



**BPJS Kesehatan**

Badan Penyelenggara Jaminan Sosial



# Klasifikasi Inefisiensi Klaim Peserta BPJS Kesehatan Menggunakan XGBoost

---

The JamMaths

# Anggota The JamMaths



Cornelius Justin S. Hadi  
Universitas Indonesia



M. Hanif Pramudya Z.  
Universitas Indonesia



Tulus Setiawan  
Universitas Indonesia

# Table of Contents

01

Data  
Preparation

02

Data  
Cleaning

03

Feature  
Selection

04

Feature  
Engineering

05

Modelling



# 01

## Data Preparation

# Dataset yang Digunakan

sampling\_healthkathon2022.csv

	id	id_peserta	dati2	typefaskes	usia	jenkel	pisat	tgldatang	tglpulang	jenispej	...
0	165666	486	17	KL	48	P	1.0	2018-07-25T17:00:00.000Z	2018-07-25T17:00:00.000Z	2	...
1	1010828	520	17	A	63	L	1.0	2019-05-27T17:00:00.000Z	2019-05-30T17:00:00.000Z	1	...
2	166042	523	17	KL	53	P	1.0	2019-07-16T17:00:00.000Z	2019-07-16T17:00:00.000Z	2	...
3	168937	549	17	KL	54	P	1.0	2019-10-17T17:00:00.000Z	2019-10-17T17:00:00.000Z	2	...
4	1005899	549	17	A	53	P	1.0	2018-04-18T17:00:00.000Z	2018-04-18T17:00:00.000Z	2	...

merupakan data kunjungan peserta JKN ke fasilitas kesehatan rujukan tingkat lanjut

Jumlah baris :  
**11401882**

Jumlah Kolom :  
**22**

sampling\_healthkathon2022\_diagnosa.csv

	id	diag	levelid
0	6	O06.9	1
1	57	J02.9	1

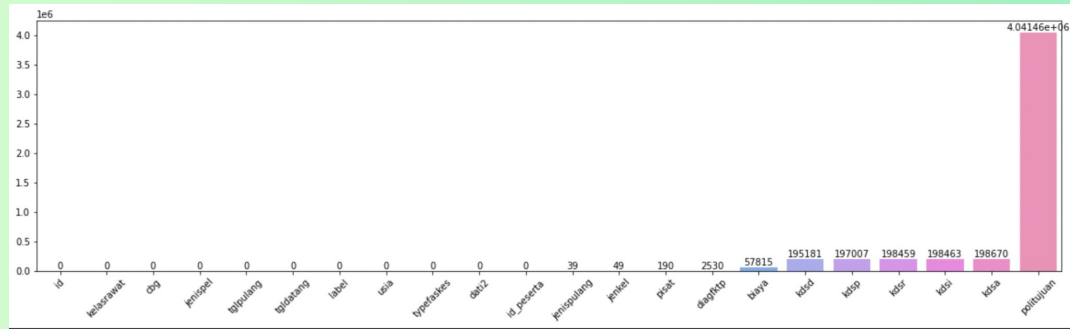
merupakan data yang berisi diagnosa penyakit peserta, di mana dalam satu kunjungan peserta bisa memiliki lebih dari satu diagnosa.

Jumlah baris :  
**11401882**

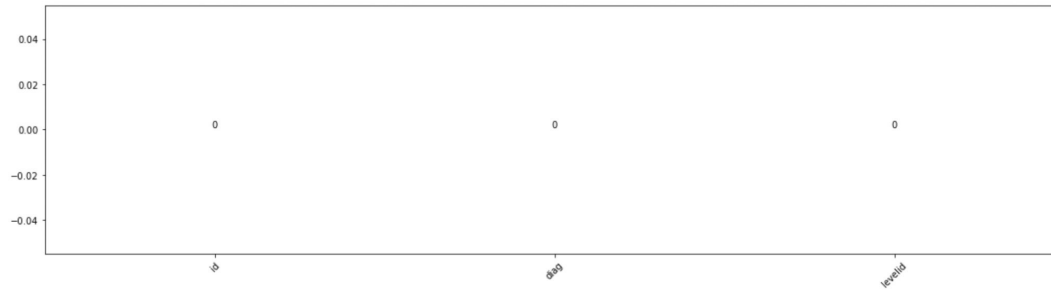
Jumlah Kolom :  
**3**



## Missing Value pada sampling\_healtkathon2022.csv



## Missing Value pada sampling\_healthkathon2022\_diagnosa.csv



## Data Duplikat pada sampling\_healthkathon2022.csv

```
df.duplicated().sum()
```

14

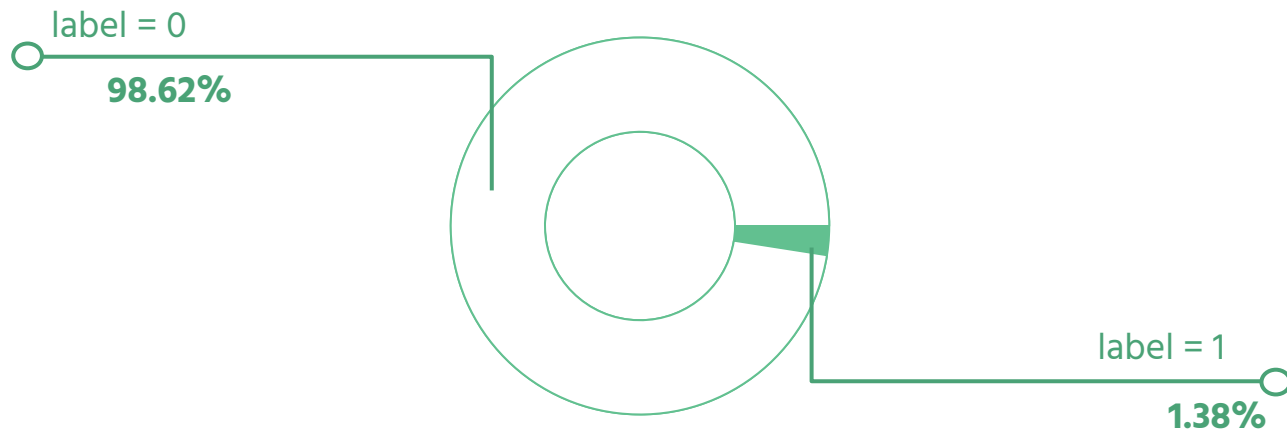
## Data Duplikat pada sampling\_healthkathon2022\_diagnosa.csv

```
df_diagnosa.duplicated().sum()
```

56

# Counts pada Kolom label

(Dataset `sampling_healthkathon2022.csv`)







# 02

## Data Cleaning



# Dataset sampling\_healthkathon2022.csv

1

Meng-handle Nilai *Null*

2

Meng-handle Data Duplikat

3

Menyamakan Format  
Penamaan Karakter

# Meng-handle Nilai Null

## Kolom Kategorik

'jenispulang', 'jenkel', 'pisat',  
'diagfktp', 'kdsa', 'kdsp', 'kdsr',  
'kdsi', 'kdsd'

### Handle :

ubah menjadi modus kolom kategorik tsb.

### Syntax :

```
SimpleImputer(strategy='most_frequent')
```



## Kolom Numerik

'biaya'

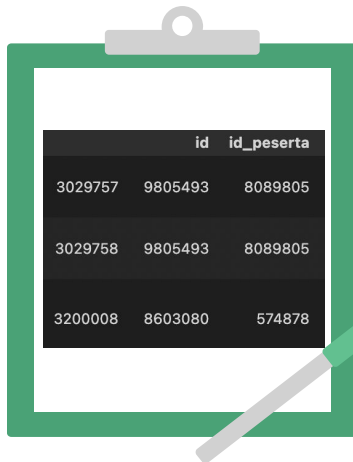
### Handle :

ubah menjadi median biaya tsb..

### Syntax :

```
SimpleImputer(strategy='median')
```

# Meng-handle Data Duplikat



	id	id_peserta
3029757	9805493	8089805
3029758	9805493	8089805
3200008	8603080	574878

Melakukan metode drop data duplikat

syntax:  
`df.drop_duplicates()`



	id	id_peserta
3029757	9805493	8089805

# Menyamakan Format Penamaan Karakter

## Kolom Kategorik

'typefaskes', 'jenkel', 'politujuan',  
'diagfktpt', 'cbg', 'kdsa', 'kdsp',  
'kdsr', 'kdsi', 'kdsd'

### Format Karakter :

Samakan huruf kapital

### Syntax :

```
df[obj_col].str.upper()
```



## Kolom Kategorik

'typefaskes', 'jenkel',  
'politujuan', 'diagfktpt', 'cbg',  
'kdsa', 'kdsp', 'kdsr', 'kdsi',  
'kdsd'

### Format Karakter :

Hilangkan Whitespaces

### Syntax :

```
df[obj_col].str.strip()
```

# Dataset

## sampling\_healthkathon2022\_diagnosa.csv

1

Meng-*handle* Data Duplikat

2

Menyamakan Format Karakter Kolom “diag”

3

Mengambil hanya diagnosa primer (baris dengan levelid = 1)

4

Diagnosa sekunder digunakan untuk membuat kolom berisi frekuensi diagnosa sekunder masing-masing id

## Hasil dari Proses Cleaning Dataset sampling\_healthkathon2022\_diagnosa.csv

	id	diag	diag_sekunder_counts
0	6	O06	0
1	57	J02	0
2	91	R10	0
3	109	R18	0
4	111	N81	1

Ket :

diag = diagnosa primer dari FKRTL



# 03

## Feature Selection





dati2

301

Kenapa fitur `dati2` di-*drop* atau dihilangkan dari dataset?

233

- + Data kategorik yang memiliki *unique values* terlalu banyak

118

- ➕ Tidak memiliki tingkatan lebih besar atau lebih kecil pada *unique values* nya

**+** Encoding dari Kabupaten/Kota di Indonesia

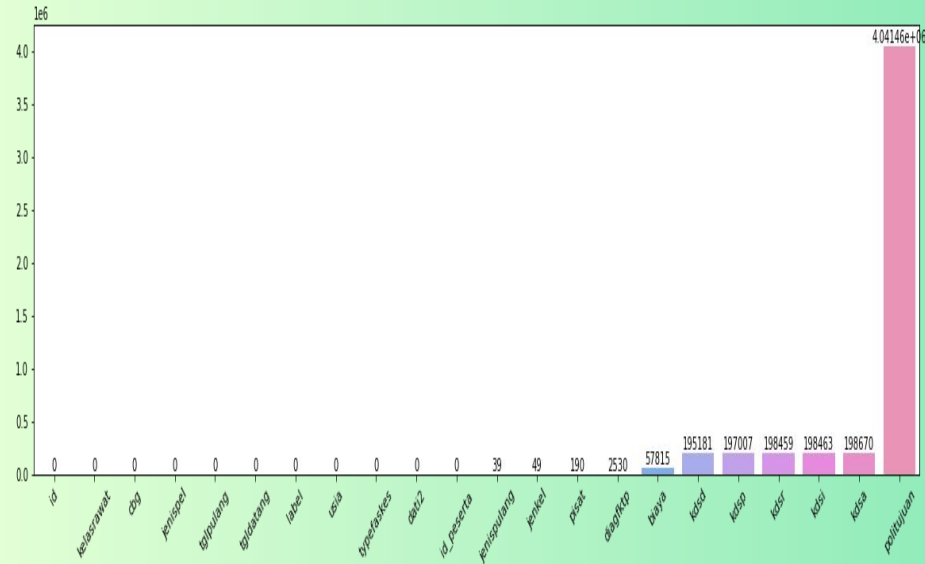
101

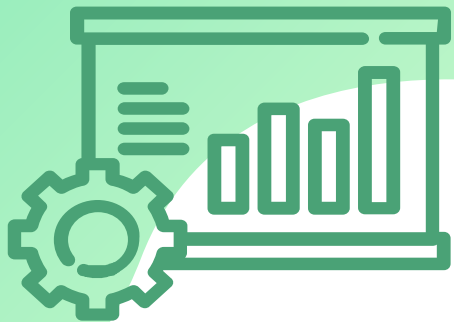
- Semakin banyak feature ketika dilakukan metode One Hot Encoding

12

- Membuat proses komputasi menjadi mahal karena semakin lama

## Kenapa fitur politujuan di-drop dari dataset?





# 04

## Feature Engineering



# Membuat Kolom Lama Perawatan Peserta

(jika nilai tiap baris  $> 0$ , maka peserta rawat inap)

```
7  tgl datang    11401882 non-null object
8  tgl pulang    11401882 non-null object
```

Ubah menjadi  
tipe datetime

```
15 tgl datang    datetime64[ns, UTC]
16 tgl pulang    datetime64[ns, UTC]
```

tgl pulang - tgl datang

lama\_rawat

0

3

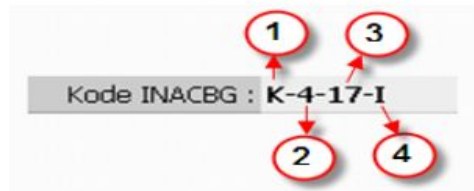
0

0

0

# Memisahkan Nilai-Nilai pada Kolom "cbg", "kdsa", "kdsp", "kdsr", "kdsi", "kdsd" (sesuai struktur kodenya) Menjadi Kolom Baru

Gambar 1  
Struktur Kode INA-CBG



Keterangan :

1. Digit ke-1 merupakan CMG (*Casemix Main Groups*)
2. Digit ke-2 merupakan tipe kasus
3. Digit ke-3 merupakan spesifik CBG kasus
4. Digit ke-4 berupa angka romawi merupakan *severity level*

Sumber: Permenkes RI Nomor 27 Tahun 2014 Tentang Petunjuk Teknis Sistem INA-CBGs

'Q-5-42-0'

Pisahkan tiap digit,  
lalu masukkan ke  
dalam kolom baru

cbg_CMG	cbg_tipekasus	cbg_spesifikkasus	cbg_severity
Q	5	42	0

# Membuat Dua Kolom Baru Berdasarkan 'diagfktp'

diagfktp
L02.8

Pisah menjadi 2 digit  
dengan mengacu pada  
website untuk  
mengecek diagnosa



diagfktp_letter	diagfktp_num
L	2

## ICD-10 Version:2016

Search

- ▼ XII Diseases of the skin and subcutaneous tissue
  - ▼ L00-L08 Infections of the skin and subcutaneous tissue
    - L00 Staphylococcal scalded skin syndrome
    - ▶ L01 Impetigo
    - ▶ L02 Cutaneous abscess, furuncle and carbuncle

# Merge Dataframe Utama dengan Dataframe Diagnosa

Dataframe utama (left)

diagfktp_letter	diagfktp_num
L	2
R	23
E	10
H	54
M	54



Dataframe diagnosa (right)

id	diagfkrtl_letter	diagfkrtl_num	diagfkrtl_sekunder_counts
6	O	6	0
57	J	2	0
91	R	10	0
109	R	18	0

Left-join

diagfktp_letter	diagfktp_num	diagfkrtl_letter	diagfkrtl_num	diagfkrtl_sekunder_counts
L	2	H	60.0	0.0
R	23	D	64.0	1.0
E	10	E	11.0	2.0
H	54	H	52.0	2.0
M	54	M	54.0	0.0

# Membuat Kolom Berisikan Kode ICD-10, Yang Berkaitan Dengan 'diagfktp' dan 'diagfkrtl' Masing-Masing Baris

diagfktp\_letter    diagfktp\_num

L

2

```
diagfktp_icd10 = []
for i in range(df_no_dup_prepared.shape[0]):
    letter = df_no_dup_prepared.loc[i, 'diagfktp_letter']
    num = df_no_dup_prepared.loc[i, 'diagfktp_num']
    if (letter in ['A', 'B']):
        diagfktp_icd10.append('I')
    elif (letter == 'C') or (letter == 'D' and num <= 48):
        diagfktp_icd10.append('II')
    elif letter == 'D' and num >= 50:
        diagfktp_icd10.append('III')
    elif letter == 'E':
        diagfktp_icd10.append('IV')
    elif letter == 'F':
        diagfktp_icd10.append('V')
    elif letter == 'G':
        diagfktp_icd10.append('VI')
    elif letter == 'H' and num <= 59:
        diagfktp_icd10.append('VII')
    elif letter == 'H' and num >= 60:
        diagfktp_icd10.append('VIII')
    elif letter == 'I':
        diagfktp_icd10.append('IX')
    elif letter == 'J':
        diagfktp_icd10.append('X')
    elif letter == 'K':
        diagfktp_icd10.append('XI')
    elif letter == 'L':
        diagfktp_icd10.append('XII')
```

diagfktp\_icd10

XII

diagfkrtl\_letter    diagfkrtl\_num

H

80.0

```
diagfkrtl_icd10 = []
for i in range(df_no_dup_prepared.shape[0]):
    letter = df_no_dup_prepared.loc[i, 'diagfkrtl_letter']
    num = df_no_dup_prepared.loc[i, 'diagfkrtl_num']
    if (letter in ['A', 'B']):
        diagfkrtl_icd10.append('I')
    elif (letter == 'C') or (letter == 'D' and num <= 48):
        diagfkrtl_icd10.append('II')
    elif letter == 'D' and num >= 50:
        diagfkrtl_icd10.append('III')
    elif letter == 'E':
        diagfkrtl_icd10.append('IV')
    elif letter == 'F':
        diagfkrtl_icd10.append('V')
    elif letter == 'G':
        diagfkrtl_icd10.append('VI')
    elif letter == 'H' and num <= 59:
        diagfkrtl_icd10.append('VII')
    elif letter == 'H' and num >= 60:
        diagfkrtl_icd10.append('VIII')
```

diagfkrtl\_icd10

VIII

# Preprocessing

Sebelum melakukan modelling, dilakukan **data preprocessing** dengan metode sebagai berikut

**Standard Scaling:** ['biaya', 'usia', 'lama\_rawat', 'diagfkrtl\_sekunder\_counts']

**One Hot Encoding:** ['cbg\_CMG', 'cbg\_tipekasus', 'cbg\_spesifikkasus',  
'kdsa\_CMG', 'kdsa\_tipekasus', 'kdsa\_spesifikkasus',  
'kdsp\_spesifikkasus', 'kdsr\_spesifikkasus',  
'kdsi\_spesifikkasus', 'kdsd\_spesifikkasus',  
'diagfktp\_icd10', 'diagfkrtl\_icd10', 'typefaskes']

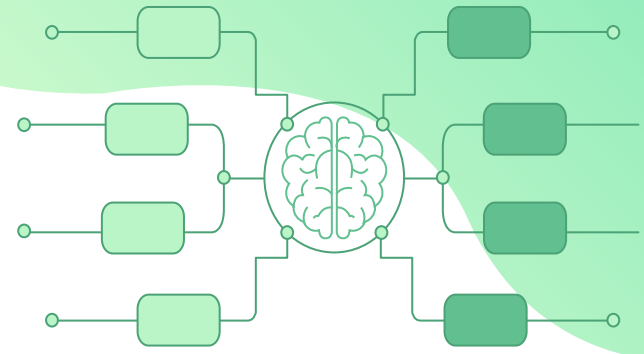
**Ordinal Encoding:** ['jenkel', 'cbg\_severity', 'kdsa\_severity',  
'kdsp\_severity', 'kdsr\_severity', 'kdsd\_severity']

Setelah preprocessing diperoleh sebanyak **776 kolom** (akibat dari One Hot Encoding)





- 
- 



# 05 Modelling

## XGBoost

eXtreme Gradient Boosting

### Imbalanced Dataset

XGBoost bekerja dengan baik pada dataset yang tidak seimbang karena pada setiap pembuatan tree, XGBoost memperbaiki error tree sebelumnya

### Fast

XGBoost dapat melakukan komputasi paralel ketika membuat tree, serta kita dapat memanfaatkan GPU

### Optimized

XGBoost menggunakan algoritma Gradient Boosting yang sudah dioptimalkan, seperti memiliki hyperparameter regularisasi untuk mencegah *overfitting*

# Optimasi Model: Hyperparameter Tuning

Hyperparameter tuning menggunakan algoritma **Bayesian Search Optimization** dengan cross-validation

Data yang diambil untuk hyperparameter tuning sebanyak 25% dari keseluruhan

```
X_train, X_test, y_train, y_test = train_test_split(X_prepared, y,
                                                    test_size=.75)
X_train.shape, y_train.shape
((2850467, 776), (2850467,))
```

Dibuat pipeline model dengan over-undersampling

```
model_pipeline = imbpipeline([
    ('over', SMOTE(sampling_strategy=.15)),
    ('under', RandomUnderSampler(sampling_strategy=.5)),
    ('xgb', xgb.XGBClassifier(n_estimators=50, objective='binary:logistic',
                             verbosity=2))
])
```

Hyperparameter yang dilakukan *tuning*:

1. Banyak data yang diover-under sampling
2. Learning rate
3. Max depth
4. Banyak subsample untuk training
5. Banyak sampel kolom untuk training
6. Nilai regularisasi L1 dan L2

```
params = {'over__sampling_strategy': Real(.1, .5, 'log-uniform'),
          'under__sampling_strategy': Real(.5, .8, 'log-uniform'),
          'xgb__learning_rate': Real(1e-2, 5e-1),
          'xgb__max_depth': Integer(6, 64),
          'xgb__subsample': Real(0.5, 1),
          'xgb__colsample_bytree': Real(0.5, 1),
          'xgb__reg_alpha': Real(1e-2, 10, 'log-uniform'),
          'xgb__reg_lambda': Real(1e-2, 10, 'log-uniform'),}
```

```
search = BayesSearchCV(model_pipeline, params, scoring='f1', cv=3, verbose=10,
                        n_jobs=-1, n_iter=20, return_train_score=True)
```

```
search.fit(X_train, y_train)
```

```
Fitting 3 folds for each of 1 candidates, totalling 3 fits
Fitting 3 folds for each of 1 candidates, totalling 3 fits
Fitting 3 folds for each of 1 candidates, totalling 3 fits
```

# Optimasi Model: Hyperparameter Tuning

## Hasil

Hyperparameter	Value
Over-sampling strategy	0.49
Under-sampling strategy	0.5
Learning rate	0.19
Max depth	50
Subsample	0.99
Column sample	0.5
L1 regularization	0.03
L2 regularization	10

# Train Model with Tuned Hyperparameter

Untuk training model, digunakan **data train sebanyak 40%** dari keseluruhan data.

Juga digunakan early stopping, jika hingga **100** tree selanjutnya skor **AUC-ROC** tidak meningkat, training akan berhenti.

```
X_train_full, X_test, y_train_full, y_test = train_test_split(X_prepared, y, test_size=.6)
X_train, X_val, y_train, y_val = train_test_split(X_train_full, y_train_full, test_size=.2)
```

```
X_train.shape, X_val.shape, X_test.shape
```

```
((3648597, 776), (912150, 776), (6841121, 776))
```

```
es = xgb.callback.EarlyStopping(rounds=100, metric_name='auc', data_name='validation_0', maximize=True,
                                save_best=True)
```

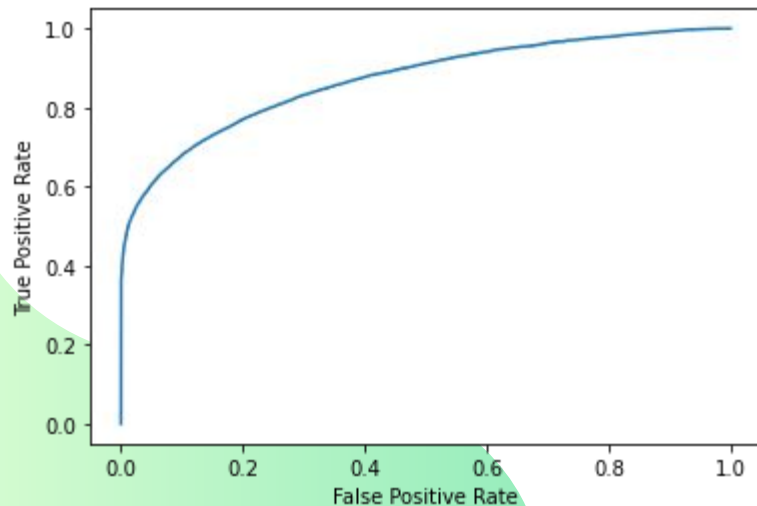
```
model_pipeline = imbpipeline([
    ('over', SMOTE(sampling_strategy=0.49999999999999994, n_jobs=-1, k_neighbors=20)),
    ('under', RandomUnderSampler(sampling_strategy=0.5)),
    ('xgb', xgb.XGBClassifier(n_estimators=1000, colsample_bytree=.5, learning_rate=0.1902755128019019,
                             max_depth=50, reg_alpha=0.039755577104576896, reg_lambda=10, subsample=0.9912446182088942,
                             callbacks=[es], objective='binary:logistic', eval_metric=['auc', 'logloss'],
                             verbosity=2, n_jobs=-1))
])
```

```
model_pipeline.fit(X_train, y_train, xgb__eval_set=[(X_val, y_val)])
```

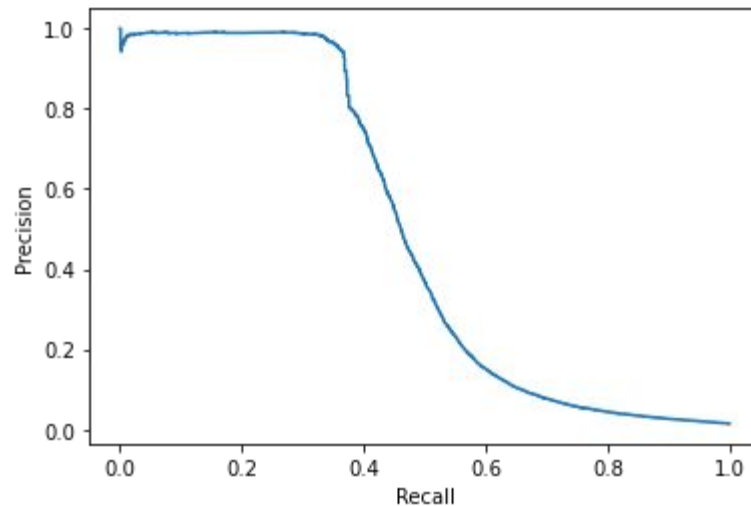
# Optimasi Model: Threshold Tuning

Dilakukan **prediksi probabilitas** pada **data validasi**, lalu probabilitas tersebut diplot menjadi ROC dan Precision-Recall Curve.

## ROC Curve



## Precision-Recall Curve



# Optimasi Model: Threshold Tuning

```
max_avg_pr = 0
max_th = 0
for i in range(len(th_pr)):
    avg_pr = (precision[i] + recall[i])/2
    if avg_pr > max_avg_pr or max_th == 0:
        max_avg_pr = avg_pr
        max_th = th_pr[i]

    if i % 100000 == 0:
        print('i =', i, 'DONE!')
```

```
i = 0 DONE!
i = 100000 DONE!
i = 200000 DONE!
i = 300000 DONE!
i = 400000 DONE!
i = 500000 DONE!
i = 600000 DONE!
```

max\_th

0.8642103

Proses mencari threshold yang **memperoleh rata-rata** precision dengan recall terbesar

Hasil **evaluasi** model dengan threshold max\_th pada data **validasi**:

	precision	recall	f1-score	support
0	0.99	1.00	1.00	899498
1	0.96	0.35	0.52	12652
accuracy			0.99	912150
macro avg	0.98	0.68	0.76	912150
weighted avg	0.99	0.99	0.99	912150

# Hasil Pengembangan Model

## Evaluasi model pada data **test**

	precision	recall	f1-score	support
0	0.99	1.00	1.00	6746952
1	0.96	0.36	0.52	94169
accuracy			0.99	6841121
macro avg	0.98	0.68	0.76	6841121
weighted avg	0.99	0.99	0.99	6841121



# Kemudahan Implementasi Model

## Scikit-learn API

Terdapat syntax XGBoost yang sama seperti menggunakan scikit-learn. Sehingga mempermudah seseorang yang terbiasa dengan *interface* scikit-learn

Model XGBoost yang sudah di-*train* sebelumnya, dapat di-*train* kembali menggunakan data baru

## Incremental Learning

## Automatically Handle Null

XGBoost dapat secara otomatis *handle* data null, baik untuk training maupun prediksi

XGBoost dapat meng-*output* langsung kelas dari data yang diprediksi

## Output 0 or 1



Thank  
You!