LAPORAN ANALISIS TGP #1
PREDICTIVE MODELLING ANALYTICS

Disusun Oleh:
Kelompok 3 Kelas PMA (C)

Anggota Kelompok:
Bayu Siddhi Mukti (5026211021)
Wanda Armadianti (5026211039)
Zahrina Candrakanti (5026211100)
Alif Destiano (5026211176)

Detail Dataset :

Dataset Kampanye Pemasaran Langsung
(*Direct Marketing*) Deposito Bank Berjangka

Jumlah Baris Data : 45211
Rentang Waktu : Mei 2008 - November 2010
Jumlah Variabel : 17

Variabel Independen :

- age : numeric = umur
- job : categorical = jenis pekerjaan
- marital : categorical = jenis status pernikahan
- education : categorical = jenis tingkat pendidikan
- default : binary = apakah memiliki kewajiban yang gagal bayar?
- balance : numeric = saldo tabungan (euro)
- housing : binary = apakah memiliki kredit rumah?
- loan : binary = apakah memiliki kredit pribadi?
- contact : categorical = jenis kontak komunikasi
- day : numeric = tanggal kontak terakhir
- month : categorical = bulan kontak terakhir
- duration : numeric = durasi kontak terakhir (detik)
- campaign : numeric = jumlah kontak ke klien pada kampanye saat ini
- pdays : numeric = jumlah hari terlewat setelah kontak terakhir ke klien pada kampanye sebelumnya
- previous : numeric = jumlah kontak ke klien pada kampanye sebelumnya
- poutcome : categorical = jenis hasil dari kampanye sebelumnya

Variabel Dependen :

- subscribe : binary = apakah klien berlangganan deposito berjangka?

Link Google Colab :
https://colab.research.google.com/drive/1mlGI--PePUxT9JRmU6Kw_OwN57RpmN3x?usp=sharing

PROSES PENGERJAAN

1. Praproses Data

A. Import Data

- Import beberapa library untuk menjalankan beberapa argumen/perintah coding

```
# Import Library Python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

- Import Data Excel yang akan dianalisis dengan memasukkannya menjadi sebuah variabel baru yaitu variabel **df** untuk dicetak menjadi output tabel. Untuk perintah import data sendiri terdiri dari perintah **pd.read_csv** yang digunakan sebagai Pembaca File CSV ke phyton dari library pandas, dan perintah **separator** digunakan untuk pemisah string pada setiap data yang ada.

```
# Import data dari CSV menjadi data frame
df = pd.read_csv('Data utk TGP #1.csv', sep = ';')
df
```

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	subscribe
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
45206	51	technician	married	tertiary	no	825	no	no	cellular	17	nov	977	3	-1	0	unknown	yes
45207	71	retired	divorced	primary	no	1729	no	no	cellular	17	nov	456	2	-1	0	unknown	yes
45208	72	retired	married	secondary	no	5715	no	no	cellular	17	nov	1127	5	184	3	success	yes
45209	57	blue-collar	married	secondary	no	668	no	no	telephone	17	nov	508	4	-1	0	unknown	no
45210	37	entrepreneur	married	secondary	no	2971	no	no	cellular	17	nov	361	2	188	11	other	no

45211 rows × 17 columns

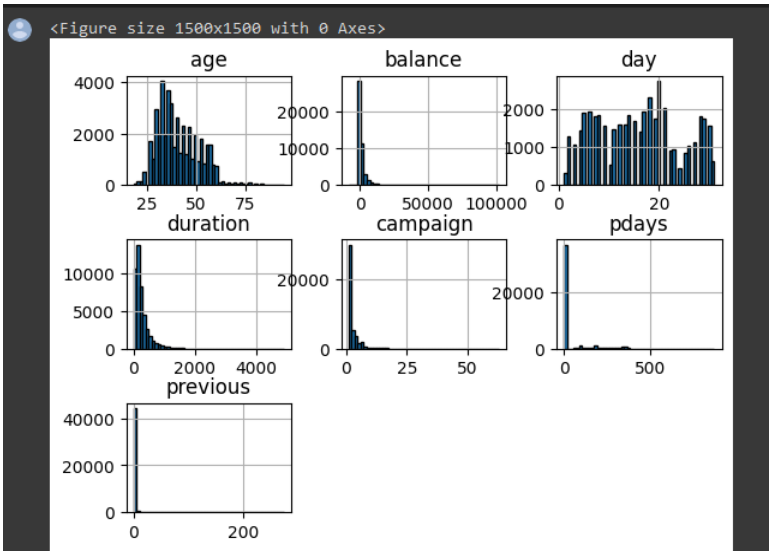
- Setelah memasukkan data untuk dibaca, data akan dibacakan overview statistiknya berisikan sebaran dataset dengan perintah **describe**

```
df.describe()
```

	age	balance	day	duration	campaign	pdays	previous
count	45211.000000	45211.000000	45211.000000	45211.000000	45211.000000	45211.000000	45211.000000
mean	40.936210	1362.272058	15.806419	258.163080	2.763841	40.197828	0.580323
std	10.618762	3044.765829	8.322476	257.527812	3.098021	100.128746	2.303441
min	18.000000	-8019.000000	1.000000	0.000000	1.000000	-1.000000	0.000000
25%	33.000000	72.000000	8.000000	103.000000	1.000000	-1.000000	0.000000
50%	39.000000	448.000000	16.000000	180.000000	2.000000	-1.000000	0.000000
75%	48.000000	1428.000000	21.000000	319.000000	3.000000	-1.000000	0.000000
max	95.000000	102127.000000	31.000000	4918.000000	63.000000	871.000000	275.000000

- Untuk melihat sebaran data awal dengan visualisasi secara histogram dapat menggunakan perintah **hist()**.

```
# menampilkan histogram
plt.figure(figsize = (15, 15))
df.hist(bins=50, edgecolor='k')
plt.subplots_adjust(hspace=0.5)
plt.show()
```



B. Mengubah Tipe Data

- Dalam Proses ini, setelah data diimpor dan dieksplor sebaran datanya, setiap data yang ada akan dicek tipe datanya dengan perintah **info**. Dalam data **"Data utk TGP #1.csv",** ditemukan tipe data object untuk teks dan tipe data int64 untuk numerik

```
# Cek tipe data kolom
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   age        45211 non-null  int64
 1   job        45211 non-null  object
 2   marital    45211 non-null  object
 3   education  45211 non-null  object
 4   default    45211 non-null  object
 5   balance    45211 non-null  int64
 6   housing    45211 non-null  object
 7   loan       45211 non-null  object
 8   contact    45211 non-null  object
 9   day        45211 non-null  int64
 10  month      45211 non-null  object
 11  duration   45211 non-null  int64
 12  campaign   45211 non-null  int64
 13  pdays      45211 non-null  int64
 14  previous   45211 non-null  int64
 15  poutcome   45211 non-null  object
 16  subscribe  45211 non-null  object
dtypes: int64(7), object(10)
memory usage: 5.9+ MB
```

- Sesuai permintaan soal, terdapat beberapa kolom yang diharuskan diubah menjadi kategorikal diantaranya "**'job', 'marital', 'education', 'contact', 'month', 'poutcome'** ". Maka kolom-kolom tersebut diubah menjadi kategorikal dengan perintah **astype**

```
[ ] # Mengubah tipe data dari kolom 'job', 'marital', 'education', 'contact', 'month', dan 'poutcome' dari object menjadi categorical
    df['job'] = df['job'].astype('category')
    df['marital'] = df['marital'].astype('category')
    df['education'] = df['education'].astype('category')
    df['contact'] = df['contact'].astype('category')
    df['month'] = df['month'].astype('category')
    df['poutcome'] = df['poutcome'].astype('category')

    # Mengubah tipe data dari kolom 'default', 'housing', 'loan', dan 'subscribe' dari object menjadi bool
    df['default'] = df['default'].astype('category')
    df['housing'] = df['housing'].astype('category')
    df['loan'] = df['loan'].astype('category')
    df['subscribe'] = df['subscribe'].astype('category')
    # df['default'] = df['default'].map({'yes':True ,'no':False})
    # df['housing'] = df['housing'].map({'yes':True ,'no':False})
    # df['loan'] = df['loan'].map({'yes':True ,'no':False})
    # df['subscribe'] = df['subscribe'].map({'yes':True ,'no':False})

    df.info()
```

```
df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 17 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   age        45211 non-null  int64
 1   job        45211 non-null  category
 2   marital    45211 non-null  category
 3   education  45211 non-null  category
 4   default    45211 non-null  category
 5   balance    45211 non-null  int64
 6   housing    45211 non-null  category
 7   loan       45211 non-null  category
 8   contact    45211 non-null  category
 9   day        45211 non-null  int64
 10  month      45211 non-null  category
 11  duration   45211 non-null  int64
 12  campaign   45211 non-null  int64
 13  pdays      45211 non-null  int64
 14  previous   45211 non-null  int64
 15  poutcome   45211 non-null  category
 16  subscribe  45211 non-null  category
dtypes: category(10), int64(7)
memory usage: 2.8 MB
```

C. Melakukan Pengecekan Nilai Null

- Agar data yang diproses nantinya akurat dan dapat diproses dengan baik, data akan dicek terlebih dahulu untuk mengetahui apakah terdapat field kosong atau yang bisa disebut missing value. Pengecekan ini dapat dilakukan dengan perintah **isnull().sum(), isnull sebagai pembaca null sedangkan sum untuk menjumlahkan field berisi null di setiap kolom,** sehingga didapatkan dari data tersebut tidak ada field yang kosong

```
[ ] # Apakah ada data yang NULL?
    jumlah_data_null = df.isnull().sum()
    print(f"Jumlah data null di tiap kolom: \n{jumlah_data_null}")
```

D. Melakukan Pengecekan pada Nilai Unik

- Untuk melakukan pengecekan nilai unik pada setiap kolom dapat dilakukan dengan membuat sebuah fungsi terlebih dahulu, yang dalam hal ini fungsi dinamakan sebagai **unique_value().** Fungsi tersebut akan melakukan pengecekan nilai unik di setiap kolom bertipe data kategorikal.

```
[ ]  # Function untuk melihat nilai unique value dari variabel categorical
     def unique_value(column):
       data = []
       for value in df[column].unique():
         data.append(value)
       print(f"Nilai dari variabel {column} = {data}")

     # Menampilkan semua unique value dari semua variabel categorical
     for column in df.select_dtypes(include='category').columns:
       unique_value(column)

     Nilai dari variabel job = ['management', 'technician', 'entrepreneur', 'blue-collar', 'unknown', 'retired', 'admin.', 'services', 'self-employed', 'unemployed', 'housemaid', 'student']
     Nilai dari variabel marital = ['married', 'single', 'divorced']
     Nilai dari variabel education = ['tertiary', 'secondary', 'unknown', 'primary']
     Nilai dari variabel default = ['no', 'yes']
     Nilai dari variabel housing = ['yes', 'no']
     Nilai dari variabel loan = ['no', 'yes']
     Nilai dari variabel contact = ['unknown', 'cellular', 'telephone']
     Nilai dari variabel month = ['may', 'jun', 'jul', 'aug', 'oct', 'nov', 'dec', 'jan', 'feb', 'mar', 'apr', 'sep']
     Nilai dari variabel poutcome = ['unknown', 'failure', 'other', 'success']
     Nilai dari variabel subscribe = ['no', 'yes']
```

E. Melakukan Pengecekan Nilai "Unknown"

- Setelah mengetahui nilai unik di beberapa kolom, terdapat beberapa kolom yang memiliki nilai unik "**Unknown**" diantaranya kolom job, education, contact, dan poutcome. Jumlah nilai unik Unknown tersebut dapat diketahui dengan membuat fungsi **check_unknown_value()** yang didalamnya terdapat perulangan untuk membaca nilai unknown di setiap data kategorikal.

```
[ ]  # Function untuk menghitung jumlah data 'unknown' di setiap variabel categorical
     def check_unknown_value():
       length = 0
       for column in df.columns:
         if length < len(column):
           length = len(column)

       print('Total data unknown di setiap variabel categorical:')

       # Menghitung jumlah data 'unknown' di setiap variabel categorical
       for column in df.select_dtypes(include='category').columns:
         unknown = (df[column] == 'unknown').sum()
         total = len(df)
         ratio = unknown/total

         text_column = column
         if len(column) < length:
           for i in range(0, (length - len(column))):
             text_column = text_column + ' '

         text_ratio = f"{unknown}/{total}"
         text_ratio_max = f"{len(df)}/{len(df)}"
         if len(text_ratio) < len(text_ratio_max):
           for i in range(0, (len(text_ratio_max) - len(text_ratio))):
             text_ratio = text_ratio + ' '

         print(f"{text_column}\t = {text_ratio}\t = {ratio}")
```

```
[ ]  check_unknown_value()

     Total data unknown di setiap variabel categorical:
     job             = 288/45211     = 0.006370131162770122
     marital         = 0/45211       = 0.0
     education       = 1857/45211    = 0.04107407489327818
     default         = 0/45211       = 0.0
     housing         = 0/45211       = 0.0
     loan            = 0/45211       = 0.0
     contact         = 13020/45211   = 0.28798301298356593
     month           = 0/45211       = 0.0
     poutcome        = 36959/45211   = 0.8174780473778506
     subscribe       = 0/45211       = 0.0
```

F. Mengganti Nilai "Unknown"

- Untuk Kasus seperti kolom job dan education yang memiliki nilai "**Unknown**" sedikit, dapat digantikan datanya dengan mencari mode-nya dengan perintah fungsi **replace_unknown_to_mode** lalu untuk hasilnya didapatkan dengan

memanggil fungsi **check_unknown_value() ,** didapatkan nilai **"Unknown"** pada kolom job dan education sebanyak 0

```
[ ]  # Function untuk mengganti unknown value menjadi data mode-nya
     def replace_unknown_to_mode(column):
       # Mendapatkan data yang paling sering muncul
       mode = df[column].value_counts().idxmax()
       # Mengganti unknown value dengan nilai mode-nya
       df[column] = df[column].replace(to_replace = 'unknown', value = mode)
```

```
# Menangani unknown value pada variabel 'job' dan 'education' (ganti value dengan mode-nya)
replace_unknown_to_mode('job')
replace_unknown_to_mode('education')

check_unknown_value()
```

```
Total data unknown di setiap variabel categorical:
job             = 0/45211       = 0.0
marital         = 0/45211       = 0.0
education       = 0/45211       = 0.0
default         = 0/45211       = 0.0
housing         = 0/45211       = 0.0
loan            = 0/45211       = 0.0
contact         = 13020/45211   = 0.28798301298356593
month           = 0/45211       = 0.0
poutcome        = 36959/45211   = 0.8174780473778506
subscribe       = 0/45211       = 0.0
```

- Untuk Kasus seperti kolom poutcome, dikarenakan nilai **"Unknown"** yang ada terdapat banyak, maka harus dihapus agar nilai **"Unknown"** tersebut tidak mempengaruhi hasil yang akan dianalisis. untuk proses ini dapat menggunakan perintah **drop**

```
# Menangani unknown value pada variabel 'poutcome' (dihapus)
df = df.drop(columns='poutcome')

check_unknown_value()
```

```
Total data unknown di setiap variabel categorical:
job             = 0/45211       = 0.0
marital         = 0/45211       = 0.0
education       = 0/45211       = 0.0
default         = 0/45211       = 0.0
housing         = 0/45211       = 0.0
loan            = 0/45211       = 0.0
contact         = 13020/45211   = 0.28798301298356593
month           = 0/45211       = 0.0
subscribe       = 0/45211       = 0.0
```

G. Melakukan Pengecekan Nilai Redundan

- Data sebaran yang memiliki nilai redundan atau duplikat sebaiknya dihilangkan agar data menjadi data yang akurat dan efektif untuk dianalisis. Untuk menghilangkan nilai duplikat atau nilai yang sama dapat menggunakan perintah **duplicated().sum(),** didapatkan nilai duplikat sebanyak 0 atau tidak ada

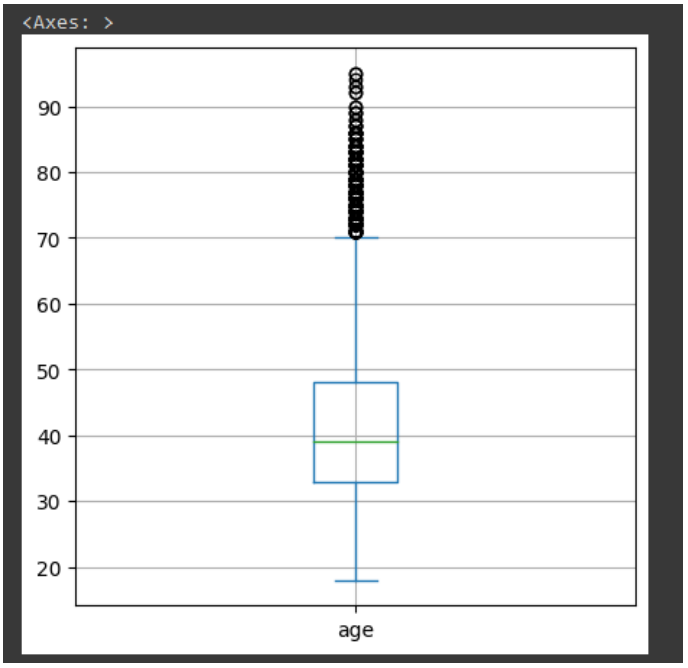
```
# Mencari jumlah baris yang nilai nya sama persis (duplikat)
jumlah_baris_duplikat = df.duplicated().sum()
print(f"Jumlah baris data duplikat = {jumlah_baris_duplikat}")
```

```
Jumlah baris data duplikat = 0
```

H. Melakukan Pengecekan Pada Outlier

- Outlier sendiri adalah data yang tidak berada pada lingkup rata-rata sebaran data atau bisa dikatakan sebagai data pencilan. Dalam Sebaran data Kampanye marketing ini didapatkan outlier di setiap kolom bertipe numerik, yang dapat divisualisasikan dengan boxplot. salah satu contohnya yaitu boxplot dari kolom age

```
# Cek apakah ada outlier pada kolom variabel 'age' menggunakan box plot
df['age'].plot(kind = 'box', figsize = (5, 5), grid = True)
```



I. Menghilangkan Outlier

- Setelah merepresentasikan outlier dalam boxplot, data yang mengandung outlier akan dihilangkan agar tidak mempengaruhi rata-rata sebaran data dengan membuat fungsi baru yang bernama **remove_outliers().** Dimana di dalam fungsi tersebut terdapat beberapa perulangan untuk mengecek nilai outlier pada setiap data numerik menggunakan Interquartile Range (IQR). Maka hasil yang didapatkan semula data sebaran awal sebanyak **45211 data,** sekarang menjadi **28069 data.**

```python
# Function untuk menghapus outlier dari seluruh kolom numerik dalam DataFrame
def remove_outliers(df):
    # Looping semua kolom dalam DataFrame
    for column in df.select_dtypes(include='number').columns:
      # if (column not in ['campaign', 'pdays', 'previous']):
        data_column = df[column]
        sorted_data = sorted(data_column)
        Q1, Q3 = np.percentile(sorted_data, [25, 75])
        IQR = Q3 - Q1
        lower_range = Q1 - (1.5 * IQR)
        upper_range = Q3 + (1.5 * IQR)

        # Mengambil nilai yang ada di dalam lower_range dan upper_range
        # Sehingga sama dengan menghapus baris yang mengandung outlier
        df = df[(df[column] >= lower_range) & (df[column] <= upper_range)]

    return df
```

```python
[ ] df = remove_outliers(df)
    df
```

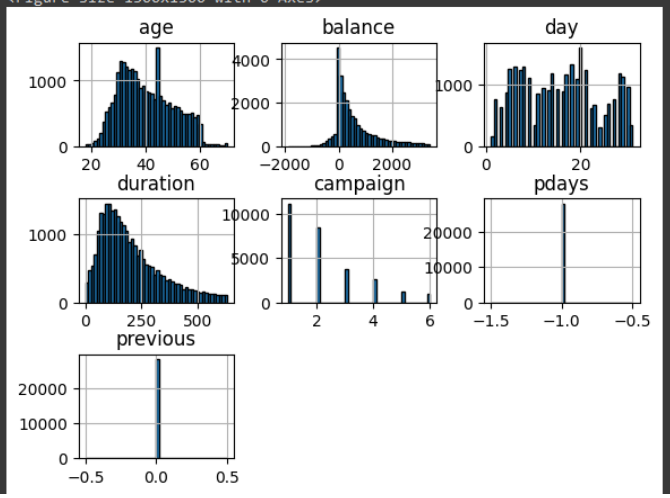
	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	subscribe
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	no
3	47	blue-collar	married	secondary	no	1506	yes	no	unknown	5	may	92	1	-1	0	no
4	33	blue-collar	single	secondary	no	1	no	no	unknown	5	may	198	1	-1	0	no
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
45198	37	management	married	tertiary	no	1428	no	no	cellular	16	nov	333	2	-1	0	no
45202	34	admin.	single	secondary	no	557	no	no	cellular	17	nov	224	1	-1	0	yes
45203	23	student	single	tertiary	no	113	no	no	cellular	17	nov	266	1	-1	0	yes
45205	25	technician	single	secondary	no	505	no	yes	cellular	17	nov	386	2	-1	0	yes
45209	57	blue-collar	married	secondary	no	668	no	no	telephone	17	nov	508	4	-1	0	no

28069 rows × 16 columns

- Dapat dilihat perbandingan histogram sebaran data awal pada praproses sebelumnya saat data masih bersifat raw dengan histogram di bawah ini saat data sudah diolah

```python
# menampilkan histogram
plt.figure(figsize = (15, 15))
df.hist(bins=50, edgecolor='k')
plt.subplots_adjust(hspace=0.5)
plt.show()
```

```
<Figure size 1500x1500 with 0 Axes>
```



J. Overview Statistics Sebaran Data Akhir

- Berikut merupakan hasil keseluruhan atau Central Tendency dari sebaran dataset kampanye marketing setelah dilakukan pengolahan data yang mencakup rata-rata, standar deviasi, hingga nilai maximum di setiap kolom dalam dataset yang baru dengan menggunakan perintah **describe**.

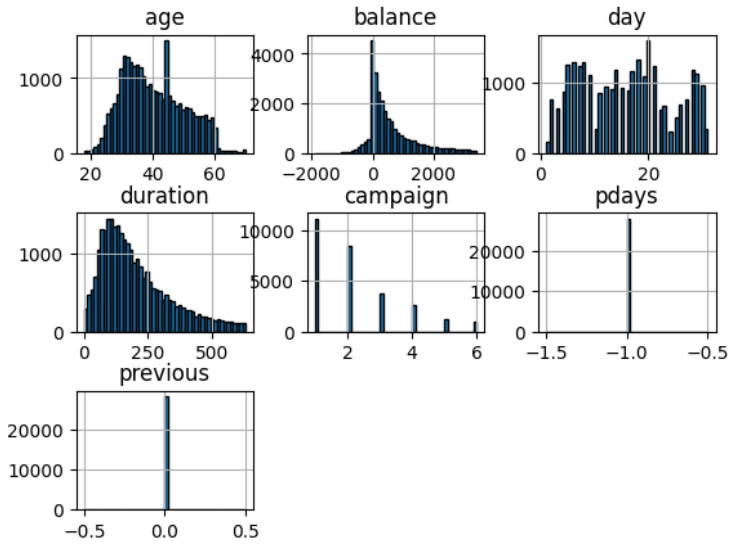
```
[ ]  df.describe()
```

	age	balance	day	duration	campaign	pdays	previous
count	28069.000000	28069.000000	28069.000000	28069.000000	28069.000000	28069.0	28069.0
mean	40.363461	602.524315	15.770744	204.185543	2.189284	-1.0	0.0
std	9.883276	824.619548	8.357648	137.645813	1.340697	0.0	0.0
min	18.000000	-1884.000000	1.000000	0.000000	1.000000	-1.0	0.0
25%	32.000000	31.000000	8.000000	101.000000	1.000000	-1.0	0.0
50%	39.000000	316.000000	16.000000	168.000000	2.000000	-1.0	0.0
75%	48.000000	916.000000	22.000000	276.000000	3.000000	-1.0	0.0
max	70.000000	3412.000000	31.000000	634.000000	6.000000	-1.0	0.0

2. Analisis Univariate dan Visualisasinya

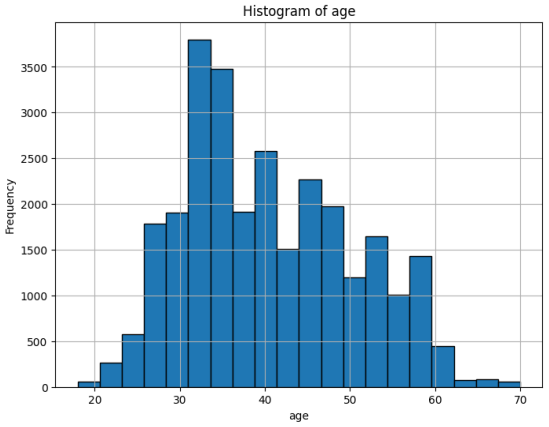
Analisis Univariate adalah salah satu jenis analisis yang berfokus pada distribusi dan karakteristik dari setiap atribut dalam dataset. Dari data yang sudah melalui tahap pra proses, analisis pertama yang dilakukan adalah menampilkan histogram untuk semua kolom dalam DataFrame sebagai berikut

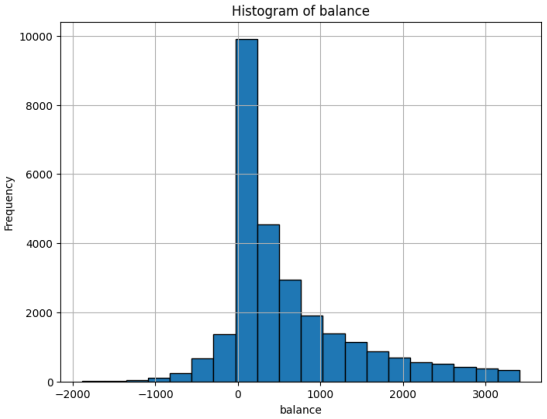
```
# menampilkan histogram
plt.figure(figsize = (15, 15))
df.hist(bins=50, edgecolor='k')
plt.subplots_adjust(hspace=0.5)
plt.show()
```

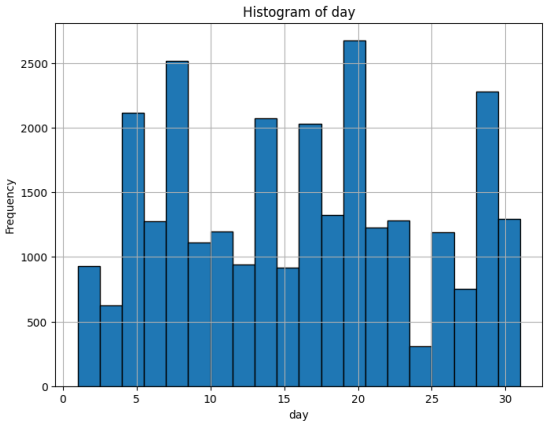


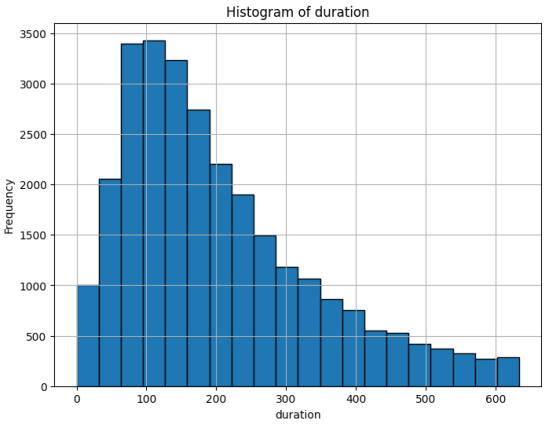
Berikut merupakan tahap kedua dalam analisis univariat dimana histogram dibuat lebih rinci, yaitu histogram untuk setiap kolom numerik dalam Data frame secara terpisah, dimana label sumbu x (horizontal) merupakan kolom yang dimaksud dan label sumbu y (vertikal) menunjukkan frekuensi kemunculan nilai dalam kolom. Dengan melakukan loop, histogram setiap kolom numerik memungkinkan analisis distribusi data secara lebih rinci

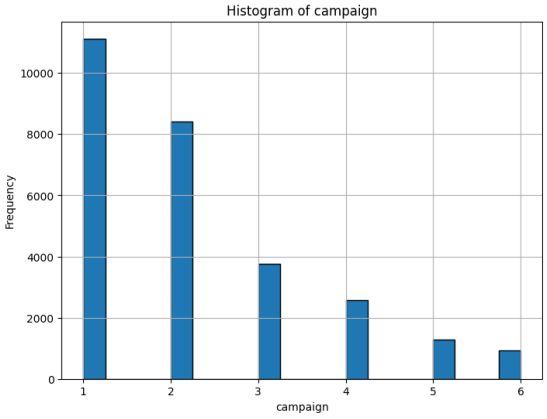
```
# Loop melalui semua kolom numerik dalam DataFrame
for column in df.select_dtypes(include='number').columns:
  print()
  # Buat histogram
  plt.figure(figsize=(8, 6))  # Ukuran gambar histogram
  plt.hist(df[column], bins=20, edgecolor='k')  # Jumlah bin bisa disesuaikan
  plt.title(f'Histogram of {column}')
  plt.xlabel(column)
  plt.ylabel('Frequency')
  plt.grid(True)
  plt.show()
```

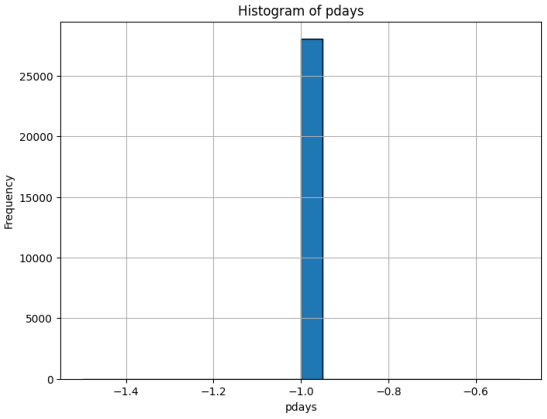


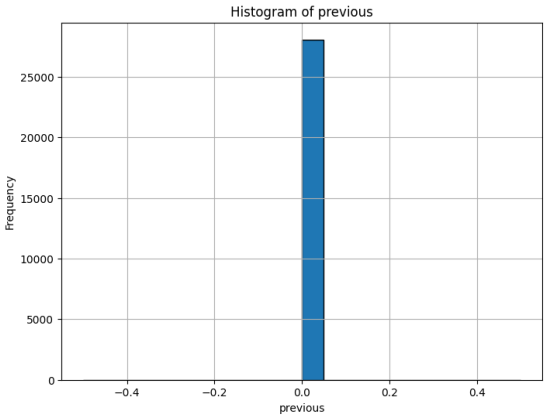


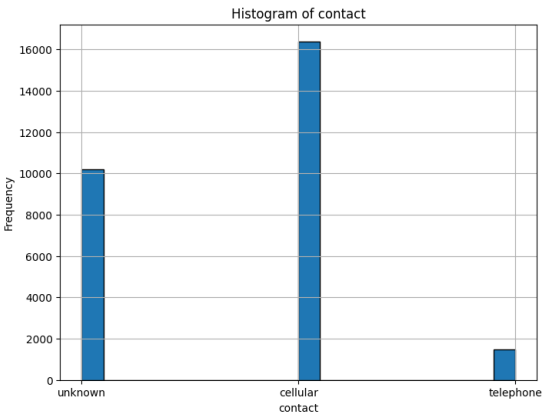


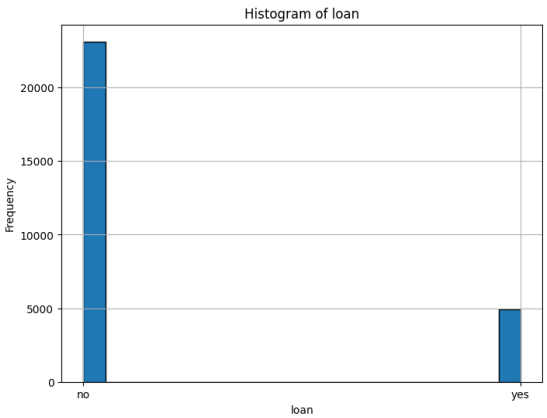


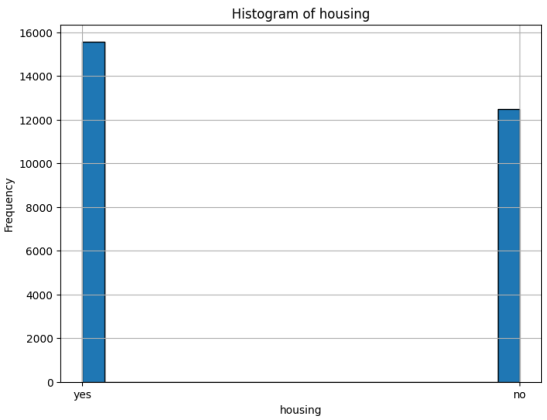

Histogram of campaign
Frequency
campaign
0
2000
4000
6000
8000
10000
1
2
3
4
5
6

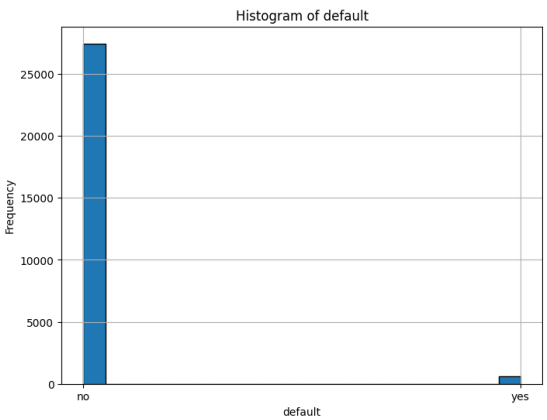

Histogram of pdays
Frequency
pdays
0
5000
10000
15000
20000
25000
−1.4
−1.2
−1.0
−0.8
−0.6

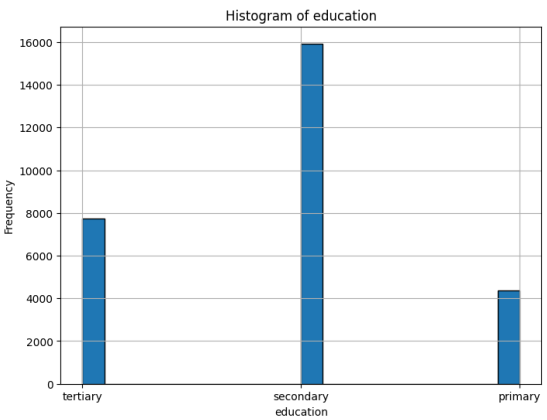

Histogram of previous
Frequency
previous
0
5000
10000
15000
20000
25000
−0.4
−0.2
0.0
0.2
0.4

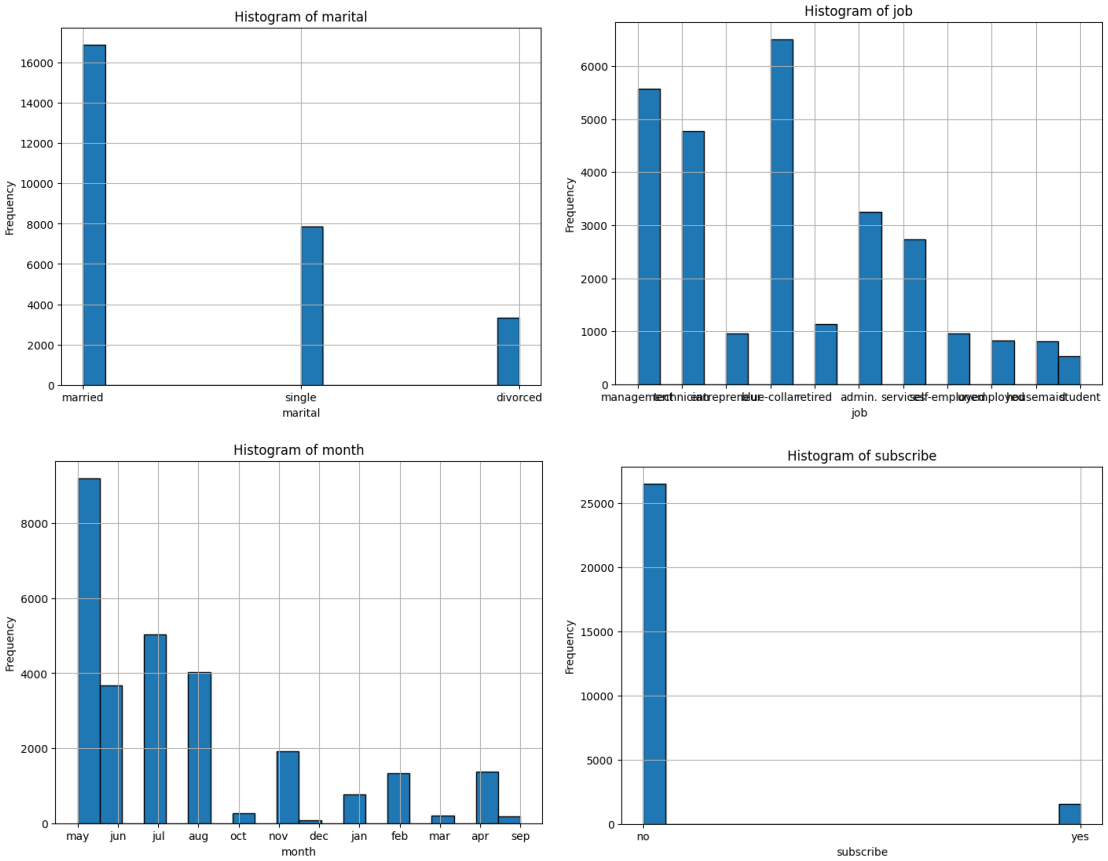

Histogram of contact
Frequency
contact
0
2000
4000
6000
8000
10000
12000
14000
16000
unknown
cellular
telephone


Histogram of loan
Frequency
loan
0
5000
10000
15000
20000
no
yes


Histogram of housing
Frequency
housing
0
2000
4000
6000
8000
10000
12000
14000
16000
yes
no


Histogram of default
Frequency
default
0
5000
10000
15000
20000
25000
no
yes


Histogram of education
Frequency
education
0
2000
4000
6000
8000
10000
12000
14000
16000
tertiary
secondary
primary

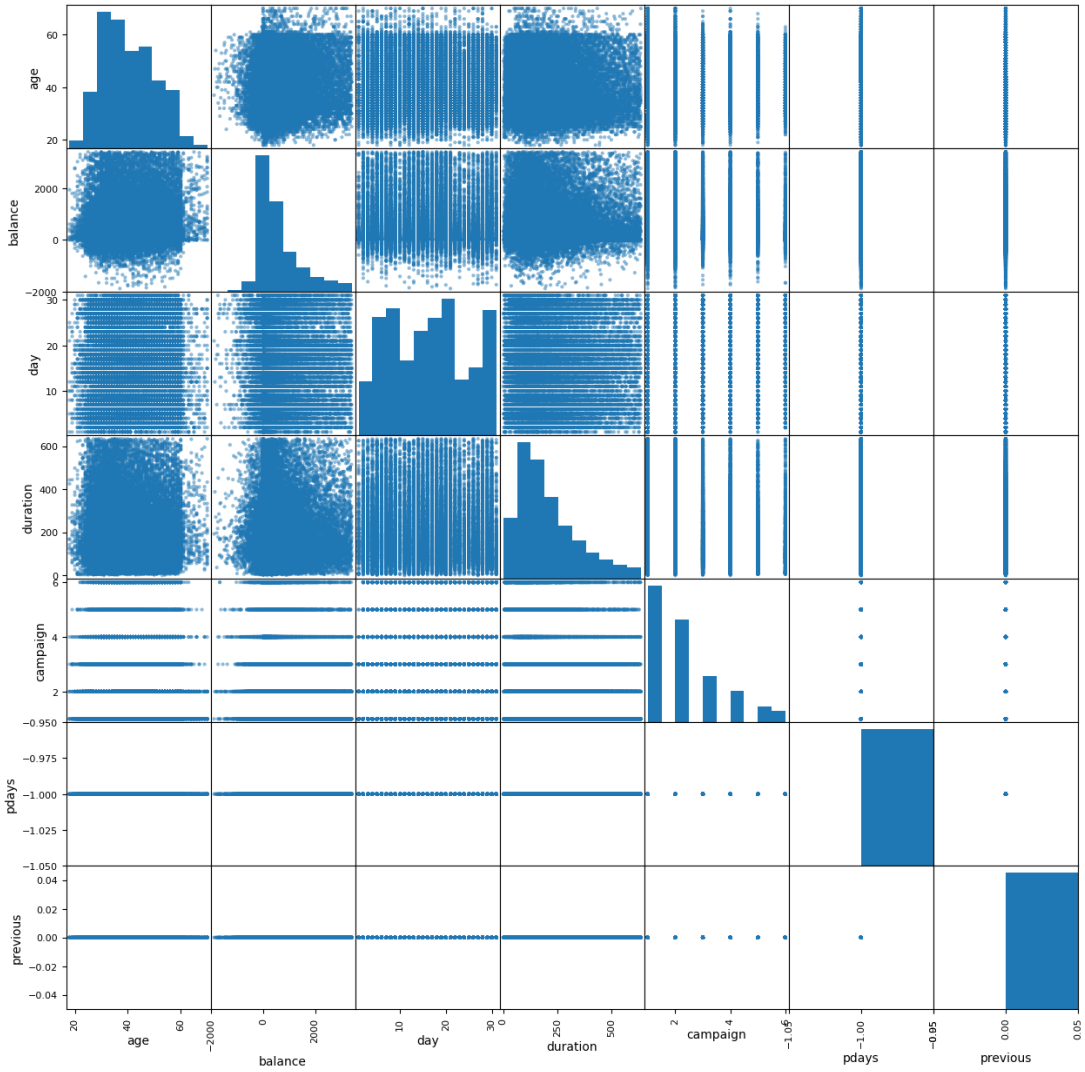


3. Analisis Bivariate dan Visualisasinya

Analisis bivariate adalah teknik statistik yang digunakan untuk mengeksplorasi hubungan atau asosiasi antara dua variabel dalam sebuah dataset. Tujuan dari analisis bivariate adalah untuk memahami bagaimana perubahan dalam suatu variabel dapat memengaruhi variabel lainnya. Analisis bivariate dilakukan dengan 2 cara, yaitu analisis antara setiap pasangan atribut independen dan setiap pasangan atribut independen dengan atribut dependen.

a. Atribut independen dan atribut independen

```
[ ]  # Scatter matrix
     pd.plotting.scatter_matrix(df, figsize = (15,15))
     plt.show()
```



Gambar di atas merupakan visualisasi dari korelasi bivariate antar atribut independen yang bertipe numeric. Atribut “pdays” dan “previous” tidak memiliki hubungan antara satu sama lain dan juga dengan atribut lain karena keduanya sama-sama hanya memiliki satu data setelah dilakukan praproses data. Angka korelasi antar atribut lainnya dapat dihitung dengan menggunakan rumus korelasi berikut.

Menghitung Korelasi Antar Atribut Independen Numeric

```
[103] # Menghitung matriks korelasi untuk semua kolom
      correlation_matrix = df.corr()

      # Menampilkan matriks korelasi
      print(correlation_matrix)
```

```
                age   balance       day  duration  campaign  pdays  previous
age        1.000000  0.081041 -0.009660 -0.051400  0.043211    NaN       NaN
balance    0.081041  1.000000  0.001830  0.018008 -0.018510    NaN       NaN
day       -0.009660  0.001830  1.000000 -0.036651  0.119843    NaN       NaN
duration  -0.051400  0.018008 -0.036651  1.000000 -0.068133    NaN       NaN
campaign   0.043211 -0.018510  0.119843 -0.068133  1.000000    NaN       NaN
pdays           NaN       NaN       NaN       NaN       NaN    NaN       NaN
previous        NaN       NaN       NaN       NaN       NaN    NaN       NaN
```

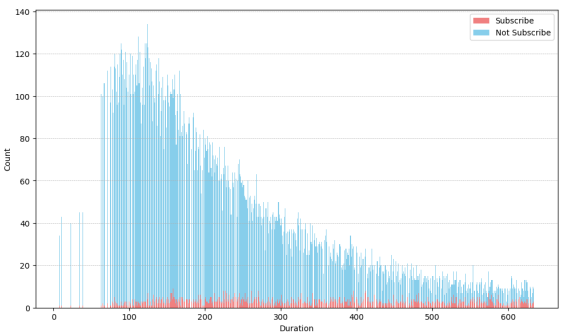
Angka korelasi digunakan untuk mengukur arah (positif atau negatif) dan kekuatan hubungan antara dua variabel, dimana angka korelasi berkisar antara -1 hingga 1. Analisis angka korelasi di atas dapat dibagi menjadi 5 kategori berikut:

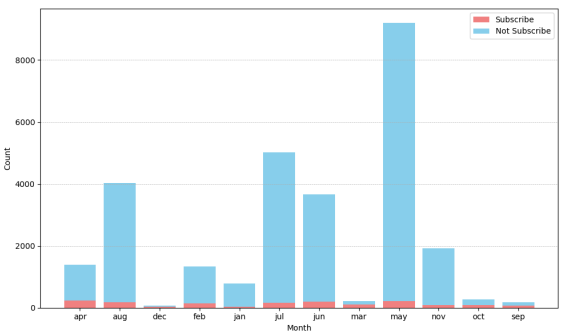
- Korelasi Positif (1): Ketika angka korelasi adalah 1, itu menunjukkan hubungan linear positif sempurna antara dua variabel. Ini berarti ketika satu variabel naik, yang lain juga naik secara linier.
- Korelasi Positif (0 hingga 1): Ketika angka korelasi berada di antara 0 dan 1 (tidak mencapai 1), itu menunjukkan hubungan positif antara dua variabel, tetapi tidak sempurna. Semakin mendekati 1, semakin kuat hubungannya.
- Tidak Ada Korelasi (0): Angka korelasi 0 menunjukkan bahwa tidak ada hubungan linier antara dua variabel. Ini berarti perubahan dalam satu variabel tidak memiliki pengaruh linier pada variabel lainnya.
- Korelasi Negatif (0 hingga -1): Ketika angka korelasi berada di antara 0 dan -1 (tidak mencapai -1), itu menunjukkan hubungan negatif antara dua variabel, tetapi tidak sempurna. Semakin mendekati -1, semakin kuat hubungannya.
- Korelasi Negatif (-1): Ketika angka korelasi adalah -1, itu menunjukkan hubungan linear negatif sempurna antara dua variabel. Ini berarti ketika satu variabel naik, yang lain turun secara linier.

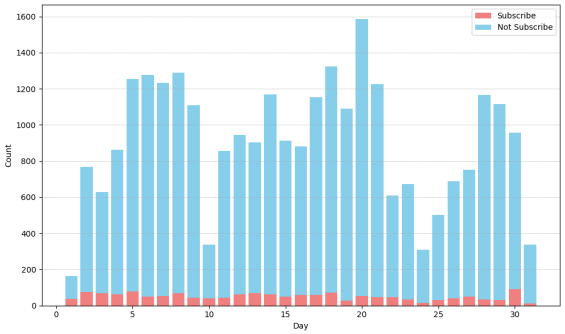
Dapat ditarik kesimpulan bahwa hubungan antara masing-masing atribut independen dengan atribut independen lainnya sangat lemah atau hampir tidak berhubungan.

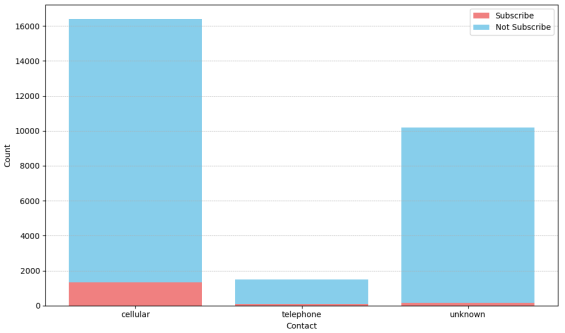
b. Atribut independen dan atribut dependen

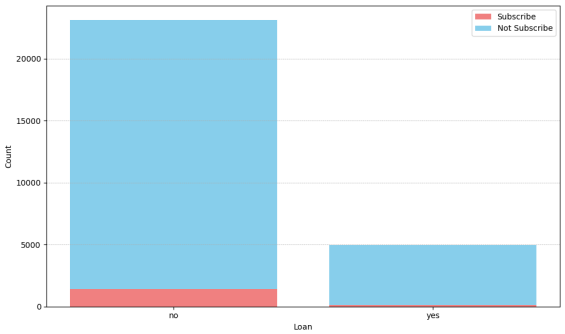
```
[ ]  def bivariate_analysis(column, type = 'bar'):
       total_subscribe = df[df['subscribe'] == 1].groupby(column)['subscribe'].count().to_frame()
       total_not_subscribe = df[df['not_subscribe'] == 1].groupby(column)['not_subscribe'].count().to_frame()
       subscribe_relationship = total_subscribe.join(total_not_subscribe)

       plt.figure(figsize=(10, 6))
       if type == 'bar':
         plt.bar(subscribe_relationship.index, subscribe_relationship['subscribe'],
                 label='Subscribe', color='lightcoral')

         plt.bar(subscribe_relationship.index, subscribe_relationship['not_subscribe'],
                 label='Not Subscribe', bottom=subscribe_relationship['subscribe'], color='skyblue')

       elif type == 'barh':
         plt.barh(subscribe_relationship.index, subscribe_relationship['subscribe'],
                  label='Subscribe', color='lightcoral')
         plt.barh(subscribe_relationship.index, subscribe_relationship['not_subscribe'],
                  label='Not Subscribe', left=subscribe_relationship['subscribe'], color='skyblue')

       plt.xlabel(column.capitalize())
       plt.ylabel('Count')
       plt.grid(True, which='both', linestyle='--', linewidth=0.5, axis='y')
       plt.legend()
       plt.tight_layout()
       plt.show()
```

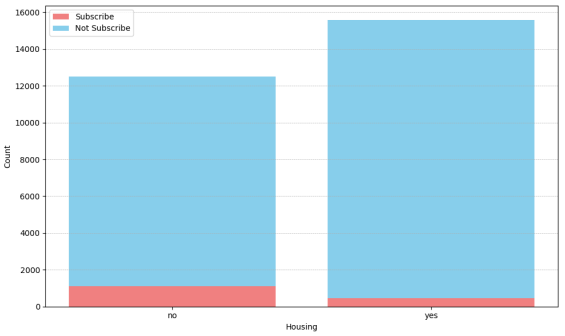


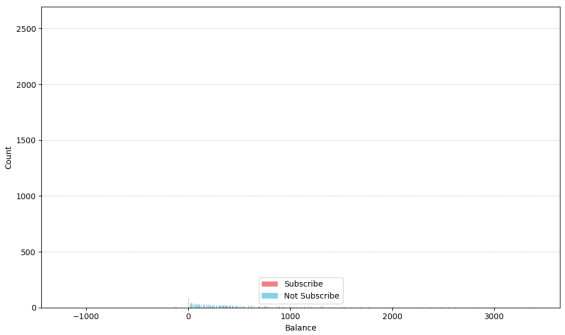


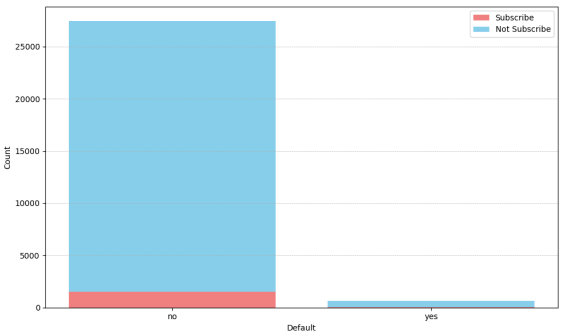

Subscribe
Not Subscribe
Count
Day

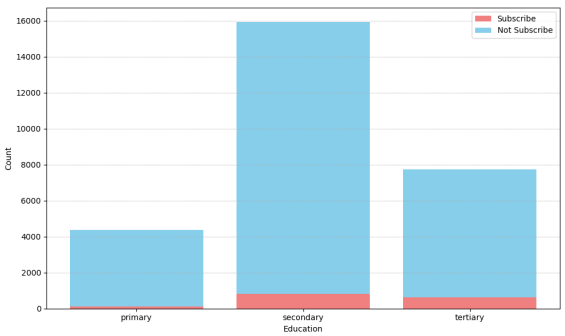

Subscribe
Not Subscribe
Count
cellular
telephone
unknown
Contact

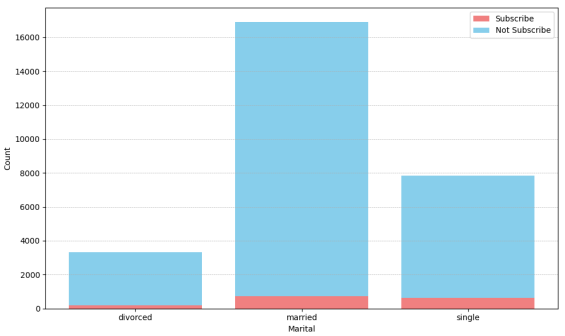

Subscribe
Not Subscribe
Count
no
yes
Loan

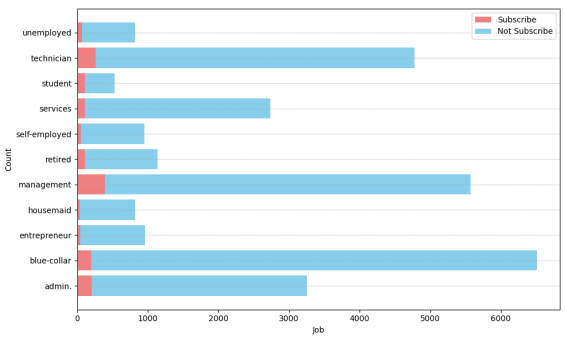

Subscribe
Not Subscribe
Count
no
yes
Housing

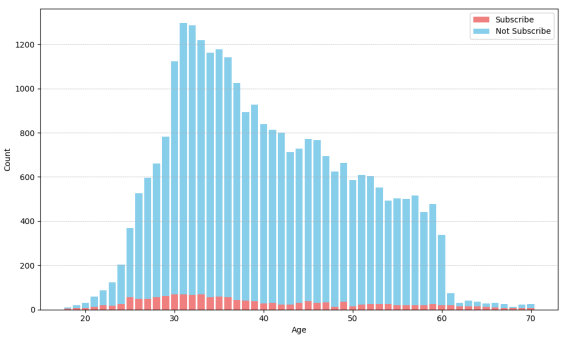

Subscribe
Not Subscribe
Count
Balance

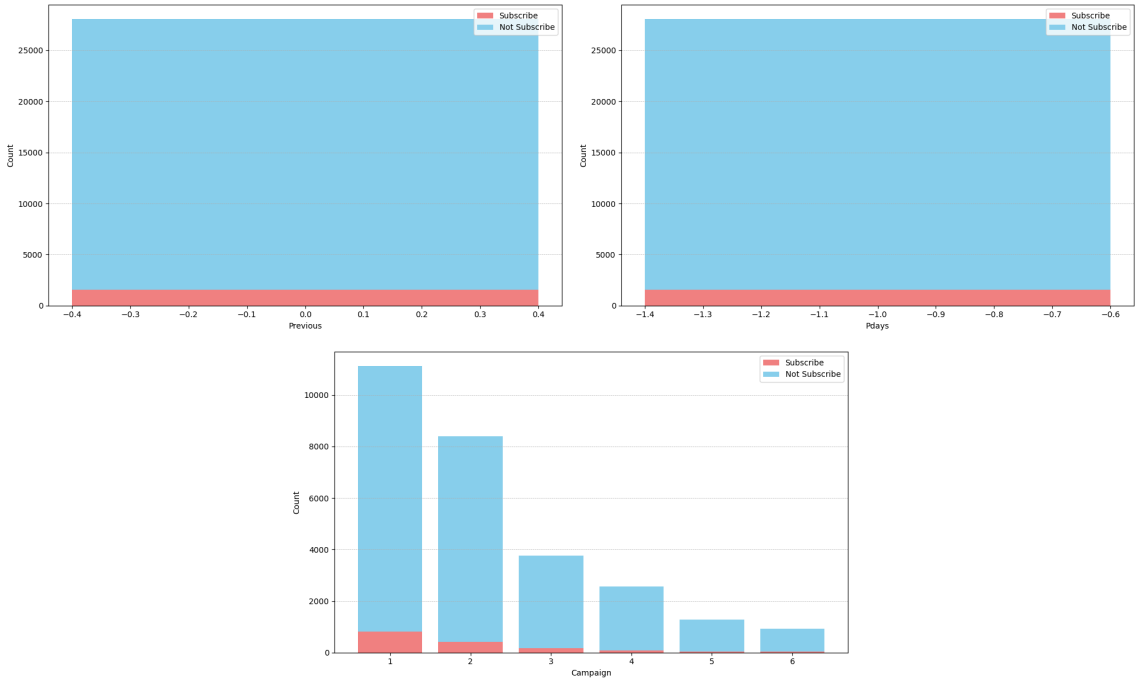

Subscribe
Not Subscribe
Count
no
yes
Default


Subscribe
Not Subscribe
Count
primary
secondary
tertiary
Education


Subscribe
Not Subscribe
Count
divorced
married
single
Marital


Subscribe
Not Subscribe
Count
unemployed
technician
student
services
self-employed
retired
management
housemaid
entrepreneur
blue-collar
admin.
Job


Subscribe
Not Subscribe
Count
Age



Gambar di atas merupakan visualisasi analisis bivariate antara masing-masing atribut independen dan atribut "subscribe", dimana yang telah melakukan *subscribe* digambarkan dengan *bar chart* merah dan yang belum *subscribe* digambarkan dengan *bar chart* biru. Berikut merupakan hasil analisisnya.

Atribut Independen	Hasil Analisis Dengan Atribut Dependen "subscribe"
duration	Durasi kontak terakhir tidak memiliki keterkaitan dengan subscribe yang dilakukan klien karena data tidak berkumpul pada rentang durasi tertentu/menyebar, sehingga tidak dapat ditemukan modus data pada durasi tertentu.
month	Klien yang terakhir dikontak pada bulan April-Juni memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
day	Klien yang terakhir dikontak pada tanggal 30 memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
contact	Klien pengguna telepon seluler memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
loan	Klien yang tidak memiliki kredit pribadi memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
housing	Klien yang tidak memiliki kredit rumah memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
balance	Klien yang memiliki balance 0-500 memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
default	Klien yang tidak memiliki kewajiban gagal bayar memiliki kecenderungan yang lebih besar untuk melakukan subscribe.

education	Klien yang menempuh pendidikan hingga sekolah menengah memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
marital	Klien yang berstatus menikah memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
job	Klien yang profesinya berkaitan dengan bidang manajemen memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
age	Klien dengan rentang usi 25-35 tahun memiliki kecenderungan yang lebih besar untuk melakukan subscribe.
previous	Jumlah kontak dengan klien pada kampanye sebelumnya tidak memiliki keterkaitan dengan subscribe yang dilakukan klien karena hanya ada satu data yang dimunculkan, yaitu -1.
pdays	Jumlah hari terlewat setelah kontak terakhir dengan klien pada kampanye sebelumnya tidak memiliki keterkaitan dengan subscribe yang dilakukan klien karena hanya ada satu data yang dimunculkan, yaitu 0.
campaign	Klien yang hanya dikontak sekali pada kampanye saat ini memiliki kecenderungan yang lebih besar untuk melakukan subscribe.