

Predict Clicked Ads Customer Classification by using Machine Learning

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



Created by:
Rheza Paleva Uyanto
uyantorheza@gmail.com
<https://www.linkedin.com/in/rheza-uyanto/>

Lulus dari Program Pendidikan Profesi Apoteker Universitas Surabaya pada tahun 2017, memiliki pengalaman praktik kefarmasian di Rumah Sakit selama 4 tahun. Kemampuan komunikasi dan managerial yang baik, dan mampu bekerja sama dalam tim ataupun secara mandiri. Sangat termotivasi dalam bidang mentoring dan pengembangan diri. Mampu dalam penggunaan Microsoft Office. Memiliki ketertarikan dalam bidang Data Analitik, sehingga mengikuti Rakamin Bootcamp Data Science Batch 24. Melalui portofolio ini, saya akan memprediksi klasifikasi customer dari *clicked ads*, untuk meningkatkan marketing campaign menggunakan Machine Learning.

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

- **Exploration Data Analysis (EDA)**

- **Dataset :** Clicked Ads Dataset
- Terdiri dari 1000 baris data dan 11 kolom data
- Terdiri dari 5 data numerical, dan 6 data kategorikal
- Ada beberapa data yang memiliki null value

```
<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
```

`df.describe()`

	Unnamed: 0	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage
count	1000.000000	987.000000	1000.000000	9.870000e+02	989.000000
mean	499.500000	64.929524	36.009000	3.848647e+08	179.863620
std	288.819436	15.844699	8.785562	9.407999e+07	43.870142
min	0.000000	32.600000	19.000000	9.797550e+07	104.780000
25%	249.750000	51.270000	29.000000	3.286330e+08	138.710000
50%	499.500000	68.110000	35.000000	3.990683e+08	182.650000
75%	749.250000	78.460000	42.000000	4.583554e+08	218.790000
max	999.000000	91.430000	61.000000	5.563936e+08	267.010000

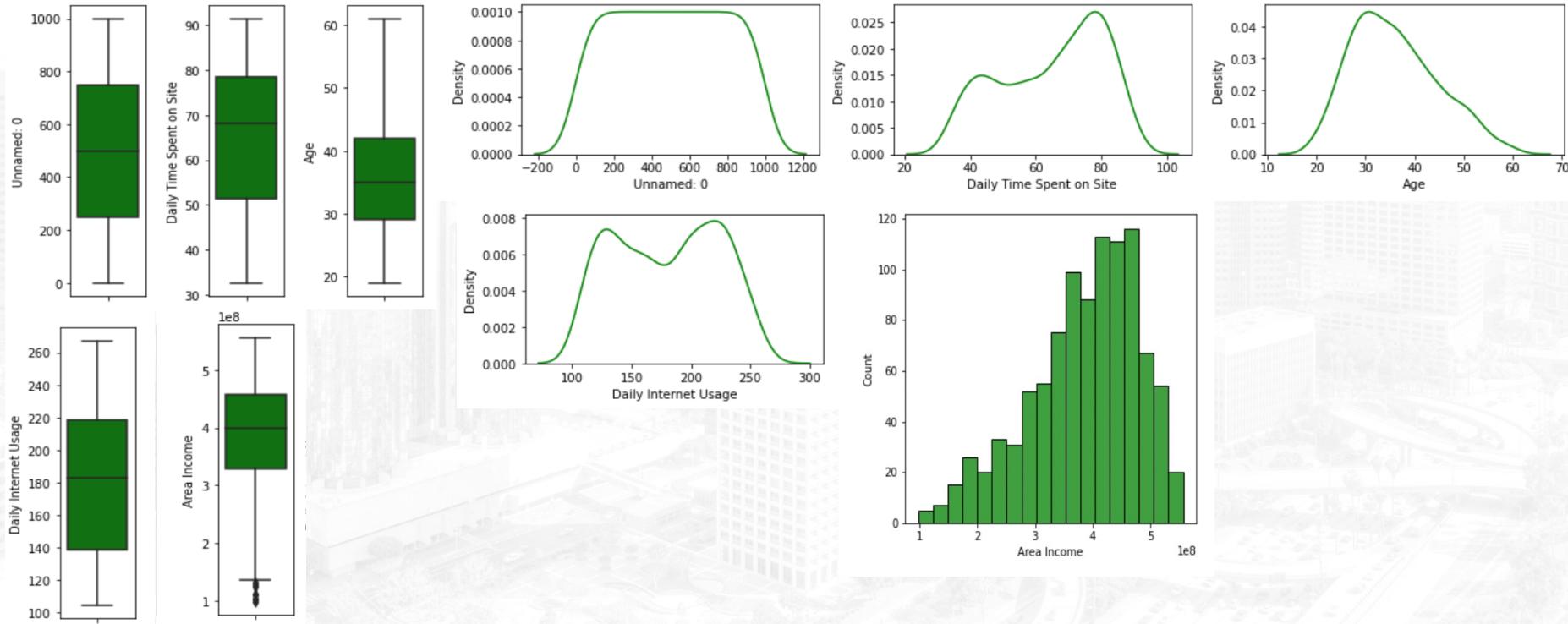
`df[cats].describe()`

	Male	Clicked on Ad	city	province	category
count	997	1000	1000	1000	1000
unique	2	2	30	16	10
top	Perempuan	No	Surabaya	Daerah Khusus Ibukota Jakarta	Otomotif
freq	518	500	64	253	112

Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Customer Type and Behaviour Analysis on Advertisement

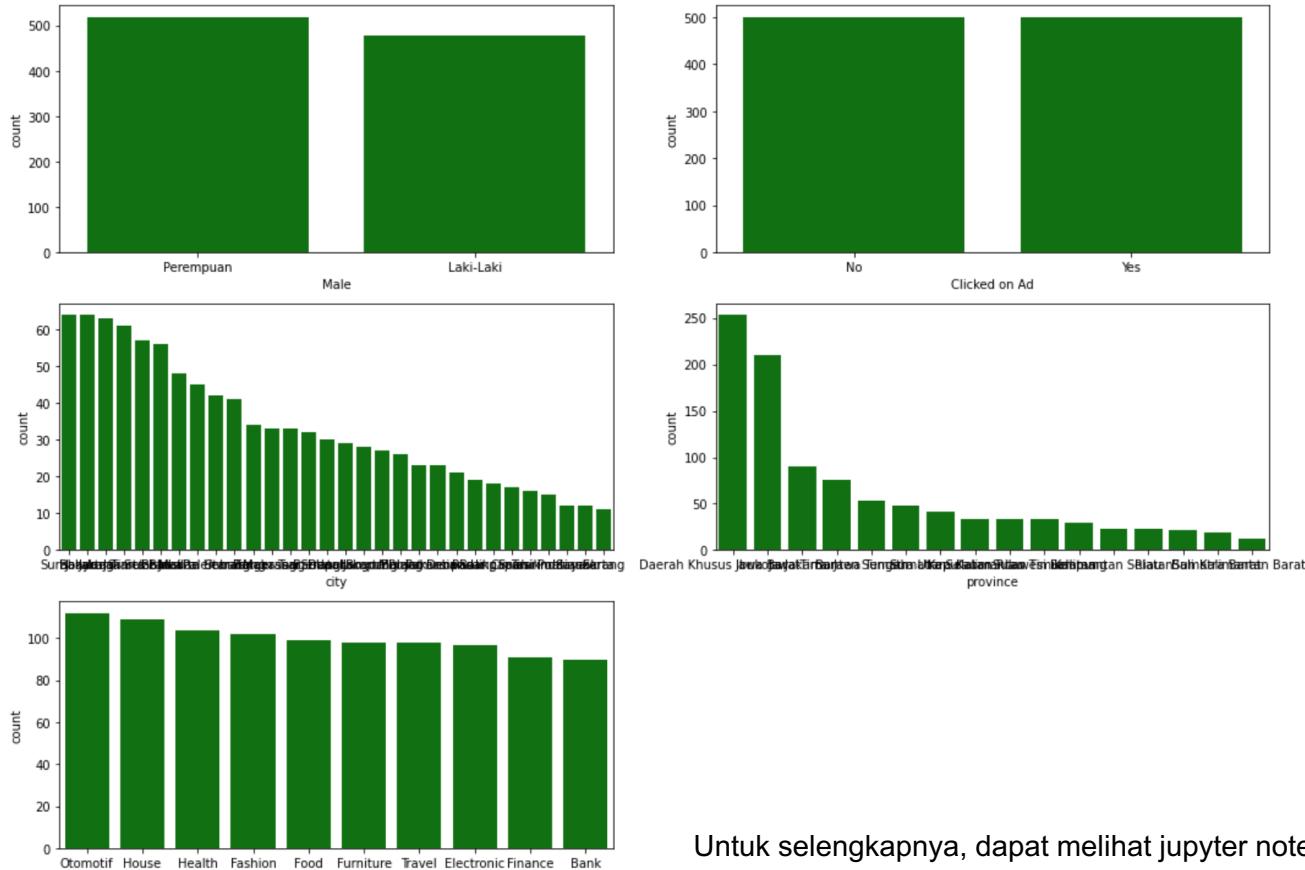
- **Univariate Analysis**
 - Data Numerical



Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Customer Type and Behaviour Analysis on Advertisement

- **Univariate Analysis**
 - Data Kategorikal

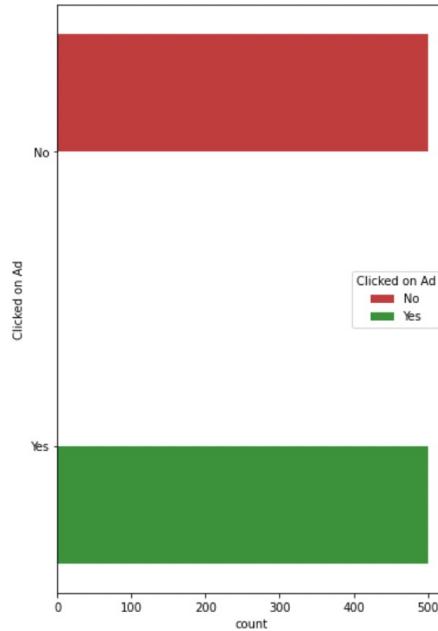
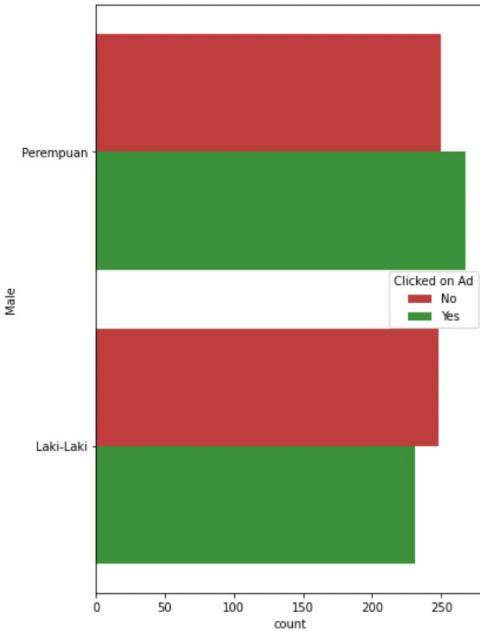


Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

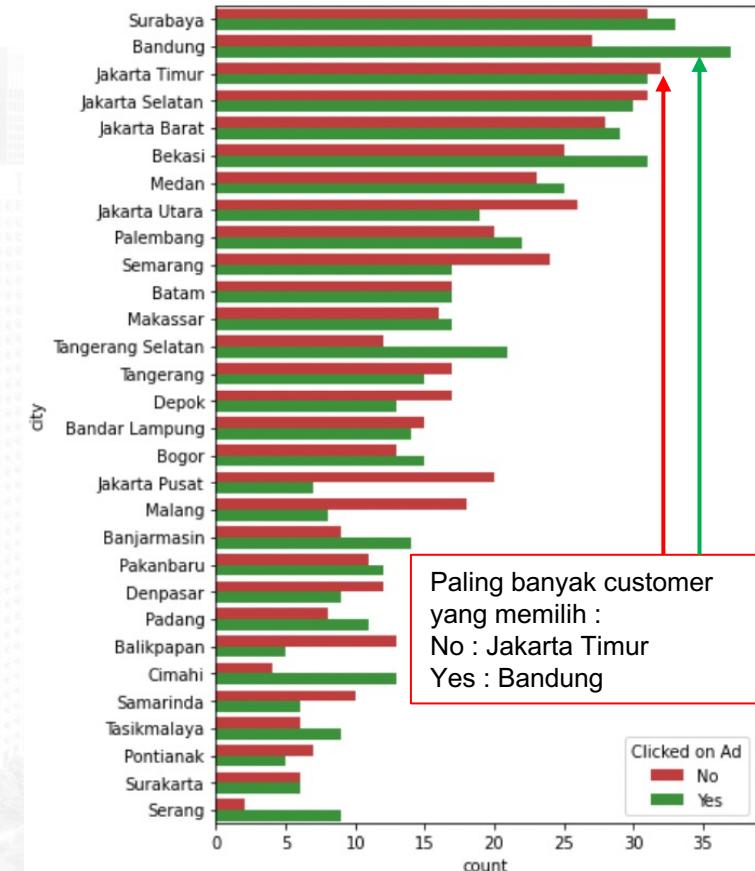
Customer Type and Behaviour Analysis on Advertisement

- Bivariate Analysis**

- Data Kategorikal



Persebaran Yes dan No, jika dilihat dari Jenis kelamin berimbang, Jumlah Yes dan No sendiri sama 50:50.

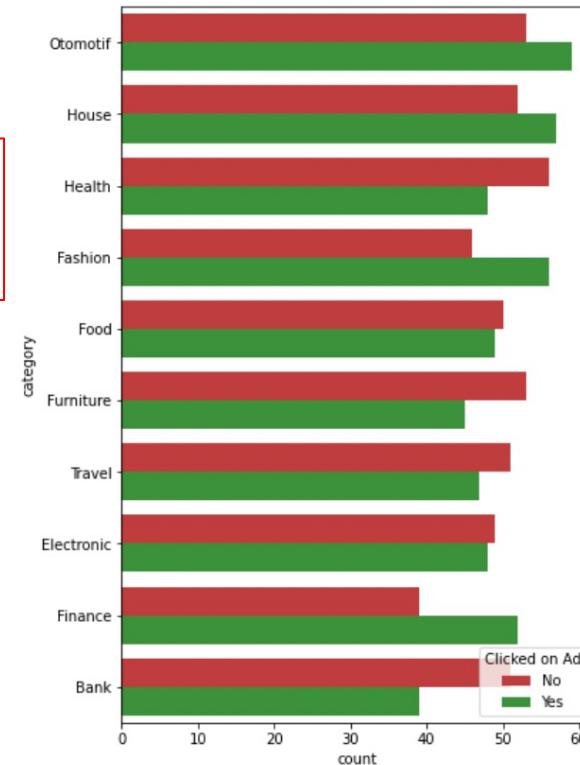
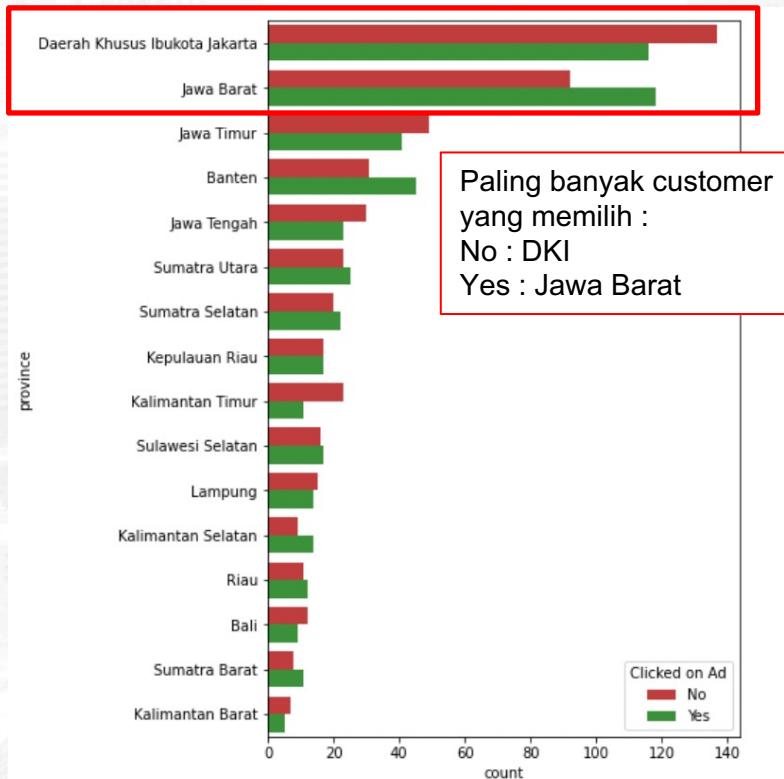


Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Customer Type and Behaviour Analysis on Advertisement

- Bivariate Analysis**

- Data Kategorikal



Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

untuk persebaran pada
kolom kategori cukup
merata

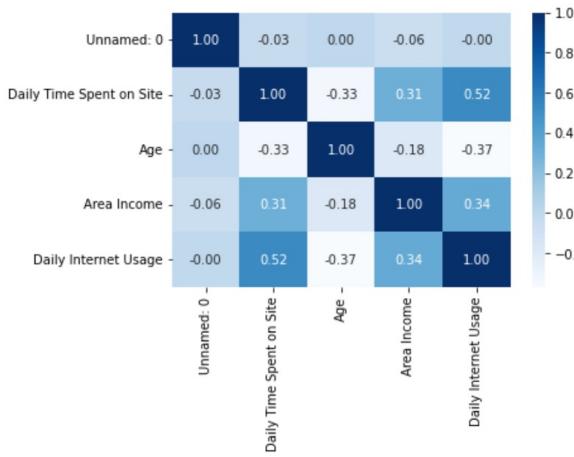
Customer Type and Behaviour Analysis on Advertisement

- **Multivariate Analysis**
 - Data Numerical

```
| df.corr()
```

	Unnamed: 0	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage
Unnamed: 0	1.000000		-0.032637	0.001835	-0.056862
Daily Time Spent on Site	-0.032637		1.000000	-0.331424	0.308266
Age	0.001835		-0.331424	1.000000	-0.179343
Area Income	-0.056862		0.308266	-0.179343	1.000000
Daily Internet Usage	-0.004842		0.518294	-0.370481	0.338080

```
sns.heatmap(df.corr(), cmap = 'Blues', annot = True, fmt = '.2f');
```



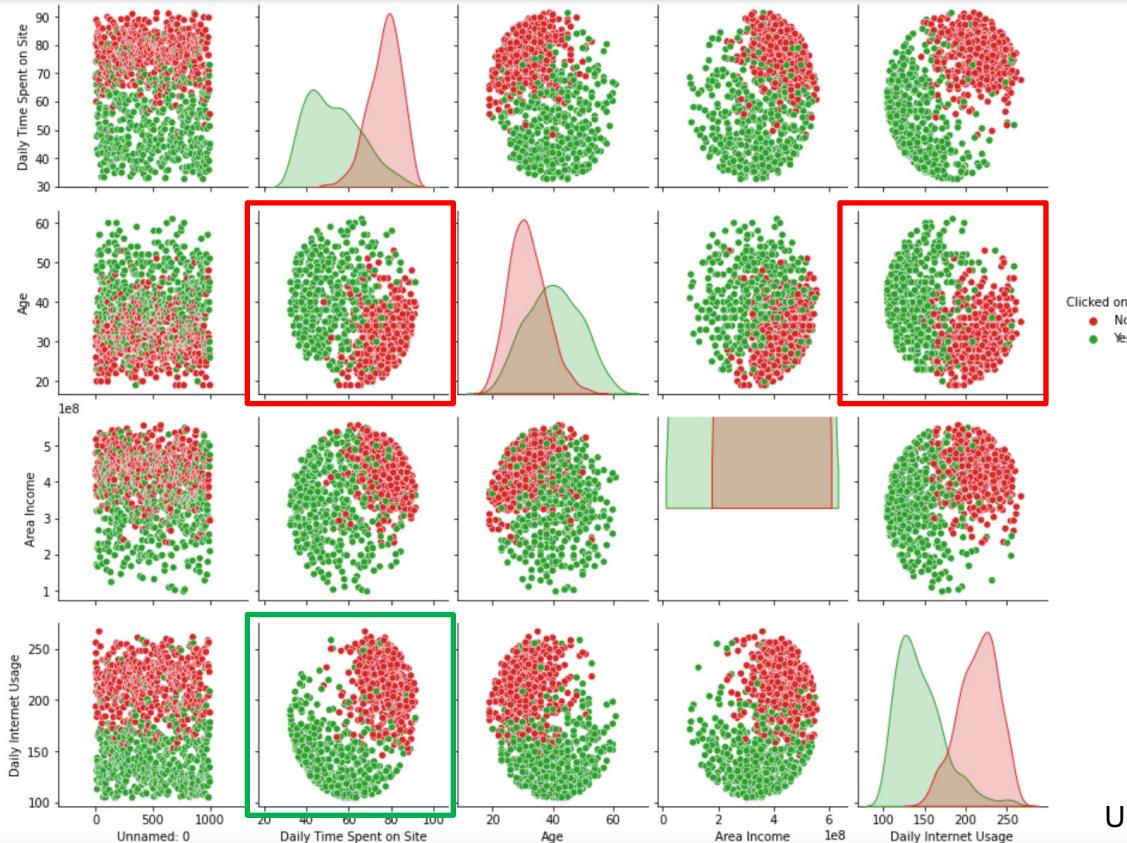
Hubungan Age – Daily Time Spent on Site dan Age – Daily Internet Usage cenderung berkorelasi negatif, Sedangkan Daily Time Spent on Site – Daily Internet Usage cenderung berkorelasi positif.

Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Customer Type and Behaviour Analysis on Advertisement

- Multivariate Analysis**

- Data Numerical



Hubungan Age – Daily Time Spent on Site dan Age – Daily Internet Usage cenderung **berkorelasi negatif**,

Pada kotak merah : Customer yang durasi internet relatif singkat, dan kuota internet yang digunakan juga sedikit, cenderung tertarik dengan iklan yang ditayangkan, dibandingkan sebaliknya. Usia tidak terpengaruh. Titik pola menyebar.

Sedangkan Daily Time Spent on Site – Daily Internet Usage cenderung **berkorelasi positif**.

Pada kotak hijau : Customer yang durasi internet yang lama juga memiliki kuota internet yang banyak, begitu juga sebaliknya. Jika dikaitkan dengan iklan. Customer yang berselancar di internet dalam durasi lama dan memiliki kuota yang banyak, cenderung tidak tertarik dengan iklan.

Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Customer Type and Behaviour Analysis on Advertisement

- Multivariate Analysis**

- Data Numerical dan Data Kategorik



Adanya korelasi antara Daily Time Spent on Site dan Daily Spent on Site dengan Click on Ad, dengan nilai 0,68 dan 0,63

Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Data Cleaning & Preprocessing

● Handling Null Value

Terdapat 4 kolom yang memiliki null-value

- Daily Time Spent on Site – diisi dengan nilai median
- Area Income – diisi dengan nilai median
- Daily Internet Usage – diisi dengan nilai median
- Male – diisi dengan modus (karena data kategorik, paling banyak adalah Laki-laki)

df.isna().sum()

```
Unamed: 0          0
Daily Time Spent on Site 13
Age                  0
Area Income          13
Daily Internet Usage 11
Male                 3
Timestamp             0
Clicked on Ad         0
city                 0
province              0
category              0
dtype: int64
```

df.isna().sum()

```
Unamed: 0          0
Daily Time Spent on Site 0
Age                  0
Area Income          0
Daily Internet Usage 0
Male                 0
Timestamp             0
Clicked on Ad         0
city                 0
province              0
category              0
dtype: int64
```

Sebelum dihandling null value

Sesudah dihandling null value

● Handling nilai duplikat

- Tidak ditemukan nilai duplikat

df.duplicated().sum()

0

Data Cleaning & Preprocessing

● Ekstrak Waktu

- Waktu diekstrak dari kolom Timestamp.
- Memanggil library datetime
- Dilakukan mengubah format Timestamp dari object menjadi date time.
- Ekstraksi dimulai, ada 4 jenis waktu yang dibutuhkan:
 - Tahun (year)
 - Bulan (month)
 - Minggu (week) 1-52 minggu
 - Hari (day) 1-7 hari
- Hasil ekstraksi diubah ke numerik agar mempermudah analisis, sebelum pemodelan

```
# import library datetime
from datetime import datetime
df['date'] = pd.to_datetime(df['Timestamp'])

#ekstrak tahun, bulan, minggu, dan hari
df['year']=(df['date'].dt.strftime('%Y'))

df['month']=(df['date'].dt.strftime('%m'))

df['week']=(df['date'].dt.strftime('%U'))

df['day']=(df['date'].dt.strftime('%w'))
#0=Sunday , 1=Monday, 2=Tuesday, 3=Wednesday, 4= Thursday, 5=Friday, 6=Saturday

#ubah ke numerik
df['year'] = pd.to_numeric(df['year'])
df['month'] = pd.to_numeric(df['month'])
df['day'] = pd.to_numeric(df['day'])
df['week'] = pd.to_numeric(df['week'])
```

Data Cleaning & Preprocessing

● Feature Encoding

Kolom yang dilakukan feature encoding antara lain :

- Male
- City
- Province
- Category

Feature encoding menggunakan One Hot Encoding karena bukan merupakan kategori yang bertingkat.

● Feature Transformation

- Clicked on Ad diubah menjadi 0 untuk No, 1 untuk Yes

● Feature Dropping

Setelah dilakukan fitur encoding, beberapa fitur didrop :

- Unnamed: 0
- Male
- Timestamp
- City
- Province
- Clicked on Ad
- Category
- date

```
df_encode=df.copy()
one_hot_cats=df_encode[['Male','city', 'province','category']]
one_hot=pd.get_dummies(one_hot_cats)

df = df.join(one_hot)
```

```
df['Clicked on Ad Label'] = np.where(df['Clicked on Ad'] == 'Yes',1,0)
```

```
df = df.drop(columns = ['Unnamed: 0','Male','Timestamp',
'city','province','Clicked on Ad','category','date']) #drop fitur
```

Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

- Split Data Train dan Data Test
 - Data dibagi menjadi X (untuk kolom selain Target) dan y (untuk kolom target)
 - Kemudian X dan y displit menjadi data train dan data test (dengan perbandingan 7:3)
 - X_train, X_test, y_train, y_test untuk data yang discaler,
 - X_train1, X_test1, y_train1, y_test1 untuk data yang tidak discaler

```
df_model = df
```

```
#Fitur dan target dipisah dalam variabel X dan y
X = df_model.drop(columns=['Clicked on Ad Label'])
y = pd.Series(df_model['Clicked on Ad Label'])
```

```
# X dan y dibagi untuk data train dan test sesuai random 7:3
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
X_train1, X_test1, y_train1, y_test1 = train_test_split(X, y, test_size=0.3, random_state=42)
```

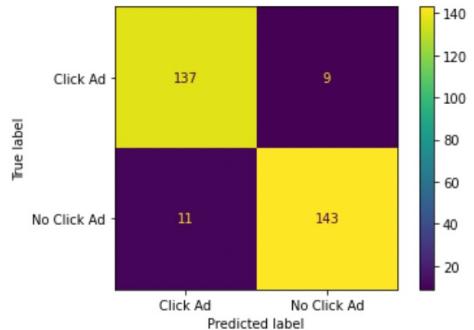
- ***Model yang digunakan antara lain :***

- *Decision Tree*
- *Logistic Regression*
- *XGBoost*
- *AdaBoost Classifier*
- *Random Forest*
- *Gradient Boosting Classifier*
- *K-Nearest Neighbors Classifier*

Data Modeling

- Model 1 : Decision Tree**

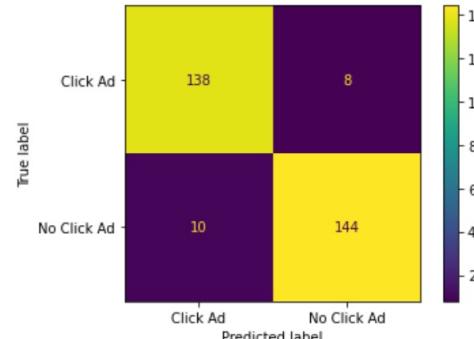
Accuracy (Test Set): 0.93
 Precision (Test Set): 0.94
 Recall (Test Set): 0.93
 F1-Score (Test Set): 0.93
 AUC: 0.93



Train score: 1.0
 Test score: 0.9333333333333333

Sebelum scaler

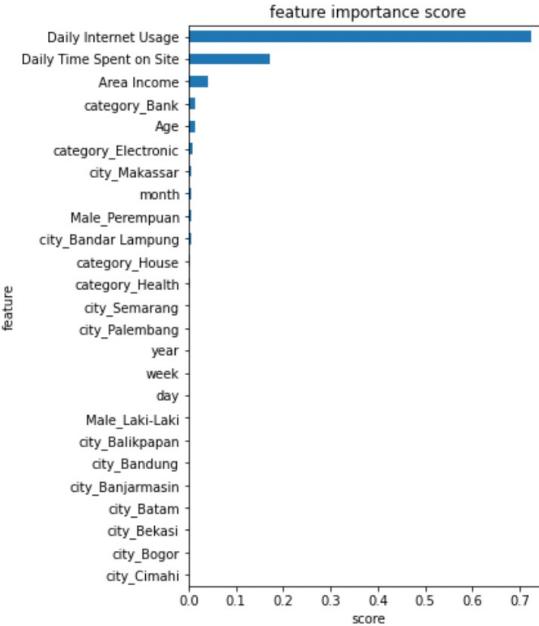
Accuracy (Test Set): 0.94
 Precision (Test Set): 0.95
 Recall (Test Set): 0.94
 F1-Score (Test Set): 0.94
 AUC: 0.94



Train score: 1.0
 Test score: 0.94

Sesudah scaler

$\text{Nilai Parameter Precision} = \text{True Positive} / (\text{True Positive} + \text{False Positive})$
 Nilai Precision meingkat sesudah discaler.

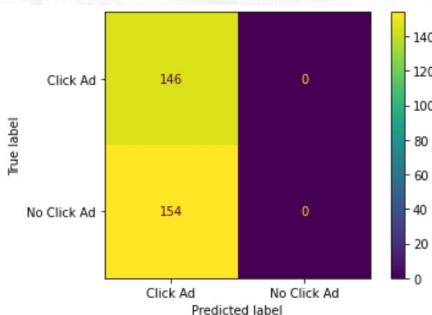


Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Data Modeling

- Model 2 : Logistic Regression**

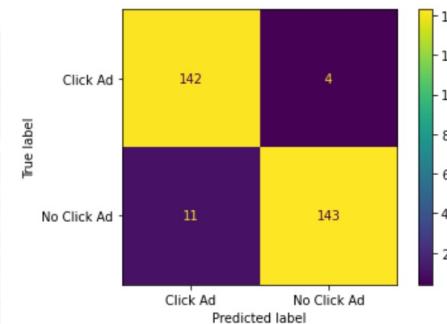
Accuracy (Test Set): 0.49
 Precision (Test Set): 0.00
 Recall (Test Set): 0.00
 F1-Score (Test Set): 0.00
 AUC: 0.73



Train score: 0.5057142857142857
 Test score: 0.4866666666666666

Sebelum scaler

Accuracy (Test Set): 0.95
 Precision (Test Set): 0.97
 Recall (Test Set): 0.93
 F1-Score (Test Set): 0.95
 AUC: 0.98



Train score: 0.98
 Test score: 0.95

Sesudah scaler

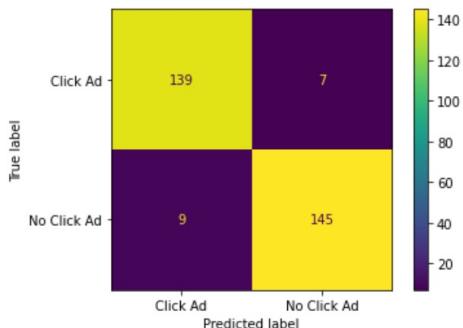
$\text{Nilai Parameter Precision} = \text{True Positive} / (\text{True Positive} + \text{False Positive})$
 Nilai Precision meingkat sesudah discaler.

Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Data Modeling

- Model 3 : XGBoost**

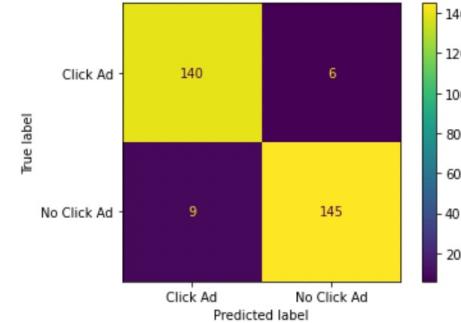
Accuracy (Test Set): 0.95
 Precision (Test Set): 0.95
 Recall (Test Set): 0.94
 F1-Score (Test Set): 0.95
 AUC: 0.98



Train score: 0.9928571428571429
 Test score: 0.9466666666666667

Sebelum scaler

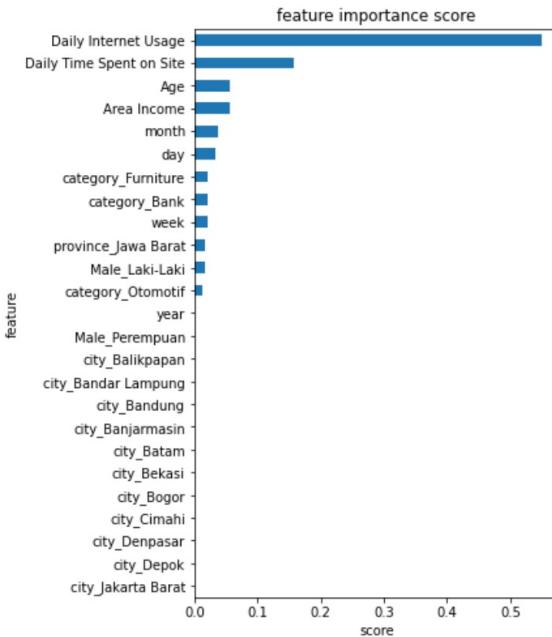
Accuracy (Test Set): 0.95
 Precision (Test Set): 0.96
 Recall (Test Set): 0.94
 F1-Score (Test Set): 0.95
 AUC: 0.98



Train score: 0.9928571428571429
 Test score: 0.95

Sesudah scaler

$\text{Nilai Parameter Precision} = \text{True Positive} / (\text{True Positive} + \text{False Positive})$
 Nilai Precision meingkat sesudah discaler.

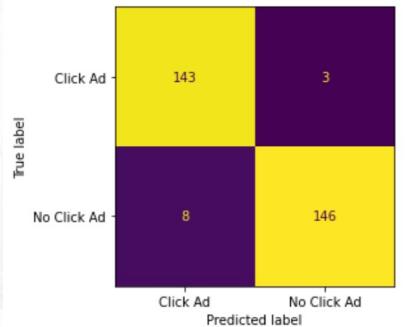


Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Data Modeling

- Model 4 : AdaBoost Classifier**

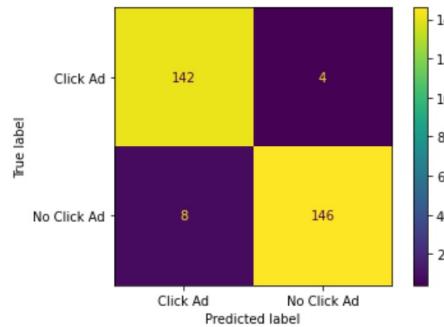
Accuracy (Test Set): 0.96
 Precision (Test Set): 0.98
 Recall (Test Set): 0.95
 F1-Score (Test Set): 0.96
 AUC: 0.98



Train score: 0.9957142857142857
 Test score: 0.9633333333333334

Sebelum scaler

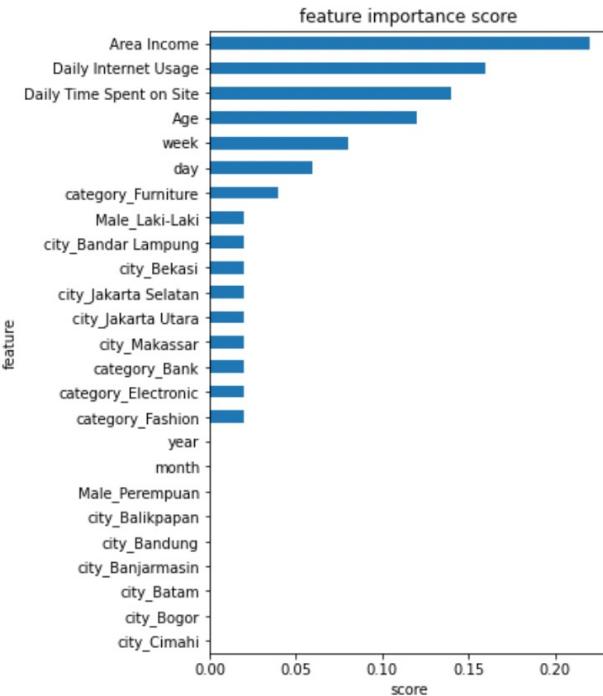
Accuracy (Test Set): 0.96
 Precision (Test Set): 0.97
 Recall (Test Set): 0.95
 F1-Score (Test Set): 0.96
 AUC: 0.99



Train score: 0.9957142857142857
 Test score: 0.96

Sesudah scaler

$\text{Nilai Parameter Precision} = \text{True Positive} / (\text{True Positive} + \text{False Positive})$
 Nilai Precision turun setelah scaler

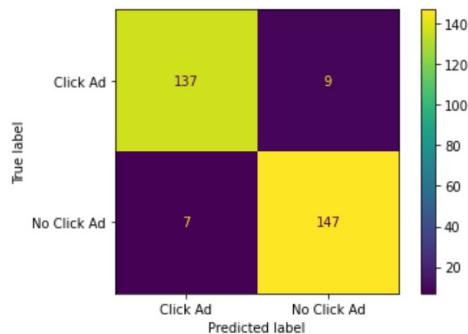


Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Data Modeling

- Model 5 : Random Forest**

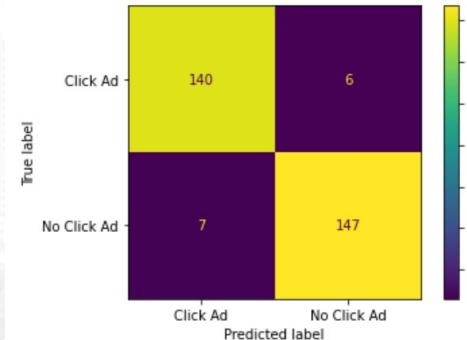
Accuracy (Test Set): 0.95
 Precision (Test Set): 0.94
 Recall (Test Set): 0.95
 F1-Score (Test Set): 0.95
 AUC: 0.98



Train score: 1.0
 Test score: 0.9466666666666667

Sebelum scaler

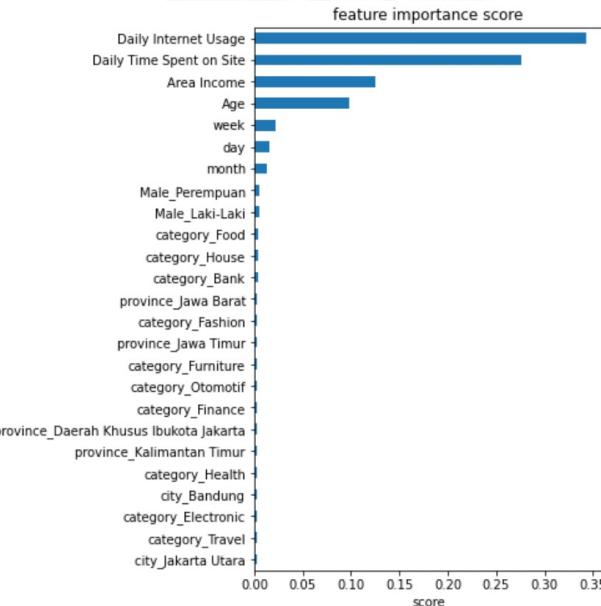
Accuracy (Test Set): 0.96
 Precision (Test Set): 0.96
 Recall (Test Set): 0.95
 F1-Score (Test Set): 0.96
 AUC: 0.98



Train score: 0.4942857142857143
 Test score: 0.5133333333333333

Sesudah scaler

$\text{Nilai Parameter Precision} = \text{True Positive} / (\text{True Positive} + \text{False Positive})$
 Nilai Precision meingkat sesudah discaler.

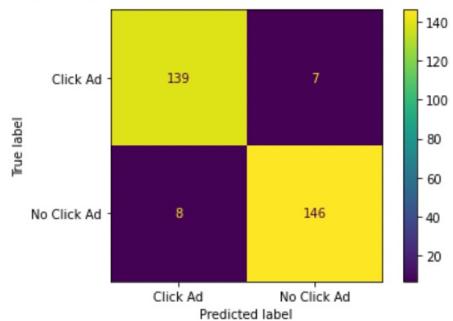


Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Data Modeling

- Model 6 : Gradient Boosting Classifier**

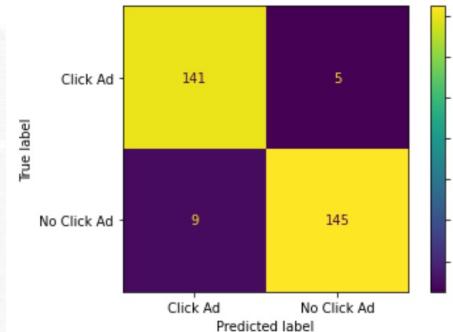
Accuracy (Test Set): 0.95
 Precision (Test Set): 0.95
 Recall (Test Set): 0.95
 F1-Score (Test Set): 0.95
 AUC: 0.98



Train score: 1.0
 Test score: 0.95

Sebelum scaler

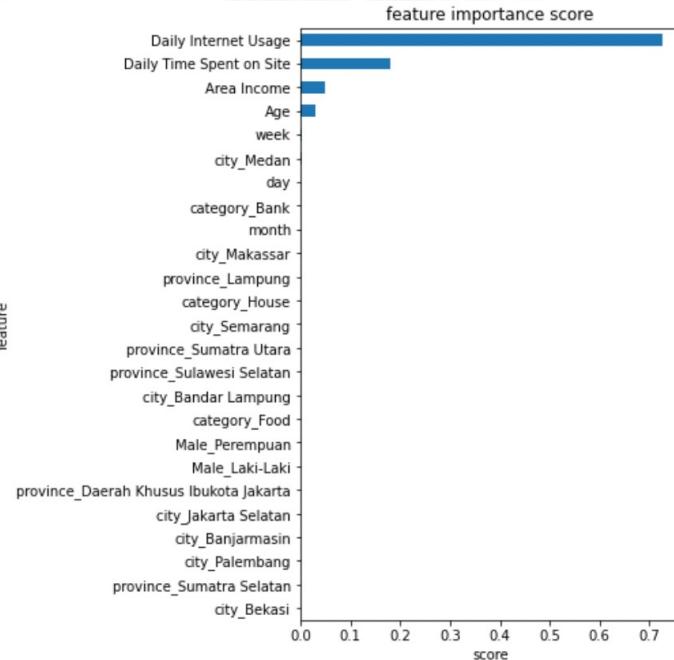
Accuracy (Test Set): 0.95
 Precision (Test Set): 0.97
 Recall (Test Set): 0.94
 F1-Score (Test Set): 0.95
 AUC: 0.98



Train score: 1.0
 Test score: 0.9533333333333334

Sesudah scaler

$\text{Nilai Parameter Precision} = \text{True Positive} / (\text{True Positive} + \text{False Positive})$
 Nilai Precision meingkat sesudah discaler.

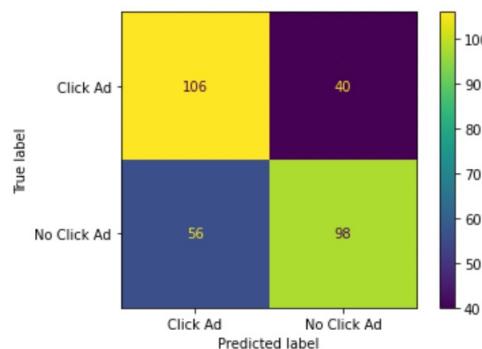


Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Data Modeling

- Model 7 : K-Nearest Neighbors Classifier**

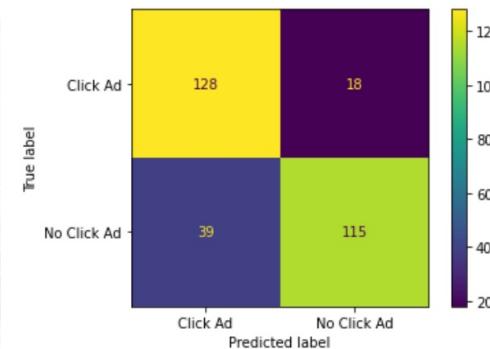
Accuracy (Test Set): 0.68
 Precision (Test Set): 0.71
 Recall (Test Set): 0.64
 F1-Score (Test Set): 0.67
 AUC: 0.70



Train score: 0.78
 Test score: 0.68

Sebelum scaler

Accuracy (Test Set): 0.81
 Precision (Test Set): 0.86
 Recall (Test Set): 0.75
 F1-Score (Test Set): 0.80
 AUC: 0.88



Train score: 0.8742857142857143
 Test score: 0.81

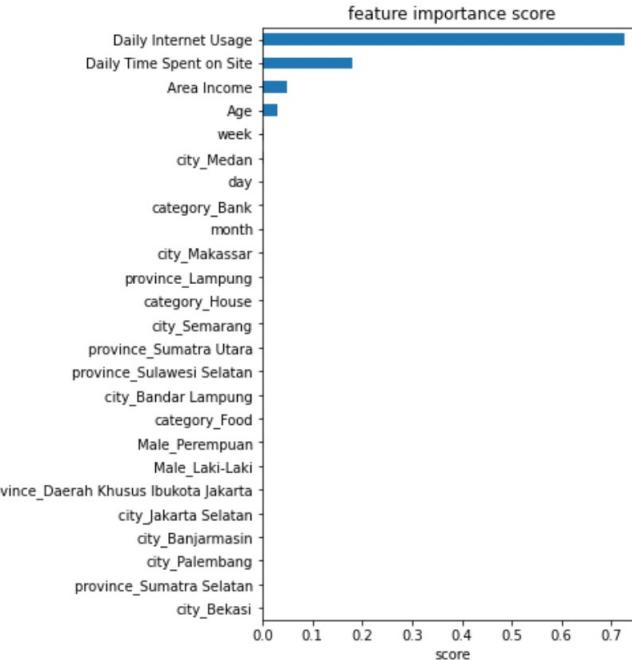
Sesudah scaler

Nilai Parameter Precision = True Positive / (True Positive + False Positive)
Nilai Precision meingkat sesudah discaler.

Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

- **Hasil Interpretasi**

- *Nilai Precision paling tinggi ditunjukkan oleh Pemodelan Gradient Boosting Classifier*
- *Nilai precision : 0,97*
- *Feature Importance tertinggi:*
 - *Daily Internet Usage*
 - *Daily Time Spent on Site*
 - *Area Income*



Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

Business Recommendation & Simulation

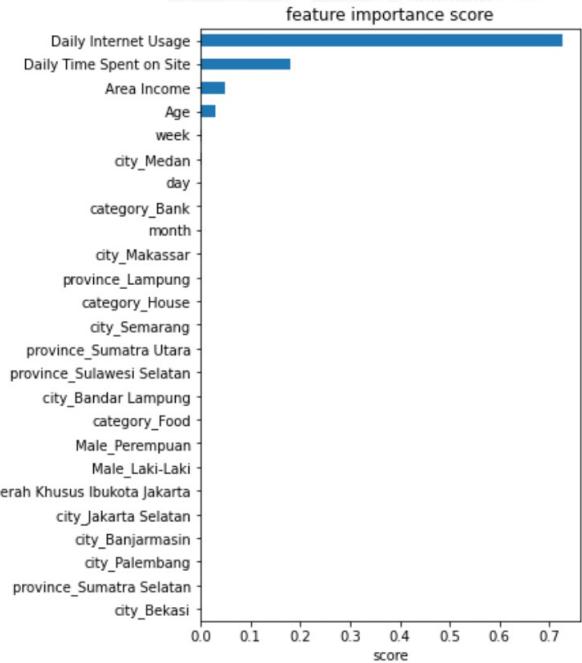
- **Feature Important** dari hasil model machine learning adalah :

- Daily Internet Usage
- Daily Time Spent on Site
- Area Income

```
df.describe()
```

Unnamed: 0	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage
count	1000.000000	987.000000	1000.000000	9.870000e+02
mean	499.500000	64.929524	36.009000	3.848647e+08
std	288.819436	15.844699	8.785562	9.407999e+07
min	0.000000	32.600000	19.000000	9.797550e+07
25%	249.750000	51.270000	29.000000	3.286330e+08
50%	499.500000	68.110000	35.000000	3.990683e+08
75%	749.250000	78.460000	42.000000	4.583554e+08
max	999.000000	91.430000	61.000000	5.563936e+08

- Daily Time Spent on Site: Lamanya tinggal disuatu situs (harian) dalam satuan menit
- Area Income: Pendapatan user dalam satuan rupiah
- Daily Internet Usage: Penggunaan internet harian dalam satuan menit
- Clicked on Ad: Click atau tidak iklan yang ditampilkan



Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

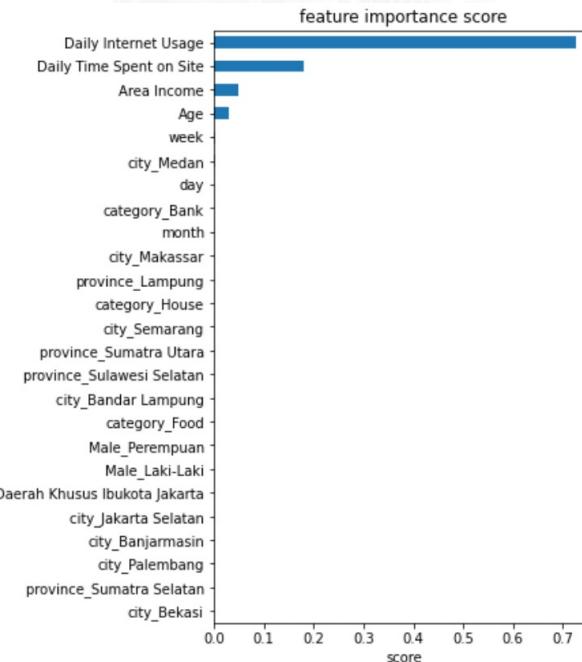
Business Recommendation & Simulation

- **Beberapa rekomendasi bisnis yang perlu dipertimbangkan:**

- Setiap hari, orang akan menghabiskan minimal 30 menit untuk berselancar dalam sebuah website dan menggunakan internet minimal selama 100 menit.
- Iklan yang ditayangkan haruslah berhubungan dengan minat user.
- Iklan harus menarik perhatian, ditayangkan 15-30 menit pertama saat user datang dalam website tersebut.
- Kota dan Provinsi tidak menjadi fitur prioritas, sehingga iklan dapat ditayangkan tanpa melihat area.
- Hari penayangan juga tidak menjadi fitur prioritas, sehingga iklan dapat ditayangkan di seluruh hari.

```
df.describe()
```

Unnamed: 0	Daily Time Spent on Site	Age	Area Income	Daily Internet Usage
count	1000.000000	987.000000	1000.000000	9.870000e+02
mean	499.500000	64.929524	36.009000	3.848647e+08
std	288.819436	15.844699	8.785562	9.407999e+07
min	0.000000	32.600000	19.000000	9.797550e+07
25%	249.750000	51.270000	29.000000	3.286330e+08
50%	499.500000	68.110000	35.000000	3.990683e+08
75%	749.250000	78.460000	42.000000	4.583554e+08
max	999.000000	91.430000	61.000000	5.563936e+08



Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

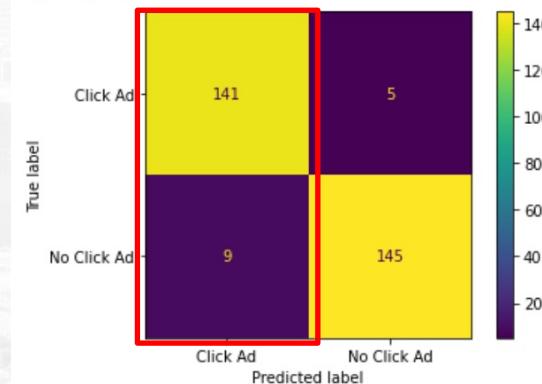
- **Simulasi Bisnis sebelum menggunakan Machine Learning**

- Jika Biaya Per klik, sumber literatur [disini](#), adalah Rp 3.000,-, dengan target peserta adalah 300 orang.
 - **Cost : 300 orang x Rp. 3.000,- = Rp. 900.000,-**
- Revenue, Peserta yang mengklik Iklan sebanyak 150 orang, dan melakukan transaksi sebanyak minimal Rp 100.000,- maka
 - **Revenue : 150 orang x Rp 100.000,- = Rp. 150.000,-**
- Profit, dengan memperhitungkan Cost dan Revenue yang didapatkan
 - **Profit = Revenue – Cost**
 - **Profit = Rp 150.000,- - Rp 900.000 = (- Rp 450.000,-) Rugi sebanyak 450.000,-**

- **Simulasi Bisnis sesudah menggunakan Machine Learning**

- Jika Biaya Per klik, sumber literatur [disini](#), adalah Rp 3.000,-, dengan target peserta difokuskan pada peserta yang berselancar di website minimal 15 menit. Dengan Machine Learning, sebanyak 150 orang diberikan Iklan
 - **Cost : 150 orang x Rp. 3.000,- = Rp. 450.000,-**
- Revenue, Dari Machine Learning, Peserta yang mengklik Iklan sebanyak 141 orang dari 150 orang, dan jika 141 orang tersebut melakukan transaksi sebanyak minimal Rp 100.000,- maka
 - **Revenue : 141 orang x Rp 100.000,- = Rp. 14.100.000,-**
- Profit, dengan memperhitungkan Cost dan Revenue yang didapatkan
 - **Profit = Revenue – Cost**
 - **Profit = Rp 14.100.000,- - Rp 450.000,- = Rp 13.650.000,-**

Accuracy (Test Set): 0.95
Precision (Test Set): 0.97
Recall (Test Set): 0.94
F1-Score (Test Set): 0.95
AUC: 0.98



Sumber Biaya Per Click : [disini](#)
Untuk selengkapnya, dapat melihat jupyter notebook [disini](#)

- **Kesimpulan**

- Penentuan model machine learning dalam pemodelan saat diperlukan
- Feature Importance juga mempengaruhi dalam pengambilan keputusan dalam bisnis, terutama dalam fokus dalam faktor tertentu yang mempengaruhi dalam proses bisnis.
- Dalam proses diatas, dengan menggunakan model machine learning dapat meningkatkan profit sampai lebih dari 100 % dengan memilih user yang berselancar di website minimal 15 menit.