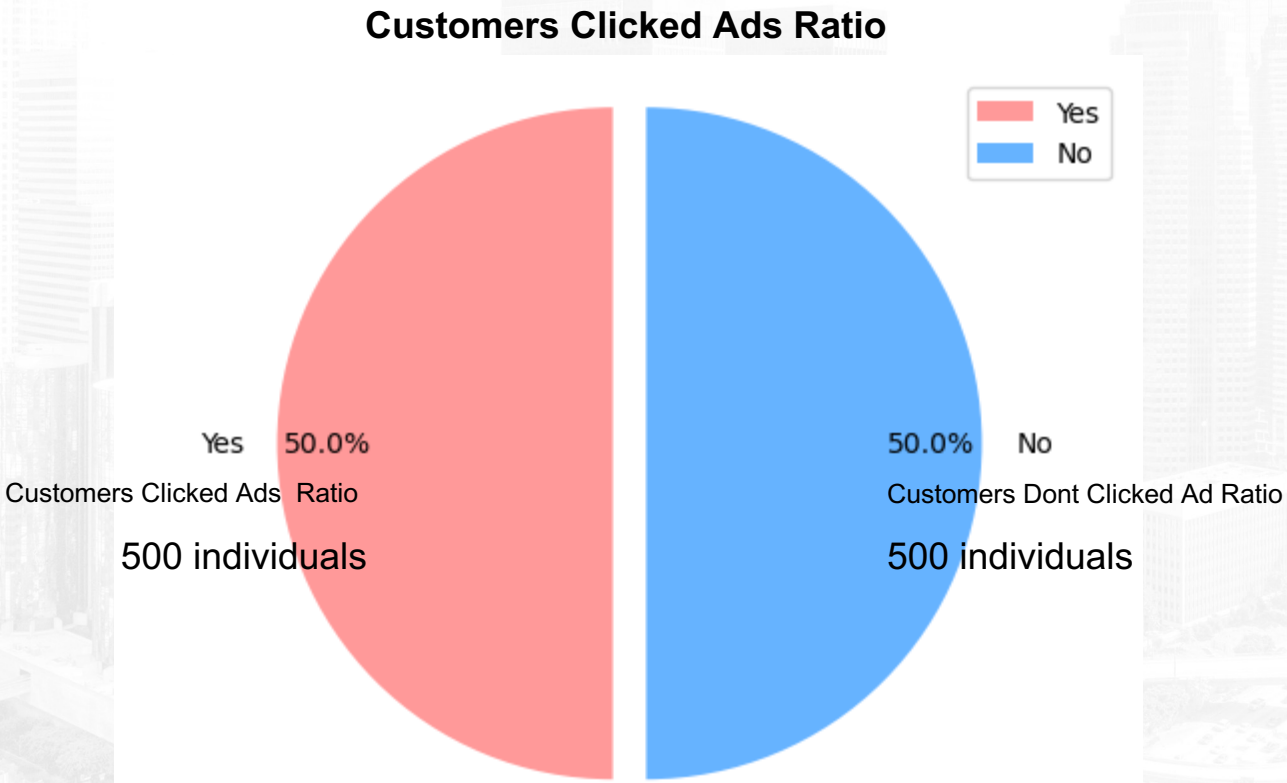
Untuk selengkapnya, dapat melihat jupyter notebook disini
https://drive.google.com/file/d/1VQKYqW9FEwGuN1WV5SJOaKI6ZlXfdzNU/view?usp=share_link

Content

Tugas 1

- EDA
- Statistical Analysis
- Cleaning Data
- Univariate Analysis
- Bivariate Analysis
- Multivariate Analysis
- Target Statistics Analysis
- Insight Summary

Customer Type and Behaviour Analysis on Advertisement



EDA

Total Row

- 1000

Columns

- 11

Duplicated

- 0

Unwanted

- Unnamed: 0

Missing Value

Daily Time Spent on Site (13)
Area Income (13)
Daily Internet Usage (11)
Male (3)

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Unnamed: 0                1000 non-null   int64
 1   Daily Time Spent on Site  987 non-null    float64
 2   Age                       1000 non-null   int64
 3   Area Income               987 non-null    float64
 4   Daily Internet Usage      989 non-null    float64
 5   Male                      997 non-null    object
 6   Timestamp                 1000 non-null   object
 7   Clicked on Ad             1000 non-null   object
 8   city                      1000 non-null   object
 9   province                  1000 non-null   object
 10  category                  1000 non-null   object
dtypes: float64(3), int64(2), object(6)
memory usage: 86.1+ KB
```

Statistical Analysis

Tabel di samping merupakan hasil analisis statistik dari DataFrame yang telah kami lakukan. Tujuan dari analisis ini adalah untuk menyajikan dan memudahkan pemahaman terhadap nilai-nilai yang terdapat dalam DataFrame.

Dari data ini kita bisa lihat bahwa Populasi didominasi Perempuan (518 orang) 51,8%. Sample terbanyak dari DKI Jakarta dan Iklan yang terbanyak di clicked Otomotif.

Numericals

	daily time spent on site	age	area income	daily internet usage	timestamp
count	987.0	1,000.0	987.0	989.0	NaT
mean	64.93	36.01	384,864,670.64	179.86	NaT
std	15.84	8.79	94,079,989.57	43.87	NaT
min	32.6	19.0	97,975,500.0	104.78	NaT
25%	51.27	29.0	328,632,990.0	138.71	NaT
50%	68.11	35.0	399,068,320.0	182.65	NaT
75%	78.46	42.0	458,355,450.0	218.79	NaT
max	91.43	61.0	556,393,600.0	267.01	NaT
median	68.11	35.0	399,068,320.0	182.65	NaT
modus	62.26	31.0	97,975,500.0	113.53	2016-05-20 12:17:00

Categoricals

	Male	Timestamp	Clicked on Ad	city	province	category
count	997	1000	1000	1000	1000	1000
unique	2	997	2	30	16	10
top	Perempuan	5/26/2016 15:40	No	Surabaya	Daerah Khusus Ibukota Jakarta	Otomotif
freq	518	2	500	64	253	112

Data Cleaning

1 Mengisi **Missing Value**

```
#statistik df

df_stat = df.describe().round(2)

median = df[nums].median().to_frame().T
median.rename(index={0:'median'}, inplace=True)


modus = df[nums].mode().iloc[0].to_frame().T
modus.rename(index={0:'modus'}, inplace=True)

df_stat = pd.concat([df_stat, median, modus], axis=0).round(2)

df_stat
```

```
df['male'].fillna('Perempuan', inplace=True)
```

2 Menghapus kolom **'Unnamed: 0'**

```
# drop Unnamed 0
df.drop('Unnamed: 0', axis=1, inplace=True)

# lowercase
df.columns = df.columns.str.lower()
```

3 Rename tipe data '**male**' & '**timestamp**'

```
df['timestamp'] = df['timestamp'].astype('datetime64')
df['male'] = df['male'].astype('int64')
```

4 Mengganti value data '**clicked on ad**' dan '**male**'

```
# rubah value
df['clicked on ad'] = df['clicked on ad'].replace({'Yes': 1, 'No': 0})
df['male'] = df['male'].replace({'Laki-Laki': 1, 'Perempuan': 0})
```

Data Cleaning

TRANSFORMASI DATAFRAME

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Unnamed: 0                1000 non-null   int64
 1   Daily Time Spent on Site  987 non-null    float64
 2   Age                       1000 non-null   int64
 3   Area Income               987 non-null    float64
 4   Daily Internet Usage      989 non-null    float64
 5   Male                      997 non-null    object
 6   Timestamp                 1000 non-null   object
 7   Clicked on Ad             1000 non-null   object
 8   city                      1000 non-null   object
 9   province                  1000 non-null   object
 10  category                  1000 non-null   object
dtypes: float64(3), int64(2), object(6)
memory usage: 86.1+ KB
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 10 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   daily time spent on site  1000 non-null   float64
 1   age                       1000 non-null   int64
 2   area income               1000 non-null   float64
 3   daily internet usage      1000 non-null   float64
 4   male                      1000 non-null   int64
 5   timestamp                 1000 non-null   datetime64[ns]
 6   clicked on ad             1000 non-null   int64
 7   city                      1000 non-null   object
 8   province                  1000 non-null   object
 9   category                  1000 non-null   object
dtypes: datetime64[ns](1), float64(3), int64(3), object(3)
memory usage: 78.2+ KB
```

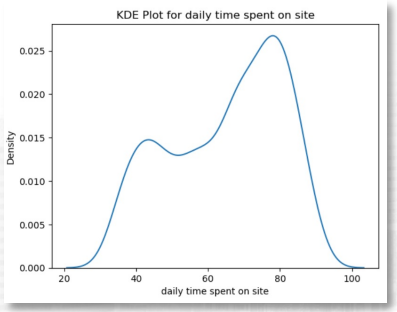
EDA

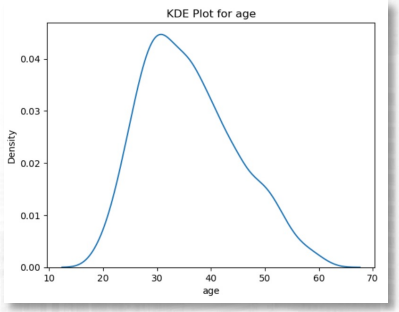
Total Row	Columns	Duplicated	Unwanted	Missing Value
• 1000	• 11	• 0	• Unnamed: 0	Daily Time Spent on Site (13) Area Income (13) Daily Internet Usage (11) Male (3)

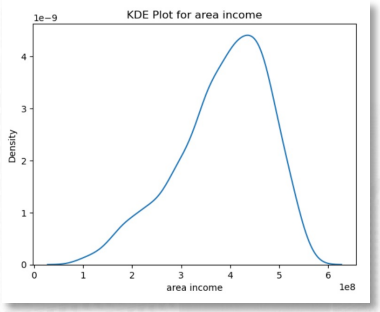
HANDLING & **CLEANING**

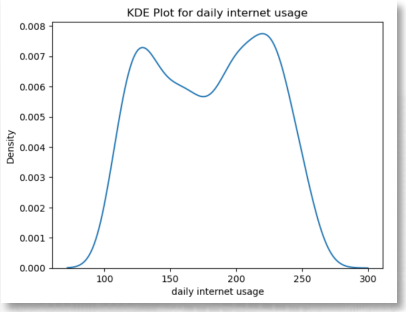
Total Row	Columns	Duplicated	Unwanted	Missing Value
• 1000	• 10	• 0	• drop	Fillna : Daily Time Spent on Site: 68,11 Area Income : 399.068.320 Daily Internet Usage: 182,65 Male : Perempuan

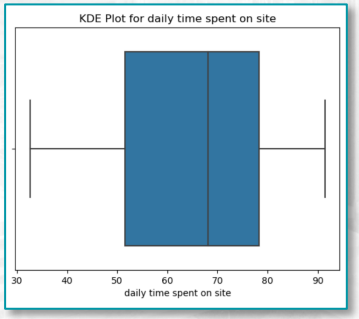
Univariate Analysis

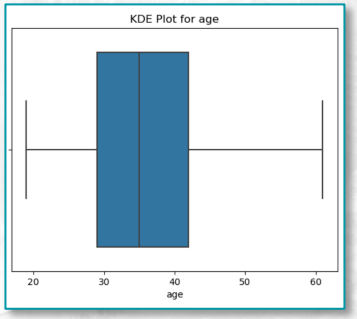


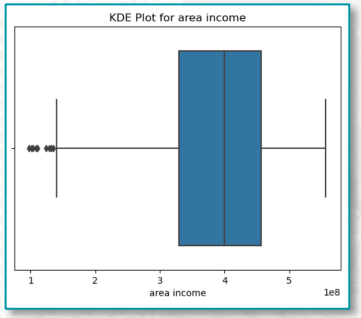


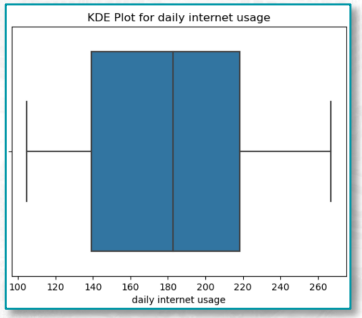




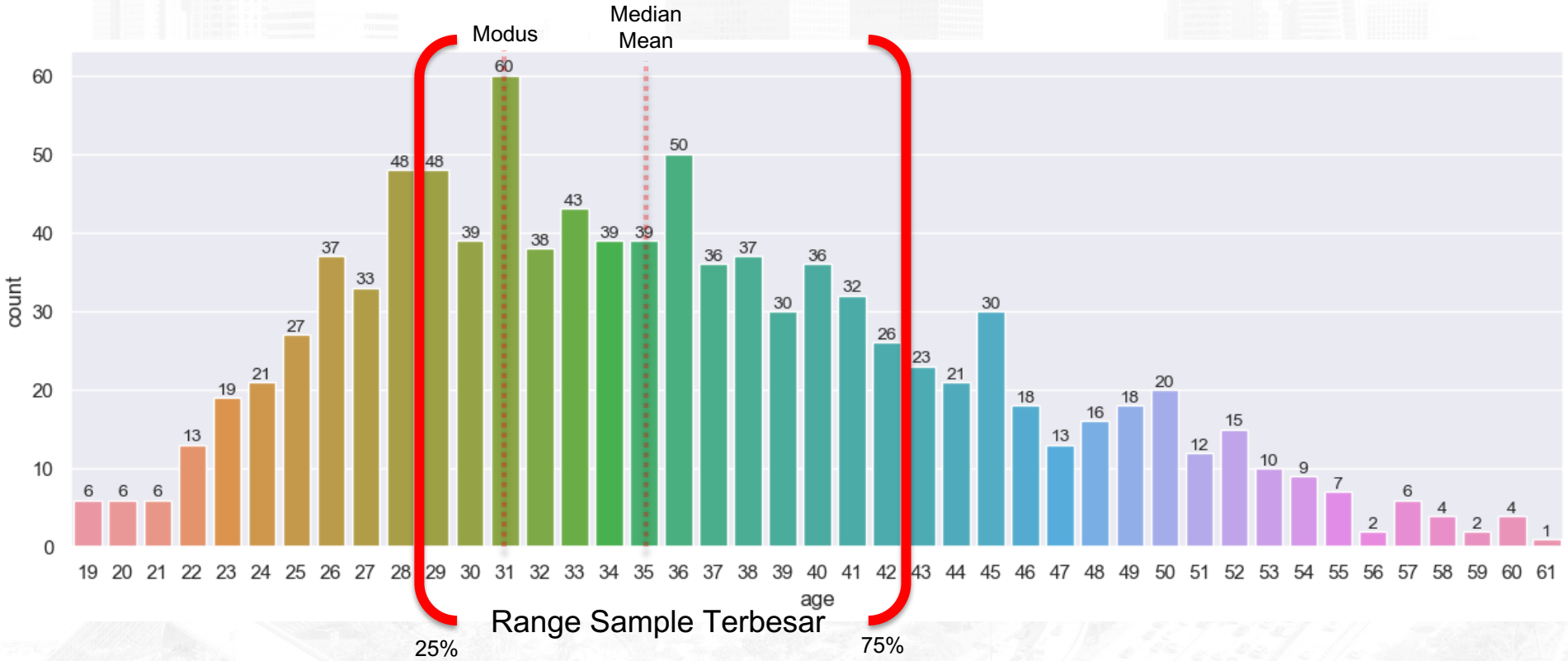




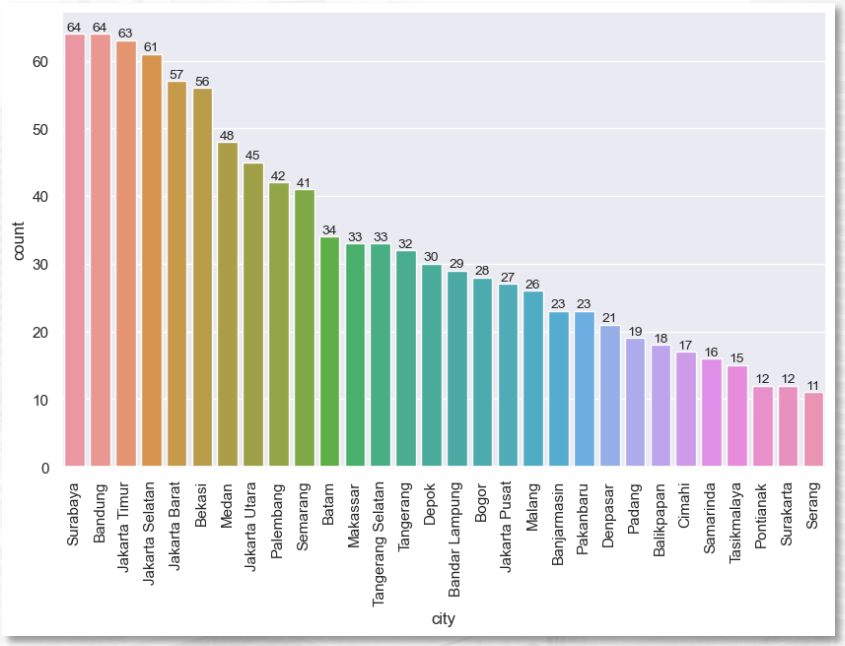


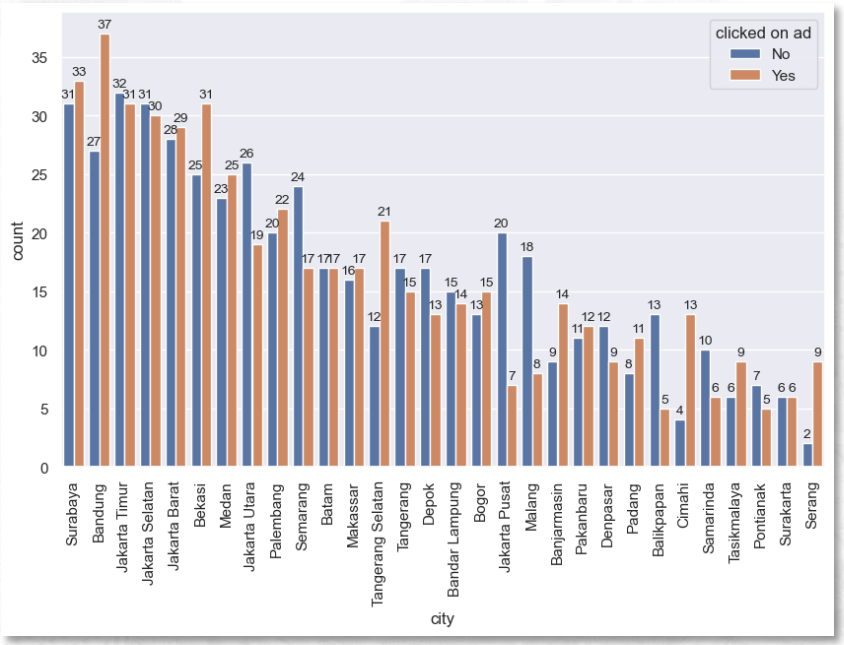


Univariate Analysis



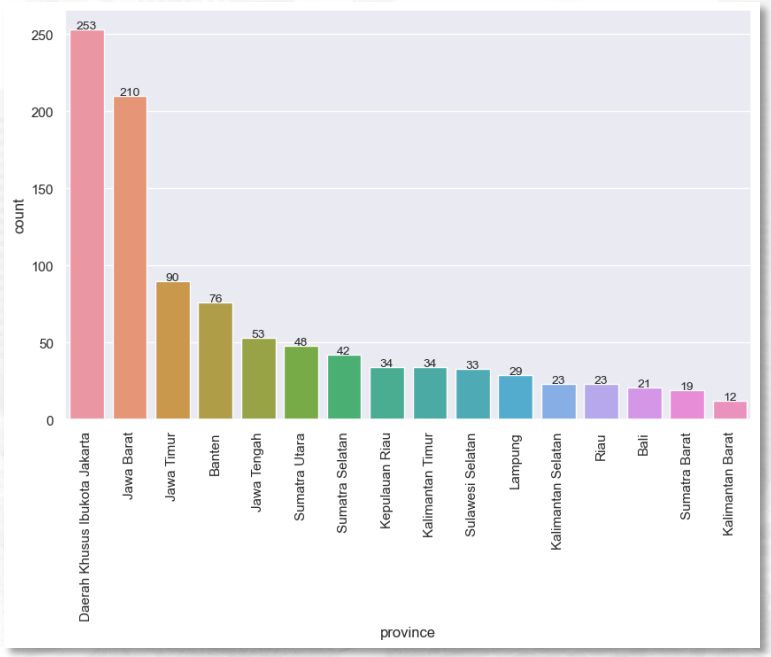
Univariate Analysis

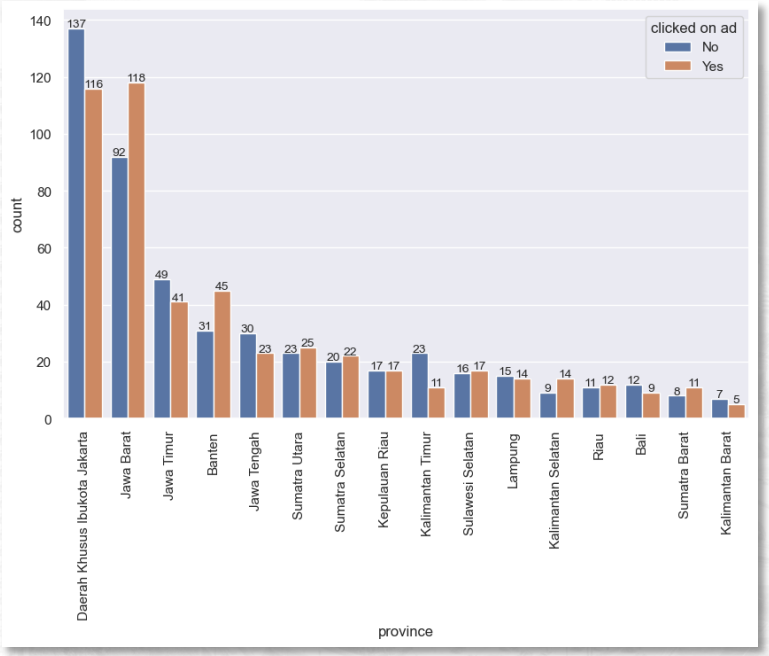




CITY

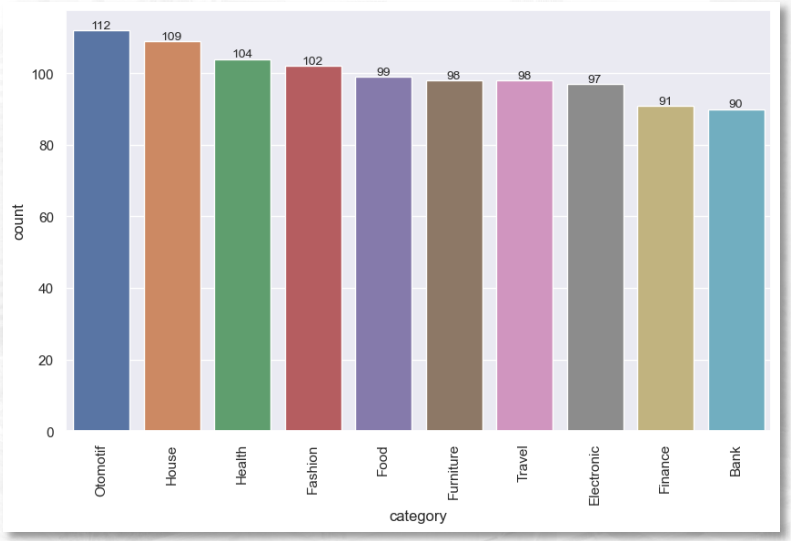
Univariate Analysis

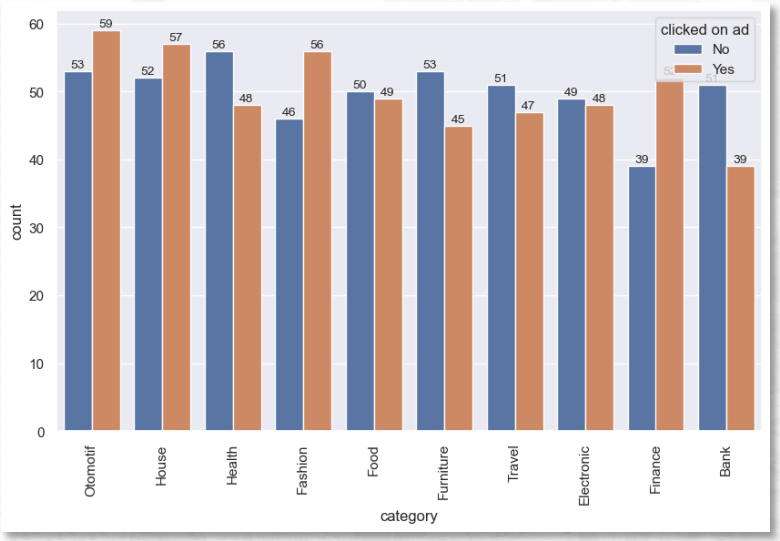




PROVINCE

Univariate Analysis

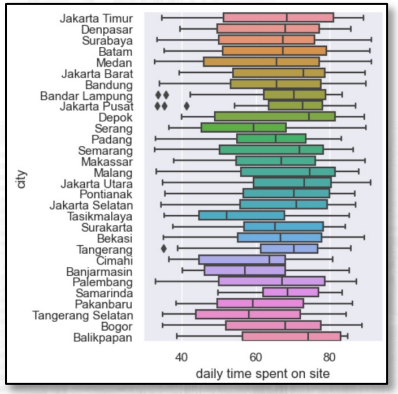


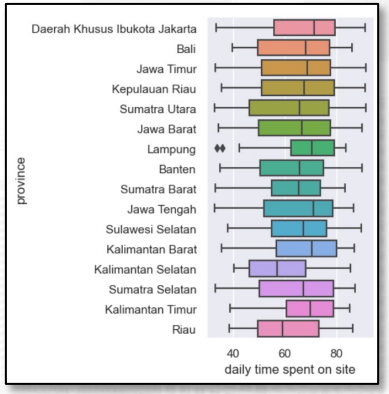


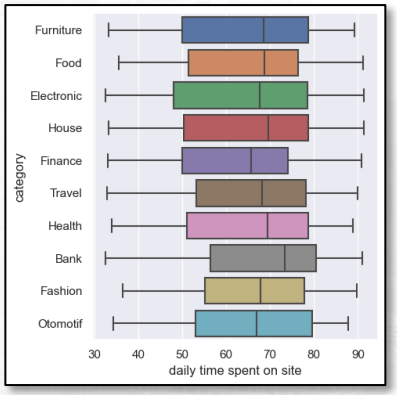
CATEGORY

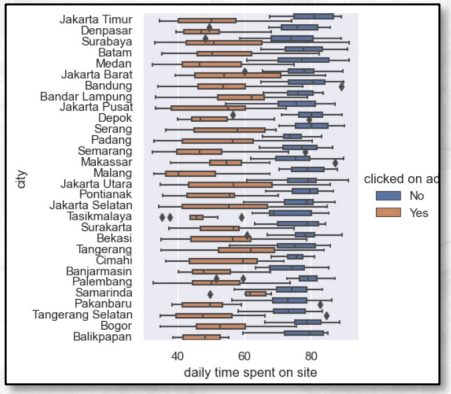
Bivariate Analysis

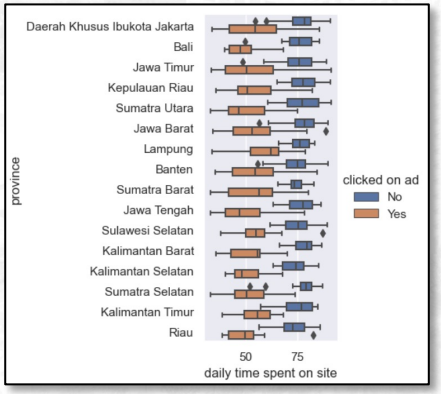
Daily time spent on site vs City, Province, Category

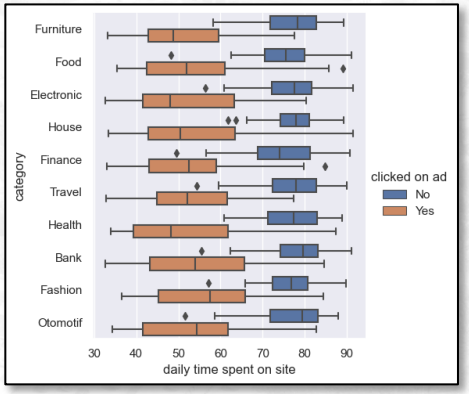






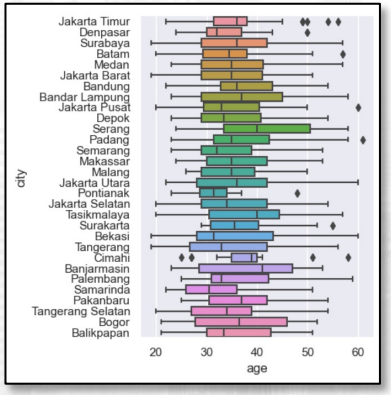


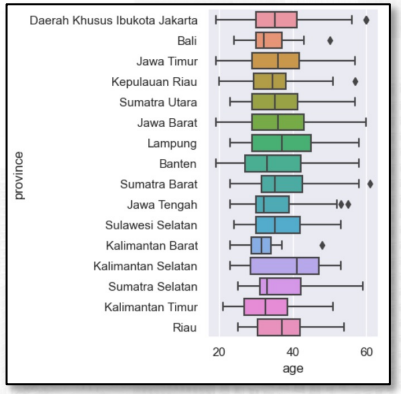


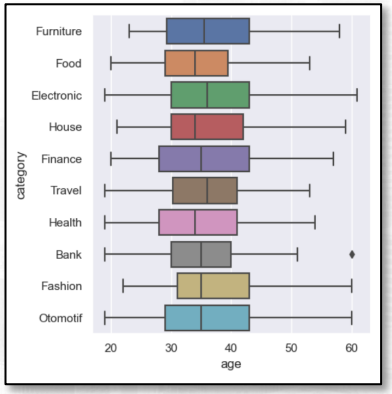


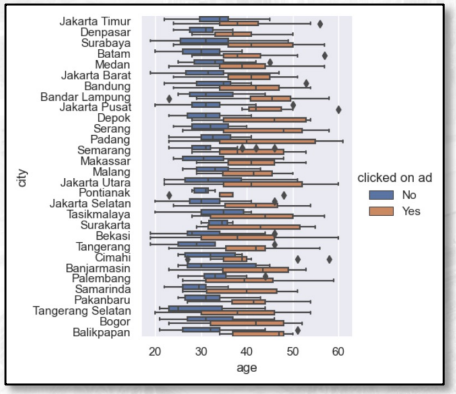
Bivariate Analysis

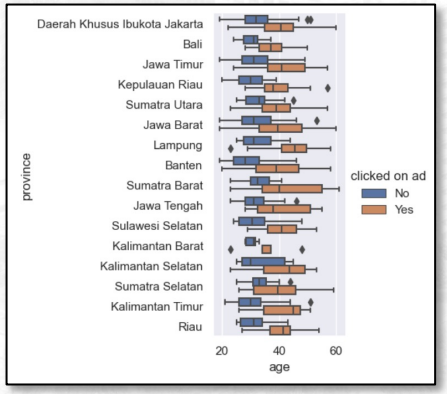
Age vs City, Province, Category

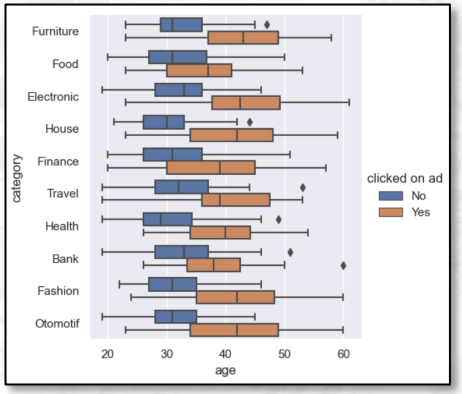






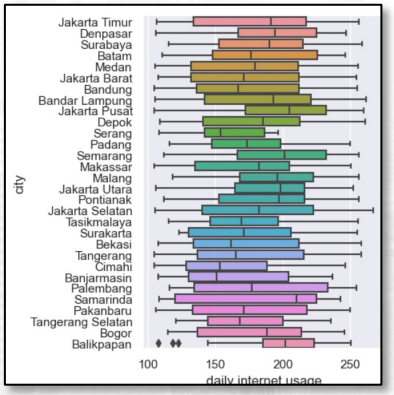


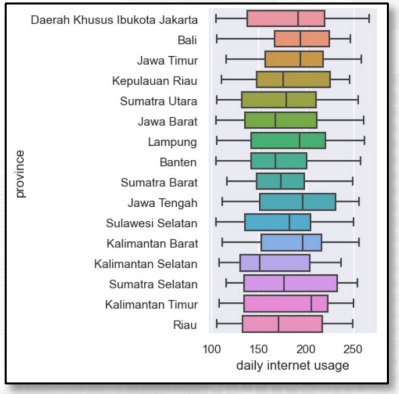


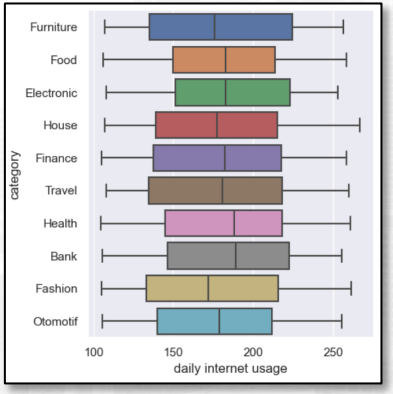


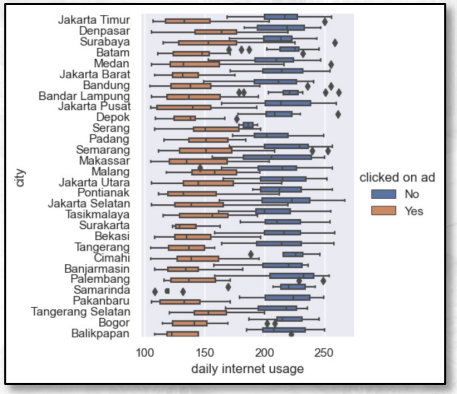
Bivariate Analysis

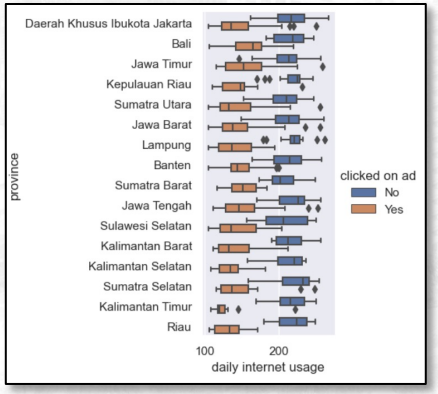
Daily internet usage vs City, Province, Category

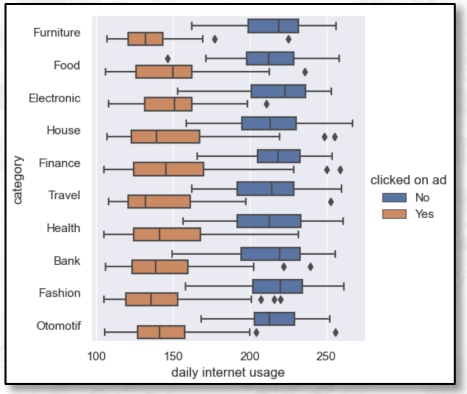




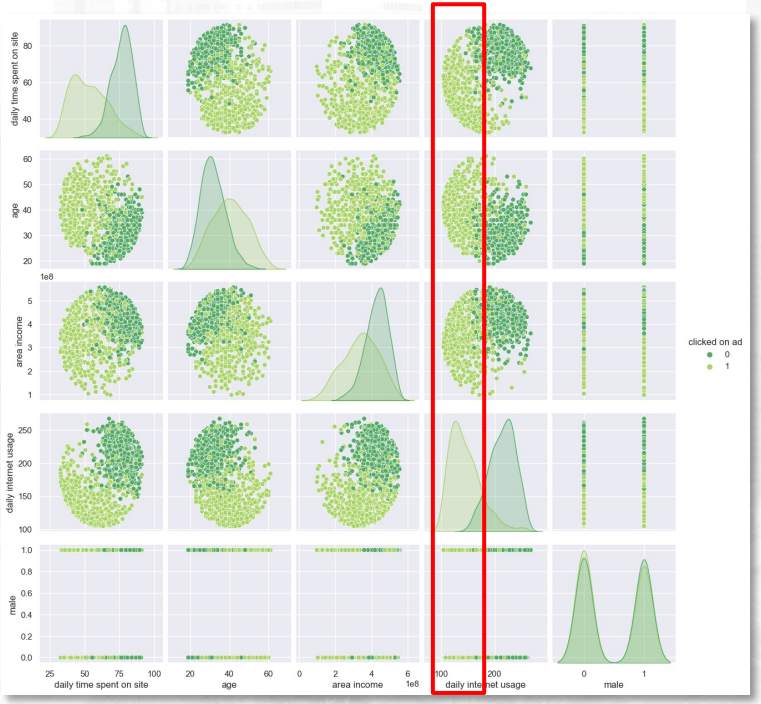








Multivariate Analysis



Target Statistics Analysis

City with clicked% ratio > 50%

	city	counts0	counts1	total city	total population	clicked%	clicked% on total counts
29	Serang	2	9	11	1000	81.82	0.9
28	Cimahi	4	13	17	1000	76.47	1.3
20	Tangerang Selatan	12	21	33	1000	63.64	2.1
23	Banjarmasin	9	14	23	1000	60.87	1.4
26	Tasikmalaya	6	9	15	1000	60.0	0.9
24	Padang	8	11	19	1000	57.89	1.1
4	Bandung	27	37	64	1000	57.81	3.7
6	Bekasi	25	31	56	1000	55.36	3.1
17	Bogor	13	15	28	1000	53.57	1.5
9	Palembang	20	22	42	1000	52.38	2.2

Target Statistics Analysis

Province with clicked% ratio > 50%

	province	counts0	counts1	total province	total population	clicked%	clicked% on total counts
13	Kalimantan Selatan	9	14	23	1000	60.87	1.4
3	Banten	31	45	76	1000	59.21	4.5
14	Sumatra Barat	8	11	19	1000	57.89	1.1
1	Jawa Barat	92	118	210	1000	56.19	11.8
7	Sumatra Selatan	20	22	42	1000	52.38	2.2
12	Riau	11	12	23	1000	52.17	1.2
5	Sumatra Utara	23	25	48	1000	52.08	2.5
9	Sulawesi Selatan	16	17	33	1000	51.52	1.7

Target Statistics Analysis

Category with clicked% ratio > 50%

	category	counts0	counts1	total category	total population	clicked%	clicked% on total counts
9	Finance	39	52	91	1000	57.14	5.2
8	Fashion	46	56	102	1000	54.9	5.6
2	Otomotif	53	59	112	1000	52.68	5.9
3	House	52	57	109	1000	52.29	5.7

Target Statistics Analysis

Category Ads with the most clicked on Province

	Jawa Barat	count
0	Food	19
1	Fashion	17
2	Otomotif	14
3	Health	13
4	House	11
5	Electronic	10
6	Travel	10
7	Finance	9
8	Furniture	9
9	Bank	6

	DKI	count
0	Otomotif	16
1	Electronic	12
2	Finance	12
3	Furniture	12
4	House	12
5	Travel	12
6	Bank	11
7	Fashion	11
8	Food	9
9	Health	9

	Banten	count
0	House	10
1	Otomotif	7
2	Fashion	5
3	Health	5
4	Travel	5
5	Furniture	4
6	Bank	3
7	Electronic	3
8	Finance	2
9	Food	1

	Jawa Timur	count
0	Travel	7
1	Food	6
2	Health	6
3	House	5
4	Fashion	4
5	Furniture	4
6	Electronic	3
7	Finance	3
8	Otomotif	2
9	Bank	1

Insight Summary

EDA :
50% clicked
50% tidak clicked

Cleaning :
- kolom **Unnamed: 0** di **drop**
- tidak ada **duplicate**
- 37 **missing value** diganti dengan **median & modus**
- **clicked on ad** diganti boolean
- **male** diganti boolean

Univariate :
- ada **outlier** bawah pada '**area income**'
- negative skewed : *daily time spent on site, area income*
- *positive skewed : age*
- normal distribution : *daily internet usage, timestamp*
- DKI Jakarta & Kalimantan Timur rata2 paling sedikit '**clicked on ad**'
- Jawa Barat paling banyak clicked on ad

Bivariate :
klik ad terjadi pada :
- **daily time spent on site** vs target : 0 - 60 menit
- **age** justru 35 ke atas
- **daily internal usage** dibawah 175 kali

Insight Summary

Multivariate :

- **Daily Internet Usage** : 0-150 paling banyak clicked ad
- **Area Income** x **Age** : dibawah 3jt paling banyak clicked ad
- **Area Income** x **daily time spent on site** : dibawah 3jt paling banyak clicked ad
- **Daily time spent on site** vs **age** : 0 - 60 banyak clicked

Correlation :

- Nilai korelasi antara "daily time spent on site" dan "clicked on ad" sebesar -0.71 menunjukkan bahwa ada korelasi negatif yang cukup kuat antara kedua variabel tersebut. yang berarti ada hubungan kuat yang cenderung negatif antara "daily time spent on site" dan "clicked on ad," ang berarti ketika salah satu variabel naik, yang lain turun secara proporsional.

Clicked Ratio on Categoricals

- Kota di jawa barat paling banyak clicked on ad
- Finance, Fashion, Otomotif, House paling banyak di clicked

Analysis Province on Ads Categoricals:

- **Jawa Barat** iklan tertinggi di klik food dan fashion
- **DKI** iklan paling tinggi otomotif
- **Banten** pada house
- **Jatim** pada travel, food

Data Cleaning & Preprocessing

- Pada tahap ***cleaning data***, tunjukan ***null*** atau ***missing value*** serta ***duplicated value*** pada dataset, serta cara penyelesaiannya.
- Tulislah pula proses **extract datetime data** sebelum dilakukan model machine learning.
- Tunjukan **Split Data** sebelum melakukan model machine learning
- Tulislah proses **feature encoding** pada tahap ini (gunakan get_dummy)
- **Source code** yang sudah kamu buat, dapat ditampilkan dan berikan link untuk mengakses file tersebut. Contohnya seperti di pojok kanan bawah.

https://drive.google.com/file/d/19OlSDAJhHGEd8Y5EUvAtTKIcejdRvBiH/view?usp=share_link

Data Cleaning

Total Row	Columns	Duplicated	Unwanted	Missing Value
• 1000	• 11	• 0	• Unnamed: 0	Daily Time Spent on Site (13) Area Income (13) Daily Internet Usage (11) Male (3)

Coding lengkap dapat dilihat di :
https://drive.google.com/file/d/19OlSDAJhHGEd8Y5EUvAtTKIcejdRvBiH/view?usp=share_link

Data Cleaning

1 Mengisi **Missing Value**

```
#statistik df

df_stat = df.describe().round(2)

median = df[nums].median().to_frame().T
median.rename(index={0:'median'}, inplace=True)


modus = df[nums].mode().iloc[0].to_frame().T
modus.rename(index={0:'modus'}, inplace=True)

df_stat = pd.concat([df_stat, median, modus], axis=0).round(2)

df_stat
```

```
df['male'].fillna('Perempuan', inplace=True)
```

2 Menghapus kolom **'Unnamed: 0'**

```
# drop Unnamed 0
df.drop('Unnamed: 0', axis=1, inplace=True)

# lowercase
df.columns = df.columns.str.lower()
```

3 Rename tipe data '**male**' & '**timestamp**'

```
df['timestamp'] = df['timestamp'].astype('datetime64')
df['male'] = df['male'].astype('int64')
```

4 Mengganti value data '**clicked on ad**' dan '**male**'

```
# rubah value
df['clicked on ad'] = df['clicked on ad'].replace({'Yes': 1, 'No': 0})
df['male'] = df['male'].replace({'Laki-Laki': 1, 'Perempuan': 0})
```

Untuk selengkapnya, dapat melihat jupyter notebook disini

Data Cleaning

TRANSFORMASI DATAFRAME

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 11 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Unnamed: 0                1000 non-null   int64
 1   Daily Time Spent on Site  987 non-null    float64
 2   Age                       1000 non-null   int64
 3   Area Income               987 non-null    float64
 4   Daily Internet Usage      989 non-null    float64
 5   Male                      997 non-null    object
 6   Timestamp                 1000 non-null   object
 7   Clicked on Ad             1000 non-null   object
 8   city                      1000 non-null   object
 9   province                  1000 non-null   object
 10  category                  1000 non-null   object
dtypes: float64(3), int64(2), object(6)
memory usage: 86.1+ KB
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 10 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   daily time spent on site  1000 non-null   float64
 1   age                       1000 non-null   int64
 2   area income               1000 non-null   float64
 3   daily internet usage      1000 non-null   float64
 4   male                      1000 non-null   int64
 5   timestamp                 1000 non-null   datetime64[ns]
 6   clicked on ad             1000 non-null   int64
 7   city                      1000 non-null   object
 8   province                  1000 non-null   object
 9   category                  1000 non-null   object
dtypes: datetime64[ns](1), float64(3), int64(3), object(3)
memory usage: 78.2+ KB
```

Untuk selengkapnya, dapat melihat jupyter notebook disini

Extract Datetime

```python
# Extract df['timestamp'] into new coloumns

df['date'] = df['timestamp'].dt.day
df['month'] = df['timestamp'].dt.month
df['months'] = df['month'].replace({1:'January',
                                    2:'February',
                                    3:'March',
                                    4:'April',
                                    5:'May',
                                    6:'June',
                                    7:'July',
                                    8:'August',
                                    9:'September',
                                    10:'October',
                                    11:'November',
                                    12:'December'
                                   })
df['year'] = df['timestamp'].dt.year
df['hour'] = df['timestamp'].dt.hour
df['minute'] = df['timestamp'].dt.minute
df['week'] = df['timestamp'].dt.isocalendar().week
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 17 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   daily time spent on site  1000 non-null   float64
 1   age                       1000 non-null   int64
 2   area income               1000 non-null   float64
 3   daily internet usage      1000 non-null   float64
 4   male                      1000 non-null   int64
 5   timestamp                 1000 non-null   datetime64[ns]
 6   clicked on ad             1000 non-null   int64
 7   city                      1000 non-null   object
 8   province                  1000 non-null   object
 9   category                  1000 non-null   object
 10  date                      1000 non-null   int64
 11  month                     1000 non-null   int64
 12  months                    1000 non-null   object
 13  year                      1000 non-null   int64
 14  hour                      1000 non-null   int64
 15  minute                    1000 non-null   int64
 16  week                      1000 non-null   int64
dtypes: datetime64[ns](1), float64(3), int64(9), object(4)
memory usage: 132.9+ KB
```

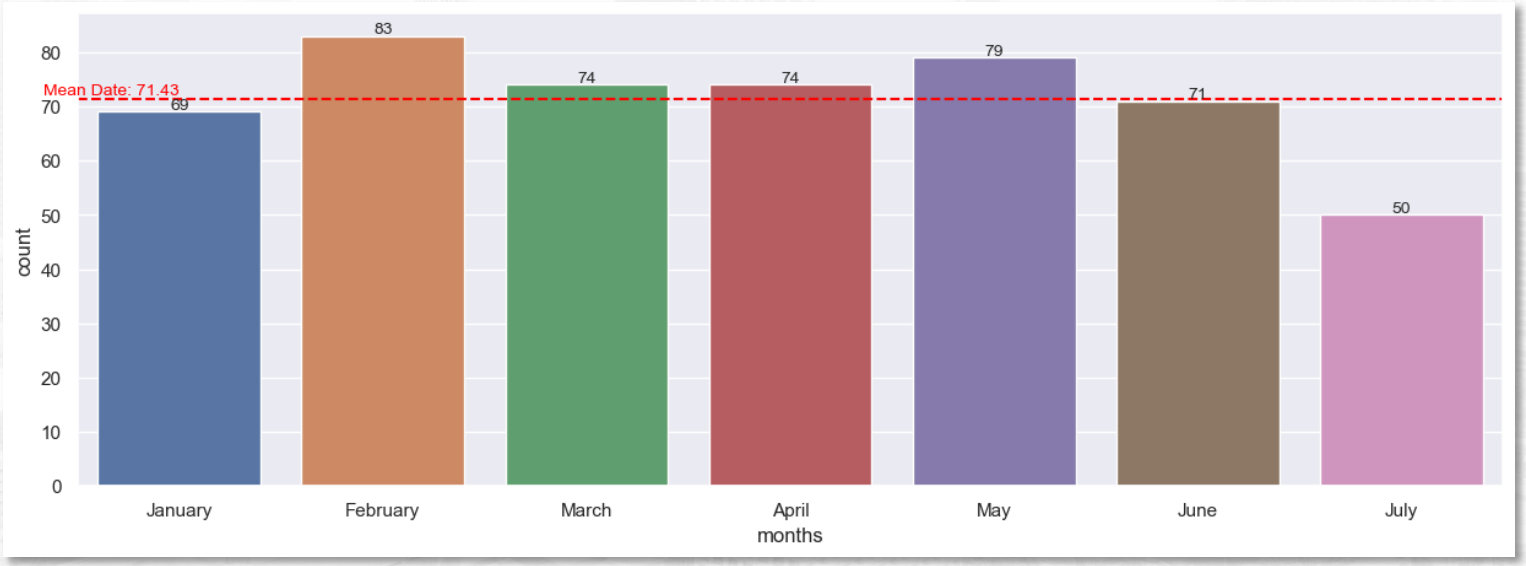
Extract Datetime

	daily time spent on site	age	area income	daily internet usage	male	timestamp	clicked on ad	city	province	category	date	month	months	year	hour	minute	week
0	68.95	35	432,837,300.0	256.09	0	2016-03-27 00:53:00	0	Jakarta Timur	Daerah Khusus Ibukota Jakarta	Furniture	27	3	March	2016	0	53	12
1	80.23	31	479,092,950.0	193.77	1	2016-04-04 01:39:00	0	Denpasar	Bali	Food	4	4	April	2016	1	39	14
2	69.47	26	418,501,580.0	236.5	0	2016-03-13 20:35:00	0	Surabaya	Jawa Timur	Electronic	13	3	March	2016	20	35	10
3	74.15	29	383,643,260.0	245.89	1	2016-01-10 02:31:00	0	Batam	Kepulauan Riau	House	10	1	January	2016	2	31	1
4	68.37	35	517,229,930.0	225.58	0	2016-06-03 03:36:00	0	Medan	Sumatra Utara	Finance	3	6	June	2016	3	36	22

Dengan telah di *extract-nya* **'timestamp'**
Bisa dianalisa dengan countplot untuk melihat trend **'clicked on ad'**

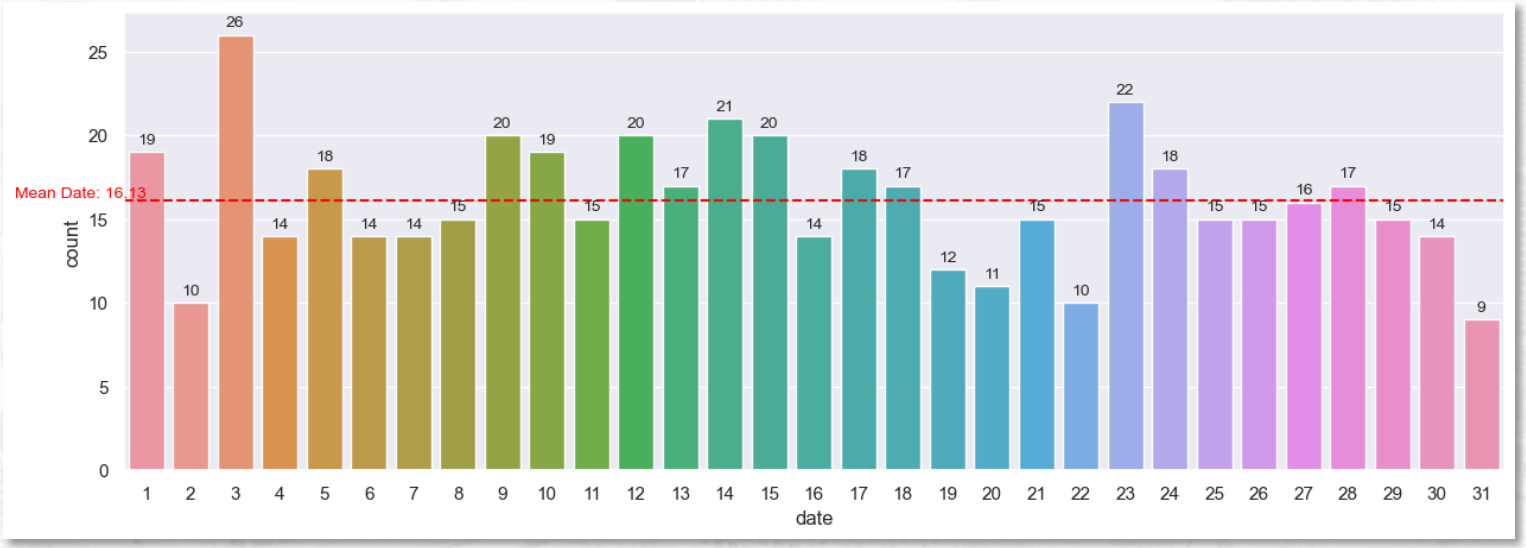
Timestamp Analysis

months to ***clicked on ad***



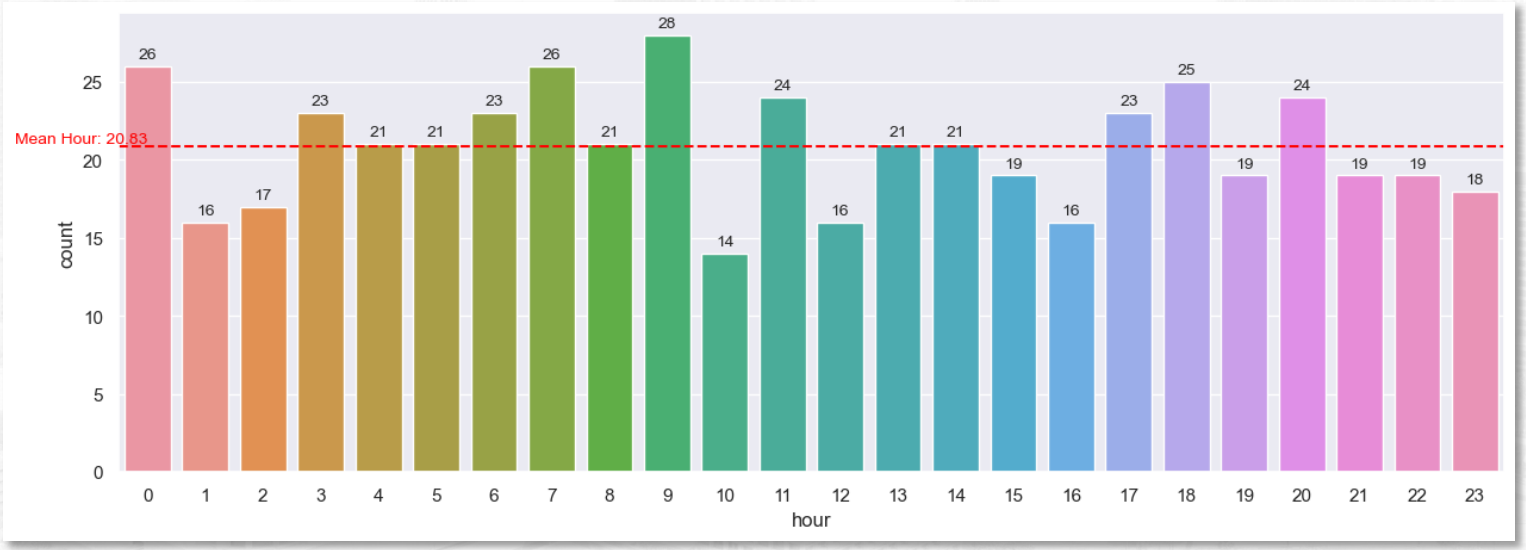
Timestamp Analysis

date to ***clicked on ad***



Timestamp Analysis

hours to ***clicked on ad***



Timestamp Analysis

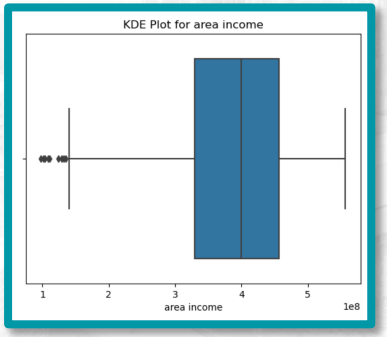
Months:
Kecendrungan Clicked on Ad tinggi pada February – May berada diatas rata-rata.

Date :
17 hari dalam sebulan berada dibawah rata-rata

Hours :
Jam 3am – 9am,1700 – 1800 dan 2000 adalah paling banyak '*clicked on ad*'

Handling Outliers

	count	mean	std	min	25%	50%	75%	max	iqr	low_thres	hi_thres	low_tr/fl	hi_tr/fl
daily time spent on site	991.0	65.03	15.79	32.6	51.53	68.11	78.39	91.43	26.86	11.240000000000002	118.68	False	False
age	991.0	35.99	8.8	19.0	29.0	35.0	42.0	61.0	13.0	9.5	61.5	False	False
area income	991.0	387,493,474.45	90,287,564.16	139,942,040.0	331,622,690.0	399,068,320.0	458,091,585.0	556,393,600.0	126,468,895.0	141,919,347.5	647,794,927.5	True	False
daily internet usage	991.0	179.88	43.78	104.78	138.79	182.65	218.55	267.01	79.76000000000002	19.149999999999963	338.19000000000005	False	False
male	991.0	0.48	0.5	0.0	0.0	0.0	1.0	1.0	1.0	-1.5	2.5	False	False
clicked on ad	991.0	0.5	0.5	0.0	0.0	0.0	1.0	1.0	1.0	-1.5	2.5	False	False
date	991.0	15.47	8.74	1.0	8.0	15.0	23.0	31.0	15.0	-14.5	45.5	False	False
month	991.0	3.81	1.93	1.0	2.0	4.0	5.0	7.0	3.0	-2.5	9.5	False	False
year	991.0	2,016.0	0.0	2,016.0	2,016.0	2,016.0	2,016.0	2,016.0	0.0	2,016.0	2,016.0	False	False
hour	991.0	11.65	6.97	0.0	6.0	12.0	18.0	23.0	12.0	-12.0	36.0	False	False
minute	991.0	29.13	17.26	0.0	14.0	30.0	43.5	59.0	29.5	-30.25	87.75	False	False
week	991.0	15.38	9.73	1.0	8.0	14.0	22.0	53.0	14.0	-13.0	43.0	False	True




```
print(f'Jumlah Outlier pada Area Income :',dx[dx['area income'] <= 141919347].shape[0],'baris')
```

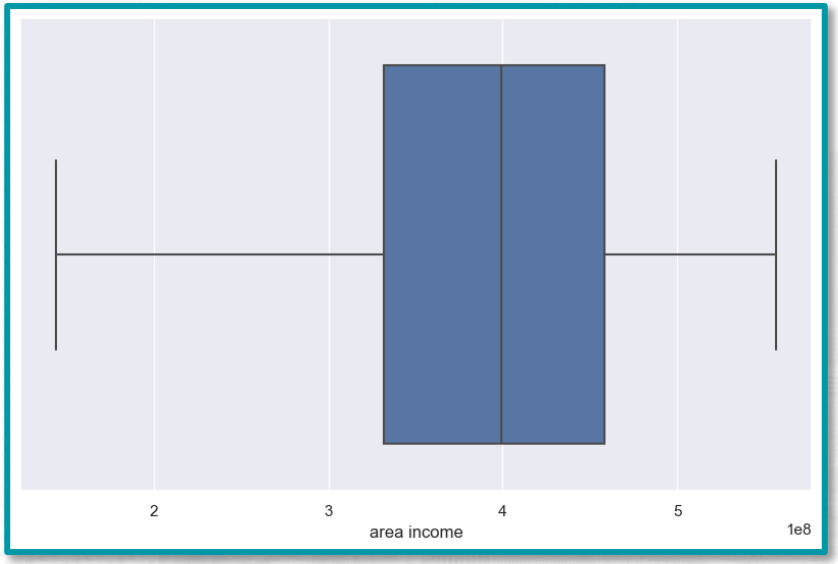
```
Jumlah Outlier pada Area Income : 10 baris
```

Handling Outliers

Penanganan outliers pada kolom **'area income**' memiliki manfaat signifikan dalam analisis data.

Dalam konteks ini, telah dilakukan filtering untuk mengidentifikasi dan menghapus 10 baris data yang mengandung outliers.

Akibatnya, total jumlah baris dalam DataFrame tanpa outliers menjadi 990 baris.




```
do = do[do['area income'] >= 141919347]
do.shape[0]
```

```
990
```

Rakamin Academy

OHE Pada Feature ‘city’

OHE City

```
dh = pd.get_dummies(di['city'], prefix_sep='_', prefix='city')
dh.head()
```

	city_Balikpapan	city_Bandar Lampung	city_Bandung	city_Banjarmasin	city_Batam	city_Bekasi	city_Bogor	city_Cimahi	city	karta Pusat
0	0	0	0	0	0	0	0	0		0
1	0	0	0	0	0	0	0	0		0
2	0	0	0	0	0	0	0	0		0
3	0	0	0	0	1	0	0	0		0
4	0	0	0	0	0	0	0	0		0

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 990 entries, 0 to 999
Data columns (total 46 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   daily time spent on site  990 non-null    float64
 1   age                       990 non-null    int64
 2   area income               990 non-null    float64
 3   daily internet usage      990 non-null    float64
 4   male                      990 non-null    int64
 5   clicked on ad             990 non-null    int64
 6   city                      990 non-null    object
 7   province                  990 non-null    object
 8   category                  990 non-null    object
 9   date                      990 non-null    int64
 10  month                     990 non-null    int64
 11  months                    990 non-null    object
 12  year                      990 non-null    int64
 13  hour                      990 non-null    int64
 14  minute                    990 non-null    int64
 15  week                      990 non-null    int64
 16  city_Balikpapan           990 non-null    uint8
 17  city_Bandar Lampung       990 non-null    uint8
 18  city_Bandung              990 non-null    uint8
 19  city_Banjarmasin          990 non-null    uint8
 20  city_Batam                990 non-null    uint8
 21  city_Bekasi               990 non-null    uint8
 22  city_Bogor                990 non-null    uint8
 23  city_Cimahi               990 non-null    uint8
 24  city_Denpasar             990 non-null    uint8
 25  city_Depok                990 non-null    uint8
 26  city_Jakarta Barat        990 non-null    uint8
 27  city_Jakarta Pusat        990 non-null    uint8
 28  city_Jakarta Selatan      990 non-null    uint8
 29  city_Jakarta Timur        990 non-null    uint8
 30  city_Jakarta Utara        990 non-null    uint8
 31  city_Makassar             990 non-null    uint8
 32  city_Malang               990 non-null    uint8
 33  city_Medan                990 non-null    uint8
 34  city_Padang               990 non-null    uint8
 35  city_Pakanbaru            990 non-null    uint8
 36  city_Palembang            990 non-null    uint8
 37  city_Pontianak            990 non-null    uint8
 38  city_Samarinda            990 non-null    uint8
 39  city_Semarang             990 non-null    uint8
 40  city_Serang               990 non-null    uint8
 41  city_Surabaya             990 non-null    uint8
 42  city_Surakarta            990 non-null    uint8
 43  city_Tangerang            990 non-null    uint8
 44  city_Tangerang Selatan    990 non-null    uint8
 45  city_Tasikmalaya          990 non-null    uint8
dtypes: float64(3), int64(9), object(4), uint8(30)
memory usage: 192.8+ KB
```

Rakamin Academy

Feature Encoding

Feature Encoding

```
from sklearn.feature_selection import mutual_info_classif
from sklearn.preprocessing import LabelEncoder

# define column has type Object to implement LabelEncoder
list_object = dh.select_dtypes("object").columns.tolist()
encoder = LabelEncoder()

for col in list_object:
    dh[col] = encoder.fit_transform(di[col].values.tolist())
```

Setelah dilakukan feature encoding maka bisa dilakukan Feature Selection menggunakan Mutual Information.

Feature ‘city’ di drop pada step berikutnya

```
dh = dh[['area income', 'daily internet usage', 'daily time spent on site',
       'age', 'minute', 'date', 'week', 'hour', 'province', 'category',
       'city_Jakarta Pusat', 'city_Cimahi', 'city_Serang', 'city_Malang',
       'city_Balikpapan', 'city_Tangerang Selatan', 'clicked on ad']]
```

Feature Selection

	index	MI Scores
0	area income	0.6834388781607652
1	daily internet usage	0.4881965709138747
2	daily time spent on site	0.39563970665769665
3	age	0.16134769205387492
4	minute	0.028794744293678162
5	city	0.01717555083508341
6	date	0.014052014559409333
7	week	0.008980605797282569
8	hour	0.008463198710795483
9	province	0.007365896201672998
10	category	0.003381246583892425
11	city_Jakarta Pusat	0.0032500626291318627
12	city_Cimahi	0.0026701100301096646
13	city_Serang	0.0020278380898220896
14	city_Malang	0.0019424593658465966
15	city_Balikpapan	0.0018111520427846715
16	city_Tangerang Selatan	0.0009064236690843955

Split Data Feature & Target

Standard Scaler

```
scaler = StandardScaler()
ds[featsel] = scaler.fit_transform(ds[featsel])
```

```
X = ds[featsel].drop('clicked on ad', axis=1)
y = ds['clicked on ad']
```

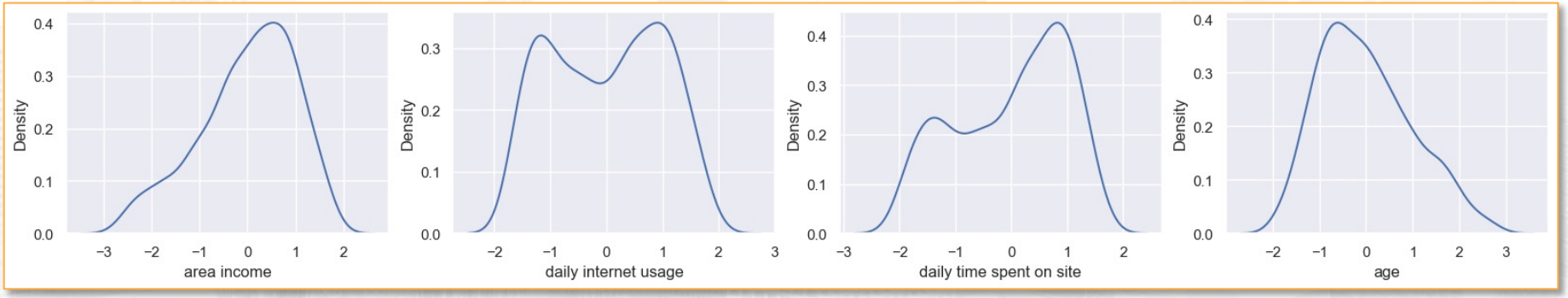
```
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.25,random_state=42)
```

```
...
Total X Train : 742
Total y Train : 742
...
Total X Test : 248
Total y Test : 248
```

Standard Scaler digunakan karena :

1. Outlier sudah dibersihkan
2. Cocok untuk algoritma yang mengasumsikan distribusi normal, seperti banyak model statistik, PCA, atau algoritma umum lainnya.
3. Mengubah data sehingga memiliki rata-rata nol dan deviasi standar satu.
4. Bisa dibilang paling fleksibel dan cukup sesuai dengan kebutuhan Modeling

Split Data Feature & Target



KDE plot pada feature diatas merefleksikan bahwa nilai ‘mean’ = 0

Data Modeling

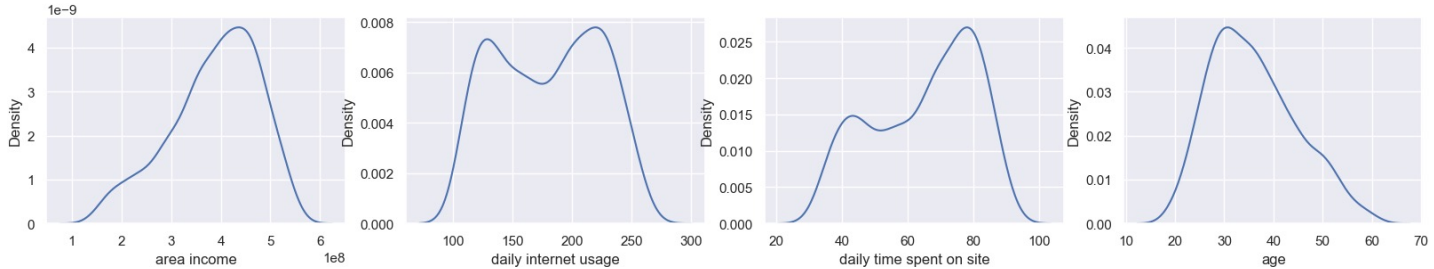
- Tulislah proses model machine learning terdiri dari
    a. **hasil *experiment 1*** (sebelum normalisasi/standardisasi),
    b. **hasil *experiment 2*** (setelah normalisasi/standardisasi).
    c. hasil tabel **confusion matrix** dari model tersebut.
    d. Daftar ***Feature Important***.

- Tulislah hasil interpretasi dari model tersebut

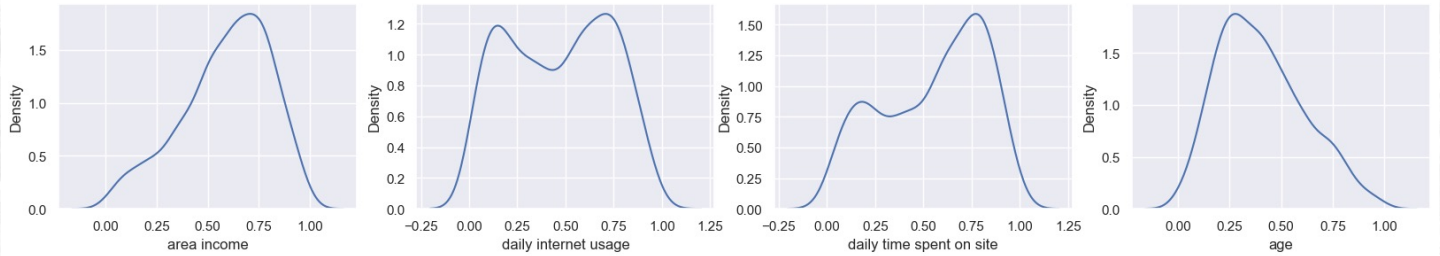
Untuk selengkapnya, dapat melihat jupyter notebook disini
https://drive.google.com/file/d/1-2C3FObN2vFwq2sTeMJTyIrmCkCpDtaF/view?usp=share_link

Komparasi Distribusi

Tanpa Scaler



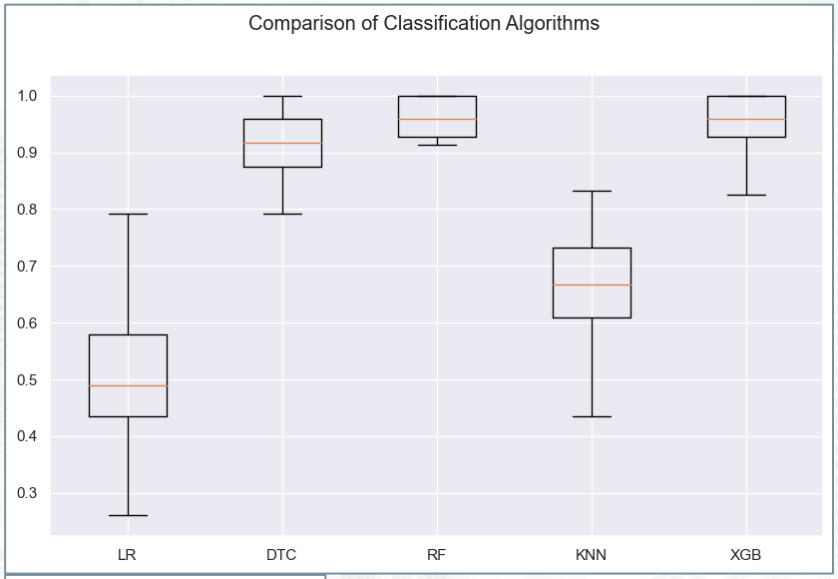
Dengan MinMaxScaler



Untuk selengkapnya, dapat melihat jupyter notebook disini
https://drive.google.com/file/d/1-2C3FObN2vFwq2sTeMJTyIrmCkCpDtaF/view?usp=share_link

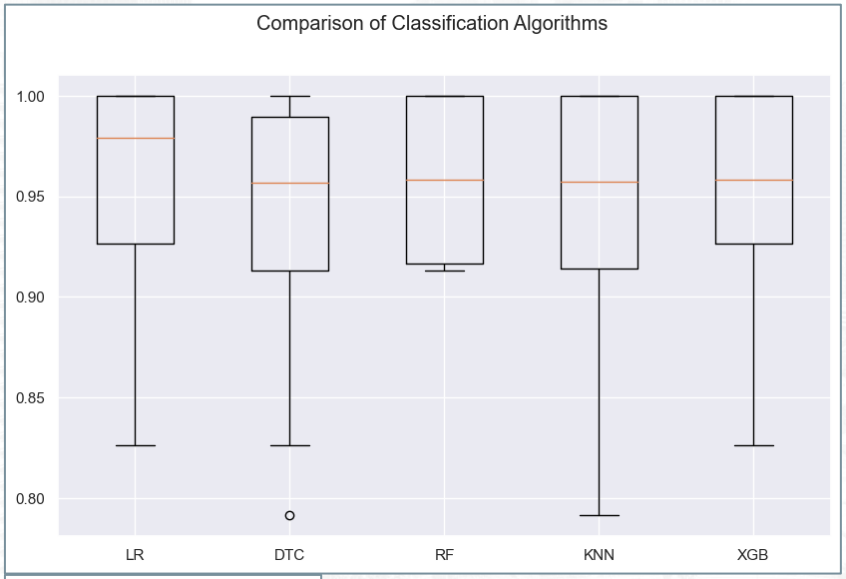
Komparasi Boxplot

Tanpa Scaler




```
LR: 0.504745 (0.109503)
DTC: 0.926285 (0.054868)
RF: 0.962560 (0.033817)
KNN: 0.656099 (0.097412)
XGB: 0.961482 (0.041719)
```

Dengan Scaler




```
LR: 0.964458 (0.042687)
DTC: 0.935257 (0.055029)
RF: 0.959498 (0.037071)
KNN: 0.946256 (0.053435)
XGB: 0.961482 (0.041719)
```

Rakamin Academy

Komparasi Confussion Matrix LogisticRegression

Tanpa Scaler

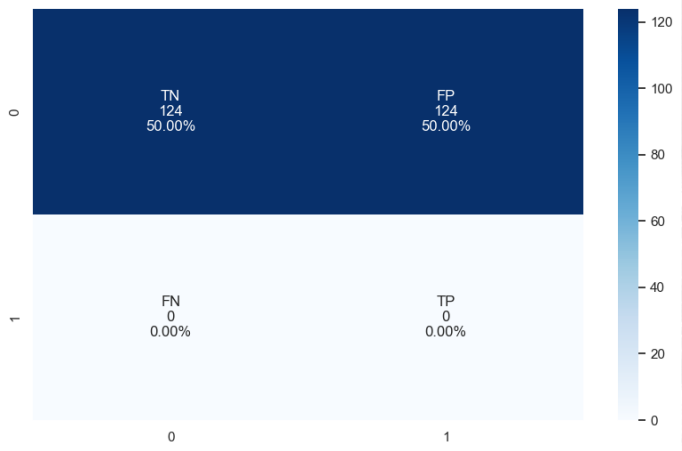
LogisticRegression()

```
Logistic regression accuracy: 0.5

Confusion Matrix:
[[124 124]
 [  0   0]]
              precision    recall  f1-score   support

           0       0.50      1.00      0.67       124
           1       0.00      0.00      0.00       124

    accuracy                           0.50       248
   macro avg       0.25      0.50      0.33       248
weighted avg       0.25      0.50      0.33       248
```



Dengan Scaler

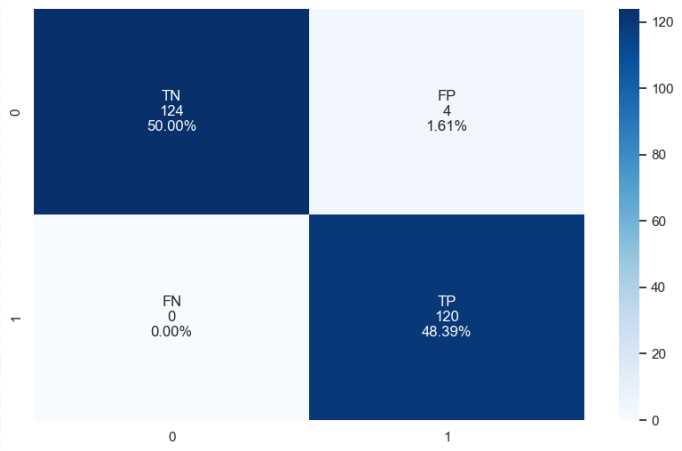
LogisticRegression()

```
Logistic regression accuracy: 0.9838709677419355

Confusion Matrix:
[[124   4]
 [  0 120]]
              precision    recall  f1-score   support

           0       0.97      1.00      0.98       124
           1       1.00      0.97      0.98       124

    accuracy                           0.98       248
   macro avg       0.98      0.98      0.98       248
weighted avg       0.98      0.98      0.98       248
```



Rakamin Academy

Komparasi Confussion Matrix DecisionTreeClassification

Tanpa Scaler

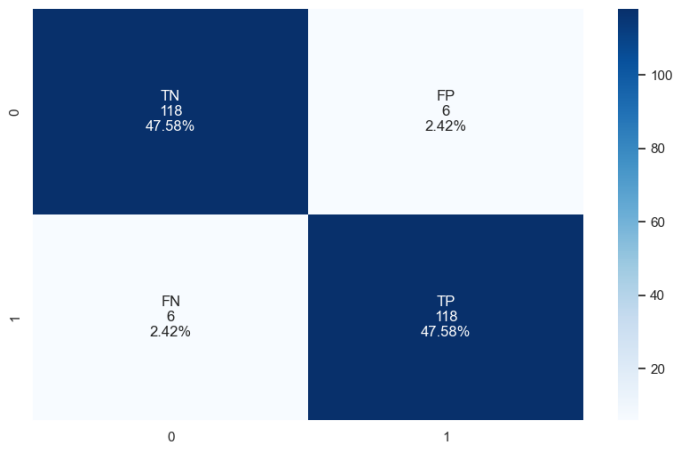
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()

DecisionTreeClassifier accuracy: 0.9516129032258065

Confusion Matrix:
[[118   6]
 [  6 118]]
              precision    recall  f1-score   support

           0       0.95      0.95      0.95       124
           1       0.95      0.95      0.95       124

    accuracy                           0.95       248
   macro avg       0.95      0.95      0.95       248
weighted avg       0.95      0.95      0.95       248
```



Dengan Scaler

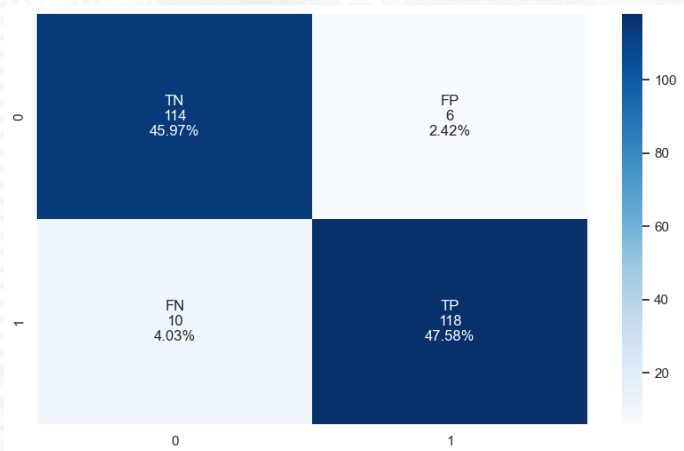
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()

DecisionTreeClassifier accuracy: 0.9354838709677419

Confusion Matrix:
[[114   6]
 [ 10 118]]
              precision    recall  f1-score   support

           0       0.95      0.92      0.93       124
           1       0.92      0.95      0.94       124

    accuracy                           0.94       248
   macro avg       0.94      0.94      0.94       248
weighted avg       0.94      0.94      0.94       248
```



Rakamin Academy

Komparasi Confussion Matrix RandomForest

Tanpa Scaler

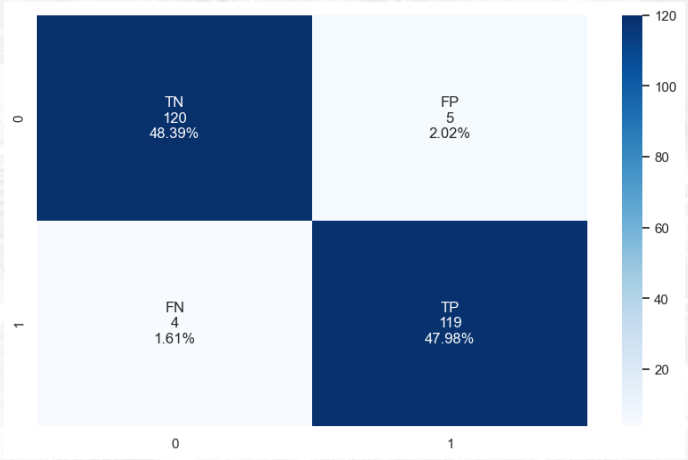
▾ RandomForestClassifier
RandomForestClassifier()

```
RandomForestClassifier accuracy: 0.9637096774193549

Confusion Matrix:
[[120   5]
 [  4 119]]
              precision    recall  f1-score   support

           0       0.96      0.97      0.96       124
           1       0.97      0.96      0.96       124

    accuracy                           0.96       248
   macro avg       0.96      0.96      0.96       248
weighted avg       0.96      0.96      0.96       248
```



Dengan Scaler

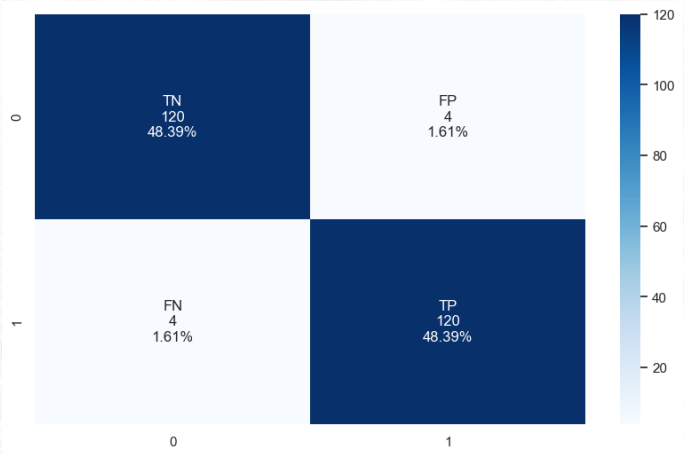
▾ RandomForestClassifier
RandomForestClassifier()

```
RandomForest accuracy: 0.967741935483871

Confusion Matrix:
[[120   4]
 [  4 120]]
              precision    recall  f1-score   support

           0       0.97      0.97      0.97       124
           1       0.97      0.97      0.97       124

    accuracy                           0.97       248
   macro avg       0.97      0.97      0.97       248
weighted avg       0.97      0.97      0.97       248
```



Komparasi Confussion Matrix KNN

Tanpa Scaler

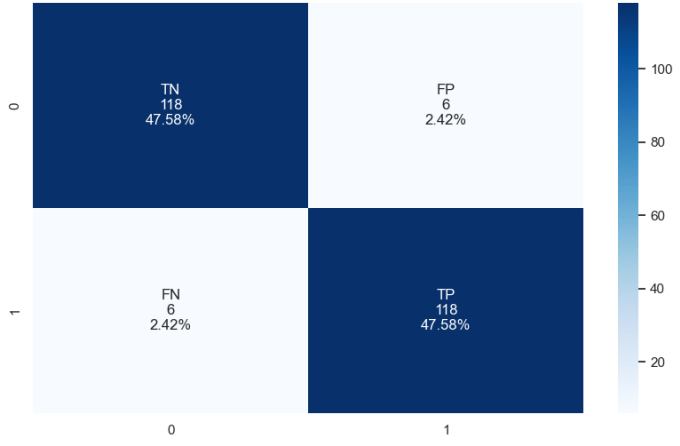
```
KNeighborsClassifier()
```

```
KNeighborsClassifier accuracy: 0.657258064516129

Confusion Matrix:
[[118   6]
 [  6 118]]
              precision    recall  f1-score   support

           0       0.63      0.76      0.69       124
           1       0.70      0.56      0.62       124

    accuracy                           0.66       248
   macro avg       0.66      0.66      0.65       248
weighted avg       0.66      0.66      0.65       248
```



Dengan Scaler

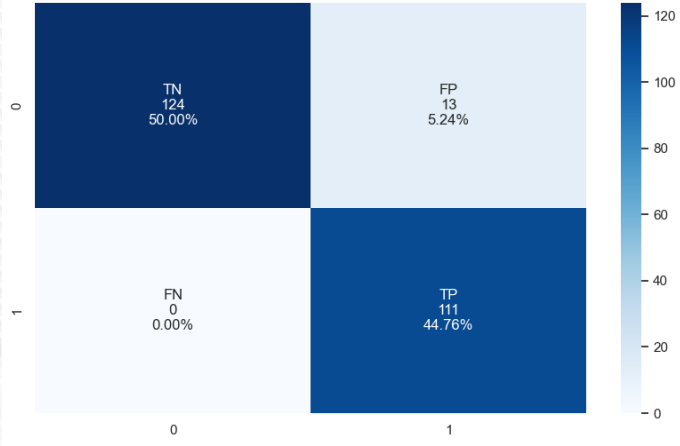
```
KNeighborsClassifier()
```

```
KNN accuracy: 0.9475806451612904

Confusion Matrix:
[[124  13]
 [  0 111]]
              precision    recall  f1-score   support

           0       0.91      1.00      0.95       124
           1       1.00      0.90      0.94       124

    accuracy                           0.95       248
   macro avg       0.95      0.95      0.95       248
weighted avg       0.95      0.95      0.95       248
```



Rakamin Academy

Komparasi Confussion Matrix XGB

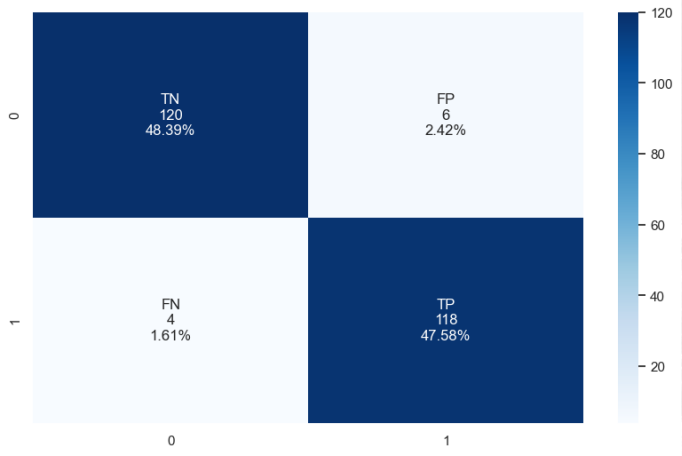
Tanpa Scaler

```
XGBClassifier accuracy: 0.9596774193548387

Confusion Matrix:
[[120   6]
 [  4 118]]
              precision    recall  f1-score   support

           0       0.95      0.97      0.96       124
           1       0.97      0.95      0.96       124

    accuracy                           0.96       248
   macro avg       0.96      0.96      0.96       248
weighted avg       0.96      0.96      0.96       248
```



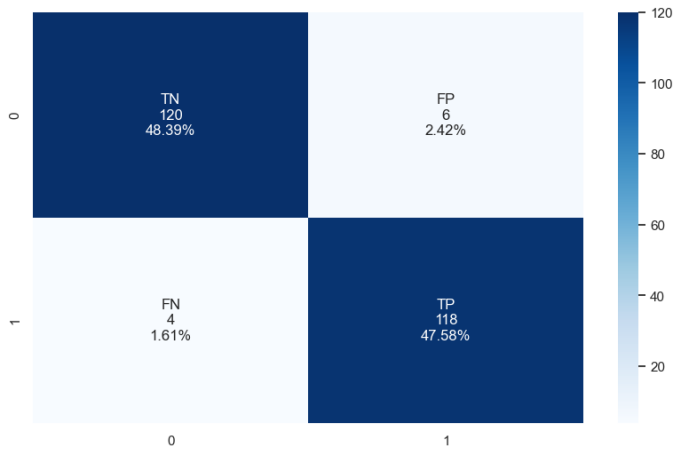
Dengan Scaler

```
XGBClassifier accuracy: 0.9596774193548387

Confusion Matrix:
[[120   6]
 [  4 118]]
              precision    recall  f1-score   support

           0       0.95      0.97      0.96       124
           1       0.97      0.95      0.96       124

    accuracy                           0.96       248
   macro avg       0.96      0.96      0.96       248
weighted avg       0.96      0.96      0.96       248
```



Predictive Model

Why i choose XGB over the rest?

XGBoost dipilih untuk menjadi *Predictive Model* dikarenakan konsistensinya baik tanpa maupun dengan *scaler,* walaupun dalam plot terlihat perbedaan distribusi.

Sebagai keluarga dari *Ensemble, Tree dan Gradient*, **XGBoost** memiliki akurasi yang lebih *reliable* dan *robust* karena menggabungkan kelebihan dari beberapa model dan menghapus kekurangannya.

Kelebihan lainnya **XGBoost** juga memiliki L1 dan L2 yang bisa mencegahnya overfitting, memiliki *Feature Importance, Handling Missing Value, Wide-Range-Used* dan kompatibel dengan *Hyperparameter*.

Feature Importance XGB

	Feature	Importance
0	daily internet usage	0.6048209071159363
1	daily time spent on site	0.18079645931720734
2	area income	0.05553075671195984
3	age	0.040533870458602905
4	province	0.03080485761165619
5	category	0.022497721016407013
6	hour	0.018578359857201576
7	minute	0.01834065653383732
8	date	0.014399674721062183
9	week	0.013696707785129547

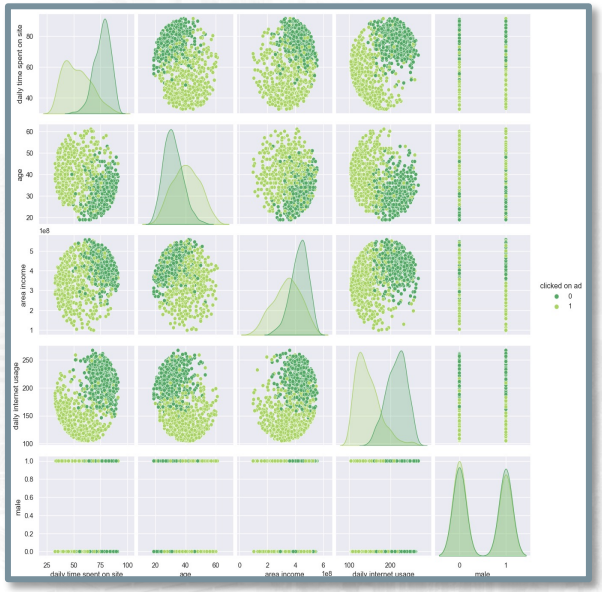
Feature Importance XGB & Mutual Information DF

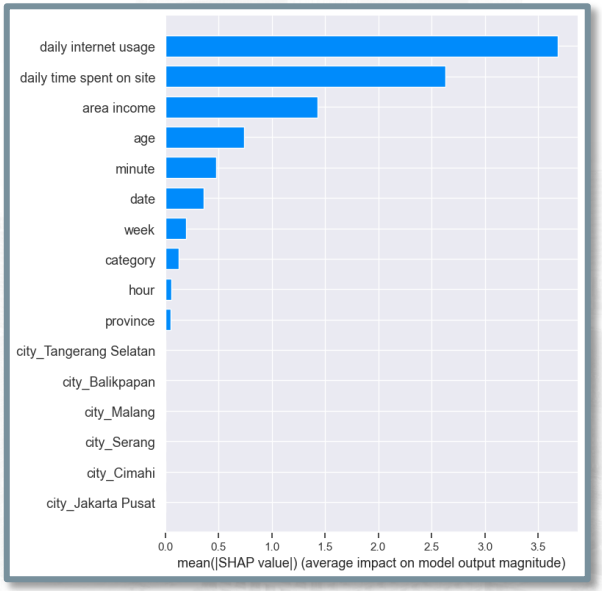
	Feature	Importance
0	daily internet usage	0.6048209071159363
1	daily time spent on site	0.18079645931720734
2	area income	0.05553075671195984
3	age	0.040533870458602905
4	province	0.03080485761165619
5	category	0.022497721016407013
6	hour	0.018578359857201576
7	minute	0.01834065653383732
8	date	0.014399674721062183
9	week	0.013696707785129547

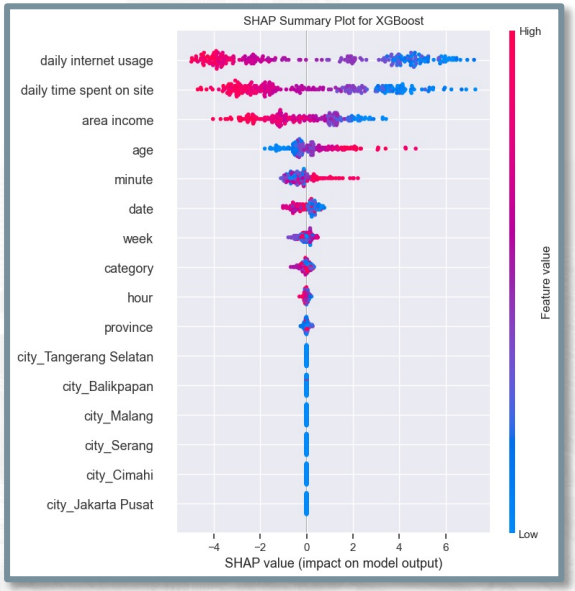
	MI Scores
area income	0.6834388781607652
daily internet usage	0.4881965709138747
daily time spent on site	0.39563970665769665
age	0.16134769205387492
minute	0.028794744293678162
city	0.01717555083508341
date	0.014052014559409333
week	0.008980605797282569
hour	0.008463198710795483
province	0.007365896201672998

Mutual Information adalah library python yang bisa digunakan untuk memudahkan **Feature Selection** sehingga bisa meminimalisir *Kutukan Dimensi* pada saat **OHE**.
MI bisa dilakukan kapan saja, **FI** setelah proses **Modeling** dan tidak semua model memiliki feature FI

Feature Importance XGB







Sebagaimana analisis sebelumnya pada pairplot, ada kesamaan bahwa pengguna yang ‘clicked on ad’ adalah yang durasi internetnya tidak lama (daily time spent on site) dan tidak sering (daily internet usage), usia relatif dewasa, dan ‘area income’ mayoritas tidak tinggi.

Interpretasi

	Feature	Importance
0	daily internet usage	0.6048209071159363
1	daily time spent on site	0.18079645931720734
2	area income	0.05553075671195984
3	age	0.040533870458602905
4	province	0.03080485761165619
5	category	0.022497721016407013
6	hour	0.018578359857201576
7	minute	0.01834065653383732
8	date	0.014399674721062183
9	week	0.013696707785129547

Setelah menggunakan XGBoost, Predictve Model Algoritma yang robust dan melihat nilai-nilainya, maka bisa disimpulkan sebagai berikut :

1. Fitur ‘daily internet usage’ adalah yang paling penting dengan dampak positif signifikan.

2. Fitur lain seperti ‘daily time spent on site’ di situs juga penting, tetapi kurang signifikan, skornya drop terlalu jauh dibanding dengan ‘daily internet usage’.

3. ‘area income’, ‘age’, ‘province’, dan (ads) ‘category’ juga berkontribusi dengan tingkat penting yang berbeda pada prediksi. ‘hours’ memiliki pengaruh sedang pada prediksi.

4. Oleh karena itu yang menjadi target utama adalah ‘daily internet usage’.

Business Recommendation & Simulation

- Tampilkan ***Feature Important*** dari hasil model machine learning
- Tulislah **rekomendasi bisnis** berdasarkan EDA dan Feature Important
- Tulislah sebuah simulasi perusahaan dalam marketing yang menunjukan ***cost, revenue, dan profit*** **sebelum dan setelah menggunakan model machine learning**. Tunjukan perbedaan dari kedua simulasi tersebut.
- Tulislah pula **simpulan** yang didapat dari proses tersebut

https://drive.google.com/file/d/16zKvTaMmJ-jJQFZ7-XysuL4TZ-sThjlT/view?usp=sharing

Business Recommendation & Simulation

Rakamin Academy

Feature Importance XGBoost

	Feature	Importance
0	daily internet usage	0.6048209071159363
1	daily time spent on site	0.18079645931720734
2	area income	0.05553075671195984
3	age	0.040533870458602905
4	province	0.03080485761165619
5	category	0.022497721016407013
6	hour	0.018578359857201576
7	minute	0.01834065653383732
8	date	0.014399674721062183
9	week	0.013696707785129547

Narasi :

Fitur penggunaan internet harian adalah yang paling penting dengan dampak positif signifikan. Waktu harian di situs juga penting, tetapi kurang signifikan. Pendapatan daerah, usia, provinsi, dan kategori juga berkontribusi dengan tingkat penting yang berbeda pada prediksi. Jam memiliki pengaruh sedang pada prediksi.

https://drive.google.com/file/d/16zKvTaMmJ-jJQFZ7-XysuL4TZ-sThjlT/view?usp=sharing

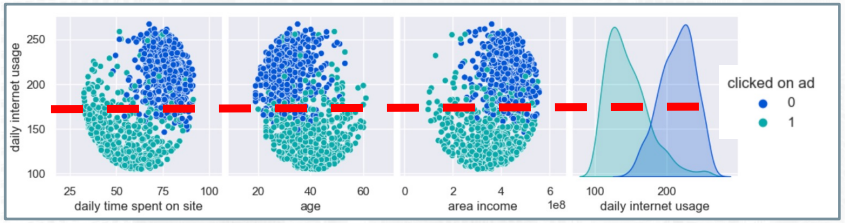
Business Recommendation & Simulation

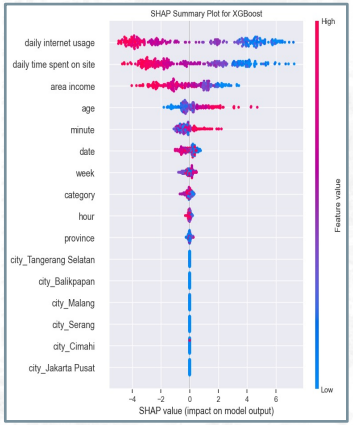
Rekomendasi Bisnis Berdasarkan EDA & Feature Importance

Sebagaimana analisis pada EDA sebelumnya yang tercermin pada heatmap korelasi bisa disimpulkan bahwa semakin tinggi ‘daily internet usage’ dan ‘daily time spent on site’ maka potensi mengklik iklan semakin sedikit bahkan tidak ada. Ini bisa diperjelas dengan diagram Shap.

Dalam pairplot tercermin bahwa angka dibawah 175 adalah yang paling berpotensi mengklik iklan terlepas dari faktor ‘age’ dan ‘area income’ (UMR) nya.

Oleh karena itu, sangat disarankan untuk mentarget iklan pada pengguna yang memiliki ‘daily internet usage’ dibawah rata-rata kelas yaitu 179,89 perhari.





	Feature	Importance
0	daily internet usage	0.6048209071159363
1	daily time spent on site	0.18079645931720734
2	area income	0.05553075671195984
3	age	0.040533870458602905
4	province	0.03080485761165619
5	category	0.022497721016407013
6	hour	0.018578359857201576
7	minute	0.01834065653383732
8	date	0.014399674721062183
9	week	0.013696707785129547

Referensi
Conversion Rate Pada Industri Global

Rata-rata tolok ukur tingkat konversi situs web e-niaga

Tingkat konversi e-niaga rata-rata adalah sekitar 2,5% hingga 3% menurut para pemimpin industri, namun itu tidak berarti ini adalah titik terbaik bagi bisnis Anda. Memiliki dasar 2,5% adalah awal yang baik, tetapi teruslah berupaya mengoptimalkannya dengan taktik tingkat konversi.

Aplikasi analitik Shopify, LittleData, melakukan survei komprehensif terhadap tingkat konversi toko Shopify dan menemukan bahwa tingkat konversi rata-rata untuk toko Shopify adalah 1,4%. Jika skor Anda di bawah 0,5%, kemungkinan besar Anda masih memiliki ruang untuk meningkatkannya, dan jika skor Anda di atas 3,3%, Anda memiliki tingkat konversi e-niaga yang sangat baik—di 20% teratas dari semua toko Shopify.

https://www.shopify.com/blog/ecommerce-conversion-rate

Referensi
Conversion Rate to Revenue

Calculating ARPC: The basic formula

The basic formula for calculating ARPC is relatively simple. To calculate ARPC, simply divide the total revenue generated by the number of conversions:

ARPC = Total Revenue / Number of Conversions

For example, let's say that a business generated $10,000 in revenue from 100 customer conversions. The ARPC for those conversions would be:

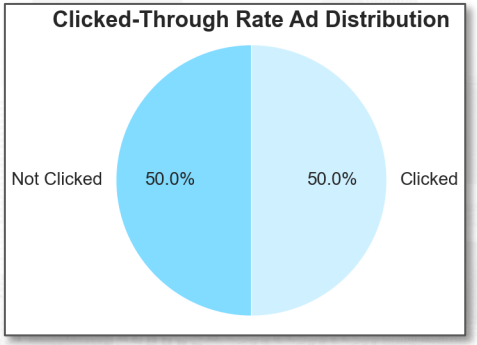
ARPC = $10,000 / 100 = $100

https://sleeknote.com/advanced/formulae/how-to-calculate-average-revenue-per-conversion-a-step-by-step-guide

Simulasi Sebelum Menggunakan Model Machine Learning Dengan Modal Yang Sama

Metrics :

CTR = Click-Through Rate,
mengukur persentase penonton iklan yang mengklik tautan atau tindakan yang diinginkan setelah melihat iklan. CTR = (Jumlah klik iklan / Jumlah tayangan iklan) x 100%. Ini mengukur tingkat interaksi dengan iklan.



CPM = Cost Per Mille,
CPM mengukur biaya per seribu tayangan iklan. Ini membantu dalam mengukur efisiensi biaya kampanye iklan

100.000
Asumsikan CPM = 100.000 untuk 1000 kali tayang. (rate IG)

Impression =
Total tayangan iklan

1000

Output :

Maka, dengan biaya 100.000 untuk 1000 tayang, perusahaan hanya mendapatkan 500 klik pada iklan.

https://bakrie.ac.id/articles/541-10-istilah-digital-marketing-yang-wajib-diketahui.html

Business Recommendation & Simulation

Simulasi Sebelum Menggunakan Model Machine Learning Dengan Modal Yang Sama

Sebelum menerapkan ML, dari 1000 impression hanya terdapat CTR 500 impression, dengan rata-rata industri conversion rate 2,5% , kalkulasi :

Total Revenue = 12.510.000
Total Cost = 100.000
Total Profit = 12.410.000

	category	count	impression	ratio%	conversion rate 2.5%	revenue
0	Otomotif	59	500	12.0	1.48	1,480,000.0
1	Fashion	56	500	11.0	1.4	1,400,000.0
2	House	57	500	11.0	1.42	1,420,000.0
3	Electronic	48	500	10.0	1.2	1,200,000.0
4	Finance	52	500	10.0	1.3	1,300,000.0
5	Food	49	500	10.0	1.23	1,230,000.0
6	Health	48	500	10.0	1.2	1,200,000.0
7	Furniture	45	500	9.0	1.12	1,120,000.0
8	Travel	47	500	9.0	1.18	1,180,000.0
9	Bank	39	500	8.0	0.98	980,000.0

```
Total Revenue: 12,510,000.0
Total Cost: 100,000
Total Profit: 12,410,000.0
```

Simulasi Setelah Menggunakan Model Machine Learning Dengan Modal Yang Sama

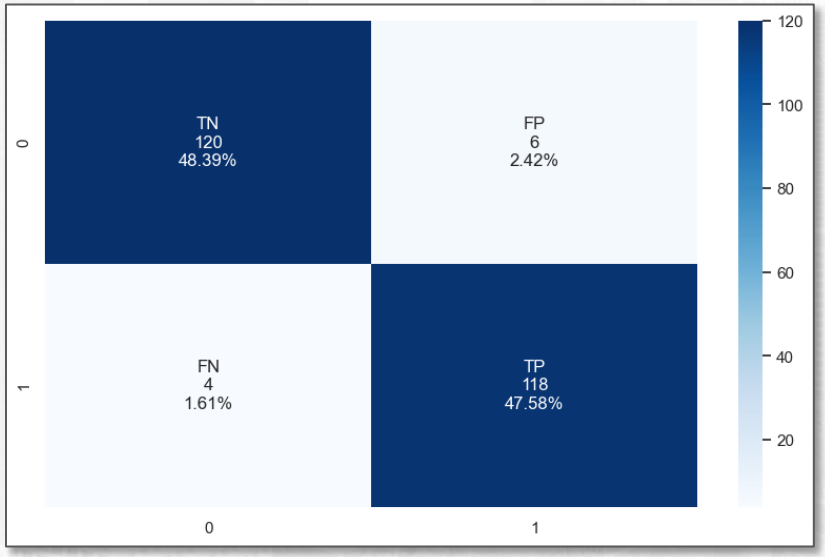
Total Visitor = 248

CTR = 100%
TP = 118 impression
TN = 120 visitor tidak diberikan iklan
FN = 4 miss, TP yang tidak mendapat iklan
FP = 6, TN yang mendapat iklan

118 visitor diberikan iklan dan semuanya clicked on ad = 100%

CPM = 100.000
Biaya 100.000 = 1000 impression
Maka 100.000 = (1000/TP) x TP
100.000 = (1000/118) x 118
100.000 = **8** x 118
100.000 = **944 impressions**

Impression = 944



Output :
Maka, dengan ML, dengan biaya yang sama, perusahaan bisa mendapatkan engagement 444 lebih banyak daripada tanpa ML.

Business Recommendation & Simulation

Simulasi Setelah Menggunakan Model Machine Learning Dengan Modal Yang Sama

Setelah menerapkan ML, terdapat 994 impression dengan rata-rata industri conversion rate 2,5% , kalkulasi :

Total Revenue = 23.600.000
Total Cost = 100.000
Total Profil = 23.500.000

	category	count	impression	ratio%	conversion rate 2.5%	revenue
0	Otomotif	113.28	944	12.0	2.83	2,830,000.0
1	Fashion	103.84	944	11.0	2.6	2,600,000.0
2	House	103.84	944	11.0	2.6	2,600,000.0
3	Electronic	94.4	944	10.0	2.36	2,360,000.0
4	Finance	94.4	944	10.0	2.36	2,360,000.0
5	Food	94.4	944	10.0	2.36	2,360,000.0
6	Health	94.4	944	10.0	2.36	2,360,000.0
7	Furniture	84.96	944	9.0	2.12	2,120,000.0
8	Travel	84.96	944	9.0	2.12	2,120,000.0
9	Bank	75.52	944	8.0	1.89	1,890,000.0

```
Total Revenue: 23,600,000.0
Total Cost: 100,000
Total Profit: 23,500,000.0
```

Business Recommendation & Simulation

Simulasi Perbandingan Cost, Revenue dan Profit

Tanpa XGBoost

	category	count	impression	ratio%	conversion rate 2.5%	revenue
0	Otomotif	59	500	12.0	1.48	1,480,000.0
1	Fashion	56	500	11.0	1.4	1,400,000.0
2	House	57	500	11.0	1.42	1,420,000.0
3	Electronic	48	500	10.0	1.2	1,200,000.0
4	Finance	52	500	10.0	1.3	1,300,000.0
5	Food	49	500	10.0	1.23	1,230,000.0
6	Health	48	500	10.0	1.2	1,200,000.0
7	Furniture	45	500	9.0	1.12	1,120,000.0
8	Travel	47	500	9.0	1.18	1,180,000.0
9	Bank	39	500	8.0	0.98	980,000.0

```
Total Revenue: 12,510,000.0
Total Cost: 100,000
Total Profit: 12,410,000.0
```

Dengan XGBoost

	category	count	impression	ratio%	conversion rate 2.5%	revenue
0	Otomotif	113.28	944	12.0	2.83	2,830,000.0
1	Fashion	103.84	944	11.0	2.6	2,600,000.0
2	House	103.84	944	11.0	2.6	2,600,000.0
3	Electronic	94.4	944	10.0	2.36	2,360,000.0
4	Finance	94.4	944	10.0	2.36	2,360,000.0
5	Food	94.4	944	10.0	2.36	2,360,000.0
6	Health	94.4	944	10.0	2.36	2,360,000.0
7	Furniture	84.96	944	9.0	2.12	2,120,000.0
8	Travel	84.96	944	9.0	2.12	2,120,000.0
9	Bank	75.52	944	8.0	1.89	1,890,000.0

```
Total Revenue: 23,600,000.0
Total Cost: 100,000
Total Profit: 23,500,000.0
```