

Banking Dataset - Marketing Strategy

Laporan Final Project

1. Muhamad Iqbal
2. A Nahda La Roiba
3. Ilham Maulana
4. Clara Natalie S
5. R. Rani Indah Salamah
6. Eka Apriyani
7. Sekar Ayu Larasati
8. Firstandy Edgar Dhafa



1. Background

Permasalahan yang ingin diselesaikan

- **INEFFECTIVE CAMPAIGN** : Hanya **11%** dari total nasabah yang **setuju untuk berlangganan** dari total nasabah yang dihubungi.
- **HIGH MARKETING COST** : Perlu **efisiensi biaya marketing** karena kondisi ekonomi yang belum pulih akibat pandemi..

Peran dalam Project

- Kami selaku **tim data Scientist** yang mensupport divisi marketing Bank, bertugas membuat **model machine learning yang bisa memprediksi pelanggan mana saja yang berpotensi berlangganan deposite** dengan tepat, karena filtering secara manual dianggap tidak memiliki akurasi yang bagus dengan banyaknya profil nasabah dengan kompleksitas tinggi.

1. Background

Goal yang Ingin Dicapai

1. Mencari target pelanggan yang **potensial** berlangganan deposito.
2. Mencari **fitur-fitur penting** yang mendasari keputusan berlangganan.
3. **Mengurangi** biaya marketing pada campaign deposito berjangka tanpa kehilangan potensi pendapatan

Objektif yang Ingin Dicapai

1. **Mencari dan menentukan algoritma Machine Learning (ML) dengan nilai evaluation metric tertinggi** untuk melakukan prediksi dan rekomendasi pelanggan yang berpotensi berlangganan term-deposit
2. Menghitung jumlah **biaya dan profit** setelah pengaplikasian rekomendasi **ML** dan membandingkannya dengan **tanpa pengaplikasian** ML dan **filtering manual**
3. Menjabarkan **rekomendasi bisnis** berdasarkan fitur-fitur penting yang mendasari rekomendasi ML.

1. Background

Business Metrics untuk Mengukur Ketercapaian Objective

- **Cost Reduction = Mengurangi** jumlah biaya yang dikeluarkan tanpa kehilangan banyak **potensi** pendapatan. Impact dihasilkan dari rekomendasi **machine learning**.

EXPLORATORY DATA ANALYSIS

Tentang Dataset

Terdapat 17 kolom (16 kolom fitur dan 1 kolom target)

| | age | job | marital | education | default | balance | housing | loan |
|-------|-----|-------------|----------|-----------|---------|---------|---------|------|
| 23547 | 32 | management | married | tertiary | no | 0 | no | no |
| 16662 | 40 | blue-collar | married | secondary | no | 3131 | yes | no |
| 11145 | 48 | management | single | tertiary | no | 0 | no | no |
| 24101 | 31 | admin. | married | secondary | no | 352 | no | no |
| 2632 | 52 | admin. | divorced | secondary | no | 26 | yes | no |

| | contact | day | month | duration | campaign | pdays | previous | poutcome | y |
|--|-----------|-----|-------|----------|----------|-------|----------|----------|----|
| | cellular | 28 | aug | 15 | 13 | -1 | 0 | unknown | no |
| | cellular | 24 | jul | 401 | 1 | -1 | 0 | unknown | no |
| | unknown | 18 | jun | 96 | 3 | -1 | 0 | unknown | no |
| | telephone | 28 | oct | 60 | 1 | -1 | 0 | unknown | no |
| | unknown | 13 | may | 215 | 1 | -1 | 0 | unknown | no |

Terdapat 2 jenis tipe data, yaitu **int64** dan **object**. Semua tipe data **sudah sesuai dengan kolom fitur**.

```
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  y           45211 non-null  object  
dtypes: int64(7), object(10)
memory usage: 5.9+ MB
```

EXPLORATORY DATA ANALYSIS

Descriptive Statistic

| | 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 |

Kolom Numerical

Nilai **mean** lebih besar nilainya bila dibandingkan dengan **median**, yang mengindikasikan bahwa kurva distribusi frekuensi **menceng kanan** atau **kemencengan positif**.

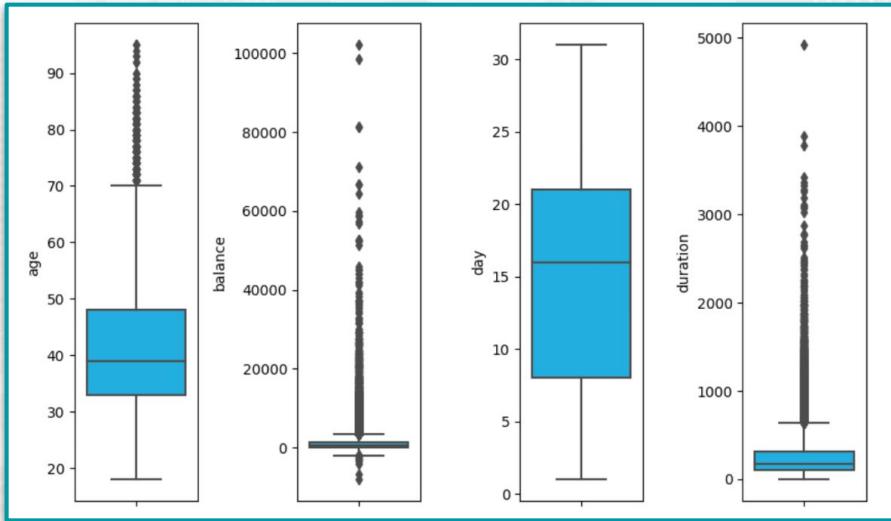
| | job | marital | education | default | housing | loan | contact | month | poutcome | y |
|--------|-------------|---------|-----------|---------|---------|-------|----------|-------|----------|-------|
| count | 45211 | 45211 | 45211 | 45211 | 45211 | 45211 | 45211 | 45211 | 45211 | 45211 |
| unique | 12 | 3 | 4 | 2 | 2 | 2 | 3 | 12 | 4 | 2 |
| top | blue-collar | married | secondary | no | yes | no | cellular | may | unknown | no |
| freq | 9732 | 27214 | 23202 | 44396 | 25130 | 37967 | 29285 | 13766 | 36959 | 39922 |

Kolom Categorical

Sebagian besar dari para nasabah tersebut **tidak mendepositkan uang mereka pada bank sebelumnya**

Univariate Analysis

BoxPlot - Numerical Columns (1)



1. Usia (Age):

Usia responden berkisar dari 18 hingga 95 tahun dengan median usia sekitar 39 tahun. Mayoritas data terpusat antara 33 hingga 48 tahun. Terdapat beberapa outlier di sisi atas distribusi, yang merupakan variasi usia yang wajar dalam populasi.

2. Saldo Rekening (Balance):

Saldo rekening memiliki variasi yang besar, dari nilai negatif hingga lebih dari 100.000. Median saldo sekitar 448 dengan sebagian besar responden memiliki saldo rendah. Terdapat outlier ekstrem di atas yang mengindikasikan beberapa responden dengan saldo sangat tinggi.

3. Hari Terakhir Dikontak (Day):

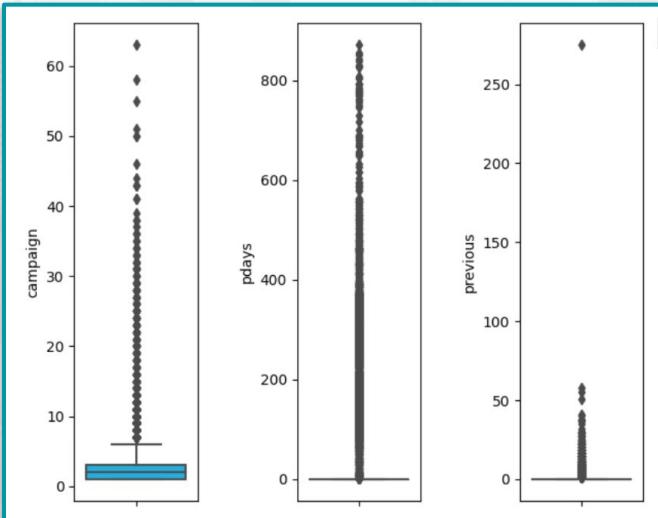
Sebagian besar responden dikontak pada hari ke-8 hingga ke-21, dan distribusi data cenderung normal tanpa outlier yang mencolok.

4. Durasi Panggilan (Duration):

Durasi panggilan bervariasi dari 0 hingga hampir 5000 detik (sekitar 82 menit). Median durasi panggilan adalah sekitar 180 detik. Terdapat outlier ekstrem yang menunjukkan panggilan dengan durasi sangat panjang.

Univariate Analysis

BoxPlot - Numerical Columns (2)



5. Jumlah Kontak Kampanye (Campaign):

Jumlah kontak berkisar dari 1 hingga 63. Median jumlah kontak sekitar 2. Terdapat beberapa outlier yang menunjukkan beberapa responden dihubungi lebih banyak daripada yang lain selama kampanye.

6. Hari Sejak Kontak Terakhir (Pdays):

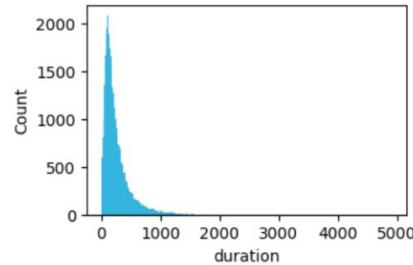
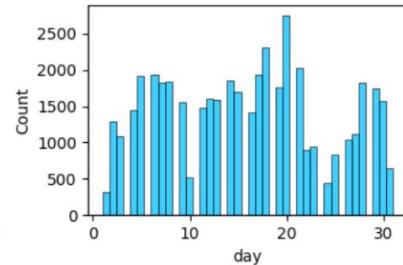
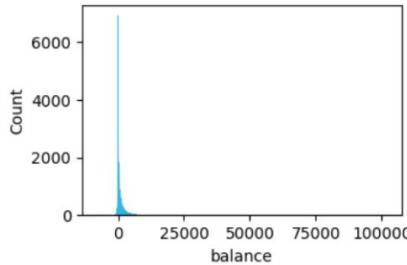
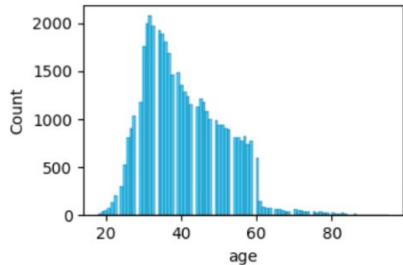
Mayoritas responden belum pernah dihubungi sebelumnya, ditunjukkan oleh median -1. Terdapat outlier ekstrem di atas yang menunjukkan sejumlah kecil responden dihubungi dalam jangka waktu sangat lama sejak kampanye sebelumnya.

7. Jumlah Kontak Sebelumnya (Previous):

Mayoritas responden tidak memiliki kontak sebelumnya, dan terdapat beberapa outlier ekstrem yang menunjukkan beberapa responden memiliki banyak kontak sebelum kampanye saat ini.

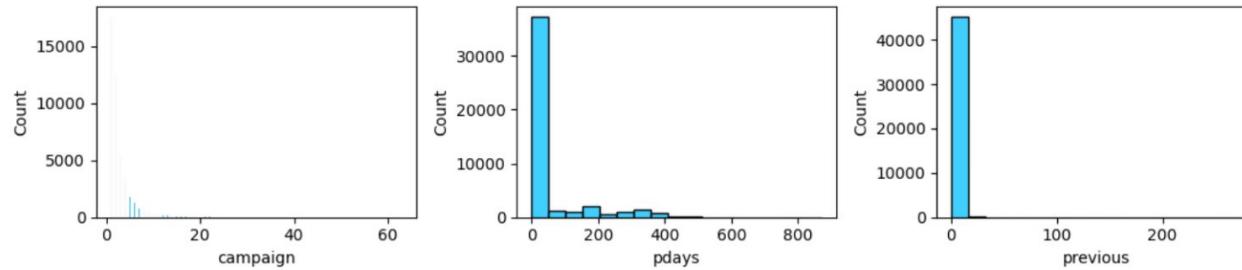
Univariate Analysis

Dalam analisis ini, fokus kami tertuju pada pemahaman lebih mendalam terhadap **karakteristik distribusi** dari fitur numerical dalam dataset.



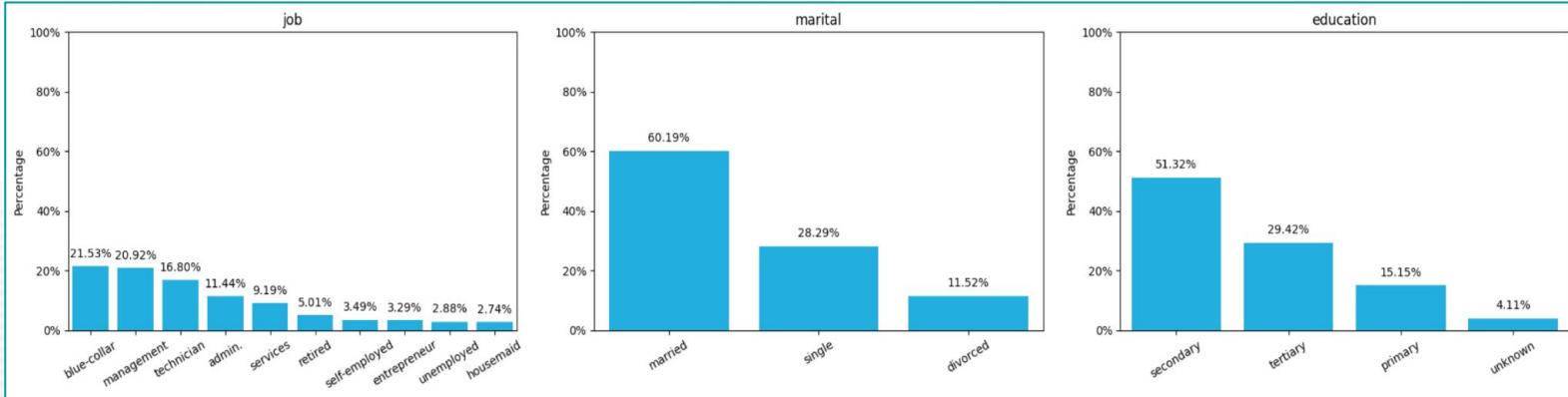
- "**age**": Distribusi umur tampaknya cukup normal, tidak ada indikasi skewness yang signifikan.
- "**balance**": Distribusi saldo tampaknya sangat skew ke kanan (positively skewed) dengan nilai maksimum yang jauh lebih tinggi dari nilai-nilai lainnya. Saat melakukan pra-pemrosesan data, dapat dilakukan penghapusan outlier yang berada di luar kisaran nilai yang masuk akal dengan menentukan batasan atas dan bawah atau melakukan transformasi data (log transformation)
- "**day**": Distribusi kolom ini tidak menunjukkan karakteristik yang mencolok.
- "**duration**": Distribusi durasi panggilan juga sangat skew ke kanan, dengan nilai maksimum yang jauh lebih tinggi dari nilai-nilai lainnya. Saat melakukan pra-pemrosesan data, perlu dilakukan penanganan outlier dengan transformasi data (misalnya log transform) atau penggunaan teknik penggantian outlier (misalnya menggunakan batas atas atau bawah yang relevan).

Univariate Analysis



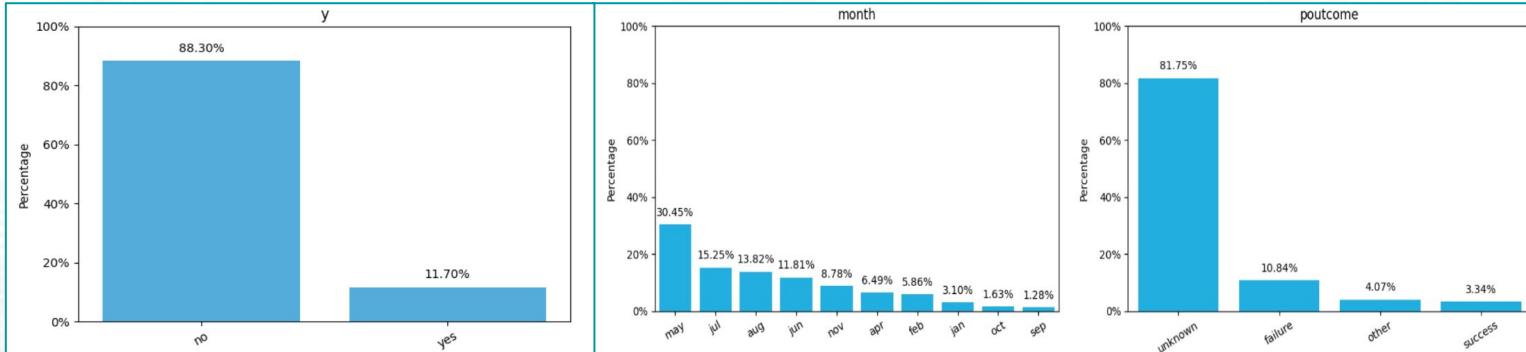
- **"campaign":** Distribusi jumlah panggilan kampanye cenderung positively skewed, dengan sebagian besar nasabah menerima panggilan dalam jumlah yang sedikit. Terdapat nilai maksimum yang jauh lebih tinggi dari nilai-nilai lainnya, menunjukkan adanya beberapa nasabah yang menerima panggilan kampanye dalam jumlah yang sangat banyak.
- **"pdays":** Distribusi nilai pdays sangat skew ke kanan, dengan sebagian besar nilai berada pada -1 (non-called). Saat melakukan pra-pemrosesan data, nilai -1 dapat diganti dengan nilai yang lebih bermakna seperti NaN untuk menandai klien yang tidak pernah dihubungi sebelumnya.
- **"previous":** Distribusi jumlah kontak sebelum kampanye saat ini juga sangat skew ke kanan, dengan nilai maksimum (275) yang jauh lebih tinggi dari nilai-nilai lainnya. Saat melakukan pra-pemrosesan data, dapat dilakukan penghapusan outlier yang berada di luar kisaran nilai yang masuk akal

Univariate Analysis



- **Variabel "job":**
 - Blue-collar (21.53%), management (20.92%), dan technician (16.80%) adalah tiga pekerjaan paling umum dalam dataset ini.
 - Housemaid (2.74%), unemployed (2.88%), entrepreneur (3.29%), dan self-employed (3.49%) adalah pekerjaan yang paling jarang ditemui dalam dataset ini.
- **Variabel "marital":**
 - Mayoritas responden (60.19%) dalam dataset ini adalah yang sudah menikah.
 - Sebesar 28.29% responden adalah single (belum menikah), dan 11.52% adalah yang bercerai.
- **Variabel "education":**
 - Sekitar setengah dari responden (51.32%) memiliki pendidikan tingkat menengah (secondary).
 - Sekitar 29.42% memiliki pendidikan tingkat atas (tertiary).
 - Sekitar 15.15% hanya memiliki pendidikan dasar (primary).
 - Ada juga sejumlah kecil responden (4.11%) yang pendidikan mereka tidak diketahui (unknown)

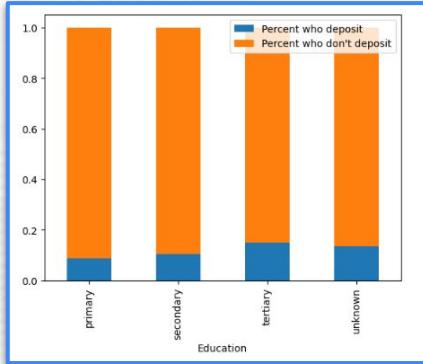
Univariate Analysis



- **Variabel "y":**
 - Mayoritas (88.30%) responden tidak berlangganan deposito atau layanan yang ditawarkan.
 - Hanya sekitar 11.70% yang berlangganan.
- **Variabel "month":**
 - Bulan terbanyak dalam dataset ini adalah Mei (30.45%).
 - Bulan-bulan lainnya memiliki persentase yang lebih rendah, seperti Juli (15.25%), Agustus (13.82%), dan Juni (11.81%).
- **Variabel "poutcome":**
 - Sebagian besar responden (81.75%) memiliki hasil pemasaran sebelumnya yang tidak diketahui (unknown).
 - Hanya sekitar 10.84% yang mengalami kegagalan (failure).
 - Persentase lainnya termasuk "other" (4.07%) dan "success" (3.34%).

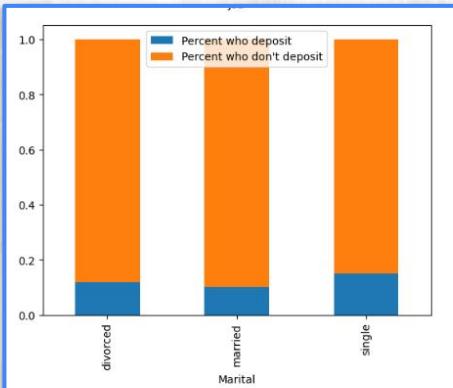
Multivariate Analysis

Stacked Bar Chart - Categorical Columns



Feature 'Education'

Jika dilihat dari conversion ratenya, maka dapat disimpulkan bahwa semakin tinggi pendidikan nasabah, peluang convertnya juga semakin tinggi (Tertiary atau biasanya tingkat perguruan tinggi). Pendidikan terakhir yang tinggi juga dapat mencerminkan stabilitas keuangan dan minat yang lebih besar dalam produk atau layanan yang ditawarkan, yang pada akhirnya dapat meningkatkan kemungkinan nasabah untuk melakukan tindakan konversi. Hal tersebut mungkin karena semakin tinggi pendidikan formal, semakin tinggi pula iterasi keuangan seseorang.

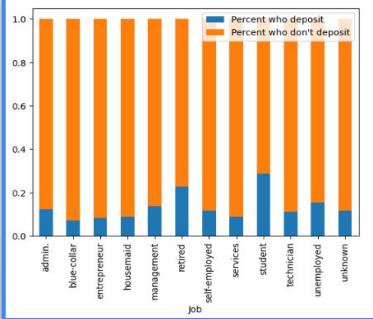


Feature 'Marital'

Jika dilihat dari marital status, nasabah dengan status 'single' memiliki persentase paling tinggi meskipun tidak begitu jauh dibanding yang lain, mungkin karena banyak dari nasabah single memiliki spare uang lebih untuk bisa di investasikan. Namun, perbedaan conversion rate tidak terlihat signifikan.

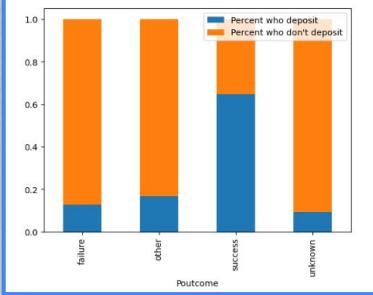
Multivariate Analysis

Stacked Bar Chart - Categorical Columns



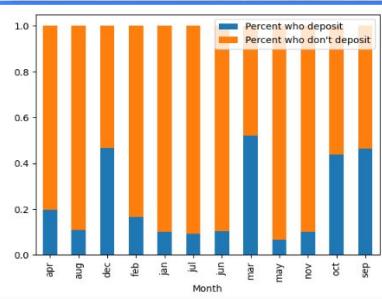
Feature 'Job'

Secara conversion rate, nasabah yang berkesibukan sebagai pelajar memiliki persentase ketertarikan tertinggi untuk berlangganan deposito. Hal ini mungkin karena berlangganan deposito menjadi target yang ingin dilakukan saat pertama kali membuka rekening di bank (mungkin berpikir untuk investasi sejak dulu). Ini juga berarti peluang untuk nasabah pelajar untuk convert adalah yang tertinggi. Untuk retired juga persentasenya kedua tertinggi, hal ini mungkin nasabah menyimpan tabungan pensiunnya pada deposito berjangka.



Feature 'Poutcome'

Hasil sukses pada campaign terakhir tentu menjadi tanda yang sesuai dengan paling banyaknya nasabah yang berlangganan, unggul jauh dari hasil lainnya.

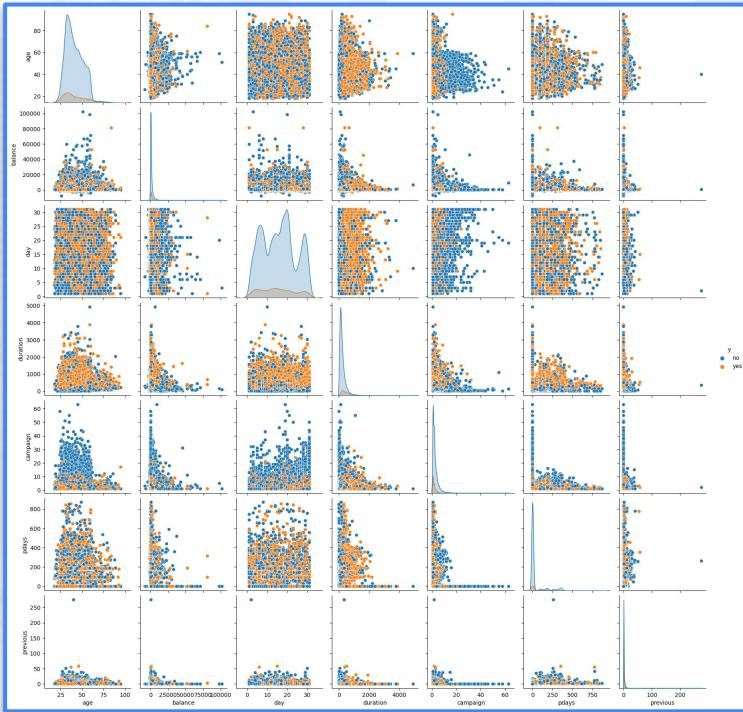


Feature 'month'

Bulan maret menjadi bulan terakhir kontak yang dilakukan dengan hasil konversi berlangganan yang tertinggi. Bulan September, Oktober dan Desember juga cukup tinggi. Sisanya rendah.

Multivariate Analysis

Scatter plot **tidak bisa menunjukkan pola yang membedakan** antara nasabah yang mendeposikan uangnya, dengan yang tidak.



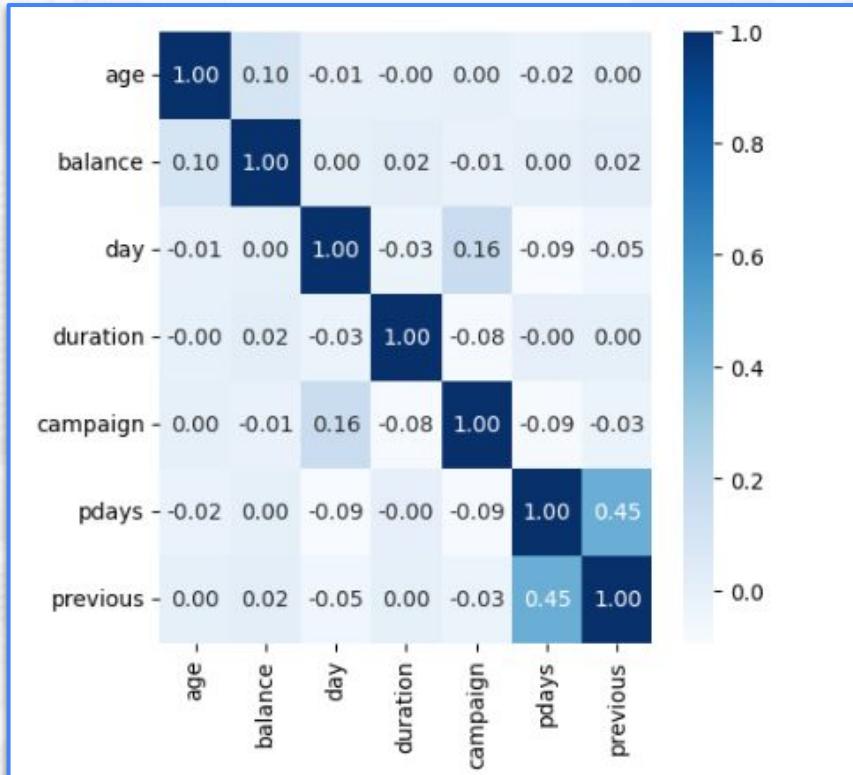
Heatmap & Pairplot - Numerical Columns

Observation :

- Pada hubungan feature 'age' dan 'duration', terlihat ada pola yang menarik dimana rata-rata nasabah yang tidak deposit memiliki nilai durasi yang rendah, sedangkan semakin lama durasi, kemungkinan deposit menjadi semakin tinggi tidak peduli berapapun usianya.
- Hubungan feature 'day' dan 'duration' juga mirip dengan yang di atas, dimana semakin lama durasi semakin tinggi kemungkinan nasabah untuk deposit tidak peduli di hari apa kontak terakhir dilakukan.
- Hubungan feature 'age' dan 'campaign' juga memiliki pola yang sedikit menarik, dimana semakin banyak campaign atau kontak yang dilakukan justru kecenderungan nasabah untuk berlangganan deposit berjangka semakin kecil tidak peduli berapapun usia nasabah.

Multivariate Analysis

Heatmap - Numerical Columns



Observations :

- Berdasarkan heatmap korelasi untuk feature numerik di atas, terlihat bahwa feature 'pdays' dan 'previous' memiliki korelasi yang masuk dalam kategori moderate correlation (0.45).
- Selain korelasi 2 feature yang telah disebutkan, sisanya berkorelasi lemah atau sangat lemah.

Multivariate Analysis

Tindak Lanjut

Feature 'pdays' dipertimbangkan untuk tidak dimasukkan dalam analisis sedangkan feature 'previous' dipertahankan. Hal ini karena dari sisi persebaran data, 'pdays' memiliki banyak outliers yang lebih ekstrim, memiliki nilai mayoritas -1 (kurang relevan untuk merepresentasikan nasabah yang belum pernah dihubungi) dan juga memiliki standar deviasi yang sangat besar ketimbang 'previous'.

Feature 'duration' juga dipertimbangkan untuk tidak dimasukkan dalam analisis dikarenakan banyaknya nasabah yang belum pernah dihubungi di campaign sebelumnya sehingga banyak nilai 0, namun ini butuh penyelidikan lebih lanjut.

Data Cleansing (Handle missing values)

```
[4] 1 # Missing values
2 missing_values = df.isna().sum()
3 missing_values

age          0
job          0
marital      0
education    0
default      0
balance      0
housing      0
loan         0
contact      0
day          0
month        0
duration     0
campaign     0
pdays        0
previous     0
poutcome     0
y             0
dtype: int64
```

Tidak ada missing values yang terdapat pada dataset, sehingga tidak perlu untuk melakukan tindak lebih lanjut.

Data Cleansing (Handle duplicated data)

```
[5] 1 # Menghitung jumlah data yang duplikat
 2 duplicate_count = df.duplicated().sum()
 3 print("Jumlah data yang duplikat:", duplicate_count)
```

```
Jumlah data yang duplikat: 0
```

Tidak ada data yang duplikat pada dataset, sehingga tidak perlu untuk melakukan tindak lebih lanjut.

Data Cleansing (Feature transformation)

| age | job | marital | education | default | balance | housing | loan | contact | day | month | duration | campaign | pdays | previous | poutcome | y |
|-----|-------------|----------|-----------|---------|---------|---------|------|---------|-----|-------|----------|----------|-------|----------|----------|-----|
| 53 | services | divorced | primary | no | -291 | yes | yes | unknown | 7 | may | 591 | 1 | -1 | 0 | unknown | yes |
| 49 | services | married | secondary | no | -8 | yes | no | unknown | 8 | may | 1119 | 1 | -1 | 0 | unknown | yes |
| 43 | blue-collar | married | primary | no | -192 | yes | no | unknown | 8 | may | 1120 | 2 | -1 | 0 | unknown | yes |
| 32 | blue-collar | married | secondary | yes | -1 | yes | no | unknown | 9 | may | 653 | 1 | -1 | 0 | unknown | yes |
| 28 | blue-collar | single | secondary | no | -197 | yes | no | unknown | 9 | may | 2016 | 2 | -1 | 0 | unknown | yes |

Pemutusan untuk mengubah nilai **negatif** pada saldo menjadi nilai **absolut** dikarenakan terdapat beberapa pelanggan dengan nilai negatif yang memiliki hasil **target (y)** yang bernilai '**yes**'. Dalam hal ini mungkin terjadi **kesalahan input** dikarenakan pelanggan dengan saldo negatif tidak mungkin atau tidak memenuhi syarat untuk membuka term deposit.

Before

| | age | balance | day | duration | campaign | pdays | previous |
|-------|--------------|---------------|--------------|--------------|--------------|---------|----------|
| count | 36954.000000 | 36954.000000 | 36954.000000 | 36954.000000 | 36954.000000 | 36954.0 | 36954.0 |
| mean | 40.932430 | 1318.788846 | 16.145424 | 257.726119 | 2.921957 | -1.0 | 0.0 |
| std | 10.430218 | 3039.557077 | 8.372554 | 262.256406 | 3.325791 | 0.0 | 0.0 |
| min | 18.000000 | -8019.000000 | 1.000000 | 0.000000 | 1.000000 | -1.0 | 0.0 |
| 25% | 33.000000 | 55.000000 | 9.000000 | 101.000000 | 1.000000 | -1.0 | 0.0 |
| 50% | 39.000000 | 414.000000 | 17.000000 | 177.000000 | 2.000000 | -1.0 | 0.0 |
| 75% | 49.000000 | 1358.000000 | 22.000000 | 318.000000 | 3.000000 | -1.0 | 0.0 |
| max | 95.000000 | 102127.000000 | 31.000000 | 4918.000000 | 63.000000 | -1.0 | 0.0 |

After

| | age | balance | day | duration | campaign | pdays | previous |
|-------|--------------|---------------|--------------|--------------|--------------|---------|----------|
| count | 36954.000000 | 36954.000000 | 36954.000000 | 36954.000000 | 36954.000000 | 36954.0 | 36954.0 |
| mean | 40.932430 | 1375.862451 | 16.145424 | 257.726119 | 2.921957 | -1.0 | 0.0 |
| std | 10.430218 | 3014.151556 | 8.372554 | 262.256406 | 3.325791 | 0.0 | 0.0 |
| min | 18.000000 | 0.000000 | 1.000000 | 0.000000 | 1.000000 | -1.0 | 0.0 |
| 25% | 33.000000 | 123.000000 | 9.000000 | 101.000000 | 1.000000 | -1.0 | 0.0 |
| 50% | 39.000000 | 457.000000 | 17.000000 | 177.000000 | 2.000000 | -1.0 | 0.0 |
| 75% | 49.000000 | 1373.000000 | 22.000000 | 318.000000 | 3.000000 | -1.0 | 0.0 |
| max | 95.000000 | 102127.000000 | 31.000000 | 4918.000000 | 63.000000 | -1.0 | 0.0 |

Data Cleansing (Feature encoding “Target”)

▼ Label Encode Variable Target (y)

```
[ ] #dataframe before label encoding 'y'
df.tail()
```

| | age | job | marital | education | default | balance | housing | loan | contact | day | month | duration | campaign | pdays | previous | poutcome | y |
|-------|-----|-------------|----------|-----------|---------|---------|---------|------|-----------|-----|-------|----------|----------|-------|----------|----------|-----|
| 45203 | 23 | student | single | tertiary | no | 113 | no | no | cellular | 17 | nov | 266 | 1 | -1 | 0 | unknown | yes |
| 45205 | 25 | technician | single | secondary | no | 505 | no | yes | cellular | 17 | nov | 386 | 2 | -1 | 0 | unknown | yes |
| 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 |
| 45209 | 57 | blue-collar | married | secondary | no | 668 | no | no | telephone | 17 | nov | 508 | 4 | -1 | 0 | unknown | no |

```
▶ from sklearn.preprocessing import LabelEncoder

# Initialize LabelEncoder
encoder = LabelEncoder()

# Fit and transform the 'y' column
df['y'] = encoder.fit_transform(df['y'])

# Print the encoded DataFrame
df.tail()
```

| | age | job | marital | education | default | balance | housing | loan | contact | day | month | duration | campaign | pdays | previous | poutcome | y |
|-------|-----|-------------|----------|-----------|---------|---------|---------|------|-----------|-----|-------|----------|----------|-------|----------|----------|---|
| 45203 | 23 | student | single | tertiary | no | 113 | no | no | cellular | 17 | nov | 266 | 1 | -1 | 0 | unknown | 1 |
| 45205 | 25 | technician | single | secondary | no | 505 | no | yes | cellular | 17 | nov | 386 | 2 | -1 | 0 | unknown | 1 |
| 45206 | 51 | technician | married | tertiary | no | 825 | no | no | cellular | 17 | nov | 977 | 3 | -1 | 0 | unknown | 1 |
| 45207 | 71 | retired | divorced | primary | no | 1729 | no | no | cellular | 17 | nov | 456 | 2 | -1 | 0 | unknown | 1 |
| 45209 | 57 | blue-collar | married | secondary | no | 668 | no | no | telephone | 17 | nov | 508 | 4 | -1 | 0 | unknown | 0 |

Untuk feature ‘y’, kami melakukan label encoding karena feature tersebut akan kami jadikan target, dimana untuk “yes” di encode 1 dan “no” di encode 0.

Data Cleansing (Feature encoding “predictor”)

Default, housing, loan adl fitur binary sehingga drop first column

```
encodee1 = ['default', 'housing', 'loan']

for e in encodee1:
    ohe = pd.get_dummies(df[e], prefix=e, drop_first=True)
    df = pd.concat([df, ohe], axis=1) # Concatenate the one-hot encoded columns

df = df.drop(encodee1, axis=1) # Drop the original categorical columns

print(df)
```

| default_no | default_yes | housing_no | housing_yes | loan_no | lo |
|------------|-------------|------------|-------------|---------|----|
| 1 | 0 | 1 | 0 | 1 | |
| 1 | 0 | 1 | 0 | 0 | |
| 1 | 0 | 1 | 0 | 1 | |
| 1 | 0 | 1 | 0 | 1 | |
| 1 | 0 | 1 | 0 | 1 | |

Job, marital, education, contact, poutcome adalah multiple class sehingga tidak drop first column

```
encodee2 = ['job', 'marital', 'education', 'contact', 'poutcome']

for e in encodee2:
    ohe = pd.get_dummies(df[e], prefix=e, drop_first=False) # Drop the first column
    df = pd.concat([df, ohe], axis=1) # Concatenate the one-hot encoded columns

df = df.drop(encodee2, axis=1) # Drop the original categorical columns

print(df)
```

| contact_cellular | contact_telephone | contact_unknown | poutcome_failure | poutcome_other | poutcome_success | poutcome_unknown |
|------------------|-------------------|-----------------|------------------|----------------|------------------|------------------|
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 0 | 1 | 0 | 0 |

Feature Engineering

Feature Engineering (Feature extraction)

month_quartal

```

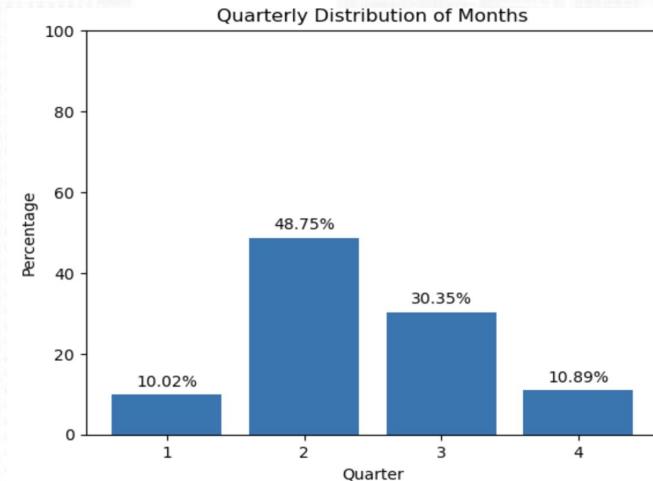
quartal_counts = df['month_quartal'].value_counts().sort_index()
total_respondents = len(df)
month_percentage = quartal_counts / total_respondents * 100

# Create the bar plot with percentage labels
plt.bar(month_percentage.index, month_percentage.values)
plt.xlabel('Quarter')
plt.ylabel('Percentage')
plt.title('Quarterly Distribution of Months')

# Adding percentage labels on top of the bars
for i, percentage in enumerate(month_percentage):
    plt.text(i + 1, percentage + 2, f'{percentage:.2f}%', ha='center')

plt.xticks(month_percentage.index)
plt.ylim(0, 100)
plt.show()

```



Fitur month_quartal diekstraksi dari fitur month, dengan pertimbangan untuk melihat banyaknya nasabah yang dihubungi pada quarter tertentu. Hasilnya, quarter terbanyak jatuh pada quarter 2, yang berarti nasabah paling banyak dihubungi pada bulan April, Mei, Juni.

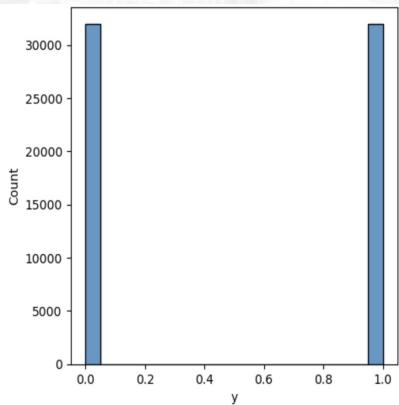
Handling Class Imbalance (y)

```
# x nya adalah atribut yang me
# y nya adalah target itu sena
X = df.drop(['y'], axis = 1)
y = df['y']

print(X.shape)
print(y.shape)
```

(45211, 23)
(45211,)

Menentukan variabel dataset



Hasil akhir menunjukkan jumlah yang seimbang



```
[ ] from sklearn.model_selection import train_test_split
from imblearn.over_sampling import SMOTE
from collections import Counter

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

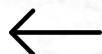
# Apply SMOTE to the training data
smote = SMOTE(random_state=42)
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)

# Print class distribution before and after applying SMOTE
print("Class distribution before SMOTE:", Counter(y_train))
print("Class distribution after SMOTE:", Counter(y_train_resampled))
```

Pertama, kita mengimpor perpustakaan yang diperlukan untuk menggunakan SMOTE dan memeriksa distribusi kelas. Kemudian, kita memuat dataset dan membaginya menjadi subset train dan test. Penerapan SMOTE pada data train menggunakan metode fit_resample.



```
print(pd.Series(y_train_resampled).value_counts())
0    31970
1    31970
Name: y, dtype: int64
```



Hasil akhir menunjukkan jumlah yang sama antara kedua kelas, yaitu 'yes' dan 'no' dengan jumlah yang sama yaitu 26846. Jumlah total sampel dapat meningkat secara signifikan, karena SMOTE menghasilkan sampel sintetik dengan melakukan interpolasi antar sampel yang ada di kelas minoritas, dan dengan melakukan hal tersebut, berpotensi membuat sampel baru dalam jumlah besar.

Feature Engineering (Scaling Dataset)

Robust Scaler

```
from sklearn.preprocessing import RobustScaler

scaler = RobustScaler()
columns_to_scale = ['age', 'balance']

scaler.fit(X_train_resampled[columns_to_scale])
for i, df in enumerate(dflist):

    data_to_scale = df[columns_to_scale]

    scaled_data = scaler.transform(data_to_scale)

    df[columns_to_scale] = scaled_data
```

Robust scaler perlu digunakan pada kolom '**age**' dan '**balance**' untuk mengatasi distribusi skewed. Dengan menggunakan median dan IQR, robust scaler dapat mengurangi pengaruh outlier pada proses penskalaan dan membantu menghasilkan data yang memiliki distribusi yang lebih terpusat dan terdistribusi secara merata.

Feature Engineering (Scaling Dataset)

Standard Scaler

```
from sklearn.preprocessing import StandardScaler

standard_scaler = StandardScaler()
columns_to_standard_scale = [i for i in df.columns.to_list() if not (i in columns_to_scale or i in encodeel or i in
standard_scaler.fit(X_train_resampled[columns_to_standard_scale])
for i, df in enumerate(dflist):
    data_to_standard_scale = df[columns_to_standard_scale]

    standard_scaled_data = standard_scaler.transform(data_to_standard_scale)

df[columns_to_standard_scale] = standard_scaled_data
```

| | age | balance | housing_yes | loan_yes | marital_divorced | marital_married | marital_single | education_primary |
|-------|--------------|--------------|--------------|--------------|------------------|-----------------|----------------|-------------------|
| count | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 |
| mean | 0.172718 | 0.495253 | 0.341063 | 0.207687 | 0.147120 | 0.221982 | 0.075593 | 0.184208 |
| std | 0.624633 | 1.870249 | 1.018940 | 1.231688 | 1.203571 | 0.979170 | 1.039963 | 1.211495 |
| min | -1.176471 | -5.267199 | -0.798786 | -0.330313 | -0.287100 | -0.982081 | -0.577591 | -0.327773 |
| 25% | -0.294118 | -0.297297 | -0.798786 | -0.330313 | -0.287100 | -0.982081 | -0.577591 | -0.327773 |
| 50% | 0.058824 | -0.066339 | 1.251900 | -0.330313 | -0.287100 | 1.018246 | -0.577591 | -0.327773 |
| 75% | 0.588235 | 0.535627 | 1.251900 | -0.330313 | -0.287100 | 1.018246 | 1.731329 | -0.327773 |
| max | 3.352941 | 62.390049 | 1.251900 | 3.027432 | 3.483111 | 1.018246 | 1.731329 | 3.050888 |

| | poutcome_success | poutcome_unknown | duration_bin_0-5mins | duration_bin_5-10mins | duration_bin_10-15mins | duration_bin_15-20mins | duration_bin_>=20mins |
|--|------------------|------------------|----------------------|-----------------------|------------------------|------------------------|-----------------------|
| | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 | 45211.000000 |
| | -0.030864 | 0.361357 | 0.507913 | 0.098136 | 0.020939 | 0.030967 | 0.032019 |
| | 0.923567 | 0.806997 | 0.892014 | 1.085965 | 1.043033 | 1.115674 | 1.151152 |
| | -0.202598 | -1.346485 | -0.948583 | -0.425430 | -0.225039 | -0.124113 | -0.097246 |
| | -0.202598 | 0.742674 | -0.948583 | -0.425430 | -0.225039 | -0.124113 | -0.097246 |
| | -0.202598 | 0.742674 | 1.054204 | -0.425430 | -0.225039 | -0.124113 | -0.097246 |
| | 4.935884 | 0.742674 | 1.054204 | 2.350565 | 4.443677 | 8.057141 | 10.283218 |

Standard Scaler perlu digunakan pada kolom sisanya untuk mengubah skala data menjadi standar dan sesuai dengan asumsi distribusi normal. data akan diubah sehingga memiliki rata-rata nol dan simpangan baku satu, sesuai dengan distribusi normal standar.

Feature Engineering (Feature selection)

- Feature '**pdays**' dipertimbangkan untuk **tidak dimasukkan** dalam modelling karena memiliki korelasi tinggi dengan fitur 'previous' dan juga memiliki nilai **majoritas -1** dan **standar deviasi** yang sangat besar.
- Feature '**age**' dan '**balance**' dibuang karena kurang berkontribusi terhadap model
- Feature '**previous**' dibuang karena bias dengan feature '**campaign**'
- Feature '**day**' dibuang karena tidak memberikan info penting
- Feature '**education**' dan '**contact**' dibuang karena berisi nilai "**unknown**" yang tidak terdefinisi

RFE (using GLM, RF, XGB)

```

import numpy as np
from sklearn.feature_selection import RFE
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier

# Generate a sample dataset

# Create a logistic regression model
model_glm = LogisticRegression()
model_rfa = RandomForestClassifier(random_state=69)

# Create the RFE model and specify the number of features to select (e.g., 10)
#0.7980.8x di n_features=28
#0.7880.971639 di n_features=26
rfe_rfa = RFE(model_rfa, n_features_to_select=26)

# Fit the RFE model to the data
rfe_rfa.fit(X_train_resampled, y_train_resampled)

# Get the selected features
selected_features_glm = np.where(rfe_glm.support_)[0]
selected_features_rfa = np.where(rfe_rfa.support_)[0]

print("Selected Features with RFE :", selected_features_rfa)

```



| # | Column | Non-Null Count | Dtype |
|----|------------------|----------------|---------|
| 0 | housing | 63930 non-null | int64 |
| 1 | loan | 63930 non-null | int64 |
| 2 | duration | 63930 non-null | float64 |
| 3 | campaign | 63930 non-null | float64 |
| 4 | job_admin. | 63930 non-null | int64 |
| 5 | job_blue-collar | 63930 non-null | int64 |
| 6 | job_management | 63930 non-null | int64 |
| 7 | job_technician | 63930 non-null | int64 |
| 8 | marital_divorced | 63930 non-null | int64 |
| 9 | marital_married | 63930 non-null | int64 |
| 10 | marital_single | 63930 non-null | int64 |
| 11 | poutcome_failure | 63930 non-null | float64 |
| 12 | poutcome_other | 63930 non-null | float64 |
| 13 | poutcome_unknown | 63930 non-null | int64 |
| 14 | month_quartal_1 | 63930 non-null | int64 |
| 15 | month_quartal_2 | 63930 non-null | int64 |
| 16 | month_quartal_3 | 63930 non-null | int64 |
| 17 | month_quartal_4 | 63930 non-null | int64 |

- kami drop fitur **age, day, balance, education** dan **previous** karena tidak memberikan kontribusi yang baik pada model setelah beberapa kali percobaan.
- Kami juga drop **semua fitur contact**, karena salah satunya memiliki fitur **contact_unknown** yang secara domain knowledge tidak terdefinisi.

Modeling

```
[ ] from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier

[ ] models = {
    "logistic": LogisticRegression(solver='sag', max_iter=250),
    "RandomForest" :RandomForestClassifier(random_state=42),
    "XGBoost" : XGBClassifier(),
}
```

Kami memutuskan untuk menerapkan tiga jenis model yang berbeda dalam uji coba kami, dan nantinya kami akan melakukan evaluasi ulang terhadap ketiga model tersebut, dengan metrics yang telah ditentukan. Tiga jenis model yang akan kami gunakan adalah **Logistic Regression**, **Random Forest**, dan **XGBoost**.

Modeling (Metrics)

```
[ ] models_precision = {}
models_cm = {}
models_cv_precision_test = {}
models_cv_precision_train = {}

[ ] from sklearn.metrics import confusion_matrix, recall_score, precision_score
from sklearn.model_selection import cross_validate, StratifiedKFold

def evaluate_classifier(model_name, model):
    Y_pred = model.predict(X_test)
    models_precision[model_name] = precision_score(y_test, Y_pred)

    kfold = StratifiedKFold(n_splits=5, shuffle=True)
    precision_cv = cross_validate(model, X_train_resampled, y_train_resampled, cv=kfold, scoring='precision', return_train_score=True)
    models_cv_precision_train[model_name] = str(precision_cv['train_score'].mean())
    models_cv_precision_test[model_name] = str(precision_cv['test_score'].mean())

    models_cm[model_name] = confusion_matrix(y_test, Y_pred)
```

- **Confusion matrix** memberikan gambaran rinci tentang klasifikasi yang benar dan kesalahan yang dibuat oleh model.
- **Precision** score akan kami jadikan **metric utama**, dimana ini mengukur kemampuan model dalam mengidentifikasi semua kasus positif dengan benar, metric ini berfokus untuk **meminimalisir** adanya nilai **false positive**.



Dengan menggunakan **precision**, diharapkan false positive akan berkurang, sehingga perusahaan tidak perlu mengeluarkan **biaya lebih besar** (misal biaya telepon, campaign, karyawan) pada nasabah yang tidak tertarik pada deposit berjangka.

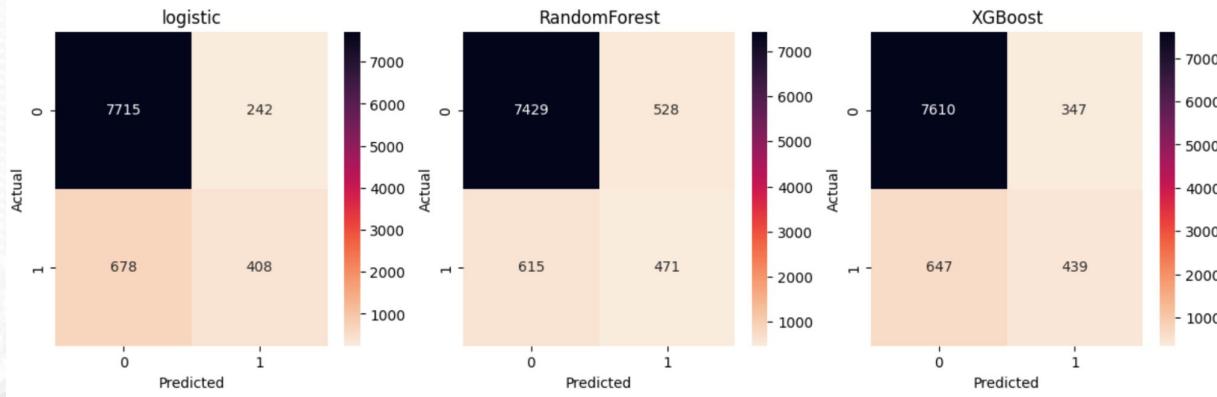
Kami **mengabaikan false negative** karena kami berasumsi bahwa nasabah yang tertarik akan langsung menghubungi kami tanpa kami harus melakukan campaign.

Modeling (Model Evaluation)

| Model | Precision Score | Precision Cross-Val (Train) | Precision Cross-Val (Test) |
|--------------|-----------------|-----------------------------|----------------------------|
| logistic | 0.627692 | 0.959004 | 0.958435 |
| RandomForest | 0.471471 | 0.996763 | 0.926028 |
| XGBoost | 0.558524 | 0.964123 | 0.946091 |

Dari ketiga algoritma yang telah dievaluasi, terdapat beberapa observasi penting yang perlu dipertimbangkan. Pertama, dalam hal precision untuk cross-validation (CV), **semua algoritma tidak menunjukkan tanda-tanda overfitting**, yang mengindikasikan bahwa model-model ini konsisten dalam kinerja mereka, khususnya, **Logistic Regression**. Dengan mempertimbangkan nilai precision score yang mencerminkan kemampuan model dalam menggambarkan kasus positif ('y'), **Logistic Regression** mengungguli dua algoritma lainnya, dengan nilai **0.627**.

Modeling (Model Evaluation)



Model logistic regression **memiliki kasus true negative cukup tinggi** daripada kedua model yang lain, walaupun ia **memprediksi true positif yang lebih rendah**. Hal tersebut dikarenakan Logistic Regression seringkali memiliki ambang batas yang lebih konservatif, yang berarti ia lebih cenderung untuk mengklasifikasikan pengamatan sebagai negatif dan berhati-hati saat membuat prediksi positif.

Namun, dalam hal **false positive**, logistic memiliki nilai paling rendah dari semua algoritma, sehingga cocok untuk minimalisir kesalahan prediksi benar.

Modeling (Hyperparameter Tuning)

| Model | Precision Score | Precision Cross-Val (Train) | Precision Cross-Val (Test) |
|-------------------------|-----------------|-----------------------------|----------------------------|
| logistic | 0.627692 | 0.959004 | 0.958435 |
| RandomForest | 0.471471 | 0.996763 | 0.926028 |
| XGBoost | 0.558524 | 0.964123 | 0.946091 |
| Tuned_LogisticRegressor | 0.623704 | 0.956765 | 0.956652 |

Setelah melakukan hyperparameter tuning, nilai precision mengalami sedikit **peningkatan namun tidak signifikan**. Oleh karena itu, kami mempertimbangkan untuk kembali menggunakan **model awal** dalam hal precision.

Modeling (Adjusting Threshold)

```

import matplotlib.pyplot as plt
from sklearn.metrics import precision_recall_curve

# Assuming you have already trained and evaluated your model
# Make predictions on the test data
y_pred_proba = models["logistic"].predict_proba(X_test)

# Adjusted threshold (choose the threshold that worked best for you)
adjusted_threshold = target_thres # Modify this threshold as needed

# Calculate precision, recall, and thresholds for the adjusted threshold
precision, recall, thresholds = precision_recall_curve(y_test, y_pred_proba[:, 1])

# Find the index of the threshold that is closest to the adjusted threshold
closest_threshold_index = (abs(thresholds - adjusted_threshold)).argmin()

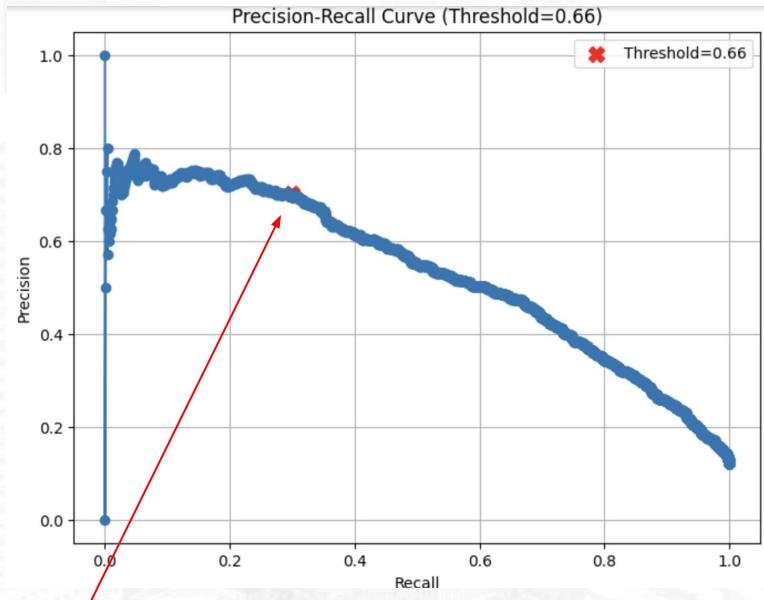
# Plot the Precision-Recall Curve with the adjusted threshold
plt.figure(figsize=(8, 6))
plt.plot(recall, precision, marker='o', linestyle='-' )
plt.xlabel('Recall')
plt.ylabel('Precision')
plt.title(f'Precision-Recall Curve (Threshold={adjusted_threshold:.2f})')

# Highlight the point on the curve corresponding to the adjusted threshold
plt.scatter(recall[closest_threshold_index], precision[closest_threshold_index], c='red', marker='x', s=100, label=f'Threshold={adjusted_threshold:.2f}')
plt.legend()
plt.grid(True)
plt.show()

```

Dalam rangka memaksimalkan precision, kami menggunakan **precision-recall curves**. Dimana kami **meningkatkan precision** dan menurunkan nilai recall dengan cara menyesuaikan nilai **threshold**.

Kami menggunakan Threshold dengan nilai **0.66**



Modeling (Adjusting Threshold)

Precision Recall Threshold

| | | |
|----------|----------|----------|
| 0.701299 | 0.298343 | 0.662721 |
| 0.700651 | 0.297422 | 0.664309 |
| 0.702174 | 0.297422 | 0.664853 |
| 0.701525 | 0.296501 | 0.666774 |
| 0.700873 | 0.295580 | 0.667835 |

```
#pred_8 = np.where(Y_pred>0.676661, 1, 0)
pred_8 = np.where(Y_pred>target_thres, 1, 0)
precision_8 = precision_score(y_test, pred_8)
precision_8
```

0.7004608294930875

Dengan mencari nilai **precision 0.70**, nilai **threshold minimal** yang tersedia adalah sebesar **0.662721**. Alhasil, nilai **precision naik menjadi 0.7012** dengan recall 0.2983 atau 0.3.

Modeling Interpretation

Result Interpretation (Cost efficiency)

- **Dataset awal**, dari 45211 nasabah, total campaign dilakukan 124.956 kali.
- **filtering manual** menggunakan (4 fitur) loan, housing, contact dan outcome, didapatkan sebanyak 998 nasabah.
- **Menggunakan ML**, didapatkan 2.480 nasabah.

Jika asumsi campaign dilakukan **2 kali** per nasabah dan menggunakan “lead cost” sebesar **25** pounds, maka:

| | Total Campaign | Price | Total Cost |
|--|----------------|-------|--|
| Before ML | 124956 | £25 | £ 3,123,900 |
| Filter Manual | (998 x 2) | £25 | £ 49,900 |
| After ML | (2480 x 2) | £25 | £ 124,000 |
| Cost Reduction After ML - before ML | | | £ 2,999,900 (96%) |
| Cost Reduction After ML - Filter manual | | | £ 74,100 (60%) *filter manual lebih mereduksi cost |

Result Interpretation (Profit Comparison)

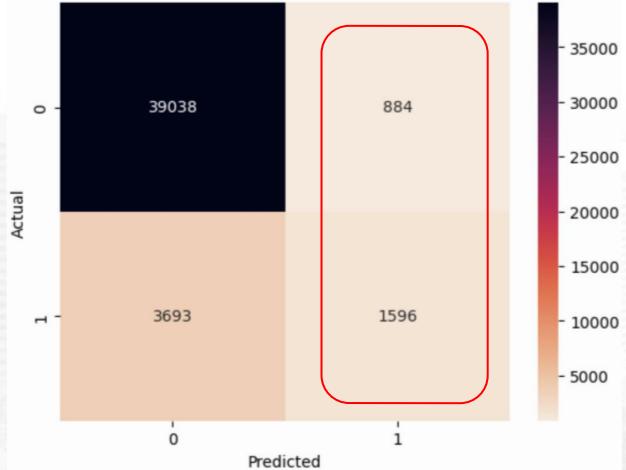
- **Dataset awal**, dari **45.211** nasabah, total nasabah deposit **5.289** nasabah.
- **filtering manual**, dari **998** nasabah yang terfilter, total deposit **704** nasabah.
- **Menggunakan ML**, dari rekomendasi **2.480** nasabah, total prediksi deposit **1.596** nasabah.

Jika asumsi nominal deposit sebesar **250 euro/216 pounds per nasabah**, maka:

| | Total Deposit | Price | Total Revenue | Total Profit |
|---------------------------------|---------------|-------|---------------|------------------|
| Before ML | 5.289 | £216 | £ 1,142,424 | -1,981,476 |
| Filter Manual | 704 | £216 | £ 152,064 | 102,164 |
| After ML | 1.596 | £216 | £ 344,736 | 220,736 |
| Profit After ML - before ML | | | | 2,202,212 (997%) |
| Profit After ML - Filter manual | | | | 118,572 (53%) |

ML mereduksi cost tanpa menghilangkan potensi pendapatan

Result Interpretation (Without Machine Learning)



INITIAL CONVERSION RATE

Conversion rate = (Jumlah Nasabah yang Berlangganan Secara **Aktual** / Jumlah Nasabah yang **Dihubungi**) * 100

$$\text{Conversion rate} = (5289 / 45211) * 100 = \sim 12\%$$

Hanya **12%** yang benar-benar berlangganan deposito berjangka.

AFTER ML CONVERSION RATE

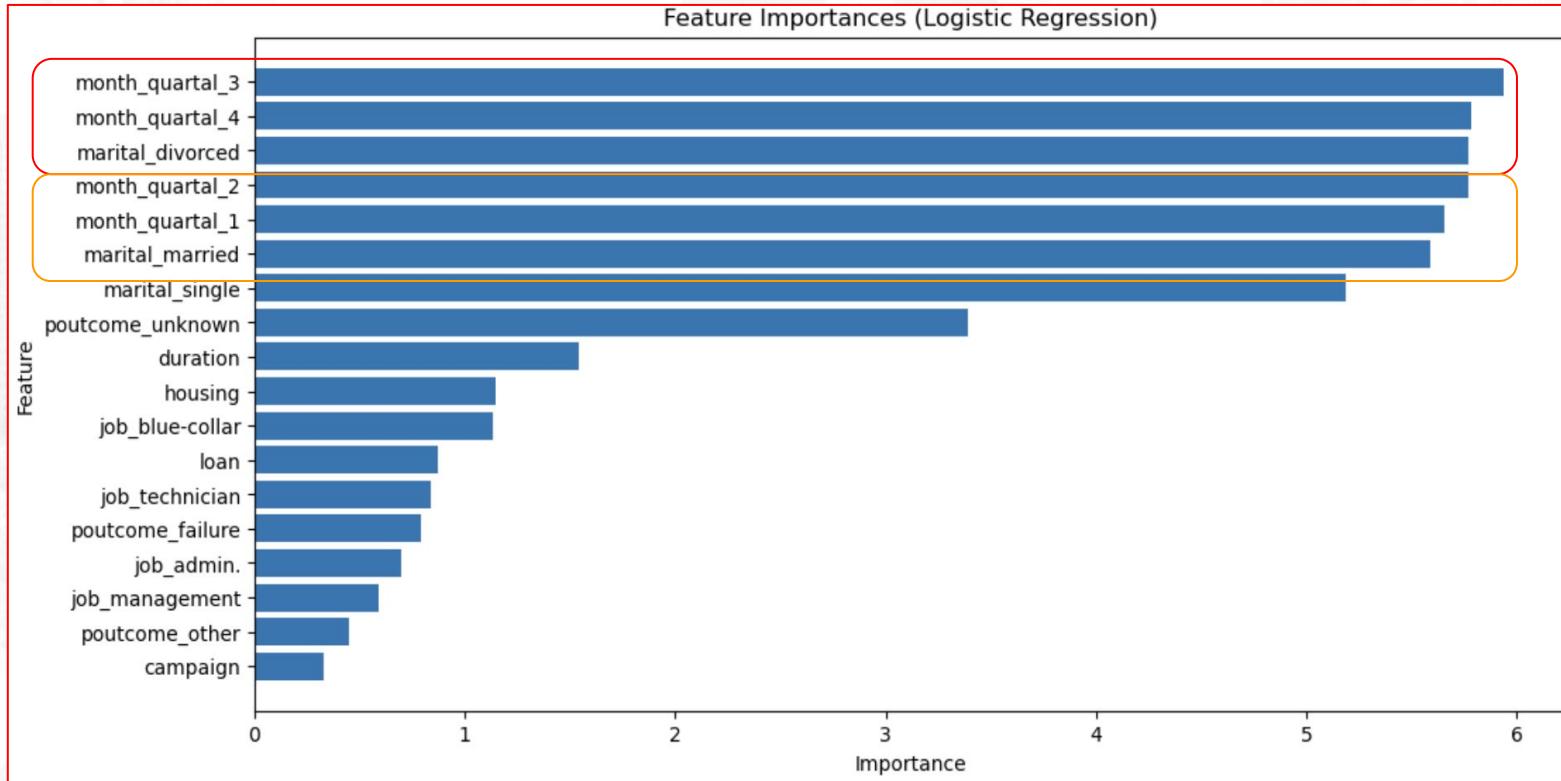
Conversion rate = (Jumlah Nasabah yang **aktual** Berlangganan / Jumlah Nasabah yang **diprediksi** berlangganan) * 100

$$\text{Conversion rate} = (1596 / \{884+1596\}) * 100 = 0.643 \sim 64\%$$

Berdasarkan prediksi model, sebanyak **64%** dari nasabah yang diprediksi benar-benar berlangganan deposito berjangka.

Feature Importance

Feature Importance



Insight

Month quartal 3 & 4 (semester 2)

- Semester 2 menjadi waktu yang **krusial dalam keputusan melakukan deposit**, hal ini bisa jadi karena semua pembayaran banyak selesai dilakukan di semester 1, sehingga semester 2 menyisakan uang yang biasanya **lebih senggang**
- Pada waktu ini juga, bank biasanya melakukan **penyesuaian** suku bunga dan juga banyak perusahaan merencanakan anggaran untuk tahun depan.

Marital Divorced

- Biasanya kategori ini merujuk kepada nasabah yang berusia **cukup dewasa** yang **telah melewati masa pernikahan**.
- Nasabah tipe ini memiliki keleluasaan dana karena banyak dari mereka yang hidup **sendiri**, sehingga bisa lebih baik spend untuk **invest perencanaan masa depan**.

Month quartal 1 & 2 (semester 1)

- Meski tidak sekrusial semester 2, semester 1 juga memiliki bulan-bulan yang bisa diperhatikan. Bulan **maret** memiliki tingkat konversi langganan deposit **tertinggi** dan bulan **april** juga **cukup banyak**, kedua bulan ini bisa menjadi perhatian sepanjang semester 1.

Marital Married

- Nasabah tipe ini biasanya memiliki **perencanaan** keuangan untuk masa depan berdua, sehingga beberapa dari mereka kemungkinan besar akan **tertarik** terhadap deposito sbg media **investasi yang aman**

Recommendation

Month

- Bank perlu melakukan **campaign masing-masing 1 kali** tiap nasabah pada **quartal 3 dan 4**
- Dalam campaign bisa menyesuaikan **promo liburan** (return lebih tinggi) dengan **pencairan** menjelang **akhir tahun** di quartal 4.
- Perlu mencoba melakukan campaign tambahan 1 kali pada bulan **maret** karena memiliki konversi tertinggi dalam langganan deposito.

Marital Divorced & Married

- Selain promo liburan, bank juga perlu memberikan **program khusus** untuk **tabungan pendidikan anak** dan **hari tua**, karena mungkin banyak nasabah dari kategori ini yang merupakan **janda memiliki anak** dan mungkin **janda tanpa tanggungan** yang biasanya butuh tabungan pensiun untuk dirinya sendiri.
- **Promo liburan** yang ditawarkan bisa **dikhususkan** juga pada pasangan yang **sudah menikah**

Thanks!

Git hub project

- <https://github.com/roguewindrunner/FinalProject-Rakamin>



Reference

- <https://www.kaggle.com/datasets/prakharrathi25/banking-dataset-marketing-targets>
- <https://immigrantinvest.com/blog/best-portugal-banks-en/>
- https://media.techtarget.com/rms/computerweekly/DowntimePDF/Buyers_guide/Telemarketing_Buyers_Guide_new.pdf