Rakamin - Data Science (Batch 37)
Explorer (Group 1)
Anggota:

1. Imam Maghfir Ramadhan
2. Marcellinus Putra Wijaya
3. Muhammad Zen Fikri
4. Puspita Ayu
5. Syaiful Adri
6. Putri Sausan
7. Nivan Dumatubun
8. Wasis Prasetyo

Stage 0 - Preparation

1. Apa problem yang mau diselesaikan dari dataset tsb? (15 poin)
Sebuah bank memiliki 10000 customer yang dimana terdapat 2037 customer (sekitar 20%) nasabah churn. Namun, bank tidak tahu faktor-faktor apa saja yang mempengaruhi nasabah churn dan karakteristik nasabah yang bagaimana yang akan berpotensi churn lebih besar.

2. Sebagai siapa kalian pada dataset tersebut? (10 poin)
Role kita dalam project ini adalah sebagai tim data scientist pada sebuah bank yang bertanggung jawab untuk memberikan saran atau rekomendasi terkait bagaimana cara mengurangi atau menurunkan nilai churn nasabah dengan mengidentifikasi faktor-faktor penyebab nasabah churn dan karakteristik nasabah yang berpotensi churn.

Pembagian tugas masing-masing individunya adalah:

- Project Manager/Leader - Imam Maghfir Ramadhan
- Data Engineer (2 orang) - Data pre-processing (Cleansing, encoding dll)
    - Nivan Dumatubun
    - Syaiful Adri
- Data Analyst (2 orang) - EDA, Insight (visualization data distribution), Business Recommendation
    - Putri Sausan
    - Puspita Ayu
- Business Intelligence (2 orang) - Visualization (showing to stakeholder), Business Recommendation
    - Wasis Prasetyo
- Data Scientist (2 orang) - Modeling, Business Recommendation
    - Muhammad Zen Fikri
    - Marcellinus Putra Wijaya

3. Apa goal yang mau dicapai? (20 poin)
Bertujuan untuk mengurangi atau menurunkan tingkat churn.

4. Apa Objective yang sesuai dengan goal tsb? (25 poin)

- Mengidentifikasi faktor-faktor yang mempengaruhi nasabah untuk churn

- Membangun model machine learning (ML) yang akurat yang dapat memprediksi nasabah yang berpotensi churn berdasarkan faktor-faktor yang ada

5. Apa business metrics yang cocok untuk mengukur ketercapaian Objective tsb? (30 poin)

- Churn Rate (Berapa banyak persentase nasabah yang exit atau berhenti?)

Stage 1 - Exploratory Data Analysis (EDA)

1. **Descriptive Statistics**

a. **Apakah ada kolom dengan tipe data kurang sesuai, atau nama kolom dan isinya kurang sesuai?**
Semua tipe data pada dataset sudah sesuai, sedangkan untuk kolom 'exited' diganti dengan nama 'Churn' agar lebih informatif.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   RowNumber        10000 non-null  int64
 1   CustomerId       10000 non-null  int64
 2   Surname          10000 non-null  object
 3   CreditScore      10000 non-null  int64
 4   Geography        10000 non-null  object
 5   Gender           10000 non-null  object
 6   Age              10000 non-null  int64
 7   Tenure           10000 non-null  int64
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64
 10  HasCrCard        10000 non-null  int64
 11  IsActiveMember   10000 non-null  int64
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```

b. **Apakah ada kolom yang memiliki nilai kosong? Jika ada, apa saja?**
Tidak terdapat missing value pada dataset

```
# cek missing value
df.isnull().sum()
```

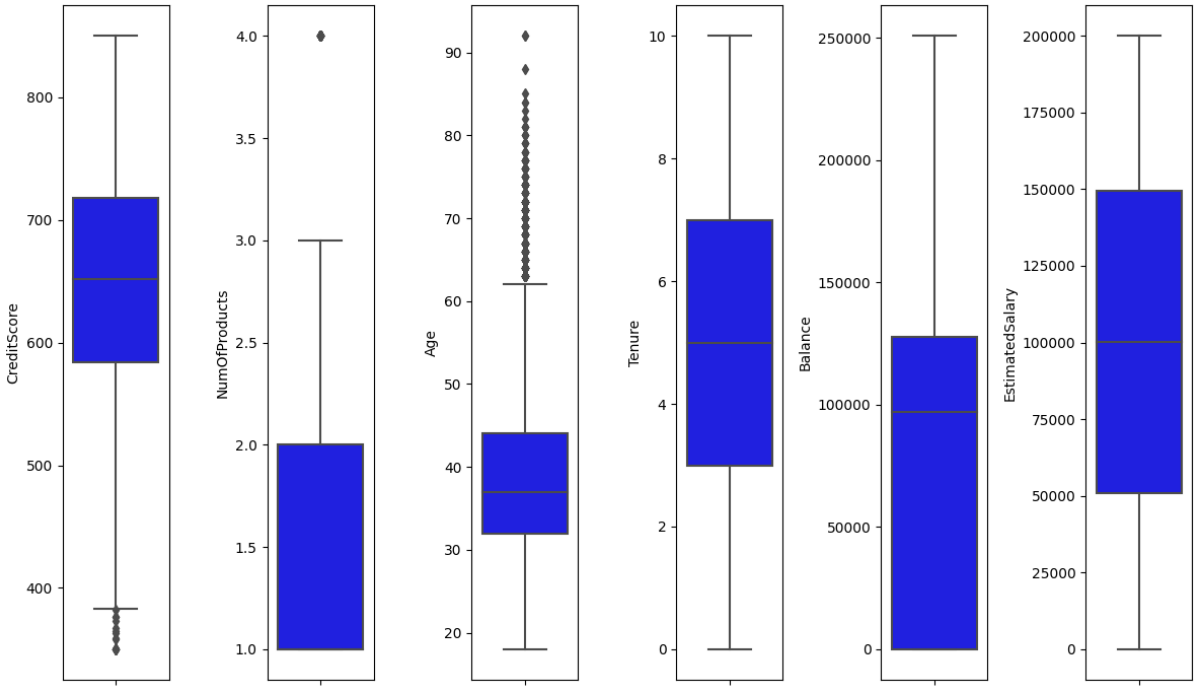
```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

c. **Apakah ada kolom yang memiliki nilai summary agak aneh? (min/mean/median/max/unique/top/freq)**

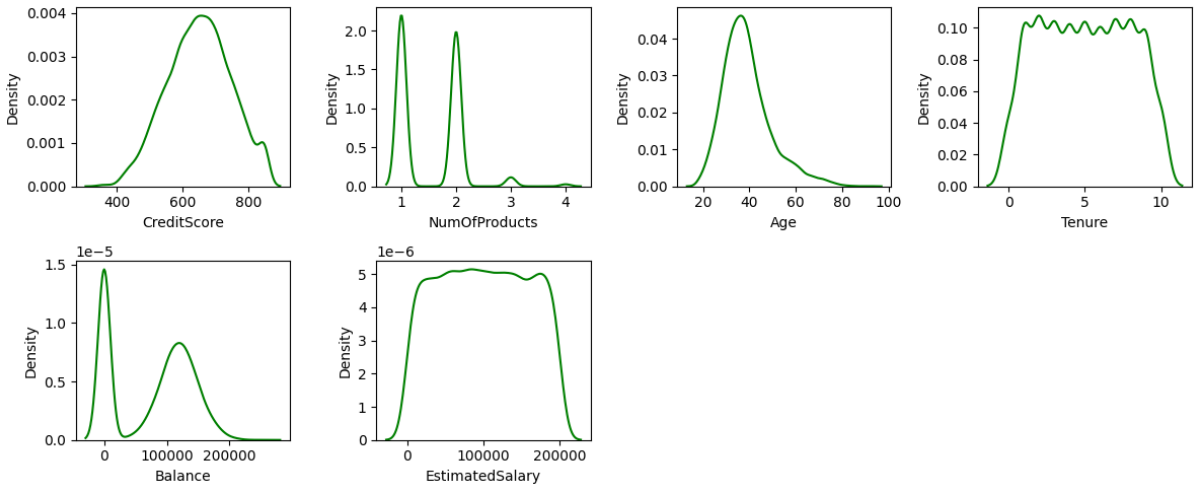
Berdasarkan data yang ada, kolom "balance" memiliki nilai summary yang cukup anomali. Hal ini dapat terlihat dari nilai mean dan mediannya yang cukup berbeda yang dimana mengindikasikan kemungkinan distribusi data dari kolom ini tidak normal.

	CreditScore	NumOfProducts	Age	Tenure	Balance	EstimatedSalary
count	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000	10000.000000
mean	650.528800	1.530200	38.921800	5.012800	76485.889288	100090.239881
std	96.653299	0.581654	10.487806	2.892174	62397.405202	57510.492818
min	350.000000	1.000000	18.000000	0.000000	0.000000	11.580000
25%	584.000000	1.000000	32.000000	3.000000	0.000000	51002.110000
50%	652.000000	1.000000	37.000000	5.000000	97198.540000	100193.915000
75%	718.000000	2.000000	44.000000	7.000000	127644.240000	149388.247500
max	850.000000	4.000000	92.000000	10.000000	250898.090000	199992.480000

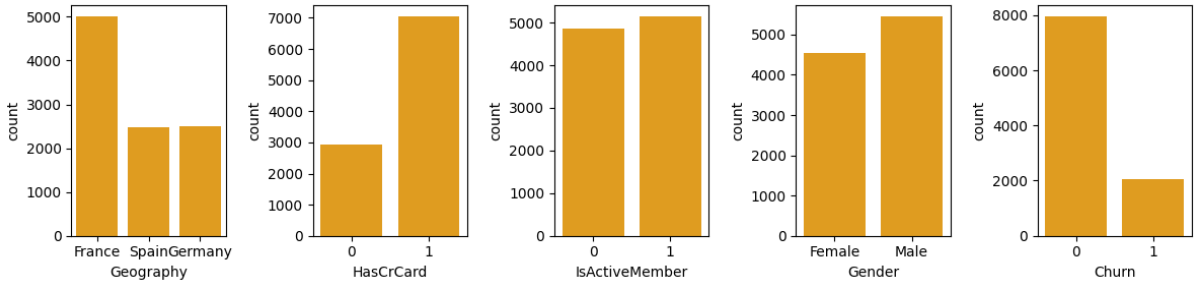
2. Univariate Analysis



- Terlihat pada boxplot chart bahwa pada kolom 'CreditScore', ‘NumOfProducts’, dan 'Age' terdapat outlier.

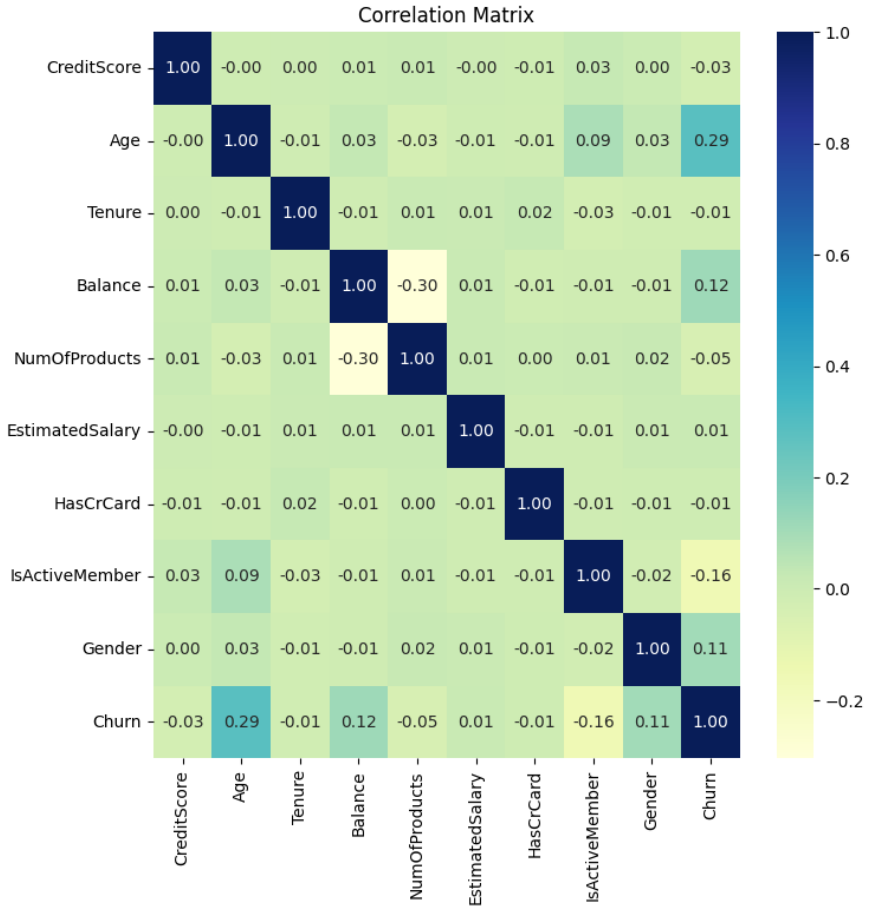


- Dari kdeplot yang menunjukkan skewness dari data tersebut, dapat terlihat bahwa sebagian besar kolom 'Age' cenderung positively skew. Sedangkan untuk kolom 'Balance' dan ‘NumOfProducts’ cenderung memiliki distribusi bimodal.



- Dari kolom2 categorical yang ada dapat terlihat bahwa kolom 'Geography' didominasi oleh France, kolom 'HasCrCard' didominasi oleh nasabah yang memiliki kartu kredit (value = 1), dan kolom 'IsActiveMember' didominasi oleh nasabah yang aktif, kolom 'Gender' didominasi oleh Male, Kolom 'Churn' lebih banyak yang tidak churn (value = 0).
- Untuk next step yang harus di follow up saat data pre-processing ialah handling data outlier, melakukan label encoding (untuk kolom 'gender', dll), kolom 'negara' (melakukan one-hot encoding), normalisasi, standarisasi, dan class imbalance.

3. **Multivariate Analysis**



a. **Bagaimana korelasi antara masing-masing feature dan label. Kira-kira feature mana saja yang paling relevan dan harus dipertahankan?**
   - Kolom "churn" memiliki korelasi positif yang moderate/menengah terhadap kolom "age" (korelasi = 0.29), "balance" (korelasi = 0.12). Sedangkan, pada kolom "IsActiveMember" (korelasi = -0.16) memiliki korelasi negative. Oleh karena itu, feature-feature tersebut harus dipertahankan

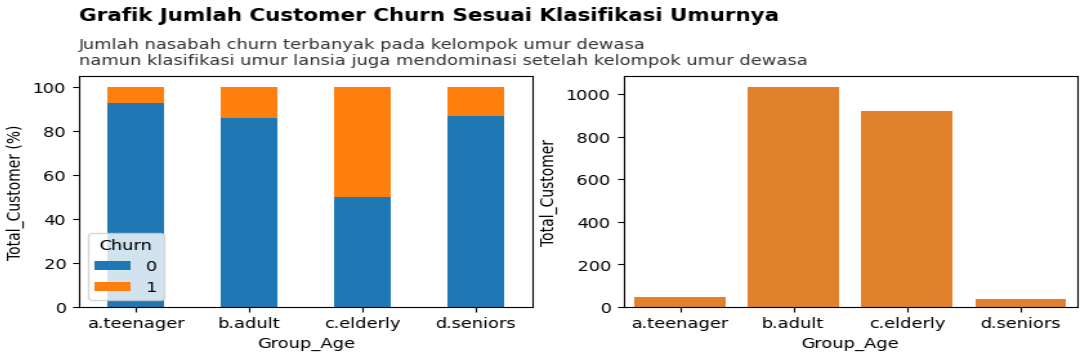
b. **Bagaimana korelasi antar-feature, apakah ada pola yang menarik? Apa yang perlu dilakukan terhadap feature itu?**
   - Feature 'balance' dan 'NumOfProduct' memiliki nilai korelasi negatif yang paling kuat di antara korelasi antar-feature lainnya. Oleh karena itu, harus dilakukan analisis lebih lanjut untuk kedua feature tersebut untuk menguji apakah terjadi multikolinearitas.

4. **Business Insight**

a. **Customer churn berdasarkan umur**

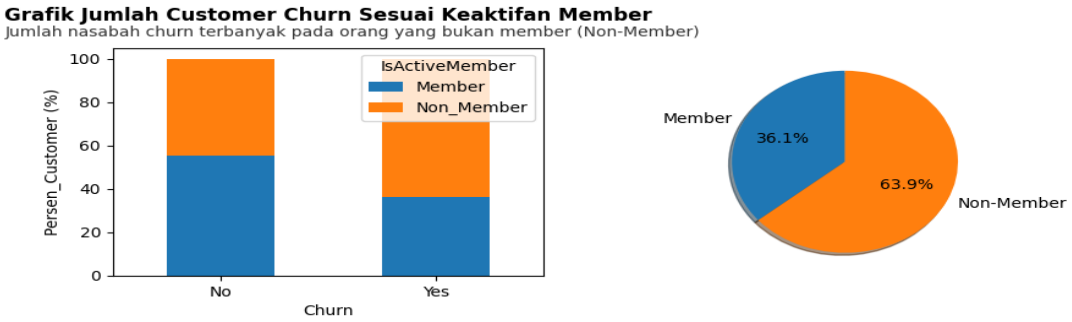
Berdasarkan Jurnal Urgensi Revisi Undang-Undang tentang Kesejahteraan Lanjut Usia, Penentuan range umur dapat dikelompokkan menjadi:

•Umur 12 - 25 = Remaja (teenager)
•Umur 26 - 45 = Dewasa (adult)
•Umur 46 - 65 = Lansia (elderly)
•Umur > 65 = Manula (seniors)



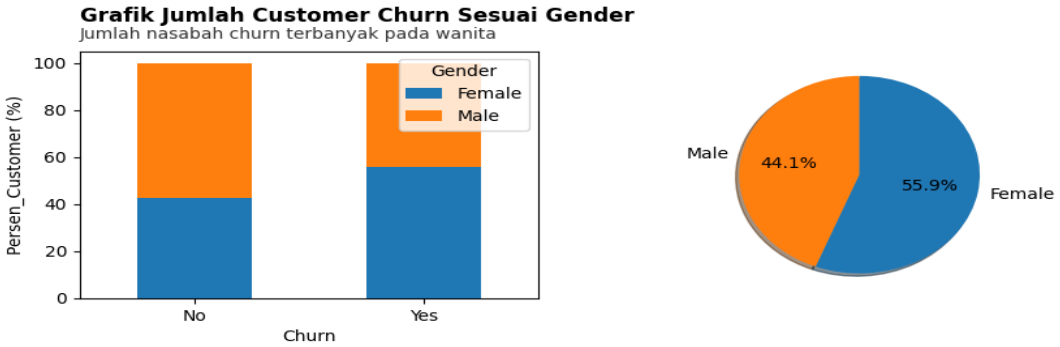
Dapat dilihat dari grafik diatas bahwa customer churn berdasarkan umur berada di kelompok **26 - 45 (Adult)** dibandingkan dengan usia yang lebih muda. Tetapi secara persentasi dari total customer per kelompok umur kelompok **46-65 (Elderly)** memiliki persentase yang tinggi.

b. Customer churn berdasarkan keaktifan member



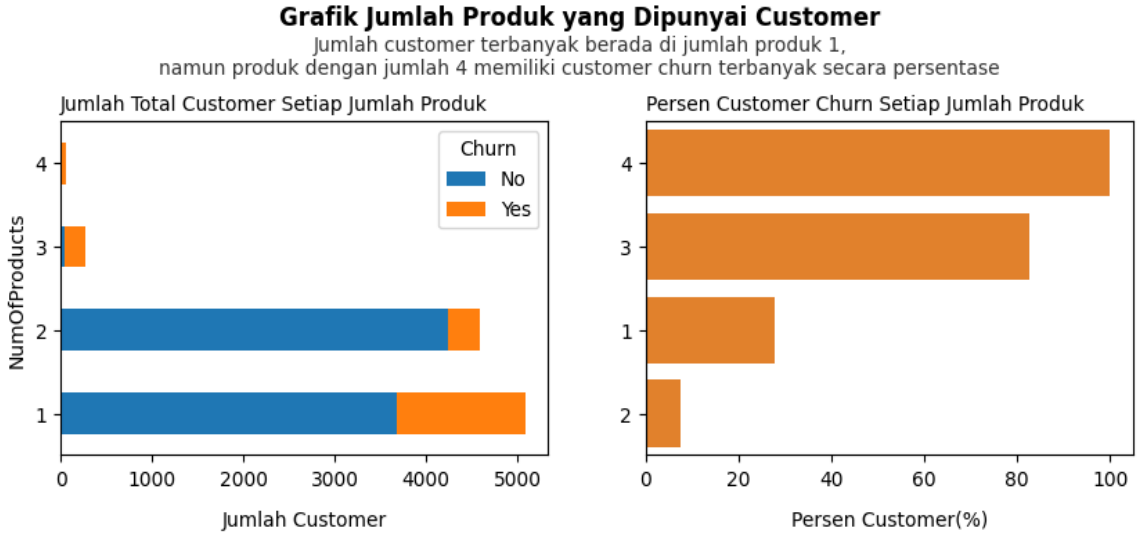
Berdasarkan grafik diatas dapat dilihat bahwa customer churn berdasarkan keaktifan member adalah **customer yang bukan merupakan member aktif**. Persentasi customer non member **63,9 %** dibandingkan dengan customer member **36,1 %**.

c. Customer churn berdasarkan gender



Berdasarkan grafik diatas didapatkan bahwa customer churn berdasarkan gender adalah **female** sebanyak 1139 customer**.** Persentase untuk Female **55,9 %** dibanding dengan Male **44,1%.**

d. Customer churn berdasarkan kepemilikan produk



Berdasarkan grafik diatas didapatkan bahwa customer bank paling banyak adalah customer yang memiliki **1 jenis produk.** Tetapi untuk secara **presentase customer bank yang churn** paling banyak adalah customer yang memiliki **4 produk**. Jumlah customer yang memiliki 4 produk adalah 60 dan semuanya churn atau 100% dari customernya churn.

Stage 2 - Preprocessing

1. Data Cleansing (50 poin)

A. Handle missing values

Dari pemeriksaan yang telah dilakukan pada dataset 'Churn_Modelling.csv' diketahui bahwa tidak ada data kosong (*missing value*). Sehingga tidak perlu dilakukan tindakan lebih lanjut untuk *handling missing value*.

```
[40] df.isna().sum() # menampilkan jumlah missing value setiap kolom

RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Churn              0
Group_Age          0
dtype: int64
```

Semua tipe data sudah sesuai dan tidak ada data kosong

B. Handle duplicated data

Dari pemeriksaan yang dilakukan pada dataset 'Churn_Modelling.csv' diketahui bahwa tidak ada data yang terduplikasi (*duplicated data*). Sehingga tidak perlu dilakukan tindakan lebih lanjut untuk *handling duplicated data*.

```
[41] df.duplicated().sum() # check data duplikat

0
```

Tidak ada data yang duplikat pada dataset

C. Handle outliers

Dalam tahapan *preprocessing* dilakukan *handle ouliers* atau pemeriksaan outliers. Tahapan ini dilakukan setelah pemisahan (split) dataset menjadi data training dan data testing. Handle outliers dilakukan pada dataframe data training numerikal. Pemeriksaan outliers menggunakan z-score (divisualisasikan dengan kdeplot) dan IQR (divisualisasikan dengan boxplot).

- Menggunakan Z-Score

```
[52] numerical = ['CreditScore','Age','Tenure','Balance','EstimatedSalary']
```

```
[53] # Using Z-score for outlier removal
     xtrain_dummy = x_train
     ytrain_dummy = y_train

     for col in numerical:
         z_scores = np.abs(stats.zscore(x_train[col]))
         filter_mask_z = (z_scores < 3)  # Adjust the threshold as needed
         x_train_after_z = xtrain_dummy[filter_mask_z]
         y_train_after_z = ytrain_dummy[filter_mask_z]

         xtrain_dummy = x_train_after_z
         ytrain_dummy = y_train_after_z
```

Setelah dilakukan outliers removal menghasilkan

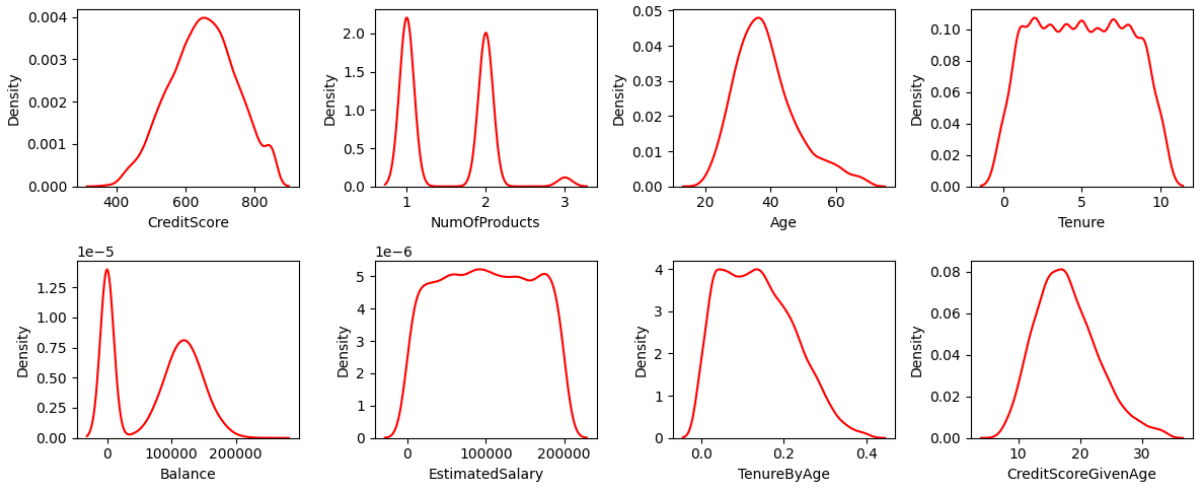
```
[54] x_train_after_z.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 7884 entries, 9254 to 7270
Data columns (total 17 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   CreditScore        7884 non-null   int64
 1   Gender             7884 non-null   int64
 2   Age                7884 non-null   int64
 3   Tenure             7884 non-null   int64
 4   Balance            7884 non-null   float64
 5   NumOfProducts      7884 non-null   int64
 6   HasCrCard          7884 non-null   int64
 7   IsActiveMember     7884 non-null   int64
 8   EstimatedSalary    7884 non-null   float64
 9   Group_Age          7884 non-null   int64
 10  Balance_Category   7884 non-null   category
 11  CreditScore_Range  7884 non-null   category
 12  Tenure_Category    7884 non-null   category
 13  Salary_Range       7884 non-null   category
 14  Geography_France   7884 non-null   uint8
 15  Geography_Germany  7884 non-null   uint8
 16  Geography_Spain    7884 non-null   uint8
dtypes: category(4), float64(2), int64(8), uint8(3)
memory usage: 732.1 KB
```

Lalu dilakukan visualisasi menggunakan kdeplot

```python
#membandingkan data setelah filter menggunakan plot
#data setelah filter z
plt.figure(figsize=(12,5))
for i in range(0, len(numerical)):
  plt.subplot(2, 4, i+1)
  sns.kdeplot(x = x_train_after_z[numerical[i]], color = 'red')
  plt.xlabel(numerical[i])
  plt.tight_layout()

plt.show()
```



- Menggunakan IQR

```python
# Using IQR for outlier removal
xtrain_dummy = x_train
ytrain_dummy = y_train

for col in numerical:
    Q1 = x_train[col].quantile(0.25)
    Q3 = x_train[col].quantile(0.75)
    IQR = Q3 - Q1
    filter_mask_iqr = ~((xtrain_dummy[col] < (Q1 - 1.5 * IQR)) | (xtrain_dummy[col] > (Q3 + 1.5 * IQR)))
    x_train_after_iqr = xtrain_dummy[filter_mask_iqr]
    y_train_after_iqr = ytrain_dummy[filter_mask_iqr]

    xtrain_dummy = x_train_after_iqr
    ytrain_dummy = y_train_after_iqr
```

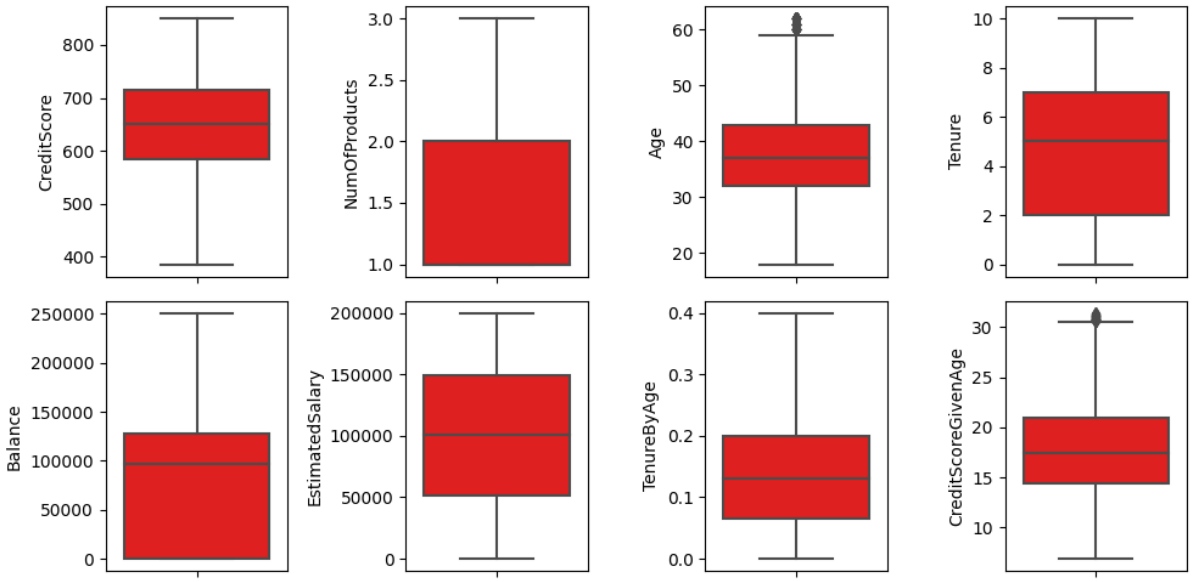
Setelah dilakukan outliers removal menghasilkan

```
x_train_after_iqr.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 7701 entries, 9254 to 7270
Data columns (total 17 columns):
 #   Column             Non-Null Count  Dtype   
---  ------             --------------  -----   
 0   CreditScore        7701 non-null   int64   
 1   Gender             7701 non-null   int64   
 2   Age                7701 non-null   int64   
 3   Tenure             7701 non-null   int64   
 4   Balance            7701 non-null   float64 
 5   NumOfProducts      7701 non-null   int64   
 6   HasCrCard          7701 non-null   int64   
 7   IsActiveMember     7701 non-null   int64   
 8   EstimatedSalary    7701 non-null   float64 
 9   Group_Age          7701 non-null   int64   
 10  Balance_Category   7701 non-null   category
 11  CreditScore_Range  7701 non-null   category
 12  Tenure_Category    7701 non-null   category
 13  Salary_Range       7701 non-null   category
 14  Geography_France   7701 non-null   uint8   
 15  Geography_Germany  7701 non-null   uint8   
 16  Geography_Spain    7701 non-null   uint8   
dtypes: category(4), float64(2), int64(8), uint8(3)
memory usage: 715.1 KB
```

Lalu dilakukan visualisasi menggunakan boxplot

```
#data setelah filter iqr
plt.figure(figsize=(10,5))
for i in range(0, len(numerical)):
  plt.subplot(1, len(numerical),i+1)
  sns.boxplot(y=x_train_after_iqr[numerical[i]], color ='red', orient='v')
  plt.tight_layout()
```



Dari kedua hasil tersebut didapatkan hasil

```
n_z       = (x_train['CreditScore'].count() - x_train_z['CreditScore'].count()) / x_train['CreditScore'].count() * 100
n_kuartil = (x_train['CreditScore'].count() - x_train_iqr['CreditScore'].count()) / x_train['CreditScore'].count() * 100
print('perbedaan jumlah data menggunakan z   : '+ str(round(n_z,2)) + '%')
print('perbedaan jumlah data menggunakan IQR : '+ str(round(n_kuartil,2)) + '%')
```

```
perbedaan jumlah data menggunakan z   : 3.19%
perbedaan jumlah data menggunakan IQR : 6.64%
```

Berdasarkan analisa outlier yang ada, dapat terlihat bahwa outlier kebanyakan terdapat pada kolom age yang dimana distribusi age dari nasabah yang ada itu wajar adanya. Oleh karena itu, proses outlier removal pada model ini tidak dilakukan dan bisa langsung menggunakan dataframe yang sudah di split untuk proses selanjutnya.

D. Feature transformation

Setelah dilakukan pemeriksaan pada outlier, maka dilakukan transformasi fitur (*feature transformation*). Fitur yang digunakan adalah normalisasi

```
[65] #data
     df_norm_train = x_train
     df_norm_test = x_test

     numerical = numerical
```

```
#menggunakan fitur normalisasi
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

# Fit the scaler on the training set and transform the training set
for col in numerical:
    df_norm_train[col] = scaler.fit_transform(df_norm_train[col].values.reshape(-1, 1))

# Use the same scaler to transform the test set (without fitting)
for col in numerical:
    df_norm_test[col] = scaler.transform(df_norm_test[col].values.reshape(-1, 1))
```

Melakukan deskriptif terhadap data train setelah dilakukan normalisasi menggunakan MinMaxScaler

```
[67] df_norm_train.describe()
```

Melakukan deskriptif terhadap data test setelah dilakukan normalisasi menggunakan MinMaxScaler

```
[68] df_norm_test.describe()
```

Jika normalisasi dibandingkan dengan menggunakan standarisasi memang normaliasi kurang robust terhadap outlier, namun karena data sudah dilakukan pengecekan outlier maka penggunaan motode transformasi normalisasi seharusnya tidak bermasalah.

E. Feature encoding

Dalam feature encoding dilakukan akan mengubah feature categorical menjadi feature numerical. Kategori yang digunakan dalam one-hot-encoding adalah Geography

```
[44] # one hot encoding
     for cat in ['Geography']:
         onehots = pd.get_dummies(df[cat], prefix=cat, drop_first = True)
         df =  df.join(onehots)
```

Menghasilkan

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 18 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   CreditScore         10000 non-null  int64
 1   Geography           10000 non-null  object
 2   Gender              10000 non-null  object
 3   Age                 10000 non-null  int64
 4   Tenure              10000 non-null  int64
 5   Balance             10000 non-null  float64
 6   NumOfProducts       10000 non-null  int64
 7   HasCrCard           10000 non-null  int64
 8   IsActiveMember      10000 non-null  int64
 9   EstimatedSalary     10000 non-null  float64
 10  Churn               10000 non-null  int64
 11  Group_Age           10000 non-null  object
 12  Balance_Category    10000 non-null  category
 13  CreditScore_Range   10000 non-null  category
 14  Tenure_Category     10000 non-null  category
 15  Salary_Range        10000 non-null  category
 16  Geography_Germany   10000 non-null  uint8
 17  Geography_Spain     10000 non-null  uint8
dtypes: category(4), float64(2), int64(7), object(3), uint8(2)
memory usage: 996.9+ KB
```

Lalu selanjutnya dilakukan label encoding untuk Gender, Group_Age, Balance_Category, CreditScore_Range, Tenure_Category, dan Salary_Range

```
[47] df = df.drop(columns =['Geography']).copy()
```

```
# label encoding
map_gender = {'Male':0,'Female':1}
map_grup_age = { 'a.teenager':0,'b.adult':1,'c.elderly':2,'d.seniors':3}
map_balance = {'Low':0, 'Medium':1, 'High':2}
map_credit_score = {'Poor':0, 'Fair':1, 'Good':2, 'Excellent':3}
map_tenure = {'Short Term':0, 'Medium Term':1, 'Long Term':2}
map_salary = {'Low':0, 'Medium':1, 'High':2, 'Very High':3}

df['Gender'] = df['Gender'].map(map_gender)
df['Group_Age'] = df['Group_Age'].map(map_grup_age)
df['Balance_Category'] = df['Balance_Category'].map(map_balance)
df['CreditScore_Range'] = df['CreditScore_Range'].map(map_credit_score)
df['Tenure_Category'] = df['Tenure_Category'].map(map_tenure)
df['Salary_Range'] = df['Salary_Range'].map(map_salary)
```

F. Handle class imbalance

Selanjutnya dilakukan pemeriksaan kesenjangan antar kelas (*Class Imbalance*). Dengan pertama menampilkan distribusi kelas sebelum oversampling

```
# Menampilkan distribusi kelas sebelum oversampling
class_distribution_before = pd.Series(y_train).value_counts()
print("Class distribution before oversampling:")
print(class_distribution_before)
```

```
Class distribution before oversampling:
0    6356
1    1644
Name: Churn, dtype: int64
```

Setelah itu dilakukan oversampling mennggunkaan SMOTE (Synthetic Minority Over-sampling Technique) dan undersamling menggunkaan RandomUnderSampler. Pemilihan SMOTE dan RandomUnderSampling digunakan karena dapat mengurangi bias dalam model

```
from imblearn.over_sampling import SMOTE
from imblearn.under_sampling import RandomUnderSampler

# Oversample Data
smote = SMOTE()
x_train_oversampled, y_train_oversampled = smote.fit_resample(x_train, y_train)

# Undersample Data
rus = RandomUnderSampler()
x_train_undersampled, y_train_undersampled = rus.fit_resample(x_train, y_train)
```

Menghasilkan output untuk distribusi oversampling dan undersampling

```
# distribusi kelas setelah oversampling
class_distribution_after_over = pd.Series(y_train_oversampled).value_counts()
print("\nClass distribution after oversampling:")
print(class_distribution_after_over)
```

```
Class distribution after oversampling:
0    6356
1    6356
Name: Churn, dtype: int64
```

+ Code + Text

```
[68] # distribusi kelas setelah undersampling
class_distribution_after_under = pd.Series(y_train_undersampled).value_counts()
print("\nClass distribution after undersampling:")
print(class_distribution_after_under)
```

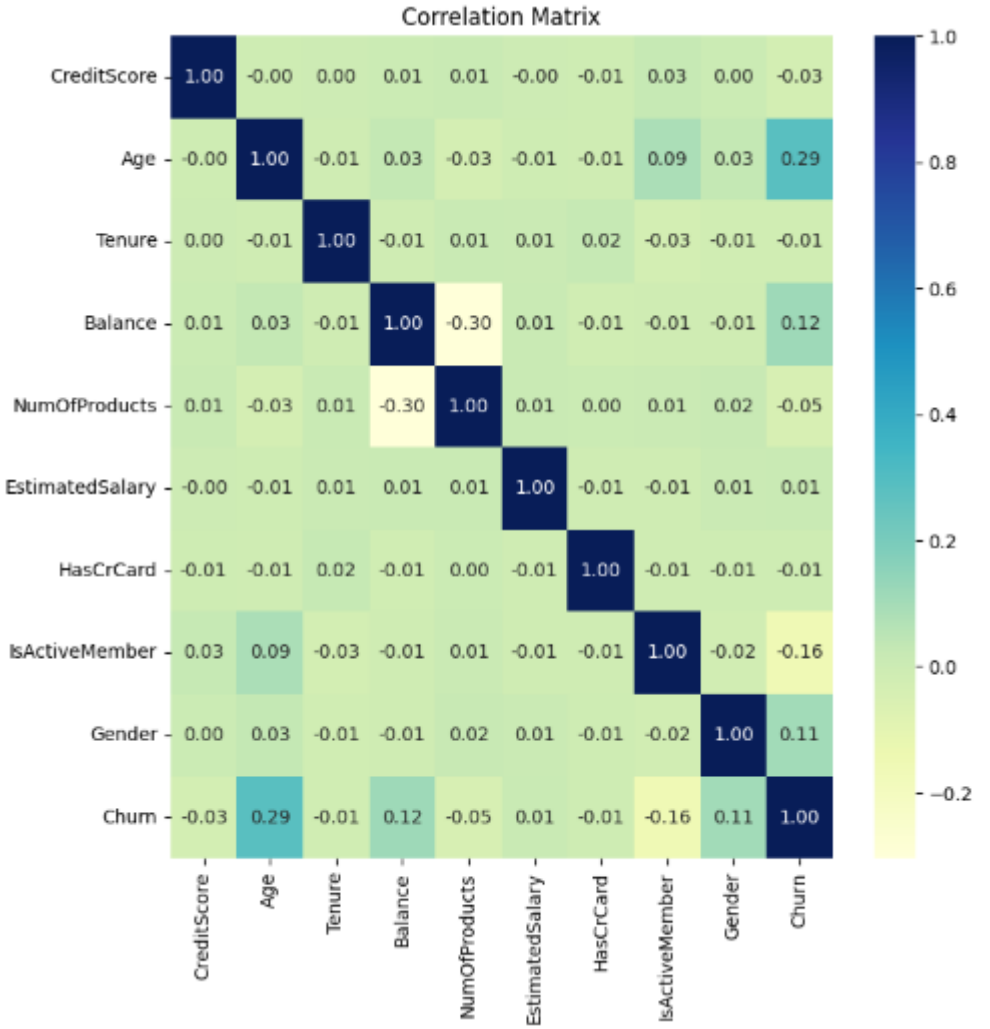
```
Class distribution after undersampling:
0    1644
1    1644
Name: Churn, dtype: int64
```

Dari kedua hasil tersebut, untuk data imbalance akan digunakan undersampling dikarenakan target prediksi category churn (value = 1) lebih sedikit. Penggunaan oversampling digunakan dapat menyebabkan bias (dikarenakan jumlah data sintesis yang akan ter-generate akan relatif banyak)

2. Feature Engineering (35 poin)

Cek *feature* yang ada sekarang, lalu lakukan:

A. Analisis Multikolinearitas



Dilihat dari *Correlation Matrix*, diketahui bahwa *feature* 'NumOfProducts' memiliki korelasi terhadap *feature* 'Balance' sebesar -0.30. Sehingga dilakukan pengujian lebih lanjut untuk membuktikan adanya multikolinearitas terhadap fitur-fitur tersebut. Salah satu cara umum untuk memeriksa multikolinearitas adalah dengan menggunakan VIF (*Variance Inflation Factor*), dimana:

Jika, VIF = 1, Multikolinearitas sangat rendah
VIF < 5, Multikolinearitas sedang
VIF > 5, Multikolinearitas ekstrim (hal ini yang harus dihindari)

Sumber:
https://www.analyticsvidhya.com/blog/2020/03/one-hot-encoding-vs-label-encoding-using-scikit-learn/#h-one-hot-encoding-vs-label-encoding

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
from statsmodels.tools.tools import add_constant

selected_columns = ['NumOfProducts', 'Balance']
df_subset = df[selected_columns]
X = add_constant(df_subset)

# Menghitung nilai VIF untuk setiap feature
vif_data = pd.DataFrame()
vif_data["Variable"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.shape[1])]

print(vif_data)
```

```
        Variable        VIF
0          const  12.445353
1  NumOfProducts   1.101959
2        Balance   1.101959
```

Dari analisa multikolinearitas antara *feature* 'NumOfProducts' dan 'Balance', diketahui nilai VIF 1.10 relatif kecil (VIF < 5) yang menunjukkan bahwa tidak ada bukti multikolinearitas antara *feature* 'NumOfProducts' dan 'Balance'. Oleh karena itu kedua *feature* tersebut tetap dipertahankan untuk pemodelan.

B. Feature selection (membuang feature yang kurang relevan atau redundan)

Dilihat dari fitur-fitur pada dataset, ada beberapa fitur yang kurang relevan terhadap target. Diantaranya adalah 'RowNumber', 'Surename', dan 'CustomerId'. Fitur-fitur tersebut hanya berisi informasi identitas customer, sehingga dilakukan tindakan penghapusan (*dropping*) pada fitur-fitur tersebut.

Dataset sebelum dilakukan penghapusan fitur yang kurang relevan terhadap target:

```
[34] df.sample(2)
```

	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Churn	Group_Age
7926	7927	15732644	Evans	567	Spain	Female	54	5	92316.31	2	1	0	158590.66	1	c.elderly
8741	8742	15762855	Hill	622	Spain	Female	23	8	0.00	2	1	1	131389.39	0	a.teenager

Dropping features yang kurang relevan terhadap target:

```
[35] #melakukan drop kolom 'RowNumber', 'Surname', 'CustomerId'
     df = df.drop(['RowNumber', 'Surname', 'CustomerId'], axis = 1) # feature 'Surname', 'RowNumber', 'CustomerID' tidak penting terhadap taget
```

Dataset setelah dilakukan penghapusan fitur-fitur yang kurang relevan terhadap target:

```
[36] df.sample(2)
```

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Churn	Group_Age
7233	571	France	Male	38	1	121405.04	1	1	1	154844.22	0	b.adult
1310	748	Germany	Female	27	2	90971.85	1	1	1	131662.47	0	b.adult

C. Feature extraction (membuat feature baru dari feature yang sudah ada)

Feature extraction ditujukan untuk mendapatkan fitur-fitur tambahan (baru) yang diturunkan dari fitur-fitur yang sudah ada. Sehingga pilihan fitur untuk melakukan *modeling* lebih banyak. Pada dataset ini kami menambahkan empat fitur baru yaitu 'Balance_Category', 'CreditScore_Range', 'Tenure_Category', 'Salary_Range'. Dimana

pembagian range setiap kategori didasari dari nilai min, quartile (1, 2, 3), dan max dari masing-masing kolom.

```
[38] # Balance Categories
     df['Balance_Category'] = pd.cut(df['Balance'], bins=[-0.001, 97199, 127644, 251000], labels=['Low', 'Medium', 'High'])

     # Credit Score Ranges
     df['CreditScore_Range'] = pd.cut(df['CreditScore'], bins=[349, 584, 652, 718, 850], labels=['Poor', 'Fair', 'Good', 'Excellent'])

     # Tenure Categories
     df['Tenure_Category'] = pd.cut(df['Tenure'], bins=[-0.001, 3, 5, 10], labels=['Short Term', 'Medium Term', 'Long Term'])

     # Salary Range
     df['Salary_Range'] = pd.cut(df['EstimatedSalary'], bins=[0, 51002, 100194, 149388, 200000], labels=['Low', 'Medium', 'High','Very High'])
```

1. **Balance_category**
   mengelompokkan *feature* 'Balance' ke dalam beberapa kategori berdasarkan range, sebagai berikut:
   a. Low = 0 - 97,198
   b. Medium = 97,199 - 127,643
   c. High = 127,644 - 251,000

   Feature 'Balance_category' memungkinkan model machine learning untuk menangkap pola yang mungkin terkait dengan tingkat saldo tertentu.
2. **CreditScore_Range**
   mengelompokkan *feature* 'CreditScore' ke dalam beberapa kategori berdasarkan range, sebagai berikut:
   a. Poor (Buruk) = 350 - 583
   b. Fair (Cukup) = 584 - 651
   c. Good (Baik) = 652 - 717
   d. Excellent (Sangat Baik) = 718 - 850

   Feature 'CreditScore_Range' dapat digunakan dalam pemodelan untuk dapat lebih mudah menangkap pola atau tren yang mungkin terkait dengan tingkat kredit tertentu.
3. **Tenure_Category**
   mengelompokkan *feature* 'Tenure' ke dalam beberapa kategori berdasarkan range, sebagai berikut:
   a. Short Term = 0 - 2
   b. Medium Term = 3 - 5
   c. Long Term = 6 - 10

   Feature 'Tenure_Category' dapat digunakan dalam pemodelan untuk mengidentifikasi pola atau tren yang berkaitan dengan lamanya waktu pelanggan menggunakan layanan.
4. **Salary_Range**
   mengelompokkan *feature* 'EstimatedSalary' ke dalam beberapa kategori berdasarkan range,sebagai berikut:
   a. Low = 0 - 51,001
   b. Medium = 51,002 - 100,193
   c. High = 100,194 - 149,387
   d. Very High = 149,388 - 200,000

   Feature 'Salary_Range' dapat digunakan untuk menyederhanakan analisis data, membuatnya lebih mudah diinterpretasi, dan memungkinkan model *machine learning* untuk menangkap pola yang terkait dengan tingkat pendapatan tertentu.

Masing-masing *range* pada *feature extraction* diperoleh dari nilai min, max, Q1, Q2, dan Q3.

Dataset setelah dilakukan *feature extraction*:

```
[39] df
```

	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Churn	Group_Age	Balance_Category	CreditScore_Range	Tenure_Category	Salary_Range
0	619	France	Female	42	2	0.00	1	1	1	101348.88	1	b.adult	Low	Fair	Short Term	High
1	608	Spain	Female	41	1	83807.86	1	0	1	112542.58	0	b.adult	Low	Fair	Short Term	High
2	502	France	Female	42	8	159660.80	3	1	0	113931.57	1	b.adult	High	Poor	Long Term	High
3	699	France	Female	39	1	0.00	2	0	0	93826.63	0	b.adult	Low	Good	Short Term	Medium
4	850	Spain	Female	43	2	125510.82	1	1	1	79084.10	0	b.adult	Medium	Excellent	Short Term	Medium
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9995	771	France	Male	39	5	0.00	2	1	0	96270.64	0	b.adult	Low	Excellent	Medium Term	Medium
9996	516	France	Male	35	10	57369.61	1	1	1	101699.77	0	b.adult	Low	Poor	Long Term	High
9997	709	France	Female	36	7	0.00	1	0	1	42085.58	1	b.adult	Low	Good	Long Term	Low
9998	772	Germany	Male	42	3	75075.31	2	1	0	92888.52	1	b.adult	Low	Excellent	Short Term	Medium
9999	792	France	Female	28	4	130142.79	1	1	0	38190.78	0	b.adult	High	Excellent	Medium Term	Low

10000 rows × 16 columns

```
[40] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype   
---  ------             --------------  -----   
 0   CreditScore        10000 non-null  int64   
 1   Geography          10000 non-null  object  
 2   Gender             10000 non-null  object  
 3   Age                10000 non-null  int64   
 4   Tenure             10000 non-null  int64   
 5   Balance            10000 non-null  float64 
 6   NumOfProducts      10000 non-null  int64   
 7   HasCrCard          10000 non-null  int64   
 8   IsActiveMember     10000 non-null  int64   
 9   EstimatedSalary    10000 non-null  float64 
 10  Churn              10000 non-null  int64   
 11  Group_Age          10000 non-null  object  
 12  Balance_Category   10000 non-null  category
 13  CreditScore_Range  10000 non-null  category
 14  Tenure_Category    10000 non-null  category
 15  Salary_Range       10000 non-null  category
dtypes: category(4), float64(2), int64(7), object(3)
memory usage: 977.3+ KB
```

D. Tuliskan minimal 4 feature tambahan (selain yang sudah tersedia di dataset) yang mungkin akan sangat membantu membuat performansi model semakin bagus (ini hanya ide saja, untuk menguji kreativitas teman-teman, tidak perlu benar-benar dicari datanya dan tidak perlu diimplementasikan)

Beberapa ide *features* yang bisa ditambahkan yang mungkin akan membantu meningkatkan performansi model, antara lain:

- **Balance-to-Salary Ratio:** Menghitung rasio antara saldo akun dan estimasi gaji pelanggan. Rasio ini dapat memberikan gambaran tentang seberapa besar bagian dari gaji yang disimpan atau diinvestasikan oleh pelanggan.
- **Average Transaction Amount:** Menghitung rata-rata jumlah transaksi per pelanggan. Ini dapat memberikan wawasan tentang seberapa sering pelanggan

berinteraksi dengan produk atau layanan, dan seberapa besar nilai transaksi yang mereka lakukan.

- **Tenure and NumOfProducts Interaction:** Fitur ini dapat memberikan model informasi tambahan tentang seberapa intensif pelanggan menggunakan produk atau layanan selama periode waktu tertentu. Dimana dapat diasumsikan bahwa semakin lama seseorang menjadi pelanggan dan semakin banyak produk yang mereka miliki, semakin kuat keterikatan mereka dengan layanan atau produk perusahaan tersebut.
- **Salary to CreditScore Ratio:** fitur ini mencerminkan seberapa besar pendapatan seseorang dibandingkan dengan skor kredit mereka. Rasio ini dapat memberikan wawasan tentang seberapa baik seseorang mengelola utang atau kredit relatif terhadap tingkat pendapatan mereka.

Namun, untuk fitur-fitur tersebut hanya bersifat rekomendasi yang mana tidak akan ditambahkan atau diimplementasikan ke dataset.

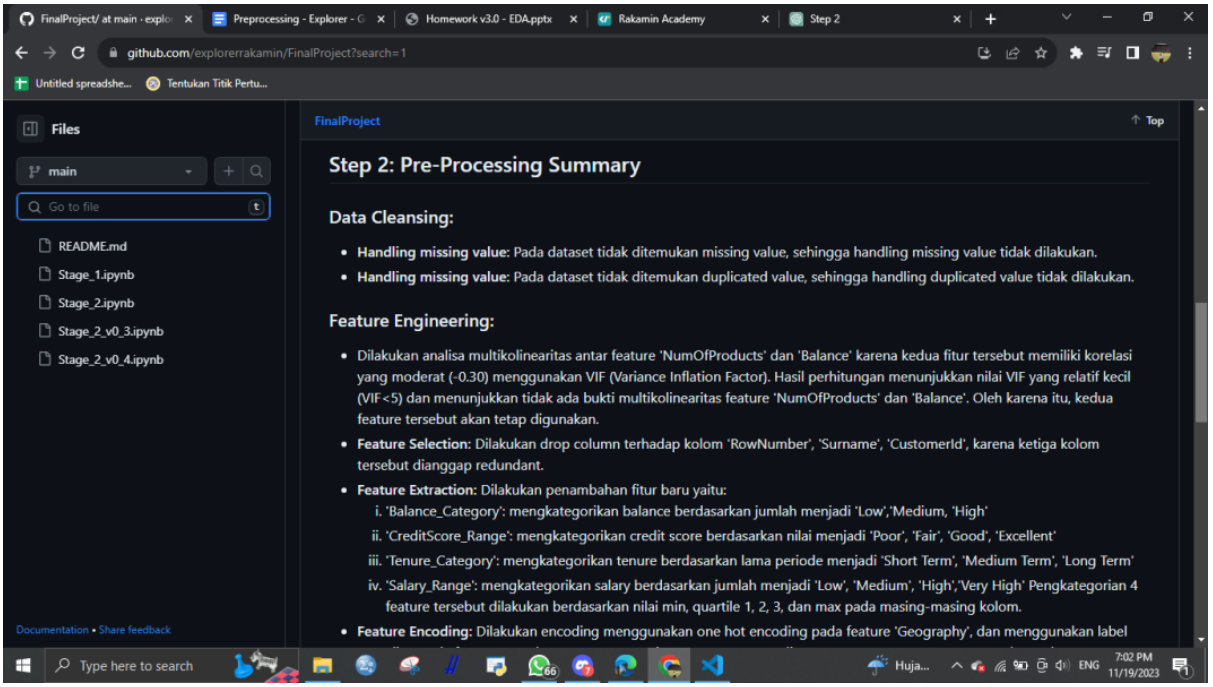
3. Git (15 poin)

Upload project teman-teman di sebuah repository git. Berkolaborasilah di Git jika ada perubahan version dari waktu ke waktu.

A. Buat Repository Git

Repository Git terlampir dalam link github
https://github.com/explorerrakamin/FinalProject.git

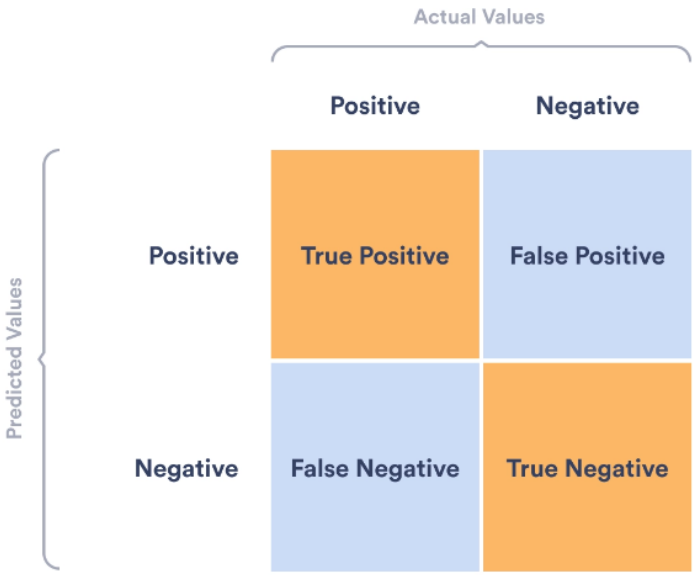
B. Upload file notebook atau file pengerjaan lainnya pada repository tersebut



Untuk file README, dapat merupakan summary dari proses data praproses yang telah dilakukan. Boleh menggunakan repositori yang sama atau membuat baru. Sedangkan, untuk file .ipynb yang terupdate menggunakan file berjudul **“Stage_2_v0_4.ipynb”**

Stage 3 - Modelling

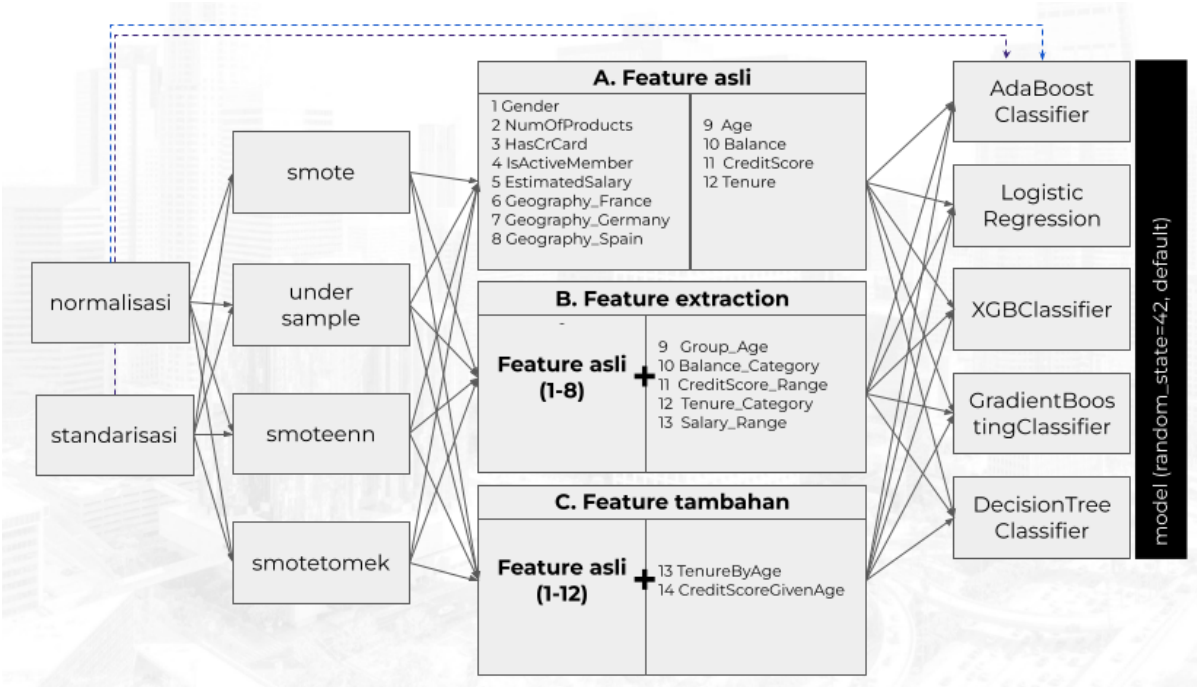
1. Pemilihan Metrik Evaluasi



Metrics evaluation yang digunakan ialah "Recall". Hal ini dikarenakan pada kasus ini bertujuan untuk mengurangi false positive (dalam kasus ini false positif itu sendiri adalah nasabah yang pada kenyataannya churn namun tidak terprediksi churn) dikarenakan cost yang dibutuhkan untuk mendapatkan customer akan lebih mahal dibandingkan untuk mempertahankannya. Oleh karena itu, model ini bertujuan untuk memprediksi nasabah yang akan churn sehingga dapat dilakukan treatment tertentu untuk mencegah customer menjadi churn.

2. Skema Model *Machine Learning*

Dalam pemilihan model *machine learning* terbaik dilakukan banyak kemungkinan yang dapat terjadi seperti terlihat pada gambar berikut :



Terlihat seperti gambar di atas dilakukan beberapa pembuatan data frame pada data train dengan total berjumlah sepuluh data frame. Untuk perlakuan data yang tidak seimbang (*imbance data*) dilakukan penambahan dua metode *imbalance data* yaitu smoteenn dan smotetomek.

Kemudian ada tiga jenis fitur yang diuji coba. Yang pertama adalah fitur asli, dalam hal ini fitur asli yang dimaksud adalah fitur bawaan yang sedari awal sudah ada, namun sudah dalam bentuk hasil encoding atau angka. Sebagai contoh seperti gender. Fitur gender sudah ada sedari awal, namun karena masih dalam bentuk *male* dan *female* maka perlu diubah menjadi *label encoding*. Hal yang sama berlaku juga pada kolom geography yang diubah menjadi *One-Hot Encoding*. Fitur kelompok kedua pada halnya merupakan kelompok pertama, namun terdapat perubahan pada kolom umur, *balance*, *credit score*, *tenure*, dan *salary*. Perubahan dalam kolom tersebut adalah sudah dalam bentuk *range* sehingga terbagi menjadi beberapa kelompok yang kemudian dilakukan *label encoding* yang telah dilaksanakan pada bagian *feature engineer* sehingga penamaan kolom tersebut menjadi *range* atau *category*. Percobaan pemodelan kelompok fitur yang terakhir adalah adanya fitur tambahan yaitu *TenureByAge* dan *CreditScoreGivenAge*. Fitur kelompok ketiga adalah fitur asli (fitur A) yang berjumlah sepuluh ditambah dua fitur yang telah disebutkan sebelumnya.

Data	*Handle Outlier*	Transformasi	Imbalance
df1_x_train	x	normalisasi	x
df2_x_train	x	standarisasi	x

df3_x_train	x	normalisasi	smote
df4_x_train	x	normalisasi	under sample
df5_x_train	x	standarisasi	smote
df6_x_train	x	standarisasi	under sample
df7_x_train	x	normalisasi	smotenn
df8_x_train	x	normalisasi	smotetomek
df9_x_train	x	standarisasi	smotenn
df10_x_train	x	standarisasi	smotetomek

Tabel di atas menunjukkan bahwa terdapat sepuluh data frame yang akan dilakukan percobaan pemodelan untuk mendapatkan data frame mana dengan *handling* berbeda-beda yang menghasilkan parameter recall train dan recall test tertinggi, namun rentang atau jarak antara matriksnya tidak terlalu jauh supaya tidak terjadi *overfitting* maupun *underfitting* yang terlalu signifikan

Untuk jenis model *machine learning* yang diuji coba berjumlah lima yaitu AdaBoost Classifier, Logistic Regression, XGBClassifier, Gradient Boosting Classifier, dan Decision Tree Classifier. Parameter model yang diatur adalah random_state bernilai 42 sedangkan parameter lainnya *default*.

3. Pemiihan Fitur dan Model Terbaik

a) Menggunakan Fitur Asli

	data	list_method	prec_train	prec_test	recall_train	recall_test	AUC_train	AUC_test
0	df9_x_train	AdaBoostClassifier	90.254	43.801	90.418	82.697	96.084	84.729
1	df7_x_train	LogisticRegression	80.729	32.242	84.405	80.153	85.866	77.819
2	df9_x_train	GradientBoostingClassifier	93.615	46.244	93.582	79.898	98.049	86.166
3	df9_x_train	LogisticRegression	80.465	33.018	82.661	79.898	85.592	77.877
4	df6_x_train	GradientBoostingClassifier	83.119	50.000	78.771	79.389	89.995	86.823
5	df4_x_train	GradientBoostingClassifier	83.119	50.000	78.771	79.389	89.995	86.823

b) Menggunakan Fitur Extraction

	data	list_method	prec_train	prec_test	recall_train	recall_test	AUC_train	AUC_test
0	df9_x_train	AdaBoostClassifier	88.704	40.293	90.436	77.099	95.300	82.292
1	df7_x_train	LogisticRegression	78.450	30.101	80.250	75.827	81.761	73.467
2	df9_x_train	LogisticRegression	77.857	30.512	78.828	74.300	80.991	73.457
3	df7_x_train	AdaBoostClassifier	91.194	45.928	90.969	71.756	95.894	82.361
4	df6_x_train	AdaBoostClassifier	76.451	43.012	71.290	70.483	82.916	81.235
5	df4_x_train	AdaBoostClassifier	76.451	43.012	71.290	70.483	82.916	81.235

c) Menggunakan Fitur Tambahan

	data	list_method	prec_train	prec_test	recall_train	recall_test	AUC_train	AUC_test
0	df9_x_train	AdaBoostClassifier	90.396	43.316	90.363	82.443	96.187	85.097
1	df9_x_train	GradientBoostingClassifier	93.716	45.911	93.564	81.425	98.051	86.315
2	df7_x_train	LogisticRegression	79.986	31.511	84.462	80.662	86.044	77.851
3	df9_x_train	LogisticRegression	79.681	31.874	83.023	79.644	85.847	77.941
4	df6_x_train	GradientBoostingClassifier	83.614	49.761	78.528	79.389	90.291	86.478
5	df7_x_train	GradientBoostingClassifier	92.922	46.637	93.644	79.389	97.758	86.211

Dari hasil ketiga kelompok fitur yang telah diuji coba, fitur yang menghasilkan lebih buruk dibandingkan dengan yang lain adalah kelompok fitur extraction dengan fitur yang dikelompokkan berdasarkan *range*nya. Kemudian, perbandingan antara kelompok fitur asli dengan fitur tambahan berdasarkan recallnya memiliki nilai yang tidak jauh berbeda. Sehingga terpilih penggunaan kelompok fitur tambahan dikarenakan jumlah fitur lebih banyak yang mungkin saja nantinya berperan dalam *feature importance*. Setelah dilakukan pemilihan kelompok fitur, dilihat hasil tertinggi nilai recall nya. Dari tabel c dipilih dua teratas yaitu data yang digunakan adalah data df9_x_train dimana dikenai *treatment* standarisasi dan smotenn, sedangkan algoritma ML Supervised yang dipakai yaitu AdaboostClassifier dan GradientBoosting Classifier.

Berikut merupakan hasil model terbaik berdasarkan matrik evaluasi tertinggi

AdaBoostClassifier (random_state=42,default)	
Parameter	Nilai (%)
recall train	90.363
recall test	82.443

GradientBoostingClassifier (random_state=42,default)	
Parameter	Nilai (%)
recall train	93.564
recall test	81.425

Terpilih terbaik terdapat dua model ML yaitu AdaBoostClassifier dan GradientBoostingClassifier. Dari gambar sebelah kiri adalah model AdaBoost Classifier menggunakan parameter *default* menghasilkan nilai recall train dan recall test masing-masing sebesar 90.363 % dan 82.443 %, kemudian gambar dari sebelah kanan terlihat dengan model Gradient Boosting Classifier menghasilkan nilai recall train 93.564 % persen dan recall test 81.425 %. Untuk nilai recall test tertinggi adalah model AdaBoost Classifier, namun dari kedua model ini sama-sama masih mengalami *over-fitting* dengan ditandainya nilai recall train yang lebih besar dibandingkan recall test dengan selisih pada AdaBoost Classifier sebesar 7.92 persen dan Gradient Boosting sebesar 12.14 persen. Oleh sebab itu perlu dilakukan *hypertuning* parameter diharapkan untuk menaikkan nilai recall dan mendapatkan parameter yang pas agar mengurangi selisih terjadinya *over-fitting*.

4. Hypertuning Parameter

Hypertuning perlu dilakukan agar diharapkan mendapat nilai *recall* yang lebih baik dari segi besaran maupun kondisi tertentu seperti *underfitting* maupun *overfitting*. Tuning yang dilakukan adalah merubah-rubah parameter yang ada dalam suatu algoritma *machine learning* dari yang sebelumnya adalah parameter yang bernilai *default*. Dalam tuning parameter ini dilakukan dengan dua metode yaitu dengan menggunakan tuning manual dan Grid Search.

4.a) Tuning Manual

Tuning manual adalah dengan mencoba-coba suatu nilai di setiap satu parameter dengan bantuan *learning curve* jika nilai yang akan dicoba lebih dari dua. Sederhananya adalah dengan melakukan looping pada setiap parameter yang menghasilkan nilai recall train, recall test, dan delta recall (selisih antara tes dan train).

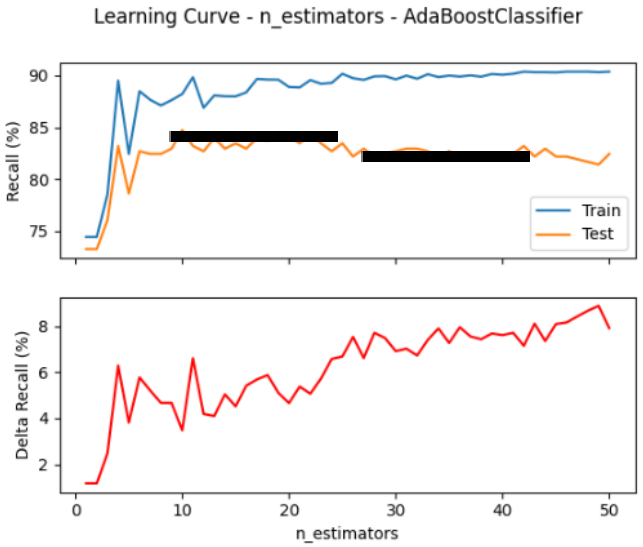
1) AdaBoostClassifier

- Tuning Algoritma : SAMME dan SAMME.R

Metrics	Data	SAMME	SAMME.R
Recall, persen	Train	88.5	90.4
	Test	82.7	82.4
Crossval, persen	Train	84.3	85.7
	Test	83.7	85.7

Secara keseluruhan SAMME.R mendapatkan hasil yang lebih baik dibandingkan dengan SAMME. Oleh sebab itu parameter pada jenis algoritma yang terpilih adalah SAMME.R

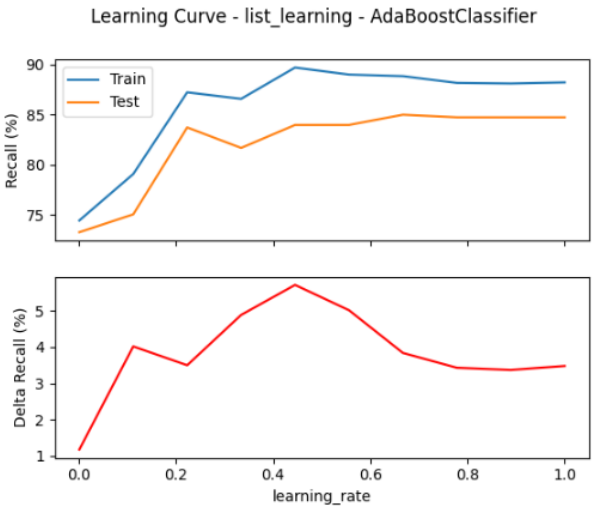
- Tuning n_estimators



Terlihat semakin tinggi nilai n_estimators nilai recall trainnya juga relatif semakin meningkat walaupun tidak signifikan. Namun, peningkatan recall train

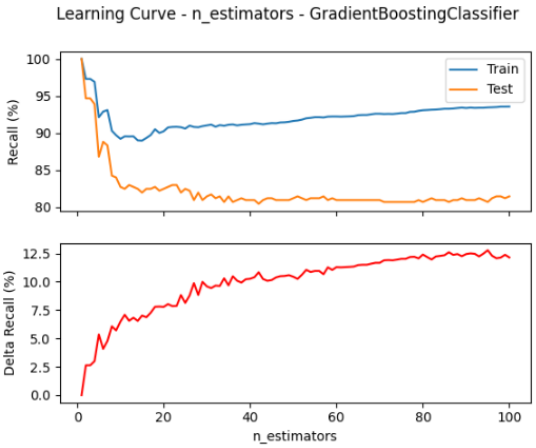
tidak dibarengi dengan peningkatan recall testnya menyebabkan delta recallnya semakin meningkat. Terlihat dari garis berwarna hitam yang berada di atas garis orange, garis hitam di sisi kiri memiliki nilai recall test yang lebih tinggi dibandingkan dengan garis hitam sebelah kanan artinya ada gap yang menandakan semakin tinggi n_estimators semakin terjadi *overfitting*. Untuk pemilihan nilai n_estimator terbaik bernilai 10 karena dilihat grafik berwarna merah memiliki nilai yang rendah. Sehingga sementara digunakan tuning sebagai berikut AdaBoostClassifier(algorithm='SAMME.R', random_state=42, n_estimators=10)

- Learning_rate



Terlihat semakin besar parameter learning_rate menyebabkan nilai recall trend nya semakin naik hingga pada learning rate > 0.7 (lebih dari 7) mengalami nilai yang statis. Artinya dengan penambahan learning rate di atas 7 tidak menaikkan nilai recall. Nilai learning_rate yang dipilih 0.8 karena pada titik ini memiliki delta recall yang kecil dan pada kondisi score yang sudah tidak fluktuatif. Sehingga digunakan tuning sebagai berikut AdaBoostClassifier (algorithm='SAMME.R', random_state=42, n_estimators=10, learning_rate=0.8)

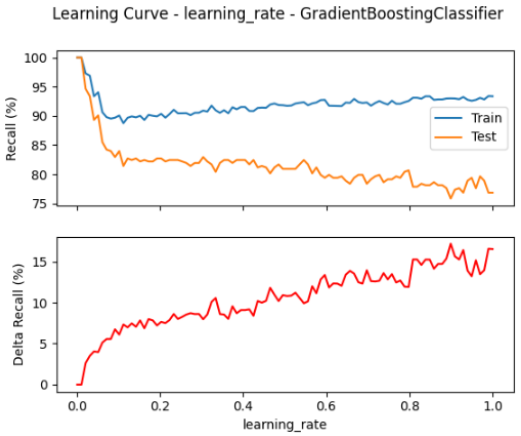
2) Gradient Boosting Classifier
   - n_estimators




```
param value: 10; train: 89.19; test: 82.7 delta: 6.489999999999995
param value: 11; train: 89.53; test: 82.44 delta: 7.090000000000003
param value: 12; train: 89.51; test: 82.95 delta: 6.560000000000002
param value: 13; train: 89.53; test: 82.7 delta: 6.829999999999998
param value: 14; train: 88.99; test: 82.44 delta: 6.549999999999997
param value: 15; train: 88.95; test: 81.93 delta: 7.019999999999996
param value: 16; train: 89.31; test: 82.44 delta: 6.870000000000045
param value: 17; train: 89.71; test: 82.44 delta: 7.269999999999996
param value: 18; train: 90.49; test: 82.7 delta: 7.789999999999992
param value: 19; train: 90.0; test: 82.19 delta: 7.810000000000002
param value: 20; train: 90.22; test: 82.44 delta: 7.780000000000001
param value: 21; train: 90.73; test: 82.7 delta: 8.030000000000001
```

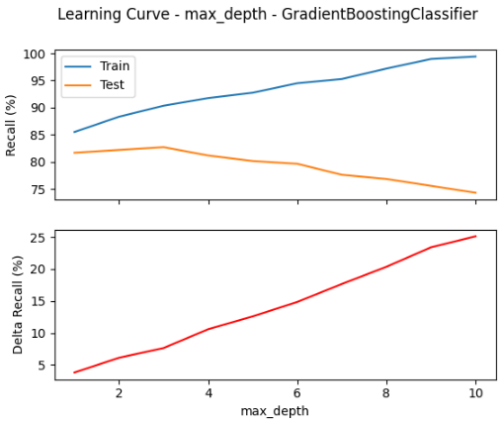
Semakin tinggi nilai n_estimators menyebabkan nilai recall train juga semakin tinggi, namun tidak diikuti dengan nilai recall testnya. Justru memiliki trend meningkat ketika nilai n_estimators sekitar kurang dari 20. Sehingga ketika nilainya lebih dari dari 20 delta recall yang ditandai dengan garis berwarna merah menjadi semakin naik. Nilai n_estimators terbaik adalah bernilai 12 karena memiliki delta yang kecil walaupun tidak sekecil n_estimators yang bernilai 10. Namun, pada n_estimators 12 memilik nilai recall train dan test yang lebih tinggi dibandingkan dengan 10. Maka digunakan tuning sebagai berikut GradientBoostingClassifier(random_state=42, n_estimators=12).

- Learning_rate



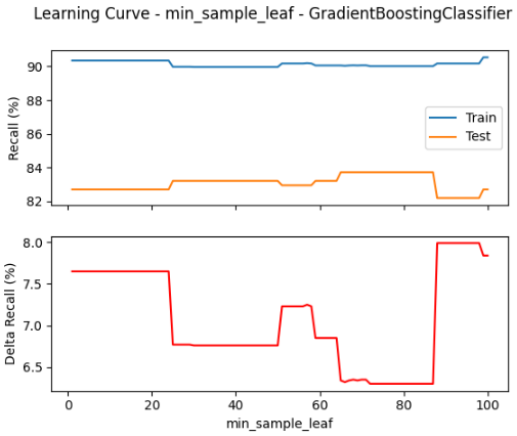
Semakin tinggi nilai *learning rate* menyebabkan nilai recall train juga semakin tinggi, namun recall testnya justru semakin turun. Pada kondisi learning rate antara 0.2 sampai 0.4 bernilai stabil yang kemudian setelah nilai 0.4 terlihat garis berwarna merah memiliki tren naik. Oleh sebab itu, pemilihan learning_rate yang terbaik adalah 0.2. Maka digunakan tuning sebagai berikut GradientBoostingClassifier (random_state=42, n_estimators=12, learning_rate=0.2).

- max_depth



Garis berwarna orange yang menandakan nilai recall test memiliki trend yang sedikit naik pada rentang nilai max_depth 1 sampai 3. nilai recall test kemudian semakin turun ketika max_depth nya lebih besar dari 3, walaupun nilai recall trainnya selalu naik. Oleh sebab itu, nilai terbaik pada parameter max_depth adalah 3 karena berada di puncak paling tinggi nilai recall testnya. Maka digunakan tuning sementara sebagai berikut GradientBoostingClassifier (random_state=42, n_estimators=12, learning_rate=0.2, max_depth=3).

- min_samples_leaf



Dapat dilihat dari grafik di atas bahwa perubahan nilai pada parameter min_sample_leaf berubah-ubah naik turun. Oleh sebab itu dipilih nilai dengan mempertimbangkan delta recall yang kecil. Dalam hal ini nilai min_sample_leaf yang diambil adalah 65. Maka digunakan tuning hyperparameter sebagai berikut GradientBoostingClassifier(random_state=42,n_estimators=12, learning_rate=0.2, max_depth=3).

4.b) Tuning GridSearch

Tuning parameter menggunakan GridSearch merupakan *library* yang diimport melalui Sklearn. Tuning ini melakukan semua kemungkinan dari parameter-parameter yang diinput dan menampilkan parameter . Artinya apabila dalam parameter x dan y di suatu algoritma ML memiliki masing-masing 3 data input yaitu 3 x dan 3 y. Dengan menggunakan GridSearchCV ini menghasilkan 9 kandidat dengan output menampilkan parameter terbaik dari kandidat tersebut. Kelemahan dari GridSearchCV adalah tidak efisien karena membutuhkan waktu running yang lama dan modeller tidak tahu pengaruh dari masing-masing parameter karena langsung menampilkan hasil akhir.

1) AdaBoost Classifier

Dilakukan gridsearch dengan parameter yang dituning adalah n_estimators dan learning_rate. Pada n_estimators menggunakan nilai 50, 100, dan 200, sedangkan pada learning_rate menggunakan nilai 0.01, 0.1, 0.2. Dari hasil GridSearchCV mendapatkan parameter terbaik n_estimators dan learning_rate masing-masing sebesar 200 dan 0.01. Dari parameter tersebut menghasilkan *score* recall train dan recall test masing-masing 90.7% dan 86.5%. Nilai recall berdasarkan Crossvalidate train dan test adalah 78.6% dan 78.3%.

2) GradientBoost Classifier

Dilakukan gridsearach dengan parameter yang dituning adalah n_estimators, learning_rate, max_depth, min_samples_split, min_sample_leaf, dan subsample. n_estimators bernilai 10, 20, dan 30. Learning_rate bernilai 0.01, 0.1, dan 0.2, max_depth 3, 4, dan 5, min_samples_split bernilai 2, 5, dan 10, min_samples_leaf 1, 2, dan 4, subsample 0.8, 0.9, dan 1. Dari parameter tersebut menghasilkan *score* recall train dan recall test masing-masing 100% dan 100%. Nilai recall berdasarkan Crossvalidate train dan test adalah 87% dan 86.2%.

4.c) Hasil Parameter Tuning Terbaik

Tabel di bawah menunjukkan nilai terbaik dari setiap parameter yang dilakukan melalu manual tuning maupun tuning menggunakan GridSearchCV

AdaBoost Classifier	**Manual**	**GridSearchCV**
algoritma	SAMME.R	SAMME.R
n_estimator	10	200
learning_rate	0.8	0.01

GradientBoosting Classifier	**Manual**	**GridSearchCV**
n_estimator	12	10
learning_rate	0.2	0.01
max_depth	3	3
min_samples_leaf	65	1
min_samples_split	default	2
subsample	default	0.8

5. Pemilihan Akhir Model

AdaBoost Classifier	Manual	GridSearchCV	Gradient Boosting	Manual	GridSearchCV
recall train (%)	88.2	90.7	recall train (%)	90.1	100
recall test (%)	84.7	86.5	recall test (%)	83.7	100
recall train (%) (cross validation)	83.4	78.6	recall train (%) (cross validation)	84.4	87
recall test (%) (cross validation)	83.2	78.3	recall test (%) (cross validation)	85.1	86.2

Dari tabel di atas terlihat perbandingan Adaboost Classifier antara tuning parameter manual dengan tuning GridSearchCV. Jika dilihat dari score recall (biasa), GridSearchCV memiliki hasil yang lebih tinggi dibandingkan dengan tuning manual yaitu dengan recall tran dan testnya masing-masing sebesar 90.7% dan 86.5%. Namun, setelah dicek menggunakan *cross validation* ternyata mendapatkan score recall yang rendah yaitu di bawah 80 persen, lebih rendah dibandingkan dengan tuning manual. Dibandingkan dengan tuning manual, hasil tuning manual pada pengukuran recall (biasa) dengan recall cross validation memiliki nilai yang relatif sama yaitu berkisar di 83% - 84% walaupun masih terjadi *overfitting* sebesar 3.5 persen. Oleh sebab itu, Adaboost Classifier terbaik adalah menggunakan hasil parameter tuning manual.

Kemudian pada Gradient Boosting Classifier, tuning manual masih terjadi overfitting yang cukup besar yaitu 6.4 persen. Dibandingkan dengan menggunakan GridSearchCV, hasil dari GridSeachCV terlihat lebih baik dengan rata-rata hasil recall yang lebih tinggi dibandingkan Gradient Boosting tuning manual. Namun, perlu dilihat lagi menggunakan *report* lebih lanjut karena GridSearchCV karena menghasilkan hasil yang sempurna yaitu nilai recall 100 persen. Untuk sementara pada model ML Gradient Boosting Classifier yang terpilih adalah tuning menggunakan GridSearchCV.

AdaBoostClassifier Hypertuning (manual)

report test :

	precision	recall	f1-score	support
0	0.95	0.69	0.80	1607
1	0.40	0.85	0.54	393
accuracy			0.72	2000
macro avg	0.67	0.77	0.67	2000
weighted avg	0.84	0.72	0.75	2000

report train :

	precision	recall	f1-score	support
0	0.84	0.83	0.83	4111
1	0.87	0.88	0.88	5531
accuracy			0.86	9642
macro avg	0.86	0.86	0.86	9642
weighted avg	0.86	0.86	0.86	9642

GradientBoostingClassifier Hypertuning (GridSearchCV)

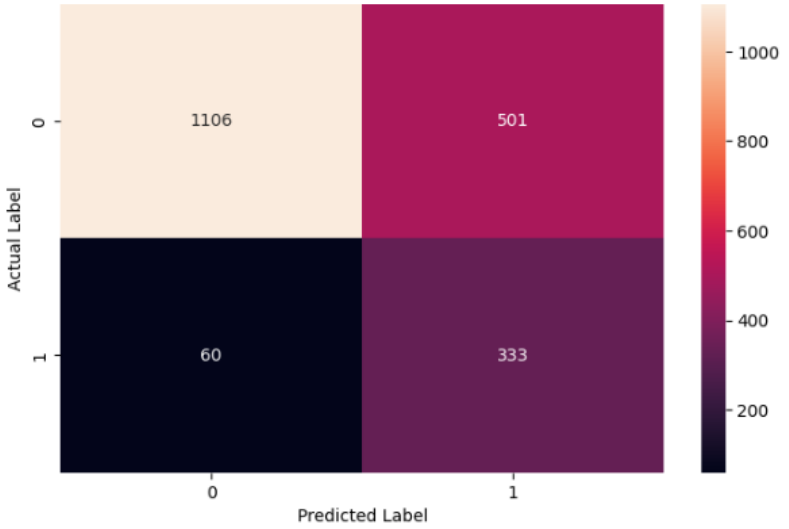
report test :

	precision	recall	f1-score	support
0	0.00	0.00	0.00	1607
1	0.20	1.00	0.33	393
accuracy			0.20	2000
macro avg	0.10	0.50	0.16	2000
weighted avg	0.04	0.20	0.06	2000

report train :

	precision	recall	f1-score	support
0	0.00	0.00	0.00	4111
1	0.57	1.00	0.73	5531
accuracy			0.57	9642
macro avg	0.29	0.50	0.36	9642
weighted avg	0.33	0.57	0.42	9642

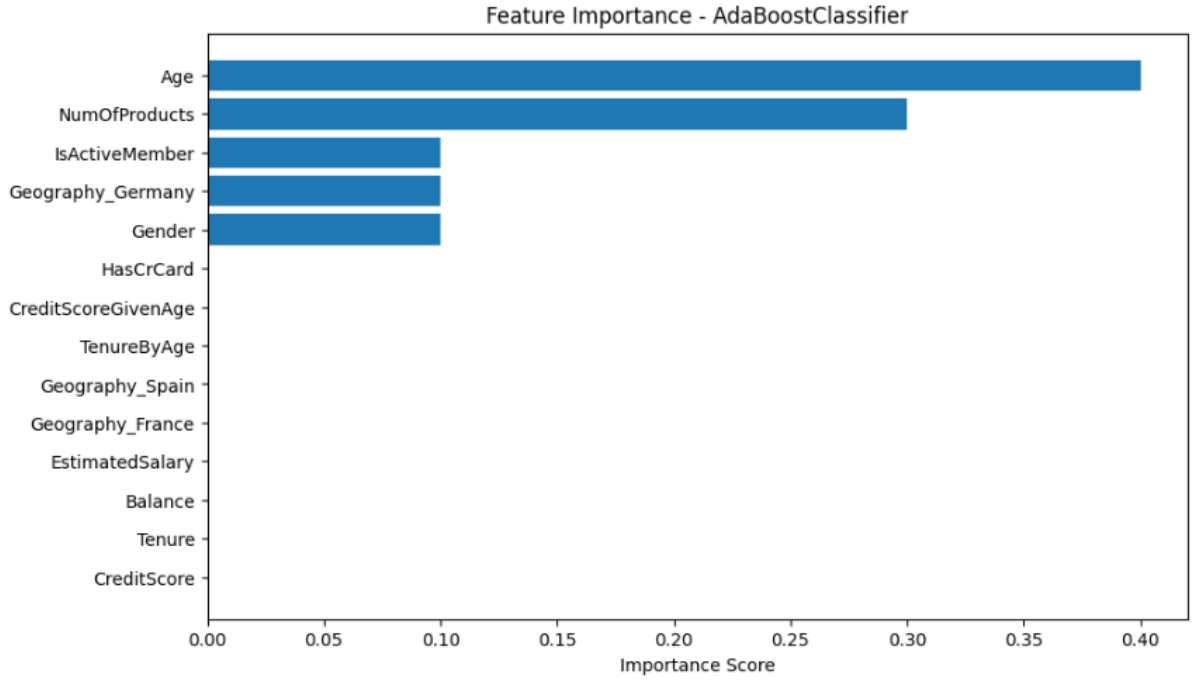
Gambar di atas merupakan laporan lanjutan yang didapatkan menggunakan *classification report*. Dapat disimpulkan bahwa setelah dibandingkan dengan GradientBoosting, Gradient Boosting memiliki recall 1 yang tinggi namun, recall 0 nya bernilai 0%. Artinya bahwa gradient boosting memprediksi sangat banyak nasabah yang churn walaupun sebenarnya nasabah tersebut tidak churn. Oleh sebab itu model terbaik secara keseluruhan adalah Adaboosting Classifier dengan tuning manual.



Gambar di atas merupakan confusion matrix dari hasil modelling yang telah dilakukan dengan recall test sebesar 84.7 persen

6. Features Importance

Setelah mendapatkan model yang paling baik dengan menggunakan algoritma AdaBoost Classifier didapatkan hasil sebagai berikut



dengan nilai pada setiap fiturnya

	Feature	Importance
0	CreditScore	0.0
3	Tenure	0.0
4	Balance	0.0
6	EstimatedSalary	0.0
7	Geography_France	0.0
9	Geography_Spain	0.0
10	TenureByAge	0.0
11	CreditScoreGivenAge	0.0
12	HasCrCard	0.0
2	Gender	0.1
8	Geography_Germany	0.1
13	IsActiveMember	0.1
5	NumOfProducts	0.3
1	Age	0.4

- Berdasarkan ***feature importance*** yang telah dibuat, dipilih empat fitur penting yang memiliki kontribusi terhadap *churn rate customer*. Diantaranya yaitu **Age** dengan

persentase kontribusi sebesar **40%** terhadap *churn rate customer*. Kedua **NumOfProducts** yang memiliki persentase kontribusi sebesar **30%**. Kemudian **IsActiveMember** dan **Gender** masing-masing memiliki persentase kontribusi sebesar **10%** terhadap *churn rate*. Keempat fitur tersebut memiliki kontribusi dalam menentukan apakah nasabah dengan nilai diatas akan memutuskan untuk churn atau tidak.

Business Insight dan Rekomendasi

Berdasarkan observasi yang dilakukan pada tahap **Exploratory Data Analysis (EDA)** dan nilai ***feature importance*** yang dihasilkan pada proses Modelling menggunakan algoritma AdaBoost Classifier, maka diketahui faktor-faktor yang dapat mempengaruhi keputusan customer untuk **Churn** adalah sebagai berikut:

- **Age (Umur Customer)**.

  Dari nilai feature importance diketahui umur memiliki kontribusi terhadap keputusan customer untuk Churn. Berdasarkan analisis data eksploratif yang telah dilakukan, diketahui customer di kelompok umur 46-65 tahun memiliki tingkat churn yang tinggi. Sehingga rekomendasi yang bisa diberikan yaitu melakukan **survei khusus** yang ditargetkan pada kelompok umur tersebut yang bertujuan untuk memahami lebih dalam mengenai kebutuhan dan preferensi mereka. Kemudian dari hasil survei tersebut, Bank dapat menyesuaikan produk dan layanan yang lebih relevan serta strategi pemasaran yang sesuai untuk mendorong retensi customer di kelompok umur tersebut.

- **NumOfProducts (Jumlah produk yang digunakan Customer)**.

  Dari nilai feature importance diketahui ‘NumOfProducts’ memiliki kontribusi terhadap keputusan customer untuk Churn. Berdasarkan analisis data eksploratif yang telah dilakukan, diketahui customer yang menggunakan 4 produk memiliki tingkat churn yang sangat tinggi (100% dari 60 customer memutuskan untuk Churn). Namun jika dilihat dari jumlah, customer yang menggunakan 1 produk memiliki jumlah Churn yang paling banyak. Sehingga rekomendasi yang bisa diberikan yaitu mengembangkan **program loyalitas** yang menawarkan insentif atau manfaat lebih ketika customer menggunakan sejumlah produk, dimana semakin banyak produk yang dimiliki customer akan semakin banyak manfaat yang didapatkan. Hal ini diharapkan dapat mendorong customer untuk meningkatkan penggunaan produk serta tertarik menggunakan lebih dari satu produk.

- **IsActiveMember (Customer yang aktif menggunakan produk dan layanan Bank)**

  Dari nilai feature importance diketahui ‘IsActiveMember’ memiliki kontribusi terhadap keputusan customer untuk Churn. Berdasarkan analisis data eksploratif yang telah

dilakukan, diketahui customer yang tidak secara aktif menggunakan produk dan layanan Bank memiliki tingkat Churn yang tinggi. Sehingga rekomendasi yang bisa diberikan yaitu mengirimkan **reminder atau notifikasi** kepada customer melalui e-mail secara rutin. Notifikasi tersebut dapat berupa penawaran program loyalitas yang diberikan kepada customer yang mendorong penggunaan kembali produk dan layanan secara aktif. Misalnya dengan memberikan reward pada aktivitas pertama kali saat menggunakan produk dan layanan kembali, serta poin pada setiap aktivitas transaksi yang dilakukan, dimana poin-poin tersebut dapat dikumpulkan untuk mendapatkan manfaat tertentu. Sehingga diharapkan dapat mendorong keaktifan nasabah dalam penggunaan produk dan layanan dan menurunkan tingkat churn.

- **Gender (Jenis kelamin Customer)**

  Dari nilai feature importance diketahui ‘Gender’ memiliki kontribusi terhadap keputusan customer untuk Churn. Berdasarkan analisis data eksploratif yang telah dilakukan, diketahui customer wanita memiliki tingkat churn yang tinggi. Sehingga rekomendasi yang bisa diberikan yaitu melakukan **evaluasi produk** dan layanan yang sudah ada untuk memastikan kesesuaiannya terhadap kebutuhan dan preferensi customer wanita. Dari hasil evaluasi yang dilakukan, Bank dapat merancang dan mengembangkan produk dan layanan khusus yang lebih relevan dengan kebutuhan dan preferensi customer wanita. Sehingga diharapkan dapat mendorong retensi customer wanita.