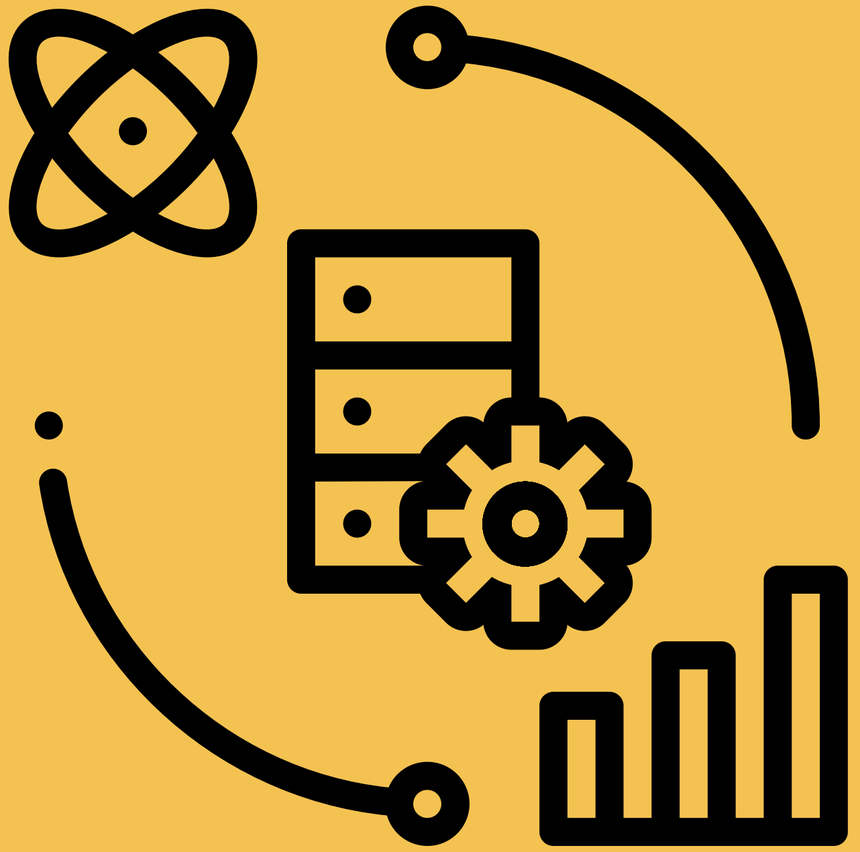


Challenge Chapter II
Group J

Dian Ifta Khana
Dimas Ari Lumintang

Prediksi *Customer Churn* Perusahaan Telekomunikasi

Import Data

Import Data

```
[3]  from google.colab import files
     data_train = files.upload()
```

Choose Files Data Train.csv

- **Data Train.csv**(text/csv) - 387621 bytes, last modified: 3/28/2023 - 100% done

```
Saving Data Train.csv to Data Train.csv
```

```
[4]  data_test = files.upload()
```

Choose Files Data Test.csv

- **Data Test.csv**(text/csv) - 69310 bytes, last modified: 3/28/2023 - 100% done

```
Saving Data Test.csv to Data Test.csv
```

```
[41] import pandas as pd
     df_data_train = pd.read_csv('Data Train.csv')
     df_data_test = pd.read_csv('Data Test.csv')
```

Tahap pertama yang dilakukan untuk memprediki *customer churn* adalah *import* data train dan data test.

#1

EXPLORATORY DATA ANALYSIS (EDA)

Investigasi awal untuk lebih memahami data yang akan dianalisis.

Menampilkan isi data yang telah kita import

```
[7] df_data_train
```

	state	account_length	area_code	international_plan	voice_mail_plan	number_vmail_messages	total_day_minutes	total_day_calls	total_day_charge	total_eve_minutes	total_eve_calls	total_eve_charge	total_night_minutes	total_night_calls
0	OH	107	area_code_415	no	yes	26	161.6	123	27.47	195.5	103	16.62	254.4	103
1	NJ	137	area_code_415	no	no	0	243.4	114	41.38	121.2	110	10.30	162.6	104
2	OH	84	area_code_408	yes	no	0	299.4	71	50.90	61.9	88	5.26	196.9	89
3	OK	75	area_code_415	yes	no	0	166.7	113	28.34	148.3	122	12.61	186.9	121
4	MA	121	area_code_510	no	yes	24	218.2	88	37.09	348.5	108	29.62	212.6	118
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
4245	MT	83	area_code_415	no	no	0	188.3	70	32.01	243.8	88	20.72	213.7	79
4246	WV	73	area_code_408	no	no	0	177.9	89	30.24	131.2	82	11.15	186.2	89
4247	NC	75	area_code_408	no	no	0	170.7	101	29.02	193.1	126	16.41	129.1	104
4248	HI	50	area_code_408	no	yes	40	235.7	127	40.07	223.0	126	18.96	297.5	116
4249	VT	86	area_code_415	no	yes	34	129.4	102	22.00	267.1	104	22.70	154.8	100

4250 rows × 20 columns

```
[8] df_data_test
```

	id	state	account_length	area_code	international_plan	voice_mail_plan	number_vmail_messages	total_day_minutes	total_day_calls	total_day_charge	total_eve_minutes	total_eve_calls	total_eve_charge	total_night_minutes	total_night_cal
0	1	KS	128	area_code_415	no	yes	25	265.1	110	45.07	197.4	99	16.78	244.7	
1	2	AL	118	area_code_510	yes	no	0	223.4	98	37.98	220.6	101	18.75	203.9	1
2	3	IA	62	area_code_415	no	no	0	120.7	70	20.52	307.2	76	26.11	203.0	
3	4	VT	93	area_code_510	no	no	0	190.7	114	32.42	218.2	111	18.55	129.6	1
4	5	NE	174	area_code_415	no	no	0	124.3	76	21.13	277.1	112	23.55	250.7	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
745	746	GA	130	area_code_415	no	no	0	119.4	99	20.30	226.3	97	19.24	202.7	1
746	747	WA	73	area_code_408	no	no	0	177.2	118	30.12	270.5	84	22.99	241.8	1
747	748	WV	152	area_code_415	no	no	0	184.2	90	31.31	256.8	73	21.83	213.6	1
748	749	DC	61	area_code_415	no	no	0	140.6	89	23.90	172.8	128	14.69	212.4	
749	750	DC	109	area_code_510	no	no	0	188.8	67	32.10	171.7	92	14.59	224.4	

750 rows × 20 columns

```
[13] df_data_train.shape

(4250, 20)
```

Dataset terdiri dari 4250 baris dan 20 kolom

```
df_data_test.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 750 entries, 0 to 749
Data columns (total 20 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   id                            750 non-null    int64
 1   state                         750 non-null    object
 2   account_length                750 non-null    int64
 3   area_code                     750 non-null    object
 4   international_plan            750 non-null    object
 5   voice_mail_plan               750 non-null    object
 6   number_vmail_messages         750 non-null    int64
 7   total_day_minutes             750 non-null    float64
 8   total_day_calls               750 non-null    int64
 9   total_day_charge              750 non-null    float64
 10  total_eve_minutes             750 non-null    float64
 11  total_eve_calls               750 non-null    int64
 12  total_eve_charge              750 non-null    float64
 13  total_night_minutes           750 non-null    float64
 14  total_night_calls             750 non-null    int64
 15  total_night_charge            750 non-null    float64
 16  total_intl_minutes            750 non-null    float64
 17  total_intl_calls              750 non-null    int64
 18  total_intl_charge             750 non-null    float64
 19  number_customer_service_calls 750 non-null    int64
dtypes: float64(8), int64(8), object(4)
memory usage: 117.3+ KB
```

Dalam Dataset terdapat data dengan type yang beragam, terdapat data berupa integer, float, dan object.

```
df_data_train.nunique()

state                           51
account_length                 215
area_code                        3
international_plan               2
voice_mail_plan                  2
number_vmail_messages           46
total_day_minutes             1843
total_day_calls                120
total_day_charge              1843
total_eve_minutes             1773
total_eve_calls                123
total_eve_charge              1572
total_night_minutes           1757
total_night_calls              128
total_night_charge             992
total_intl_minutes             168
total_intl_calls                21
total_intl_charge              168
number_customer_service_calls   10
churn                            2
dtype: int64
```

Terdapat beberapa nilai yang sangat granular, seperti pada kolom total_day_minutes; total_day_charge; total_eve_minute; total_eve_charge; total_night_minutes; total_night_charge.

Cek apakah terdapat missing value dan duplicate

```
df_data_train.isnull().sum()
```

```
state                          0
account_length                 0
area_code                      0
international_plan             0
voice_mail_plan                0
number_vmail_messages          0
total_day_minutes              0
total_day_calls                0
total_day_charge               0
total_eve_minutes              0
total_eve_calls                0
total_eve_charge               0
total_night_minutes            0
total_night_calls              0
total_night_charge             0
total_intl_minutes             0
total_intl_calls               0
total_intl_charge              0
number_customer_service_calls  0
churn                          0
dtype: int64
```

```
[17] df_data_train.duplicated().sum()
```

```
0
```

Dataset tidak memiliki missing value ataupun duplicate, sehingga tidak perlu dilakukan perbaikan terhadap dataset.

Kategorisasikan Data

```
[18] cat_train = []
     num_train = []

     for col in df_data_train.columns:
       if df_data_train[col].dtype == "object":
         cat_train.append(col)
       else:
         num_train.append(col)

     print("Kolom Categorical:", cat_train)
     print("Kolom Numerical:", num_train)
```

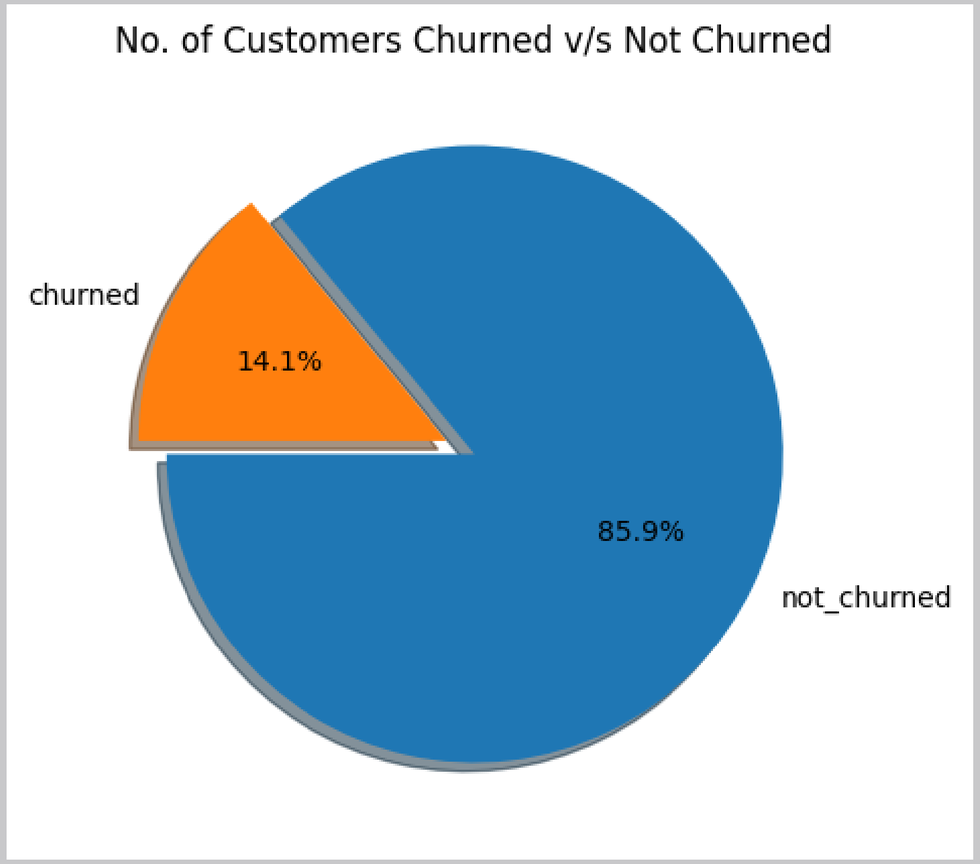
```
Kolom Categorical: ['state', 'area_code', 'international_plan', 'voice_mail_plan', 'churn']
Kolom Numerical: ['account_length', 'number_vmail_messages', 'total_day_minutes', 'total_day_calls', 'total_day_charge', 'total_eve_minutes', 'total_eve_calls', 'total_eve_charge', 'total_night_minutes', 'total_night_calls', 'total_night_charge', 'to
```

Terdapat data dengan kategorikal dan numerikal dalam dataset

Kolom Categorical : 'state', 'area_code', 'international_plan', 'voice_mail_plan', 'churn'

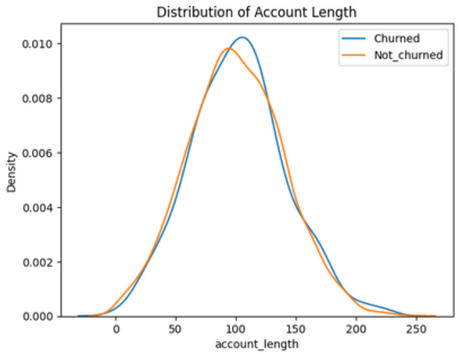
Kolom Numerical : 'account_length', 'number_vmail_messages', 'total_day_minutes', 'total_day_calls', 'total_day_charge','total_eve_minutes', 'total_eve_calls', 'total_eve_charge', 'total_night_minutes', 'total_night_calls','total_night_charge', 'total_intl_minutes', 'total_intl_calls', 'total_intl_charge', 'number_customer_service_calls'

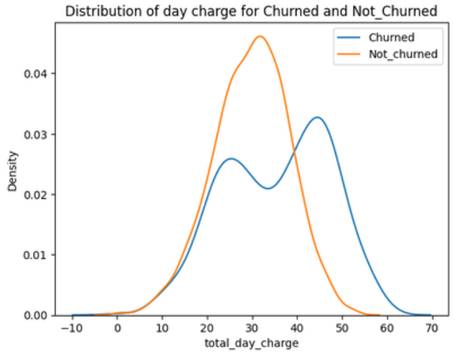
Cek kolom target

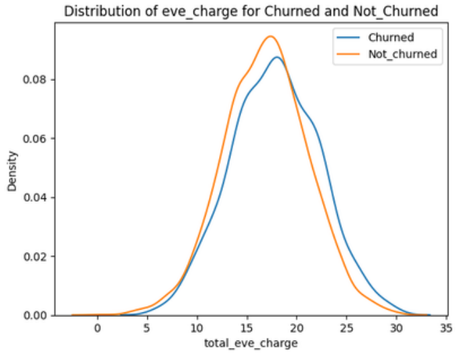


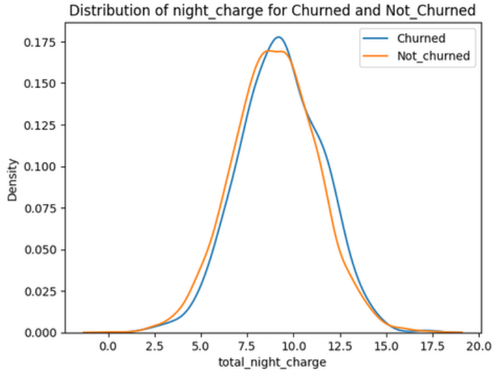
Kolom churn merupakan kolom target karena analisis yang dilakukan adalah untuk memprediksi customer churn. Data churn dan not churn pada kolom memiliki data imbalace yang cukup signifikan, sehingga perlu dilakukan balacing data.

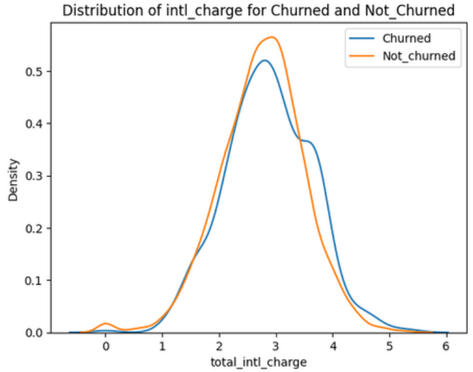
Distribusi dari beberapa features terhadap kolom churn

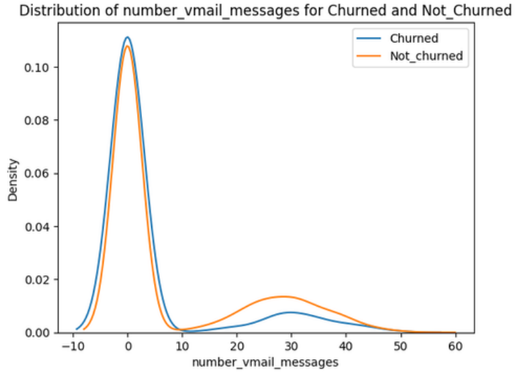






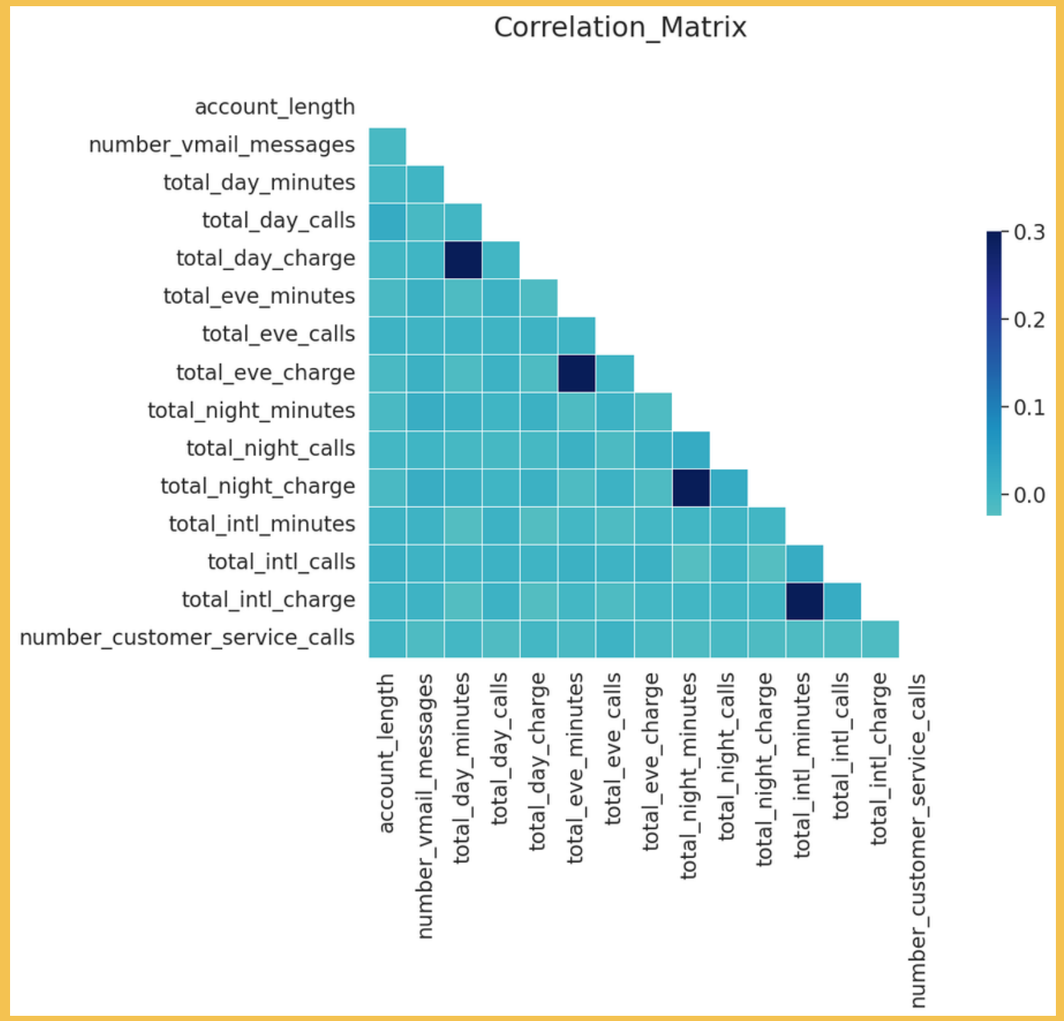






visualisasi grafik disamping dilakukan untuk mengethui perbedaan sebaran data churn dan not churn pada beberapa features. Pada features total-day_charge, Total_eve_charge, and number_vmail_messages memiliki perbedaan yang cukup signifikan.

Melihat korelasi antar *features*



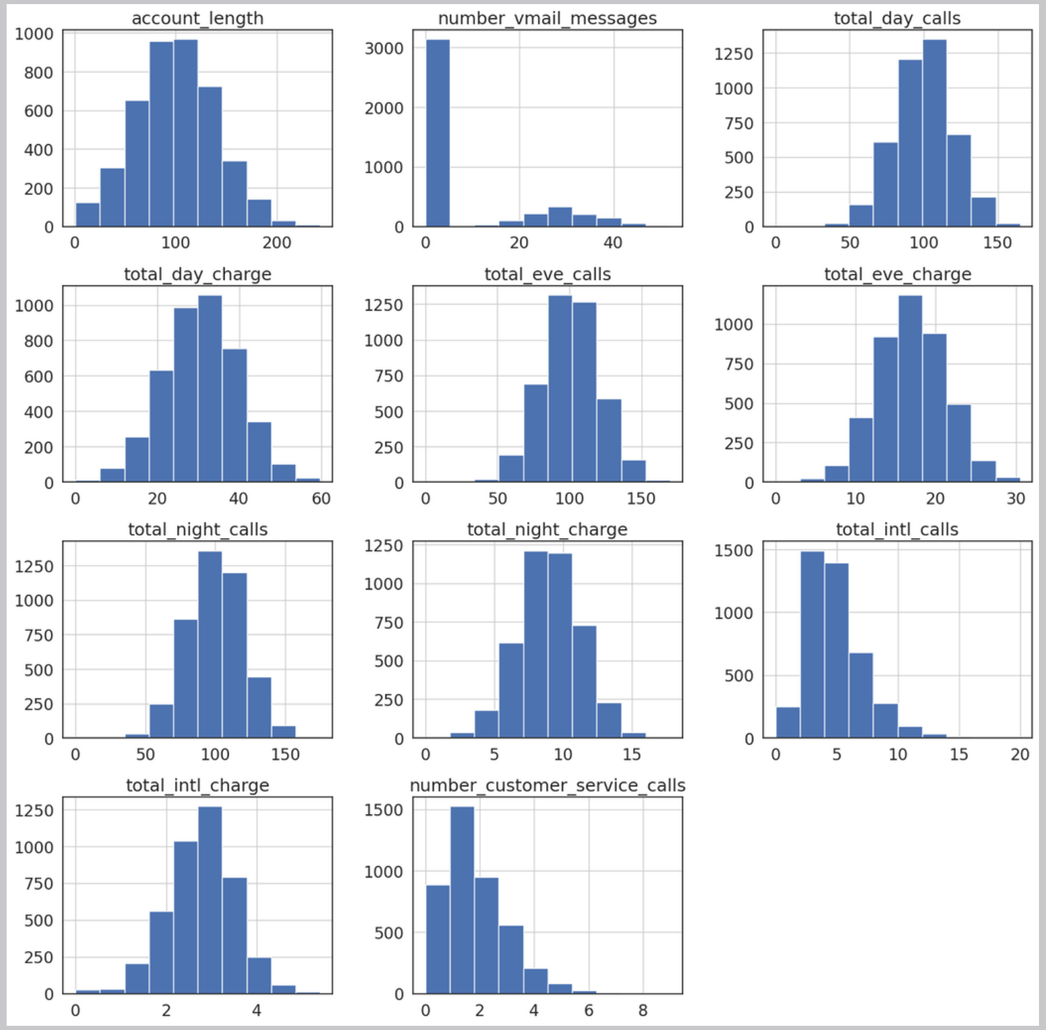
Semakin tinggi korelasi antar features, maka warnanya akan semakin gelap.

Korelasi features yang paling tinggi

- total_day_minutes, total_day_charge
- total_eve_minutes, total_eve_charge
- total_night_minutes, total_night_charge
- total_intl_minutes, total_intl_chargec

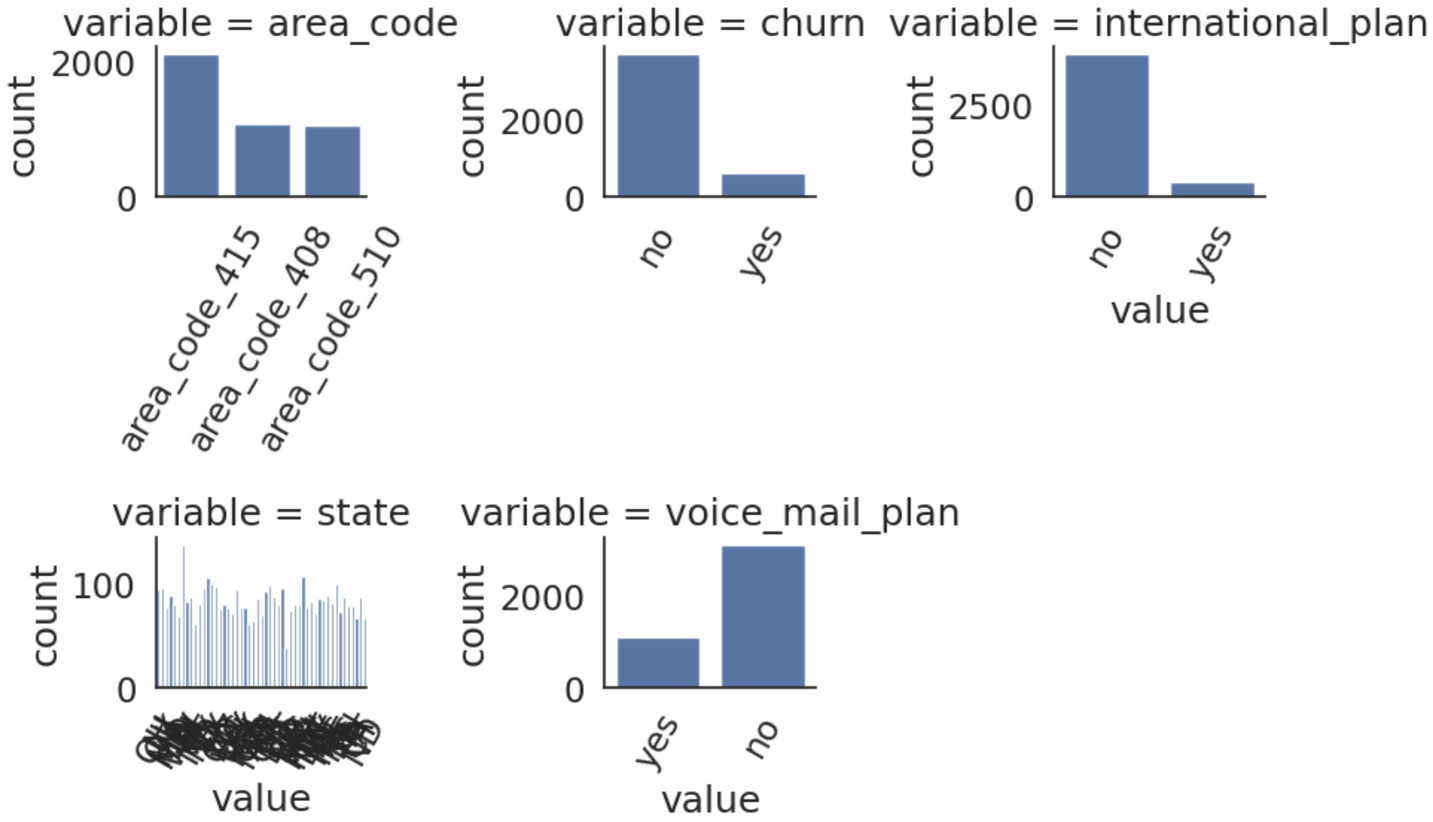
Pasangan feature yang memiliki korelasi paling tinggi tersebut dapat dilakukan drop pada salah satunya, karena tidak akan memberikan informasi baru.

Melihat disribusi features data numerik setelah dilakukan drop



Visualisasi distribusi features akan menampilkan apakah pada features tersebut data memiliki distribusi normal atau tidak, dari chart tersebut terlihat bahwa features tidak memiliki distribusi normal, sehingga perlu dilakukan standarisasi terhadap masing-masing features

Melihat disribusi features data kategorikal



Distribusi pada feeatures kategorikal memiliki perbedaan distribusi yang cukup signifikan pada setiap features, sehingga perlu dilakukan balancing data.

#2 DATA PREPROCESSING

Mempersiapkan data sebelum dilakukan modeling menggunakan machine learning

Standarization

```
[313] from sklearn.preprocessing import StandardScaler
      scaler=StandardScaler()
      data_train[num_train]=scaler.fit_transform(data_train[num_train].values)
      data_train.describe()
```

	account_length	number_vmail_messages	total_day_calls	total_day_charge	total_eve_calls	total_eve_charge	total_night_calls	total_night_charge	total_intl_calls	total_intl_charge	number_customer_service_calls
count	4.250000e+03	4250.000000	4.250000e+03	4.250000e+03	4.250000e+03	4.250000e+03	4.250000e+03	4.250000e+03	4.250000e+03	4.250000e+03	4.250000e+03
mean	-2.591391e-17	0.000000	1.864130e-16	2.169245e-16	2.081472e-16	-7.615346e-16	2.374049e-16	-4.505677e-16	-2.925764e-17	5.684342e-16	-4.932003e-17
std	1.000118e+00	1.000118	1.000118e+00	1.000118e+00	1.000118e+00	1.000118e+00	1.000118e+00	1.000118e+00	1.000118e+00	1.000118e+00	1.000118e+00
min	-2.500048e+00	-0.567911	-5.033498e+00	-3.337831e+00	-5.032413e+00	-3.984118e+00	-4.969402e+00	-3.982906e+00	-1.797300e+00	-3.717075e+00	-1.188960e+00
25%	-6.861596e-01	-0.567911	-6.502913e-01	-6.839856e-01	-6.619263e-01	-6.819738e-01	-6.888472e-01	-6.626744e-01	-5.791639e-01	-6.303097e-01	-4.263461e-01
50%	-5.951451e-03	-0.567911	4.670679e-03	3.846813e-03	-8.865085e-03	1.053414e-02	7.987245e-03	-1.717722e-03	-1.731186e-01	1.388492e-02	-4.263461e-01
75%	6.742567e-01	0.622715	6.596326e-01	6.649937e-01	6.944317e-01	6.679191e-01	6.550478e-01	6.779974e-01	6.389720e-01	6.312381e-01	3.362679e-01
max	3.596633e+00	3.301625	3.279480e+00	3.171252e+00	3.507619e+00	3.166918e+00	3.741029e+00	3.860300e+00	6.323606e+00	3.530114e+00	5.674566e+00

Standarisasi dilakukan agar diperoleh data yang terdistribusi normal, sehingga data akan lebih mudah diproses oleh machine learning. Data yang telah distandarisasi akan memiliki standart deviasi 1, hal tersebut akan memudahkan *features* untuk diproses oleh *machine learning*.

Merubah semua data kategorikal menjadi numerikal

Converting to Numerical Data

```
[316] from sklearn import preprocessing
      label_encoder = preprocessing.LabelEncoder()
      data_train['state'] = label_encoder.fit_transform(data_train['state'])
      data_train['international_plan'] = label_encoder.fit_transform(data_train['international_plan'])
      data_train['voice_mail_plan'] = label_encoder.fit_transform(data_train['voice_mail_plan'])
      data_train['area_code'] = label_encoder.fit_transform(data_train['area_code'])

      print (data_train.dtypes)
```

```
state                           int64
account_length                  float64
area_code                       int64
international_plan              int64
voice_mail_plan                 int64
number_vmail_messages           float64
total_day_calls                 float64
total_day_charge                float64
total_eve_calls                 float64
total_eve_charge                float64
total_night_calls               float64
total_night_charge              float64
total_intl_calls                float64
total_intl_charge               float64
number_customer_service_calls   float64
churn                           object
dtype: object
```

```
[317] data_train.replace(['yes','no'],[1,0],inplace=True)
```

Selanjutnya data kategorikal diubah menjadi data numerikal, karena *machine learning* hanya dapat memproses data dalam bentuk numerik. Sehingga *value* 'yes' dan 'no' pada *features* 'churn' juga diubah menjadi '1' dan' 0'. Tipe data pada dataset setelah dilakukan encode hanya terdiri dari integer dan float.

Melakukan split terhadap data train dan data set

Train Test Split

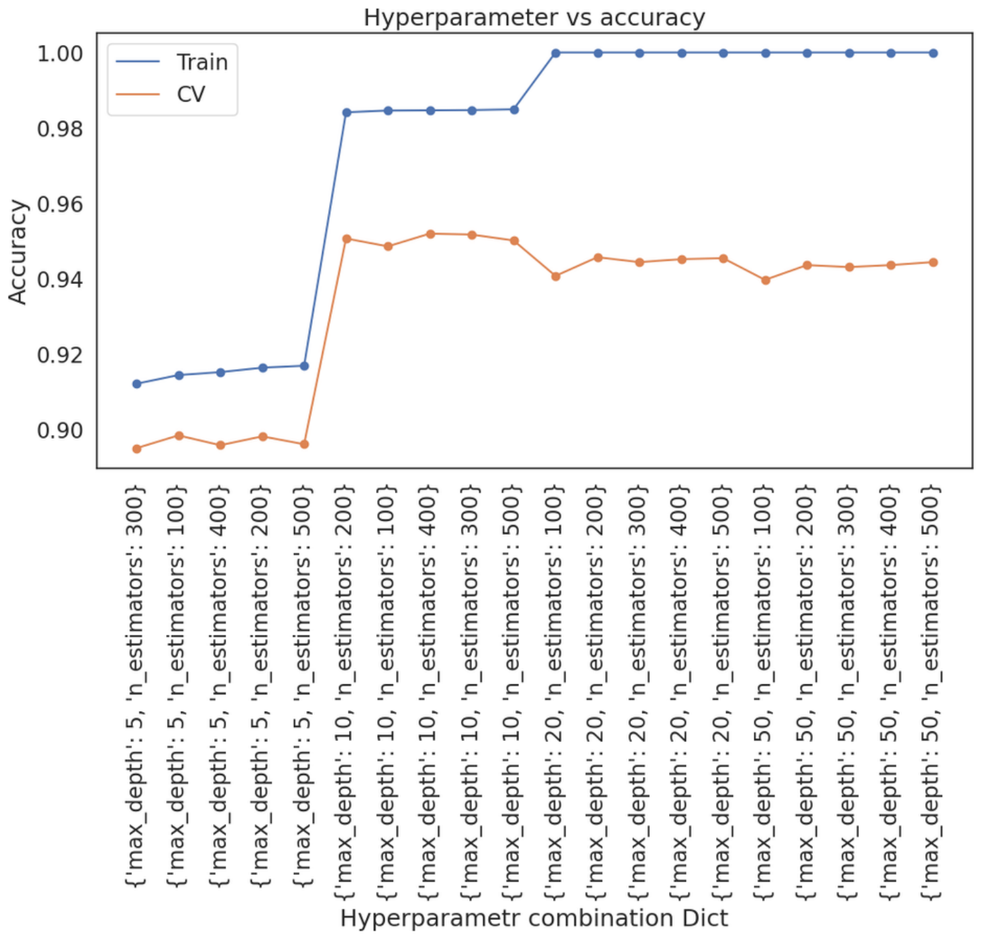
```
from sklearn.model_selection import train_test_split

Y=data_train['churn']
X=data_train.drop(['churn'],axis=1)

x_train,x_test,y_train,y_test = train_test_split(X,Y,test_size=0.10,stratify=Y,random_state=11)
print('Shape of x_train and y_train: ',x_train.shape, y_train.shape)
print('Shape of x_test and y_test: ',x_test.shape, y_test.shape)
```

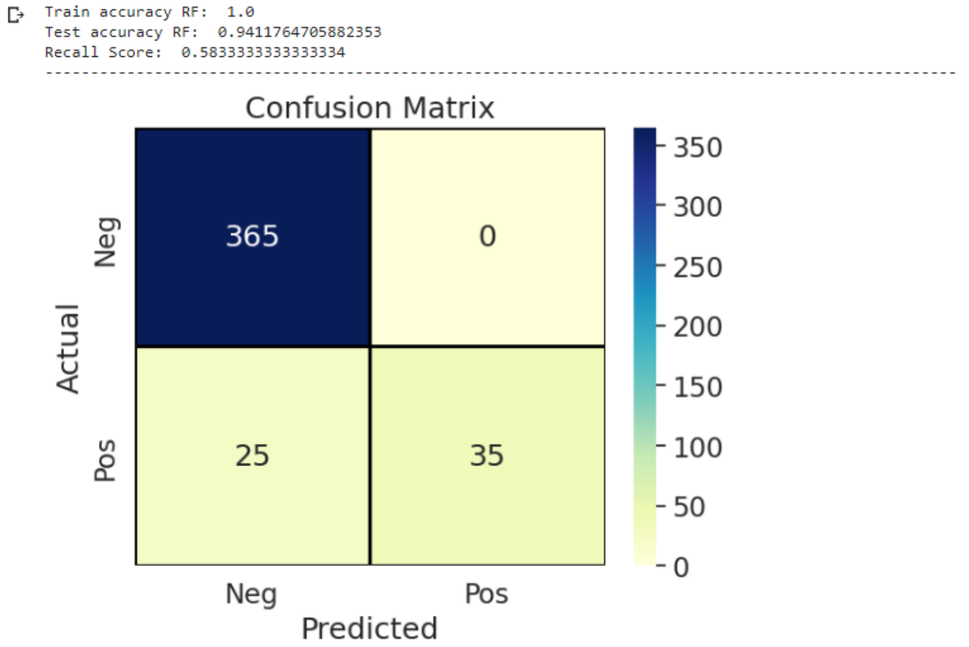
```
Shape of x_train and y_train:  (3825, 15) (3825,)
Shape of x_test and y_test:  (425, 15) (425,)
```

train_test_split sklearn dilakuakan untuk membagi data menjadi set pelatihan dan pengujian, sehingga kinerja model akan diketahui.

#3 MODELING

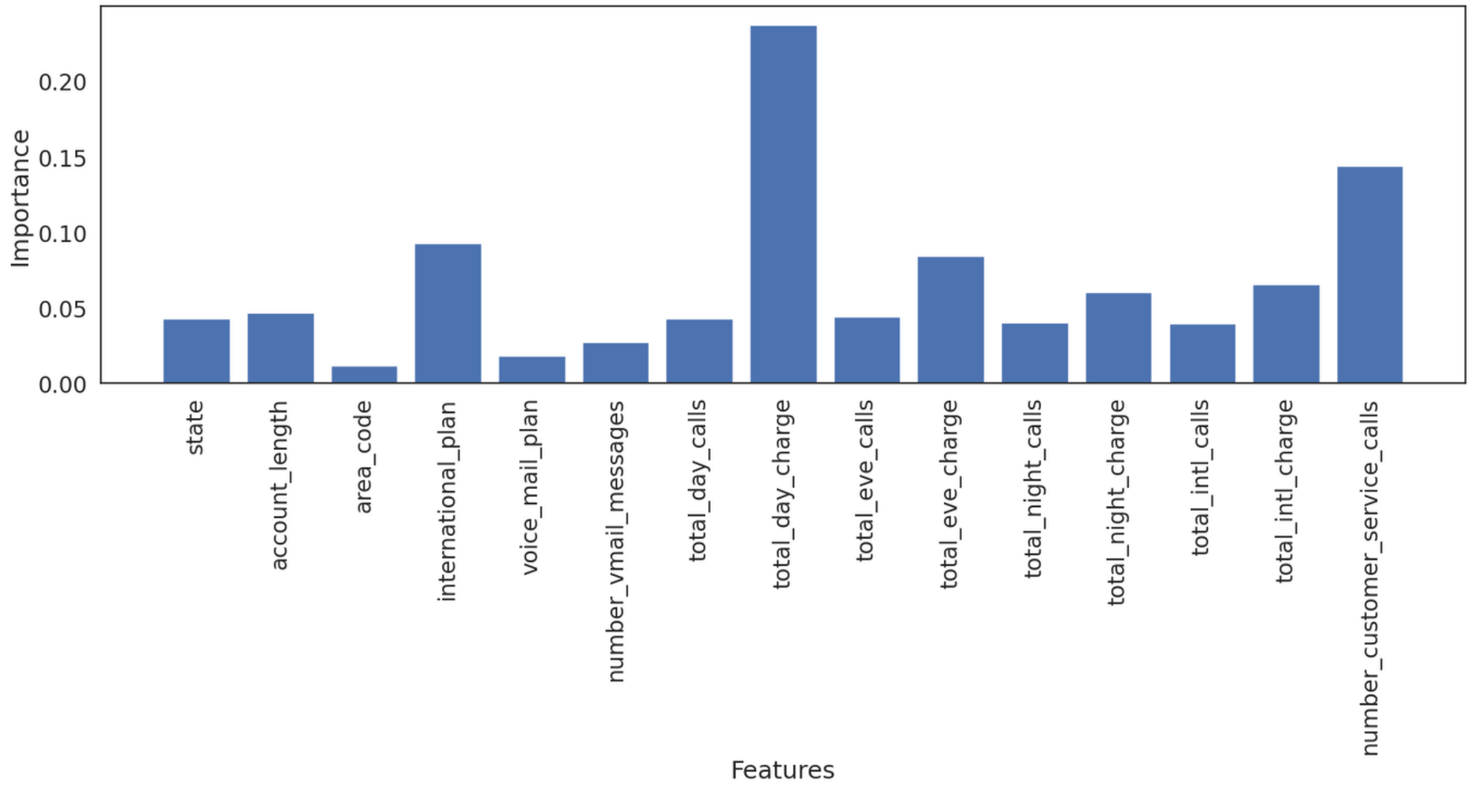
Random Forest Classifier





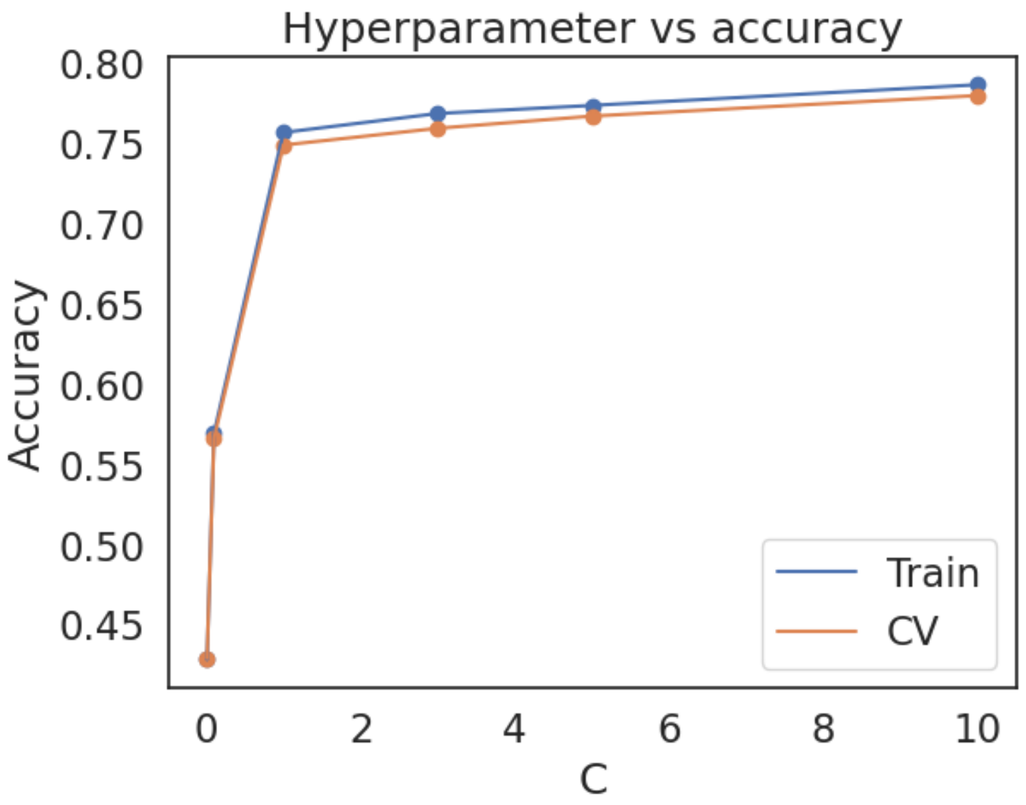
Dari confusion matrix random forest classifier terlihat bahwa terdapat 365 data yang bukan churn dan memang data tersebut bukan churn, 25 data yang sbenarnya churn tapi terprediksi sebagai bukan churn, serta 35 data yang merupakan churn dan terprediksi benar churn.

Cek features important

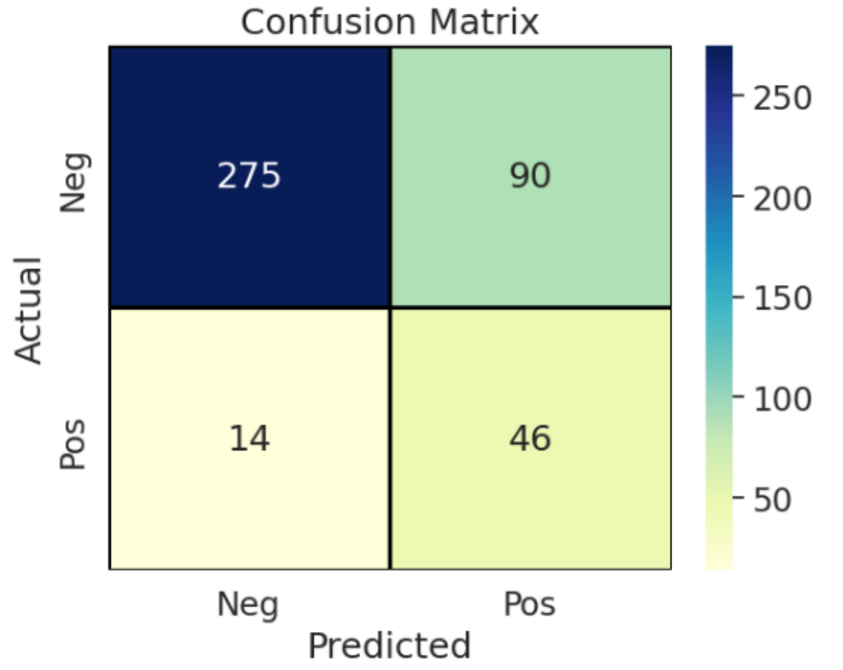


Melalui permodelan menggunakan random forest, dapat dilakukan analisis feature yang paling penting dalam menentukan 'churn' dan 'not churn'. Faktor yang paling berpengaruh terhadap prediksi churn adalah total_day_charge dan number_customer_service_calls

Support Vector Machine Classifier

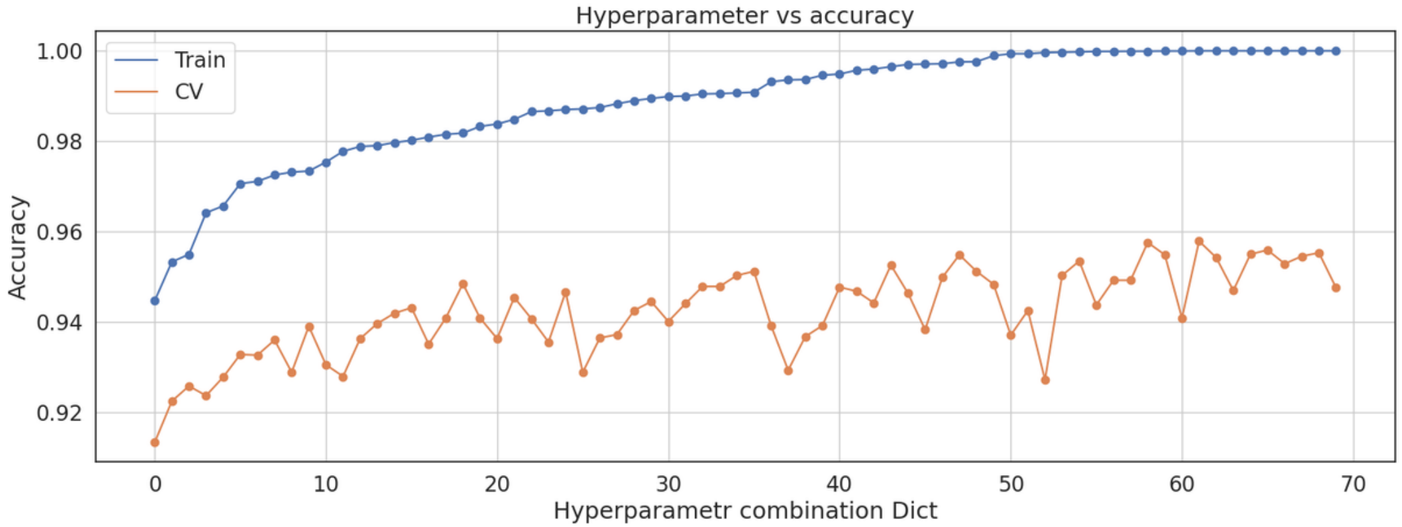



```
Train accuracy SVM:  0.7699346405228759
Test accuracy SVM:  0.7552941176470588
Recall Score:  0.7666666666666667
```

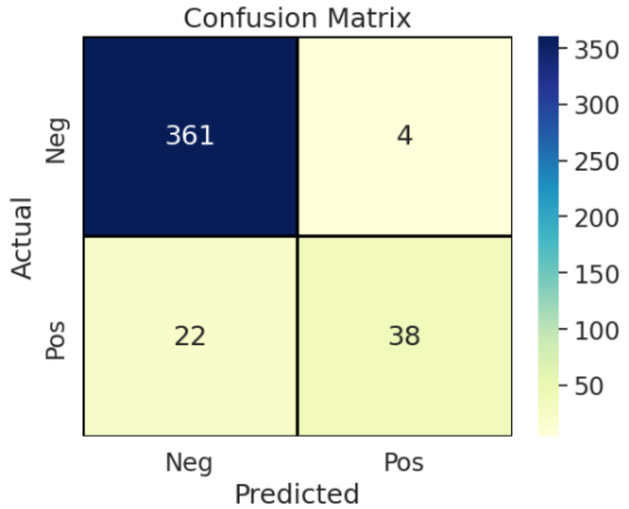


Dari confusion matrix SVM classifier terlihat bahwa terdapat 275 data yang bukan churn dan memang data tersebut bukan churn, 14 data yang sbenarnya churn tapi terprediksi sebagai bukan churn, 46 data yang merupakan churn dan terprediksi benar churn, serta 90 data bukan churn yang terprediksi sebagai churn.

XG Boost Classifier




```
Train accuracy XGB:  0.9987830848798296
Test accuracy XGB:  0.9388235294117647
Recall Score:  0.6333333333333333
```



Dari confusion matrix XG Boost classifier terlihat bahwa terdapat 361 data yang bukan churn dan memang data tersebut bukan churn, 22 data yang sbenarnya churn tapi terprediksi sebagai bukan churn, 38 data yang merupakan churn dan terprediksi benar churn, serta 4 data bukan churn yang terprediksi sebagai churn.

Sumarry

No.	Model	class imbalance	Train_accuracy	Test_Accuracy	Test_Recall_score
1.	RF	Balanced using class weights	1.0	0.941	0.583
2.	SVM	Balanced using class weights	0.77	0.755	0.767
3.	XGBoost	Balanced using SMOTE	0.999	0.939	0.633

- Grafik antara data train dan data cross validation menunjukkan nilai paling optimal untuk hyperparameter. Ketika grafik antara train dan CV terdapat gap yang tinggi dan akurasi tes tidak meningkat secara signifikan, menunjukkan bahwa model mulai overfitting.
- Nilai recall paling tinggi terdapat pada penggunaan model SVM dengan nilai recall 76%, namun memiliki nilai accuracy yang rendah.
- Permodelan menggunakan XGBoost dengan balancing data menggunakan SMOTE menghasilkan nilai recall yang cukup baik sebesar 63%, dengan nilai test accuracy 93% dan nilai train accuracy 99%.

I'ts the End of the Page, Thank you!