

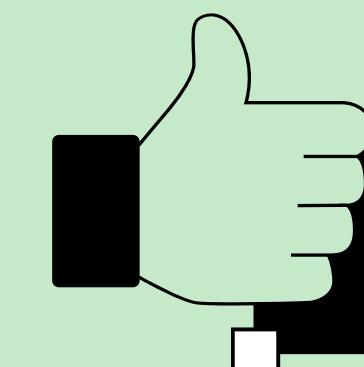
Tugas Akhir Ilmu Data

Desember 20. 2022

IMPLEMENTASI METODE TERBAIK DALAM PREDIKSI KLAIM ASURANSI MOBIL

DENGAN DECISION TREE (DTREE), ADAPTIVE BOOSTING
(ADABOOST), DAN EXTREME GRADIENT BOOSTING (XGBOOST)

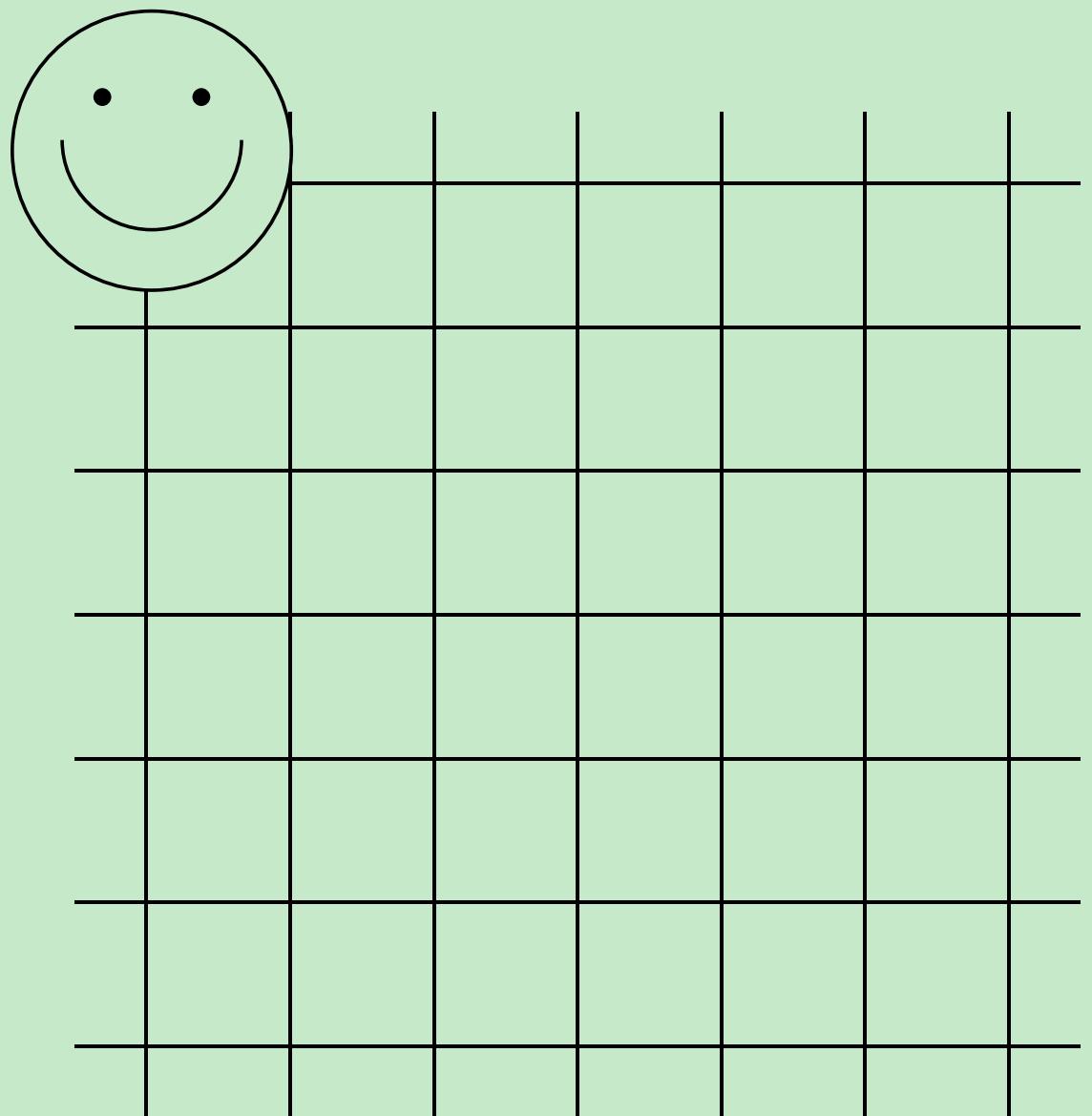
 Muhammad Iqbal AL-Fatih 2006571311
Emmanuel Justin Heumasse 2006571500
Gabriel Rico Fortino 2006571236
Muhammad Adli Rahmat Solihin 2006529184



Kelompok 12

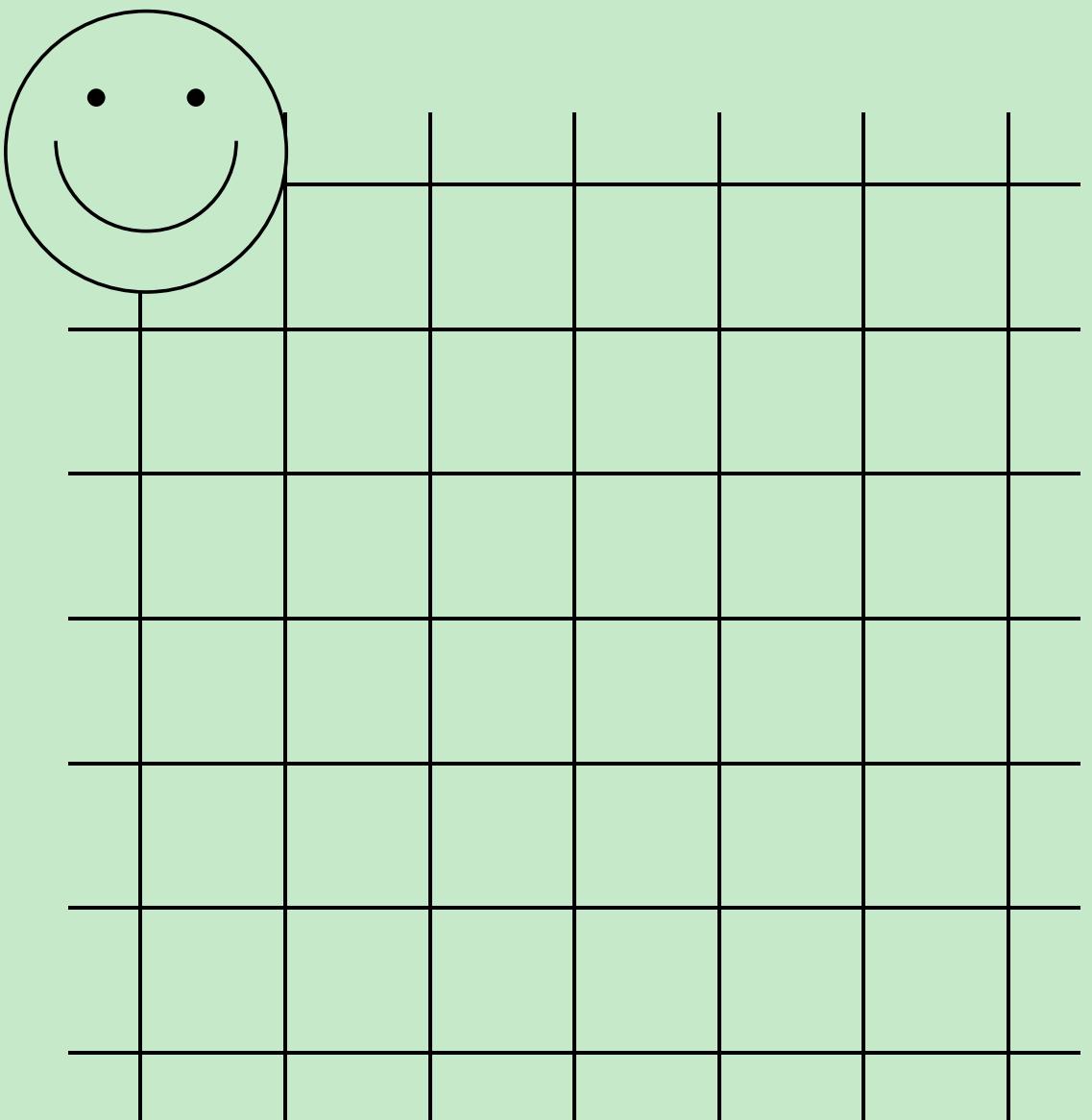
LATAR BELAKANG

Perusahaan asuransi tentunya menginginkan keuntungan dan menghindari kerugian bagi perusahaannya. Tren **peningkatan** frekuensi klaim untuk asuransi kendaraan mengakibatkan perlunya metode pengajuan klaim secara cepat dengan tetap menjaga akurasi. Prediksi yang akurat memberikan peluang untuk mengurangi kerugian finansial bagi perusahaan. **Akan dicari model dengan akurasi terbaik.**



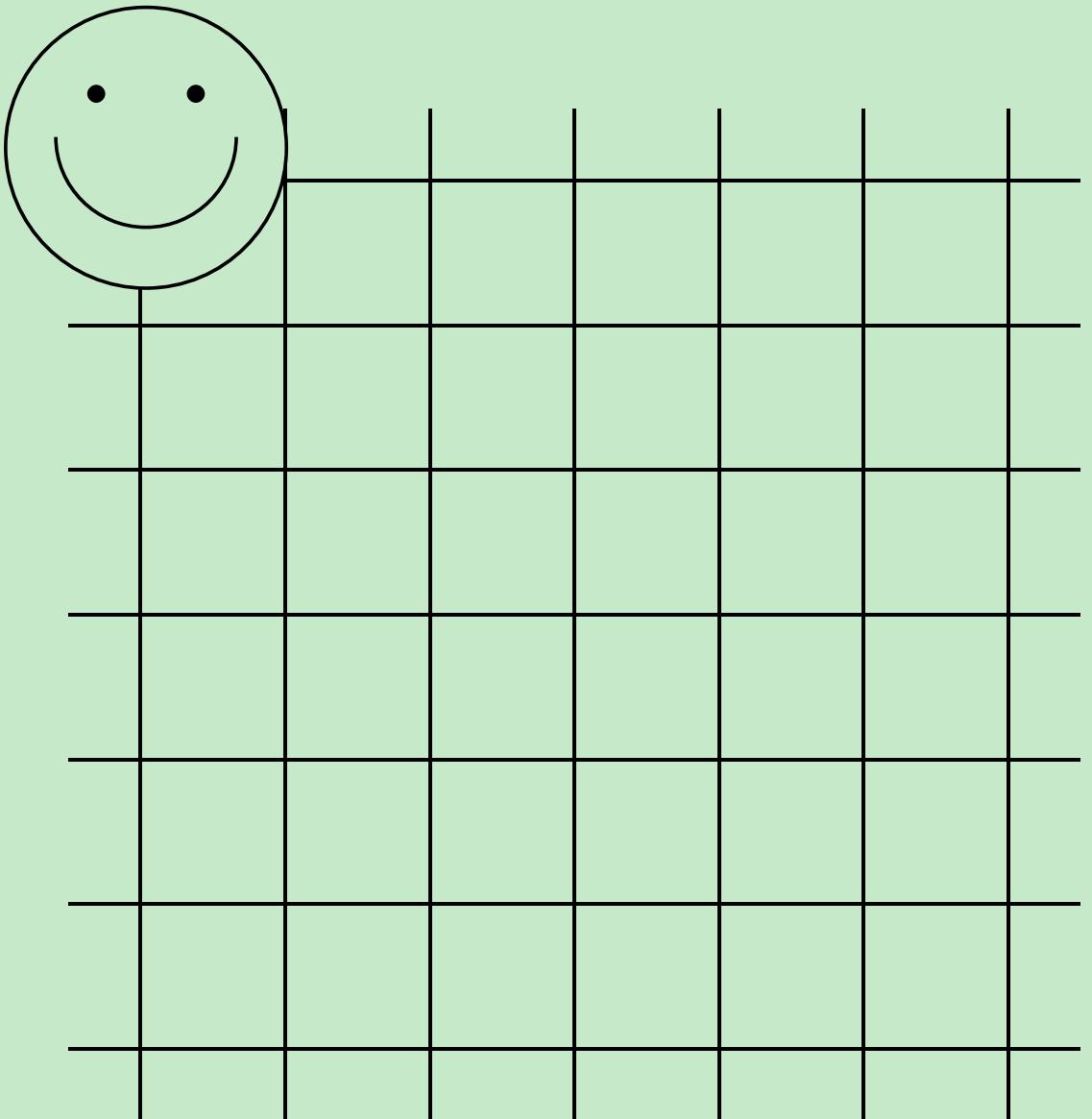
RUMUSAN MASALAH

- Bagaimana cara mengimplementasikan metode Decision Tree, AdaBoost, dan XGBoost dalam memprediksi klaim asuransi mobil?
- Di antara Decision Tree, AdaBoost, dan XGBoost, metode manakah yang lebih akurat dalam memprediksi klaim asuransi mobil?

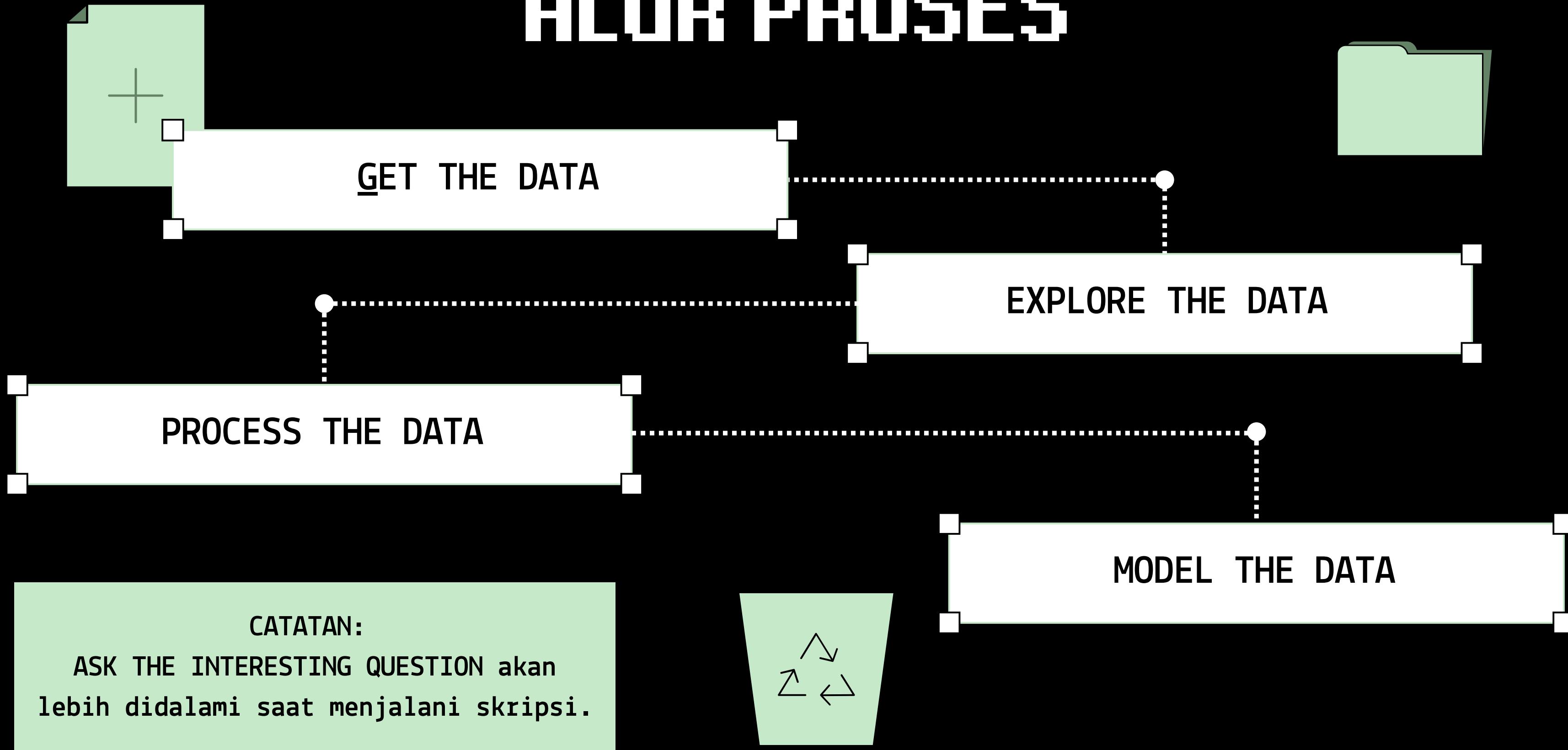


TUJUAN PENELITIAN

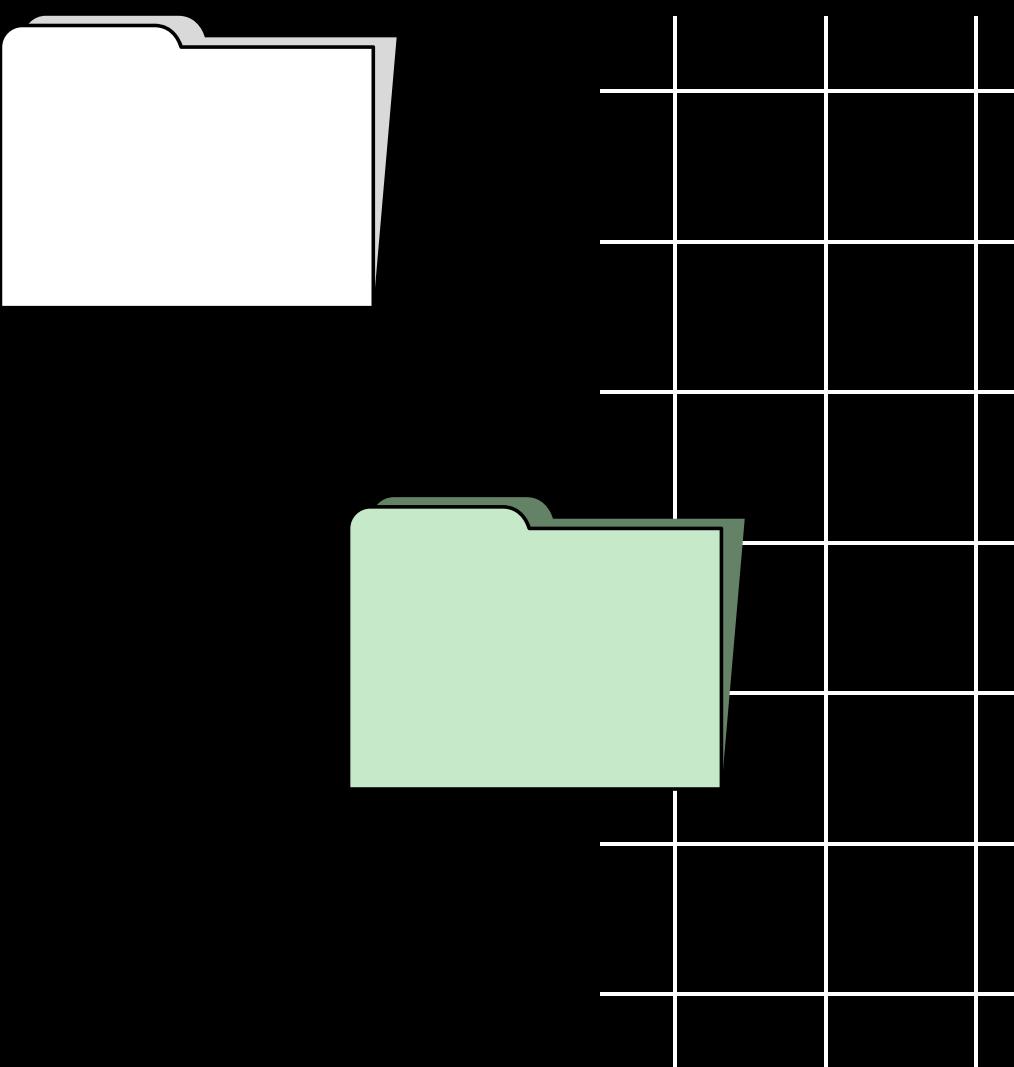
- Mengimplementasi metode Decision Tree, AdaBoost, dan XGBoost dalam memprediksi klaim asuransi mobil.
- Menganalisis dan membandingkan implementasi metode Decision Tree, AdaBoost, dan XGBoost dengan melihat akurasi dalam memprediksi klaim asuransi mobil.



ALUR PROSES



GET THE DATA



Data diambil dari Kaggle, yaitu Car Insurance Claim Prediction.

<https://www.kaggle.com/datasets/ifteshanajin/carinsuranceclaimprediction-classification?resource=download>

Fitur 'policy_id' dijadikan indeks data

Highlight two or more cells, **right-click** then
press "**Merge Cells**" to organize your table
according to your needs!

ABOUT DATASET

1. Dataset berisi informasi tentang pemegang polis yang memiliki 43 atribut sebagai variabel prediktor dan 1 variabel target yang menunjukkan apakah pemegang polis mengajukan klaim dalam 6 bulan ke depan atau tidak.
2. Nilai Nm dan Rpm pada fitur **Max_Torque** dan Bhp dan Rpm pada fitur **Max_Power** akan dipisahkan agar dapat terdefinisi.

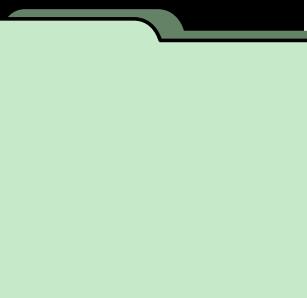
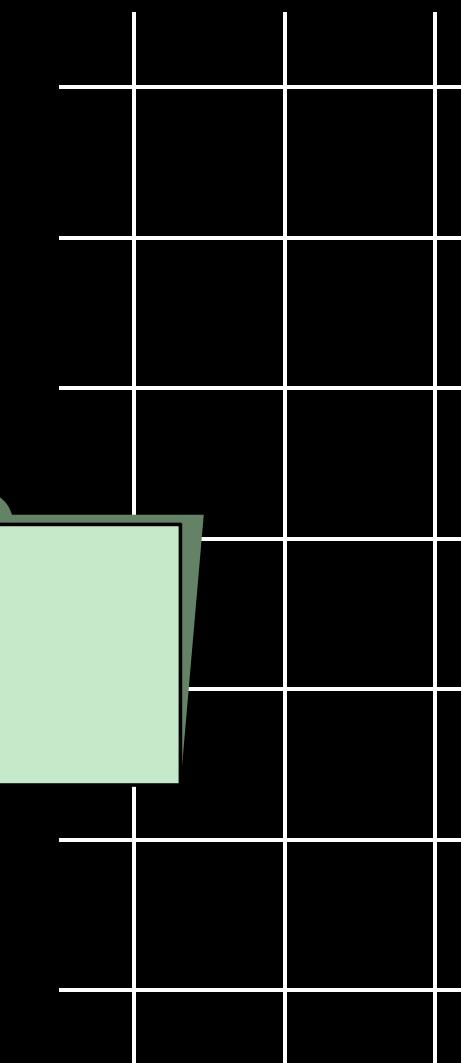
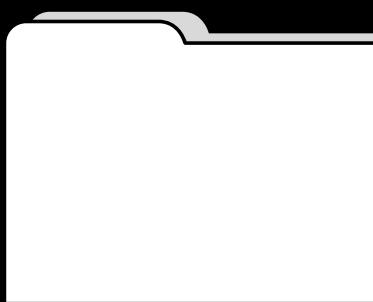
Jenis Data	
Numerik (20 Data)	Kategorik (25 Data)
<code>'policy_tenure', 'age_of_car', 'age_of_policyholder', 'population_density', 'make', 'airbags', 'displacement', 'cylinder', 'gear_box', 'turning_radius', 'length', 'width', 'height', 'gross_weight', 'ncap_rating', 'is_claim', 'max_torque_Nm', 'max_torque_rpm', 'max_power_bhp', 'max_power_rpm'</code>	<code>'area_cluster', 'segment', 'model', 'fuel_type', 'engine_type', 'is_esc', 'is_adjustable_steering', 'is_tpms', 'is_parking_sensors', 'is_parking_camera', 'rear_brakes_type', 'transmission_type', 'steering_type', 'is_front_fog_lights', 'is_rear_window_wiper', 'is_rear_window_washer', 'is_rear_window_defogger', 'is_brake_assist', 'is_power_door_locks', 'is_central_locking', 'is_power_steering', 'is_driver_seat_height_adjustable', 'is_day_night_rear_view_mirror', 'is_ecw', 'is_speed_alert'</code>

Variable	Description
policy_id	Unique identifier of the policyholder
policy_tenure	Time period of the policy
age_of_car	Normalized age of the car in years
age_of_policyholder	Normalized age of policyholder in years
area_cluster	Area cluster of the policyholder
population density	Population density of the city (Policyholder City)
make	Encoded Manufacturer/company of the car
segment	Segment of the car (A/ B1/ B2/ C1/ C2)
model	Encoded name of the car
fuel_type	Type of fuel used by the car
max_torque	Maximum Torque generated by the car (Nm@rpm)
max_power	Maximum Power generated by the car (bhp@rpm)
max_power	Maximum Power generated by the car (bhp@rpm)
engine_type	Type of engine used in the car
airbags	Number of airbags installed in the car

Variable	Description
is_esc	Boolean flag indicating whether Electronic Stability Control (ESC) is present in the car or not.
is_adjustable_steering	Boolean flag indicating whether the steering wheel of the car is adjustable or not.
is_tpms	Boolean flag indicating whether Tyre Pressure Monitoring System (TPMS) is present in the car or not.
is_parking_sensors	Boolean flag indicating whether parking sensors are present in the car or not.
is_parking_camera	Boolean flag indicating whether the parking camera is present in the car or not.
rear_brakes_type	Type of brakes used in the rear of the car
displacement	Engine displacement of the car (cc)
cylinder	Number of cylinders present in the engine of the car
transmission_type	Transmission type of the car
gear_box	Number of gears in the car
steering_type	Type of the power steering present in the car
turning_radius	The space a vehicle needs to make a certain turn (Meters)
length	Length of the car (Millimetre)
width	Width of the car (Millimetre)
height	Height of the car (Millimetre)

Variable	Description
gross_weight	The maximum allowable weight of the fully-loaded car, including passengers, cargo and equipment (Kg)
is_front_fog_lights	Boolean flag indicating whether front fog lights are available in the car or not.
is_rear_window_wiper	Boolean flag indicating whether the rear window wiper is available in the car or not.
is_rear_window_washer	Boolean flag indicating whether the rear window washer is available in the car or not.
is_rear_window_defogger	Boolean flag indicating whether rear window defogger is available in the car or not.
is_brake_assist	Boolean flag indicating whether the brake assistance feature is available in the car or not.
is_power_door_lock	Boolean flag indicating whether a power door lock is available in the car or not.
is_central_locking	Boolean flag indicating whether the central locking feature is available in the car or not.
is_power_steering	Boolean flag indicating whether power steering is available in the car or not.
is_driver_seat_height_adjustable	Boolean flag indicating whether the height of the driver seat is adjustable or not.
is_day_night_rear_view_mirror	Boolean flag indicating whether day & night rearview mirror is present in the car or not.
is_ecw	Boolean flag indicating whether Engine Check Warning (ECW) is available in the car or not.
is_speed_alert	Boolean flag indicating whether the speed alert system is available in the car or not.
ncap_rating	Safety rating given by NCAP (out of 5)
is_claim	Outcome: Boolean flag indicating whether the policyholder file a claim in the next 6 months or not.

EXPLORE THE DATA



- □ ▲
- 1. Menampilkan info data
- 2. Menampilkan 5 data pertama
- 3. Mengecek adanya missing value
- 4. Mengecek adanya duplicate value
- 5. Mengecek adanya imbalanced data

MENAMPILKAN INFO DATA

58592 OBS X 45 FITUR

```
<class 'pandas.core.frame.DataFrame'>
Index: 58592 entries, ID00001 to ID58592
Data columns (total 45 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   policy_tenure    58592 non-null   float64
 1   age_of_car        58592 non-null   float64
 2   age_of_policyholder 58592 non-null   float64
 3   area_cluster      58592 non-null   object 
 4   population_density 58592 non-null   int64  
 5   make              58592 non-null   int64  
 6   segment            58592 non-null   object 
 7   model              58592 non-null   object 
 8   fuel_type          58592 non-null   object 
 9   engine_type        58592 non-null   object 
 10  airbags            58592 non-null   int64  
 11  is_esc              58592 non-null   object 
 12  is_adjustable_steering 58592 non-null   object 
 13  is_tpms             58592 non-null   object 
 14  is_parking_sensors 58592 non-null   object 
 15  is_parking_camera   58592 non-null   object 
 16  rear_brakes_type   58592 non-null   object 
 17  displacement        58592 non-null   int64  
 18  cylinder            58592 non-null   int64  
 19  transmission_type  58592 non-null   object 
 20  gear_box            58592 non-null   int64  
 21  steering_type       58592 non-null   object 
 22  turning_radius      58592 non-null   float64
 23  length              58592 non-null   int64  
 24  width               58592 non-null   int64  
 25  height              58592 non-null   int64  
 26  gross_weight         58592 non-null   int64  
 27  is_front_fog_lights 58592 non-null   object 
 28  is_rear_window_wiper 58592 non-null   object 
 29  is_rear_window_washer 58592 non-null   object 
 30  is_rear_window_defogger 58592 non-null   object 
 31  is_brake_assist      58592 non-null   object 
 32  is_power_door_locks 58592 non-null   object 
 33  is_central_locking    58592 non-null   object 
 34  is_power_steering     58592 non-null   object 
 35  is_driver_seat_height_adjustable 58592 non-null   object 
 36  is_day_night_rear_view_mirror 58592 non-null   object 
 37  is_ecw                58592 non-null   object 
 38  is_speed_alert        58592 non-null   object 
 39  ncap_rating           58592 non-null   int64  
 40  is_claim              58592 non-null   int64  
 41  max_torque_Nm         58592 non-null   float64
 42  max_torque_rpm        58592 non-null   float64
 43  max_power_bhp          58592 non-null   float64
 44  max_power_rpm          58592 non-null   float64
dtypes: float64(8), int64(12), object(25)
memory usage: 20.6+ MB
```

MENAMPILKAN 5 DATA PERTAMA

CEK MISSING & DUPLICATE VALUE

```
policy_tenure          0 length                      0
age_of_car              0 width                       0
age_of_policyholder    0 height                      0
area_cluster            0 gross_weight                 0
population_density      0 is_front_fog_lights        0
make                    0 is_rear_window_wiper       0
segment                 0 is_rear_window_washer      0
model                   0 is_rear_window_defogger     0
fuel_type                0 is_brake_assist             0
engine_type              0 is_power_door_locks        0
airbags                  0 is_central_locking           0
is_esc                   0 is_power_steering           0
is_adjustable_steering  0 is_driver_seat_height_adjustable 0
is_tpms                  0 is_day_night_rear_view_mirror 0
is_parking_sensors       0 is_ecw                         0
is_parking_camera        0 is_speed_alert               0
rear_brakes_type         0 ncap_rating                 0
displacement             0 is_claim                     0
cylinder                  0 max_torque_Nm                0
transmission_type        0 max_torque_rpm              0
gear_box                  0 max_power_bhp                0
steering_type             0 max_power_rpm               0
turning_radius            0 dtype: int64
```

`df.isnull().sum()`

`df.duplicated().sum()`

0

KESIMPULAN :

Data Lengkap dan Unik

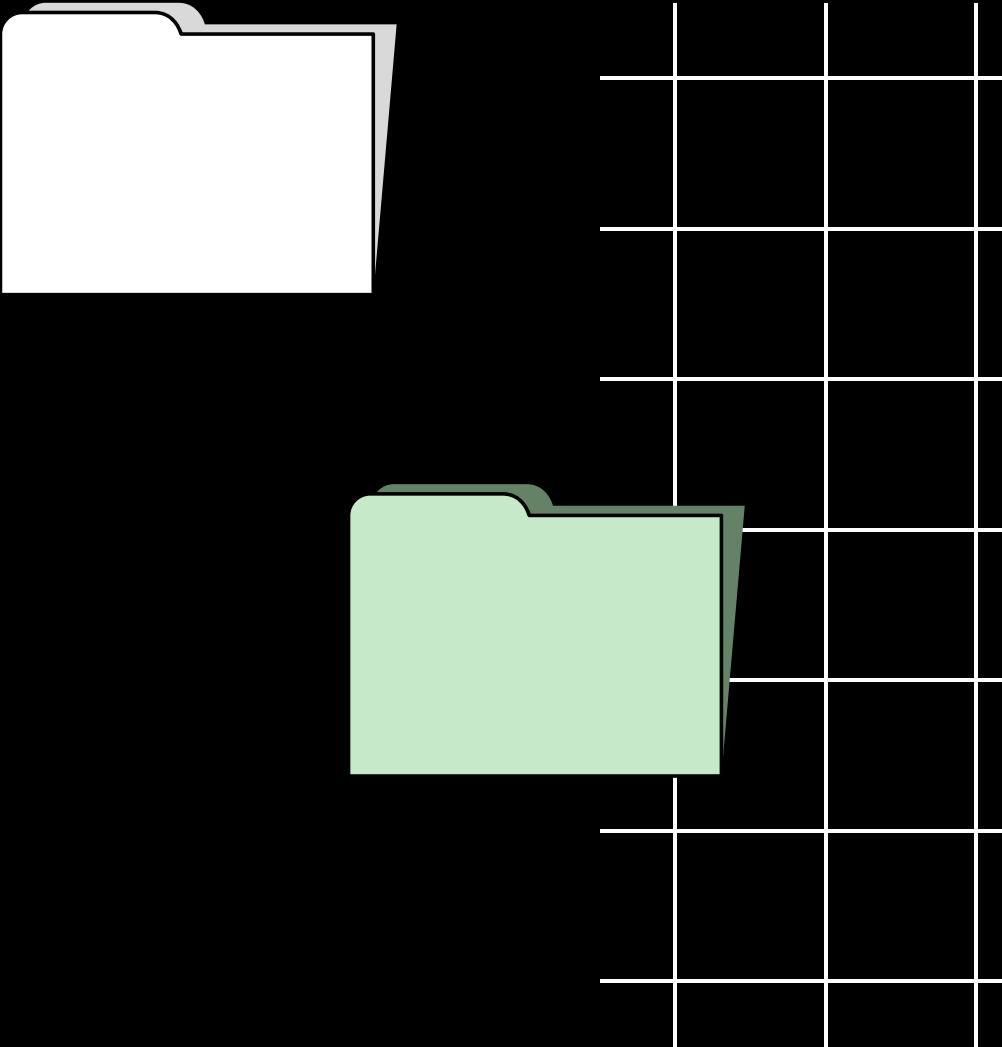
PERSEBARAN DATA RESPONS

Fitur : 'is_claim'

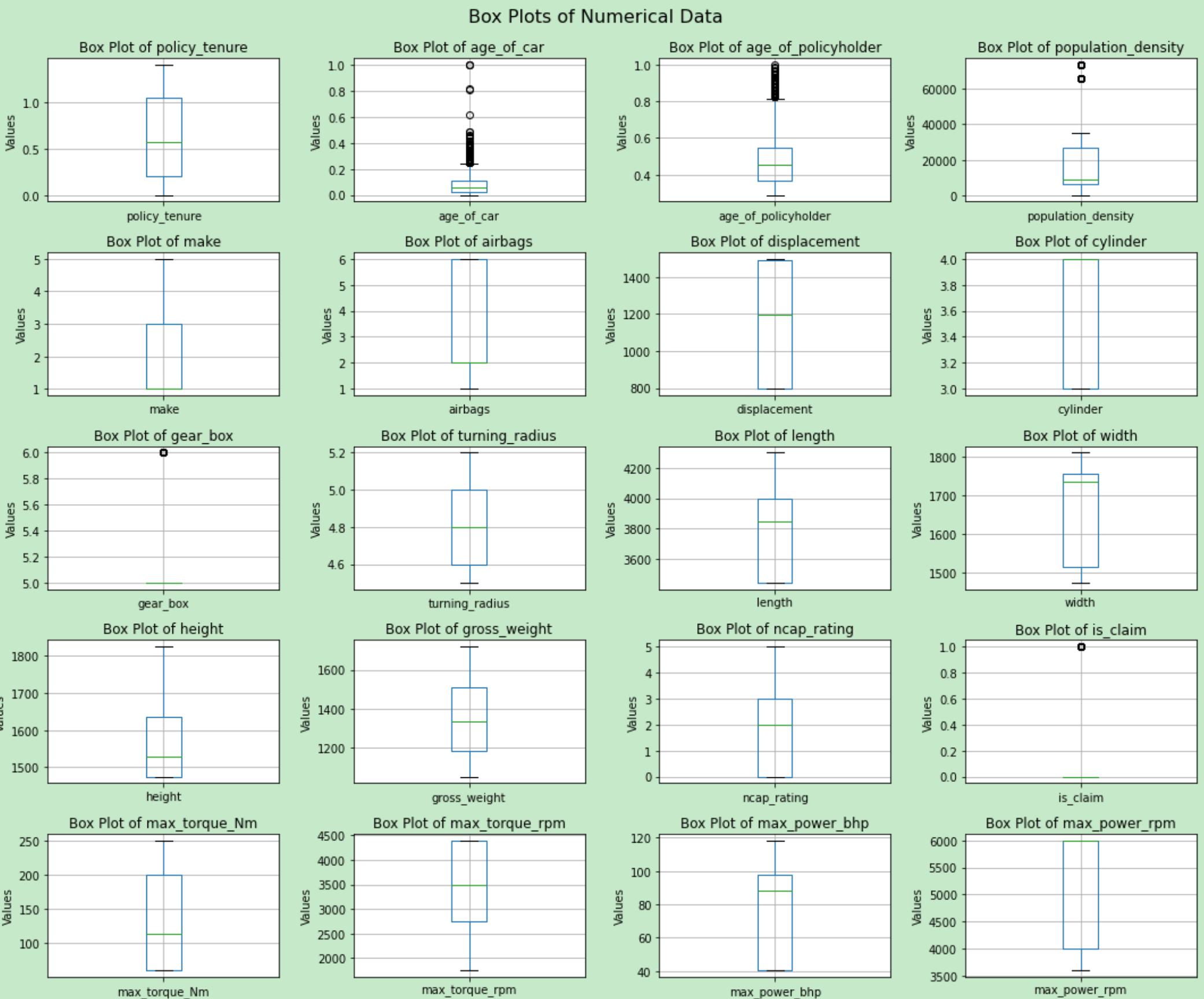
Jumlah Data	
0 (Tidak Mengajukan Klaim)	1 (Mengajukan Klaim)
54844 (93,6%)	3748 (6,4%)

KESIMPULAN : Data tidak seimbang (**IMBALANCED**) sehingga akan ditanganid engan teknik OVERSAMPLING, yaitu **SMOTEN** (Syntethic Minority Over Sampling Technique for Nominal)

PROCESSING THE DATA

- 
- 1. Melihat Persebaran Data Numerik
 - 2. Menghilangkan Outlier pada Data Numerik
 - 3. Melihat Persebaran Data Kategorik
 - 4. Mengubah Data Kategorik menjadi Numerik
 - 5. Menampilkan Deskripsi Data
 - 6. Mengecek Korelasi Antarfitur

OUTLIER DATA NUMERIK

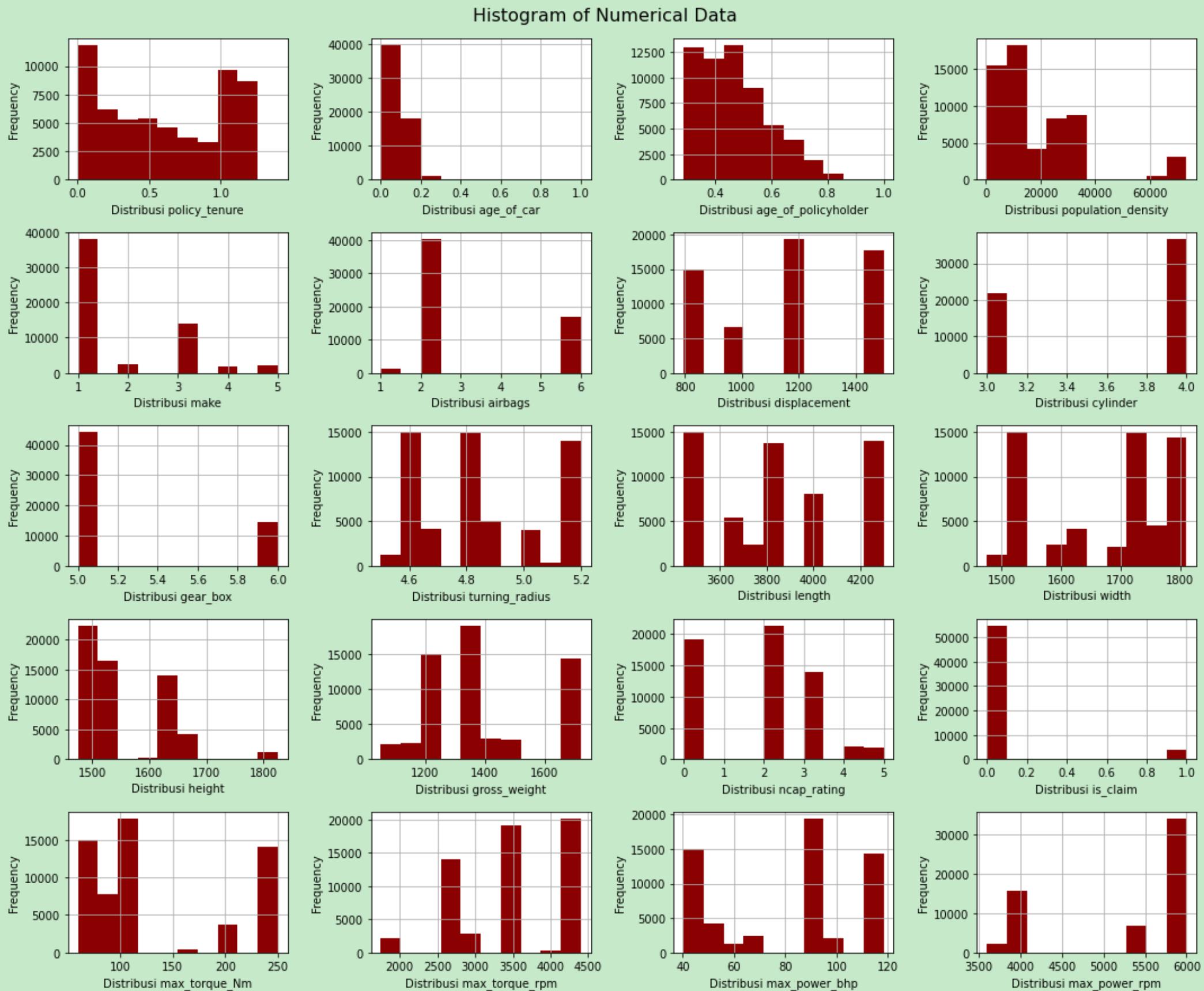


Ditemukan outlier pada fitur:

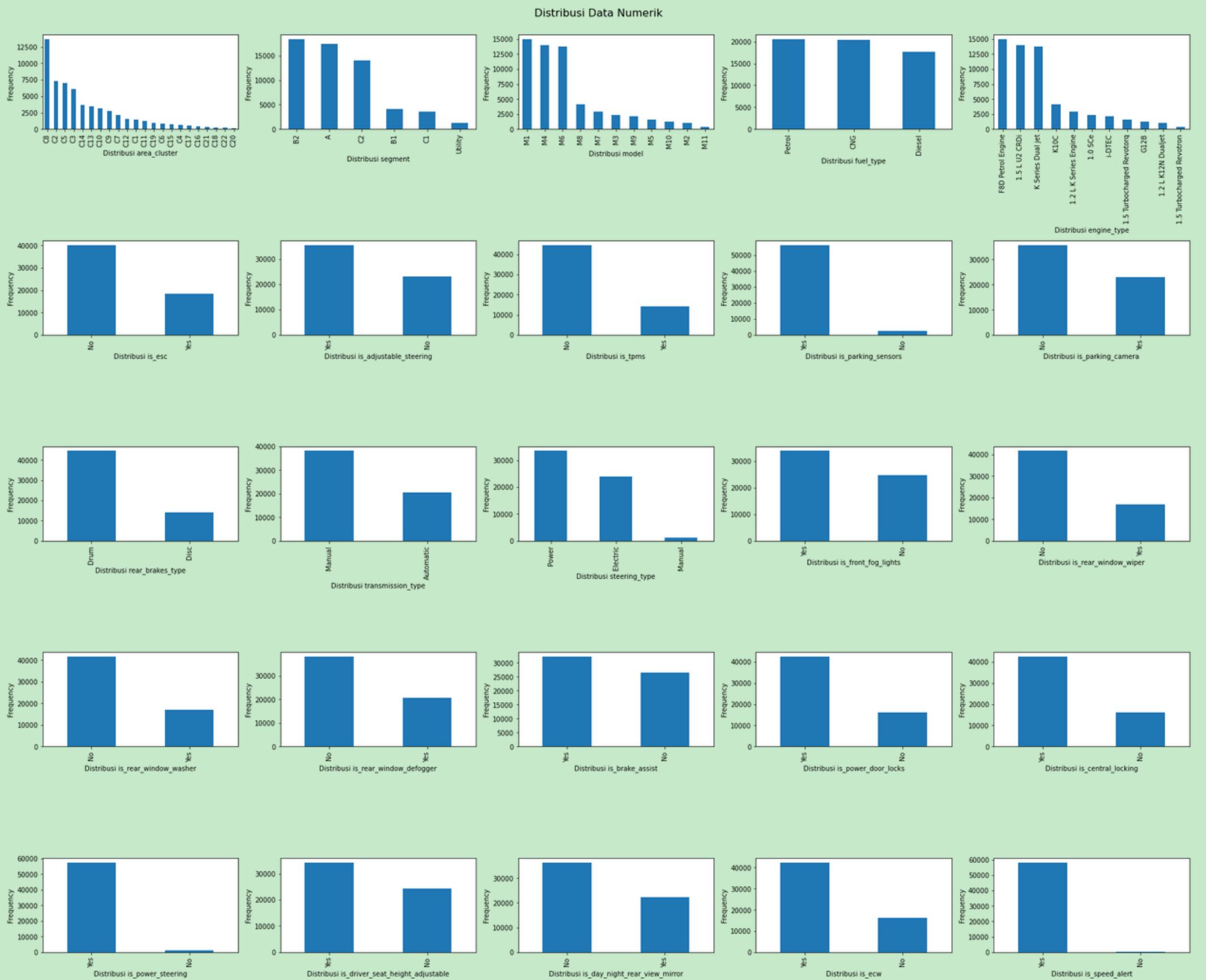
- age_of_car,
- age_of_policyholder, dan
- population_density

Akan ditangani dengan mengganti outlier yang terlalu besar dengan 99% persentil data.

PERSEBARAN DATA NUMERIK



PERSERBAAN DATA KATEGORIK



UBAH KATEGORIK MENJADI NUMERIK

area_cluster			
C22	21	C11	10
C21	20	C10	9
C20	19	C9	8
C19	18	C8	7
C18	17	C7	6
C17	16	C6	5
C16	15	C5	4
C15	14	C4	3
C14	13	C3	2
C13	12	C2	1
C12	11	C1	0

model	
M11	10
M10	9
M9	8
M8	7
M7	6
M6	5
M5	4
M4	3
M3	2
M2	1
M1	0

model		model	
1.5 Turbocharged Revotron	10	1.5 Turbocharged Revotorq	4
G12B	9	1.5 L U2 CRDi	3
i-DTEC	8	1.0 SCe	2
K10C	7	1.2 L K Series Engine	1
1.2 L K Series Engine	6	1.2 L K12N Dualjet	1
K Series Dual jet	5	F8D Petrol Engine	0

rear_brakes_type	
Drum	1
Disc	0

fuel_type	
Diesel	2
Petrol	1
CNG	0

steering_type	
Power	2
Electric	1
Manual	0

UBAH KATEGORIK MENJADI NUMERIK

is_speed_alert, is_day_night_rear_view_mirror, is_power_steering, is_power_door_locks,
is_ecw, is_esc, is_rear_window_wiper, is_parking_sensors, is_parking_camera,
is_adjustable_steering, is_rear_window_washer, is_driver_seat_height_adjustable,
is_central_locking, is_brake_assist, is_rear_window_washer, is_front_fog_lights,

Yes	1
No	0

```
Categorical Features in DataSet: 0
Index([], dtype='object')
Numerical Features in DataSet: 45
Index(['policy_tenure', 'age_of_car', 'age_of_policyholder', 'area_cluster',
       'population_density', 'make', 'segment', 'model', 'fuel_type',
       'engine_type', 'airbags', 'is_esc', 'is_adjustable_steering', 'is_tpms',
       'is_parking_sensors', 'is_parking_camera', 'rear_brakes_type',
       'displacement', 'cylinder', 'transmission_type', 'gear_box',
       'steering_type', 'turning_radius', 'length', 'width', 'height',
       'gross_weight', 'is_front_fog_lights', 'is_rear_window_wiper',
       'is_rear_window_washer', 'is_rear_window_defogger', 'is_brake_assist',
       'is_power_door_locks', 'is_central_locking', 'is_power_steering',
       'is_driver_seat_height_adjustable', 'is_day_night_rear_view_mirror',
       'is_ecw', 'is_speed_alert', 'ncap_rating', 'is_claim', 'max_torque_Nm',
       'max_torque_rpm', 'max_power_bhp', 'max_power_rpm'],
      dtype='object')
```

Kesimpulan:

Seluruh fitur pada data kategorik telah diubah menjadi data numerik.

segment

Utility	5
C2	4
C1	3
B2	2
B1	1
A	0

transmission_type

Automatic	1
Manual	0

MENAMPILKAN DESKRIPSI DATA

Dapat dilihat nilai mean, standard deviasi, minimum, 25% percentile, median, 75% percentil, dan maksimum

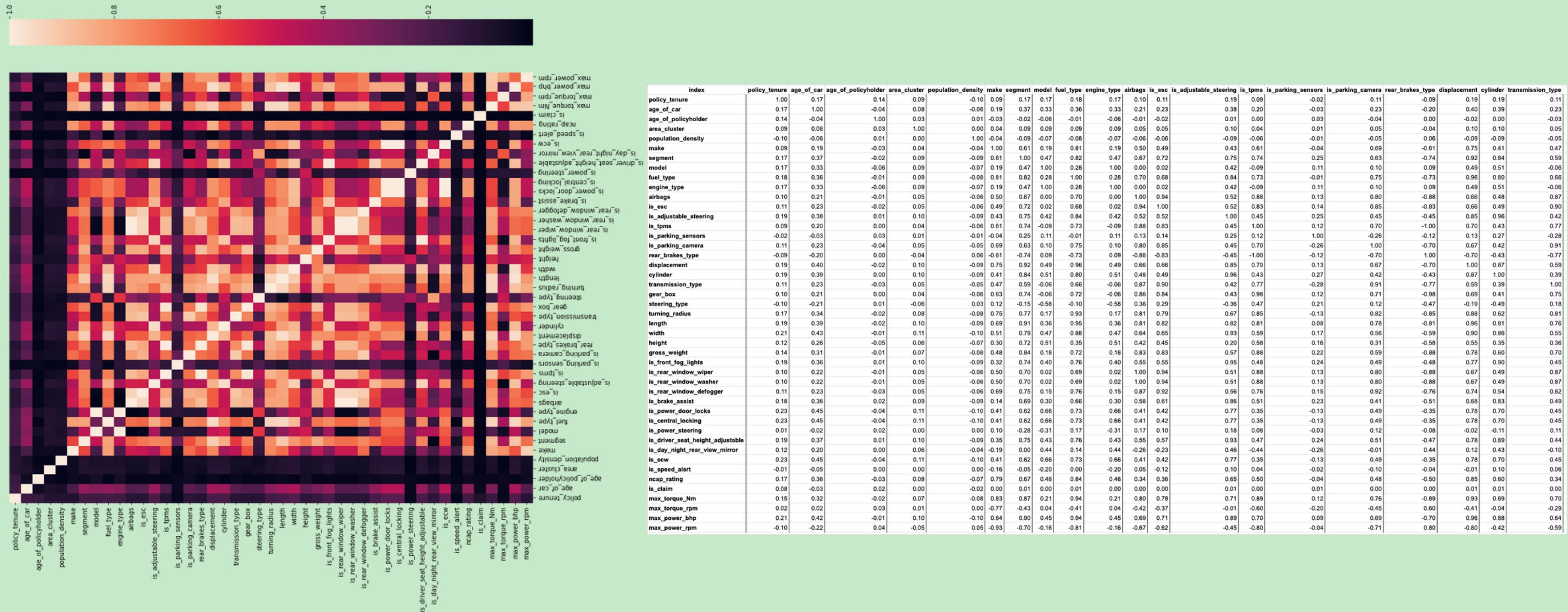
	policy_tenure	age_of_car	age_of_policyholder	area_cluster	population_density	make	segment	model	fuel_type	engine_type
count	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000
mean	0.611246	0.068879	0.469075	6.672174	18826.858667	1.763722	1.938644	3.437773	0.955625	3.437773
std	0.414156	0.054517	0.121881	4.513651	17660.174792	1.136988	1.566329	2.593467	0.804748	2.593467
min	0.002735	0.000000	0.288462	0.000000	290.000000	1.000000	0.000000	0.000000	0.000000	0.000000
25%	0.210250	0.020000	0.365385	2.000000	6112.000000	1.000000	0.000000	0.000000	0.000000	0.000000
50%	0.573792	0.060000	0.451923	7.000000	8794.000000	1.000000	2.000000	3.000000	1.000000	3.000000
75%	1.039104	0.110000	0.548077	9.000000	27003.000000	3.000000	4.000000	5.000000	2.000000	5.000000
max	1.396641	0.210000	0.788462	21.000000	73430.000000	5.000000	5.000000	10.000000	2.000000	10.000000

MENAMPILKAN DESKRIPSI DATA

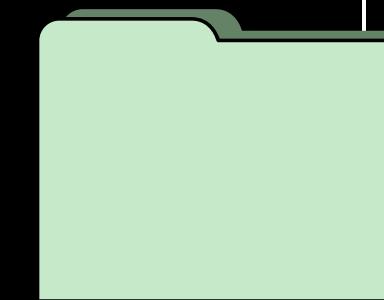
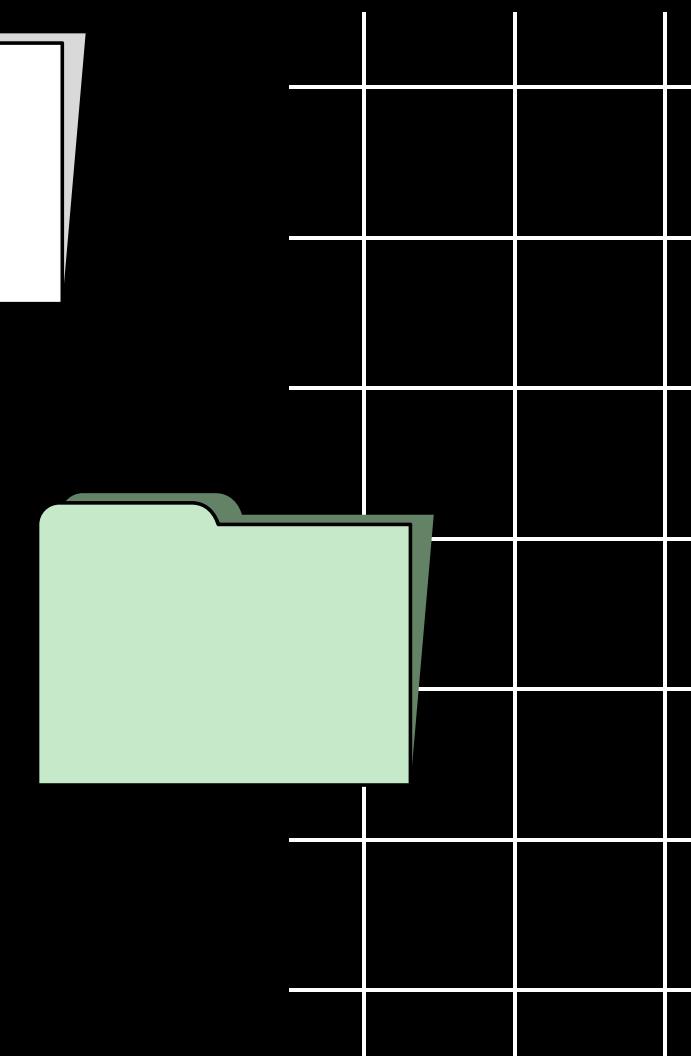
Dapat dilihat nilai mean, standard deviasi, minimum, 25% percentile, median, 75% percentil, dan maksimum

is_day_night_rear_view_mirror	is_ecw	is_speed_alert	ncap_rating	is_claim	max_torque_Nm	max_torque_rpm	max_power_bhp	max_power_rpm
58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000	58592.000000
0.380308	0.724246	0.993805	1.759950	0.063968	134.450937	3533.176031	78.976765	5307.163094
0.485467	0.446897	0.078467	1.389576	0.244698	73.146794	725.960661	27.699259	916.770819
0.000000	0.000000	0.000000	0.000000	0.000000	60.000000	1750.000000	40.360000	3600.000000
0.000000	0.000000	1.000000	0.000000	0.000000	60.000000	2750.000000	40.360000	4000.000000
0.000000	1.000000	1.000000	2.000000	0.000000	113.000000	3500.000000	88.500000	6000.000000
1.000000	1.000000	1.000000	3.000000	0.000000	200.000000	4400.000000	97.890000	6000.000000
1.000000	1.000000	1.000000	5.000000	1.000000	250.000000	4400.000000	118.360000	6000.000000

MENGECEK KORELASI DATA



MODEL THE DATA



- 1. Memisahkan Variabel Target dan Respon
- 2. Melakukan *Feature Selection*
- 3. Melakukan *Normalization*
- 4. Memisahkan Data Training dan Testing
- 5. Melakukan *Balancing Data*
- 6. Melakukan Uji Model 1: Decision Tree
- 7. Melakukan Uji Model 2: XGBoost
- 8. Melakukan Uji Model 3: Adaboost

VARIABEL PREDIKTOR

	1 X								
	policy_tenure	age_of_car	age_of_policyholder	area_cluster	population_density	make	segment	model	fuel_type
policy_id									
ID00001	0.515874	0.05	0.644231	0	4990	1	0	0	0
ID00002	0.672619	0.02	0.375000	1	27003	1	0	0	0
ID00003	0.841110	0.02	0.384615	2	4076	1	0	0	0
ID00004	0.900277	0.11	0.432692	3	21622	1	3	1	1
ID00005	0.596403	0.11	0.634615	4	34738	2	0	2	1
...
ID58588	0.355089	0.13	0.644231	7	8794	2	0	2	1
ID58589	1.199642	0.02	0.519231	13	7788	1	0	0	0
ID58590	1.162273	0.05	0.451923	4	34738	1	0	0	0
ID58591	1.236307	0.14	0.557692	7	8794	1	2	5	1
ID58592	0.124429	0.02	0.442308	7	8794	3	4	3	2

1 y

policy_id

ID00001 0

ID00002 0

ID00003 0

ID00004 0

ID00005 0

..

ID58588 0

ID58589 0

ID58590 0

ID58591 0

ID58592 0

Name: is_claim, Length: 58592,

VARIABEL
RESPON

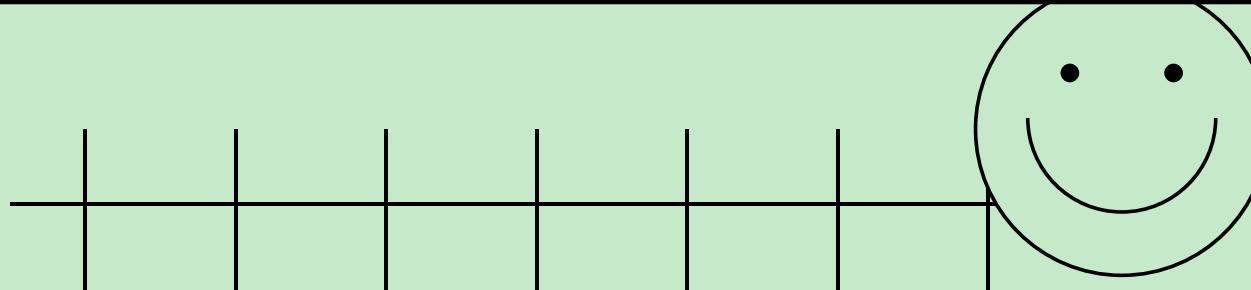
MELAKUKAN FEATURE SELECTION

```
1 #PILIH 20 FITUR TERBAIK BERDASARKAN K-VALUE
2 from sklearn.feature_selection import SelectKBest
3 kbest = SelectKBest(k=20).fit(X,y)
4 X_selected = kbest.transform(X)
5
6 kbest.get_feature_names_out()
```

```
array(['policy_tenure', 'age_of_car', 'age_of_policyholder',
       'population_density', 'fuel_type', 'is_adjustable_steering',
       'is_parking_sensors', 'displacement', 'cylinder', 'steering_type',
       'width', 'is_front_fog_lights', 'is_brake_assist',
       'is_power_door_locks', 'is_central_locking',
       'is_driver_seat_height_adjustable',
       'is_day_night_rear_view_mirror', 'is_ecw', 'is_speed_alert',
       'max_power_bhp'], dtype=object)
```

Akan dipilih 20 fitur terbaik dengan skor k-value tertinggi.

MELAKUKAN NORMALISASI NILAI X



1 X_selected

```
array([[5.15873590e-01, 5.0000000e-02, 6.44230769e-01, ...,
       0.0000000e+00, 1.0000000e+00, 4.0360000e+01],
      [6.72618514e-01, 2.0000000e-02, 3.7500000e-01, ...,
       0.0000000e+00, 1.0000000e+00, 4.0360000e+01],
      [8.41110256e-01, 2.0000000e-02, 3.84615385e-01, ...,
       0.0000000e+00, 1.0000000e+00, 4.0360000e+01],
      ...,
      [1.16227251e+00, 5.0000000e-02, 4.51923077e-01, ...,
       0.0000000e+00, 1.0000000e+00, 4.0360000e+01],
      [1.23630690e+00, 1.4000000e-01, 5.57692308e-01, ...,
       1.0000000e+00, 1.0000000e+00, 8.8500000e+01],
      [1.24428929e-01, 2.0000000e-02, 4.42307692e-01, ...,
       1.0000000e+00, 1.0000000e+00, 1.1345000e+02]])
```

SEBELUM NORMALISASI

```
1 #Standardisasi nilai X
2 from sklearn.preprocessing import StandardScaler
3 X_selected = StandardScaler().fit_transform(X_selected)
```

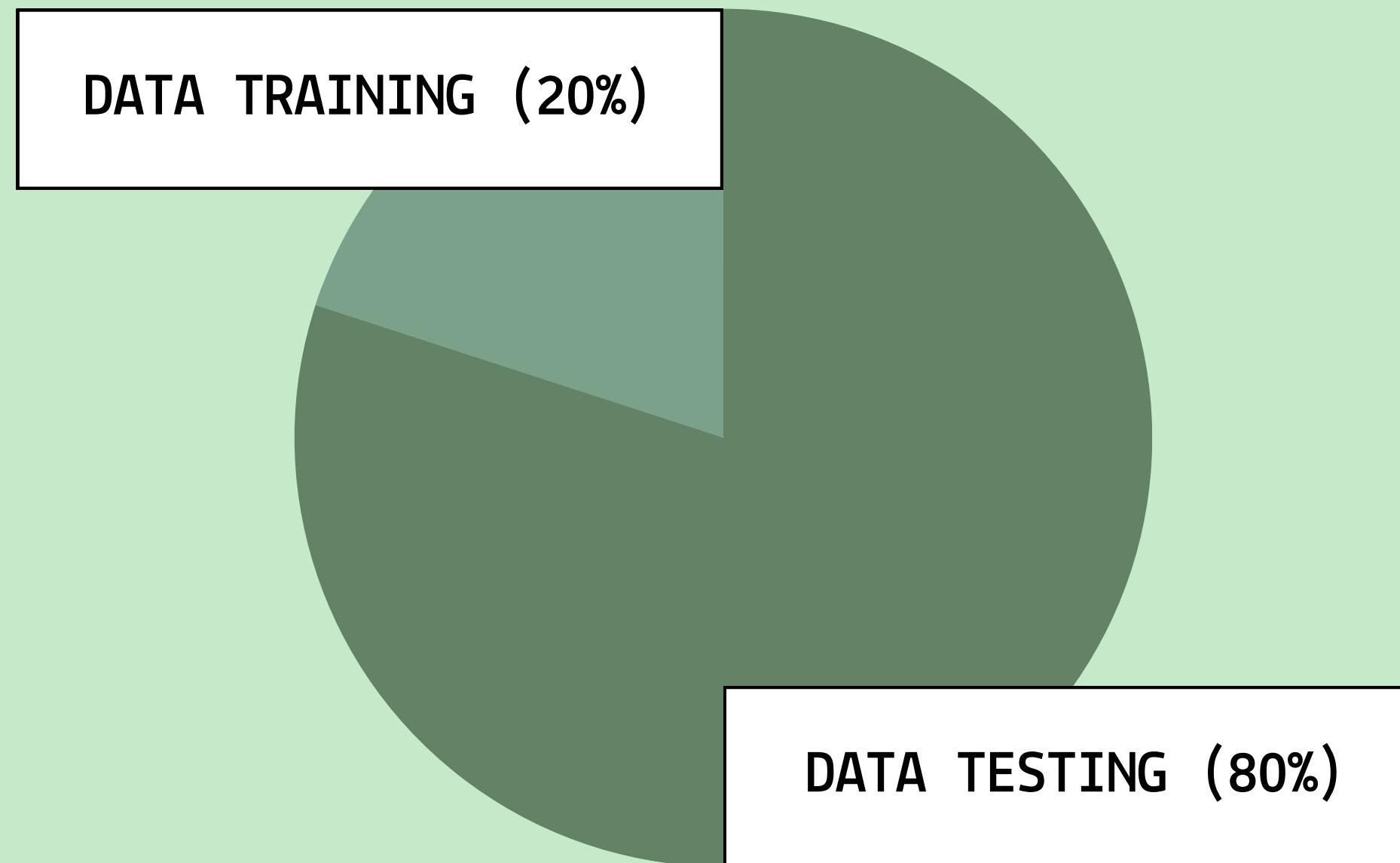
SETELAH NORMALISASI

1 X_selected

```
array([[-0.23028345, -0.34629581, 1.43711871, ..., -1.62062204,
        0.07895573, -1.39415626],
      [ 0.14818765, -0.89659227, -0.77186813, ..., -1.62062204,
        0.07895573, -1.39415626],
      [ 0.55502223, -0.89659227, -0.69297574, ..., -1.62062204,
        0.07895573, -1.39415626],
      ...,
      [ 1.33048996, -0.34629581, -0.14072904, ..., -1.62062204,
        0.07895573, -1.39415626],
      [ 1.50925096, 1.30459357, 0.72708722, ..., 0.61704702,
        0.07895573, 0.34381122],
      [-1.17545276, -0.89659227, -0.21962142, ..., 0.61704702,
        0.07895573, 1.24456503]])
```

MELAKUKAN PEMISAHAN DATA

```
data = X_train  
labels = y_train
```



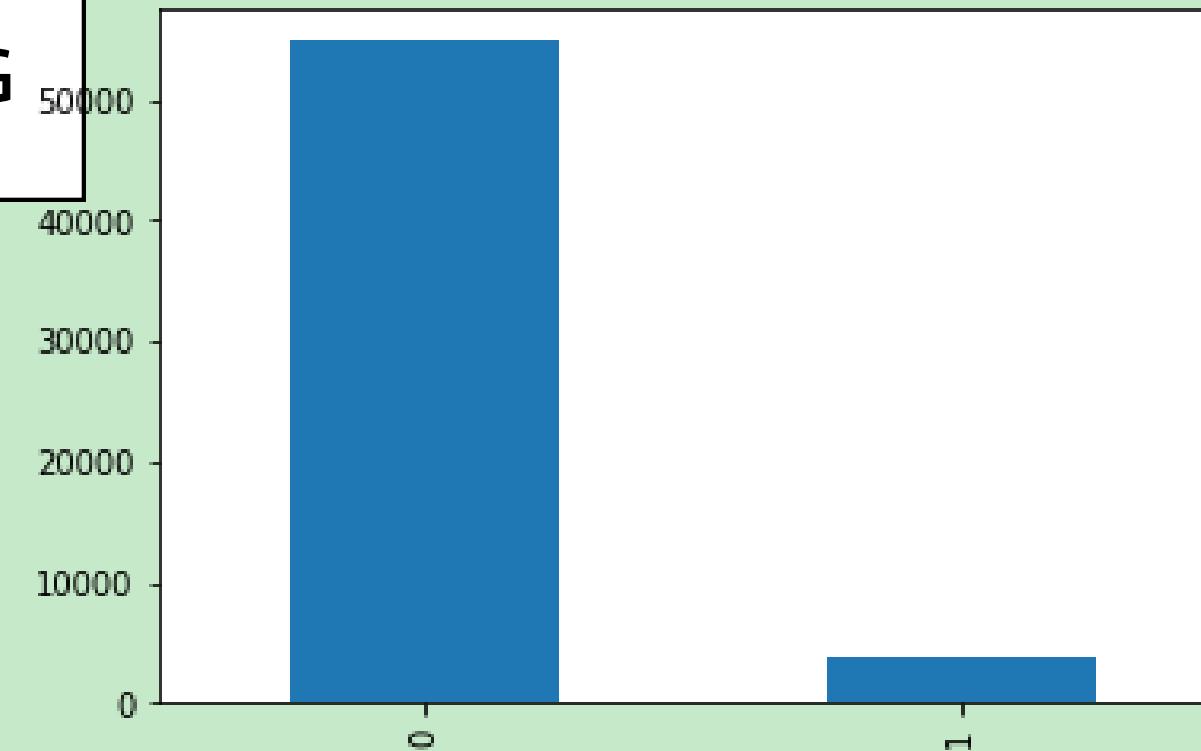
```
data = X_test  
labels = y_test
```

MENGATASI IMBALANCED DATA

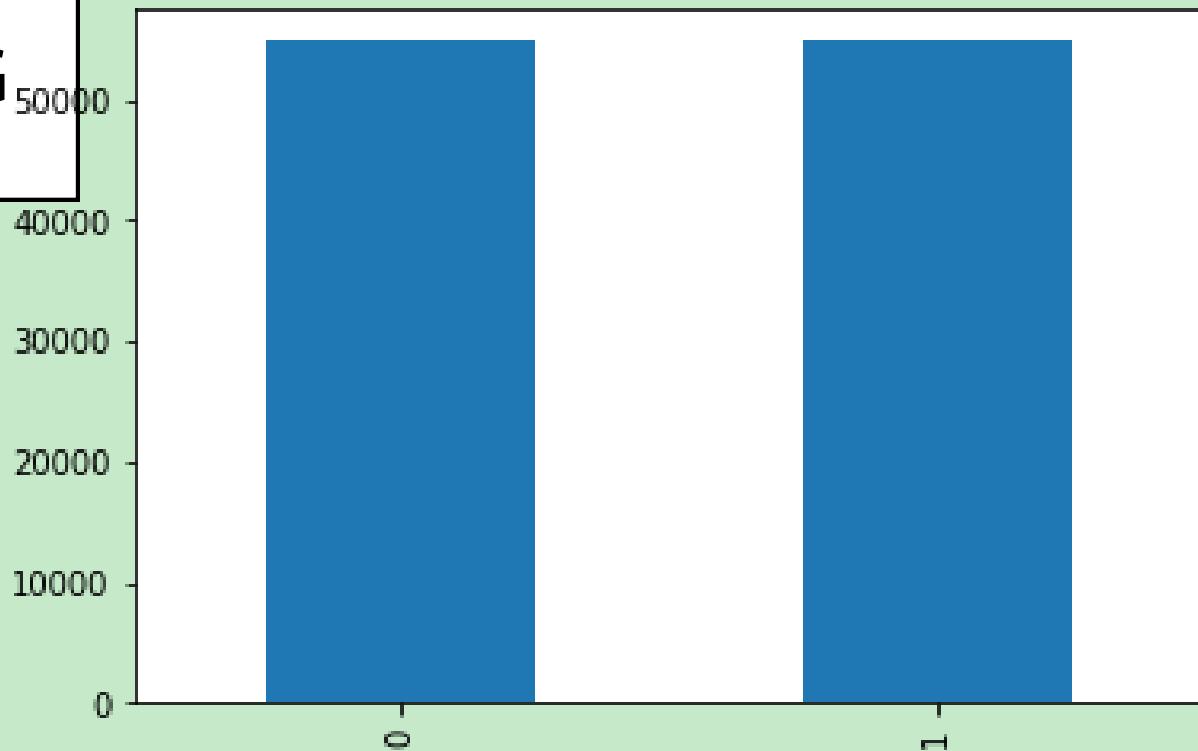
dengan SMOTEN (Synthetic Minority Over-sampling Technique for Nominal)

```
1 #Sampling Method untuk Mengatasi Imbalanced Data
2 !pip install imbalanced-learn
3 from imblearn.over_sampling import SMOTEN
4 sm = SMOTEN(sampling_strategy="minority",k_neighbors=20,n_jobs=-1)
5 X_res, y_res = sm.fit_resample(X_selected, y)
```

SEBELUM
BALANCING



SETELAH
BALANCING



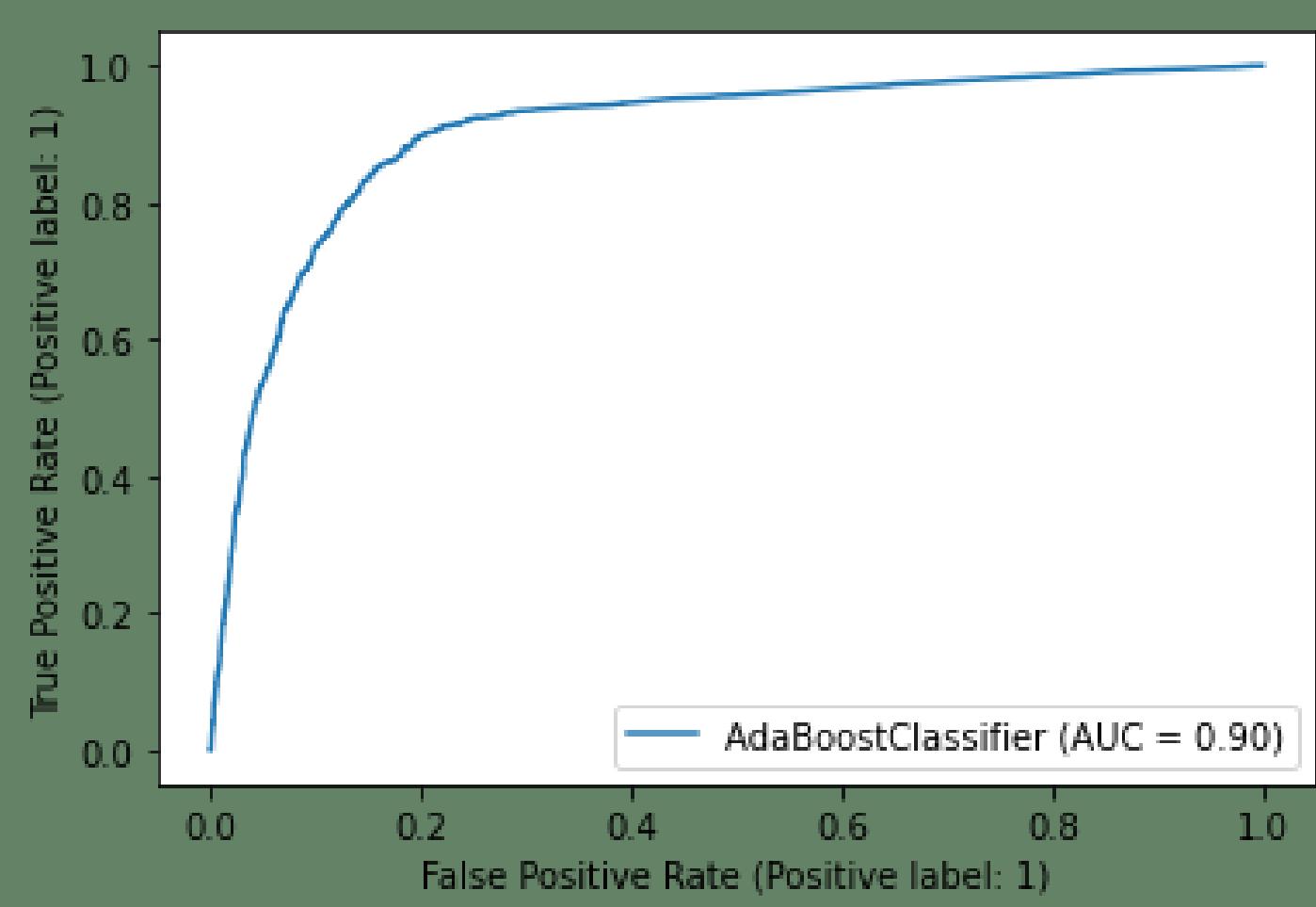
MODEL 1A : ADABoost



tanpa Hyperparameter Tuning

```
3 from sklearn.ensemble import AdaBoostClassifier  
4  
5 ada = AdaBoostClassifier()  
6 ada.fit(X_train,y_train)  
7  
8 print(classification_report(y_test,ada.predict(X_test)))  
9  
10 print(roc_auc_score(y_test,ada.predict(X_test)))  
11  
12 plot_roc_curve(ada, X_test, y_test)  
13 plt.show()
```

Rata-Rata setelah 4x Running
roc_auc_score : 0.8432552344



	PRECISION	RECALL	F1-SCORE	SUPPORT
0	0.90	0.77	0.83	10969
1	0.80	0.91	0.85	10696
ACCURACY				0.84
MACRO AVG	0.85	0.84	0.84	21938
WEIGTHED AVG	0.85	0.84	0.84	21938

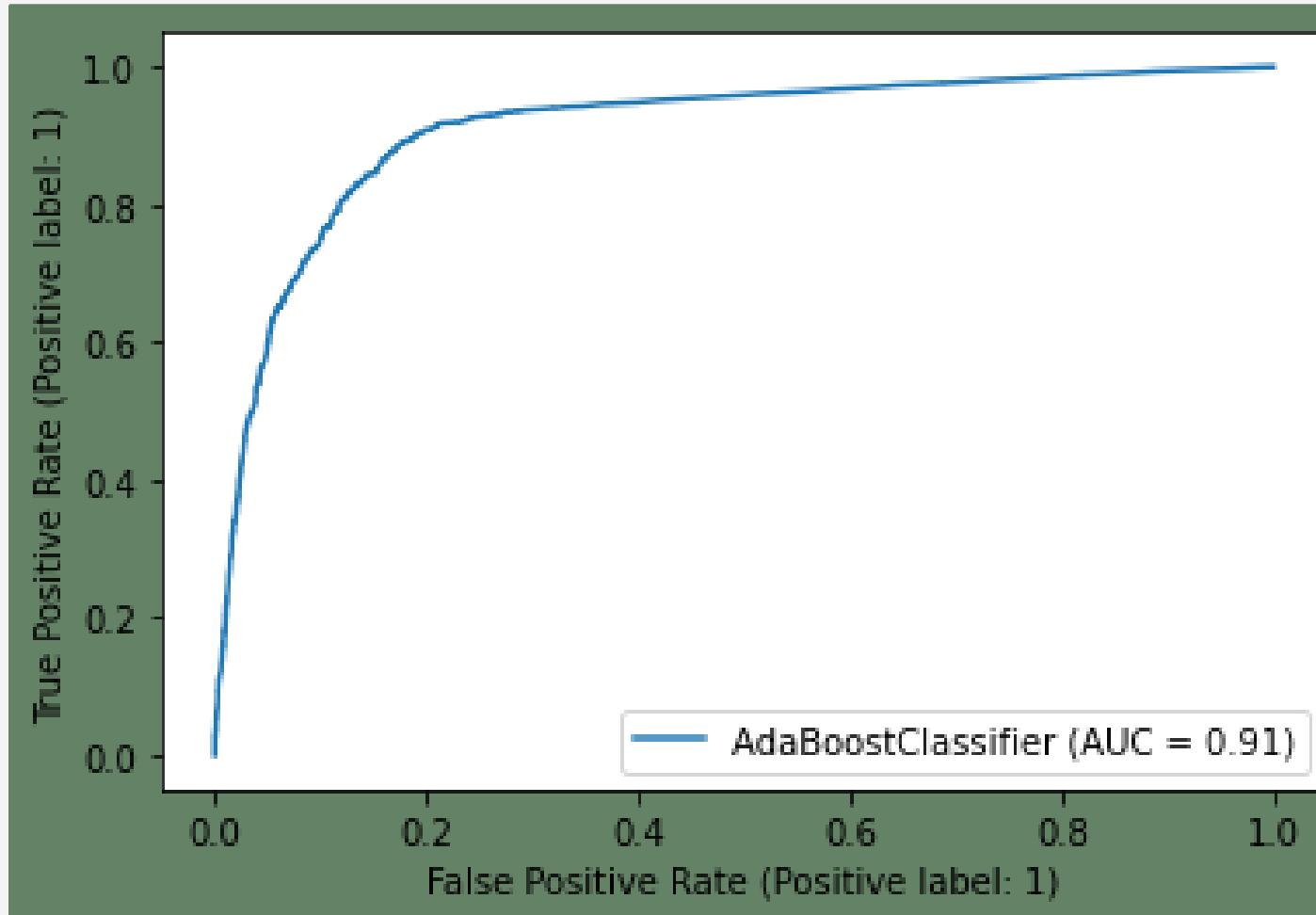
MODEL 1B : ADABOOST



dengan Hyperparameter Tuning GridSearch

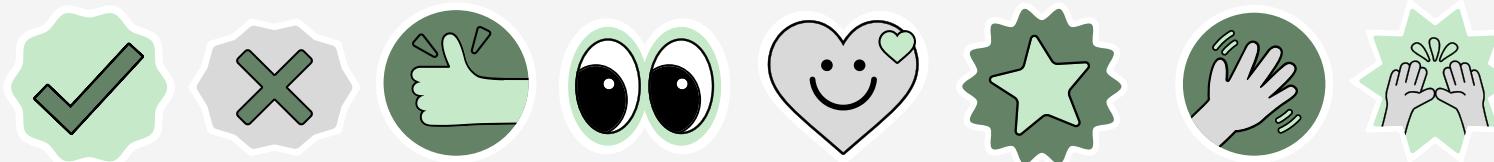
```
2 from sklearn.model_selection import GridSearchCV
3
4 model = AdaBoostClassifier()
5 param_grid1 = {
6     'learning_rate': [0.0001, 0.01, 0.1, 1.0, 1.1, 1.2],
7     'n_estimators': [10, 50, 100]
8 }
9 grid3 = GridSearchCV(estimator = model, param_grid = param_grid1,
10                      n_jobs=-1, cv = 5, scoring='accuracy')
11 grid3.fit(X_train, y_train)
12 ada_grid = grid3.best_estimator_
13
14 print(roc_auc_score(y_test, ada_grid.predict(X_test)))
15
16 print(classification_report(y_test, ada_grid.predict(X_test)))
17
18 from sklearn.metrics import plot_roc_curve
19 plot_roc_curve(ada_grid, X_test, y_test)
20 plt.show()
```

Rata-Rata setelah 4x Running
roc_auc_score : 0.8520831434



	PRECISION	RECALL	F1-SCORE	SUPPORT
0	0.89	0.80	0.84	10969
1	0.82	0.90	0.86	10969
ACCURACY				0.85
MACRO AVG	0.85	0.85	0.85	21938
WEIGTHED AVG	0.85	0.85	0.85	21938

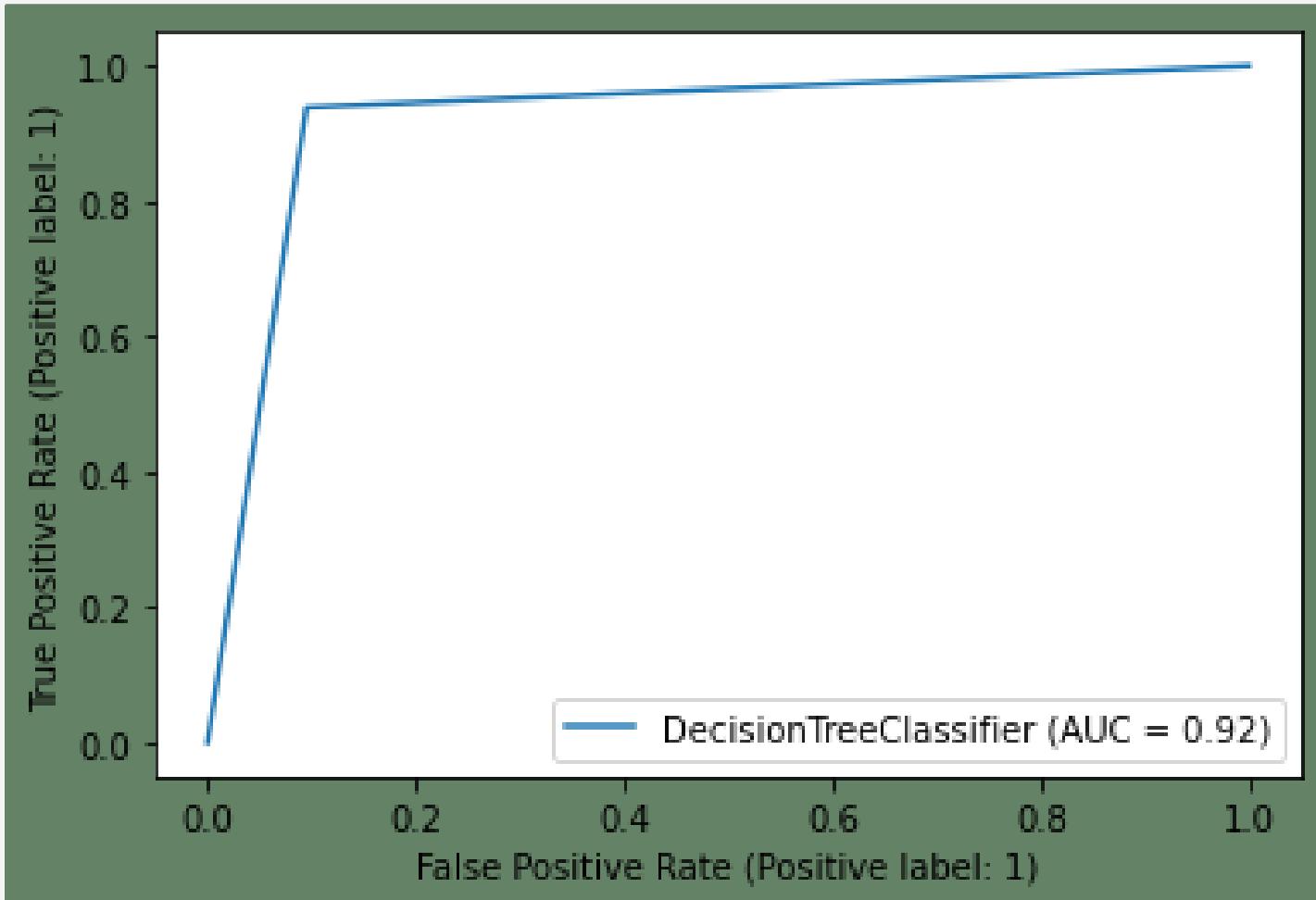
MODEL 2A : DTREE



tanpa Hyperparameter Tuning

```
3 from sklearn.tree import DecisionTreeClassifier  
4  
5 dtc = DecisionTreeClassifier()  
6 dtc.fit(X_train,y_train)  
7  
8 print(classification_report(y_test,dtc.predict(X_test)))  
9  
10 print(roc_auc_score(y_test,dtc.predict(X_test)))  
11  
12 plot_roc_curve(dtc, X_test, y_test)  
13 plt.show()
```

Rata-Rata setelah 4x Running
roc_auc_score : 0.9238611846



	PRECISION	RECALL	F1-SCORE	SUPPORT
0	0.94	0.91	0.92	10969
1	0.91	0.94	0.93	10969
ACCURACY				0.93
MACRO AVG	0.93	0.93	0.93	21938
WEIGTHED AVG	0.93	0.93	0.93	21938

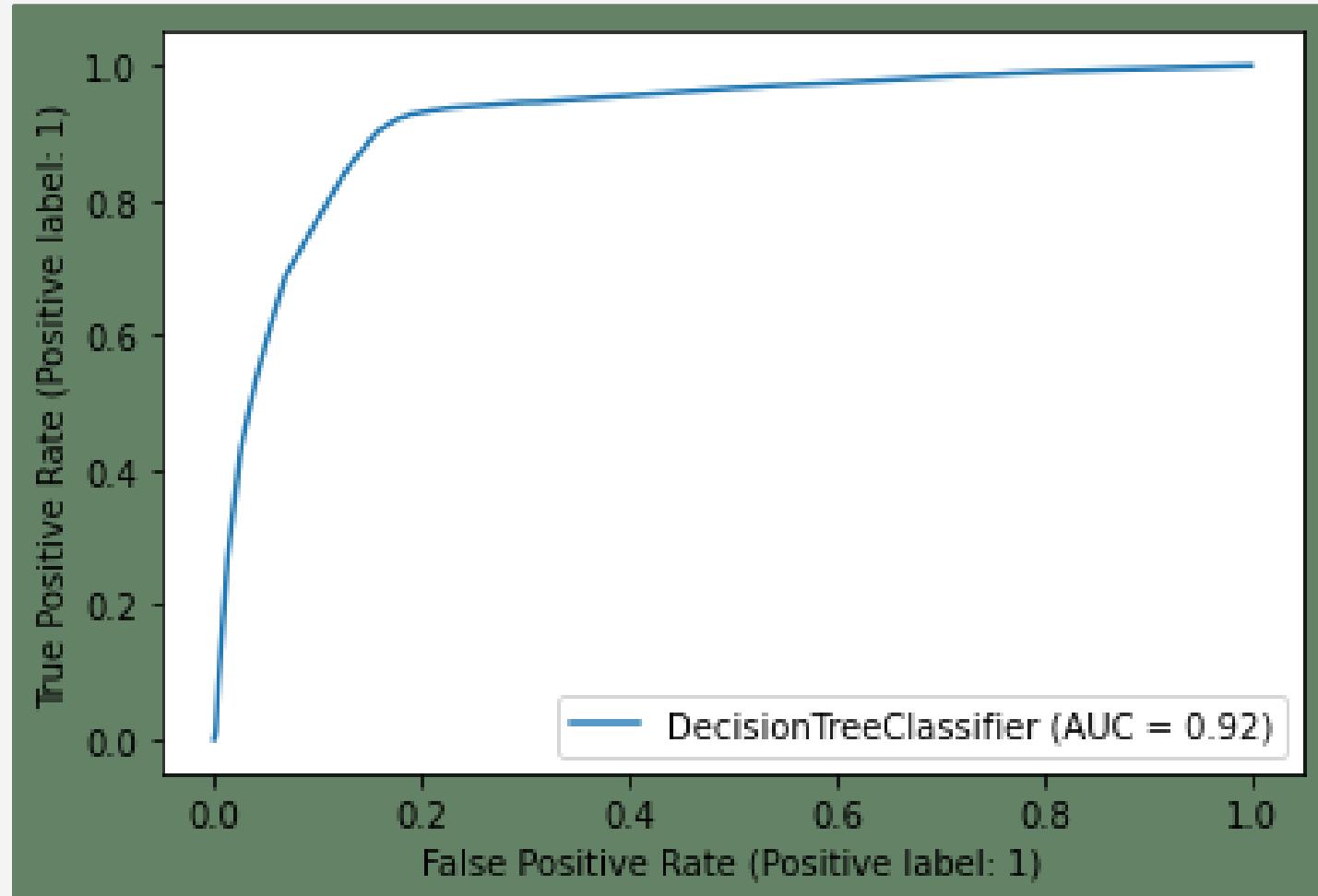
MODEL 2B : DTREE



dengan Hyperparameter Tuning GridSearch

```
2 from sklearn.model_selection import GridSearchCV
3
4 param_grid2 = {
5     'max_depth': range(3,10),
6 }
7 grid2 = GridSearchCV(DecisionTreeClassifier(class_weight = 'balanced'),
8                      cv = 5, scoring='roc_auc', param_grid = param_grid2)
9 grid2.fit(X_train, y_train)
10 dct_grid = grid2.best_estimator_
11
12 print(roc_auc_score(y_test, dct_grid.predict(X_test)))
13
14 print(classification_report(y_test, dct_grid.predict(X_test)))
15
16 from sklearn.metrics import plot_roc_curve
17 plot_roc_curve(dct_grid, X_test, y_test)
18 plt.show()
```

Rata-Rata setelah 4x Running
roc_auc_score : 0.8745859544



	PRECISION	RECALL	F1-SCORE	SUPPORT
0	0.90	0.86	0.88	10969
1	0.86	0.90	0.88	10969
ACCURACY			0.88	21938
MACRO AVG	0.88	0.88	0.88	21938
WEIGTHED AVG	0.88	0.88	0.88	21938

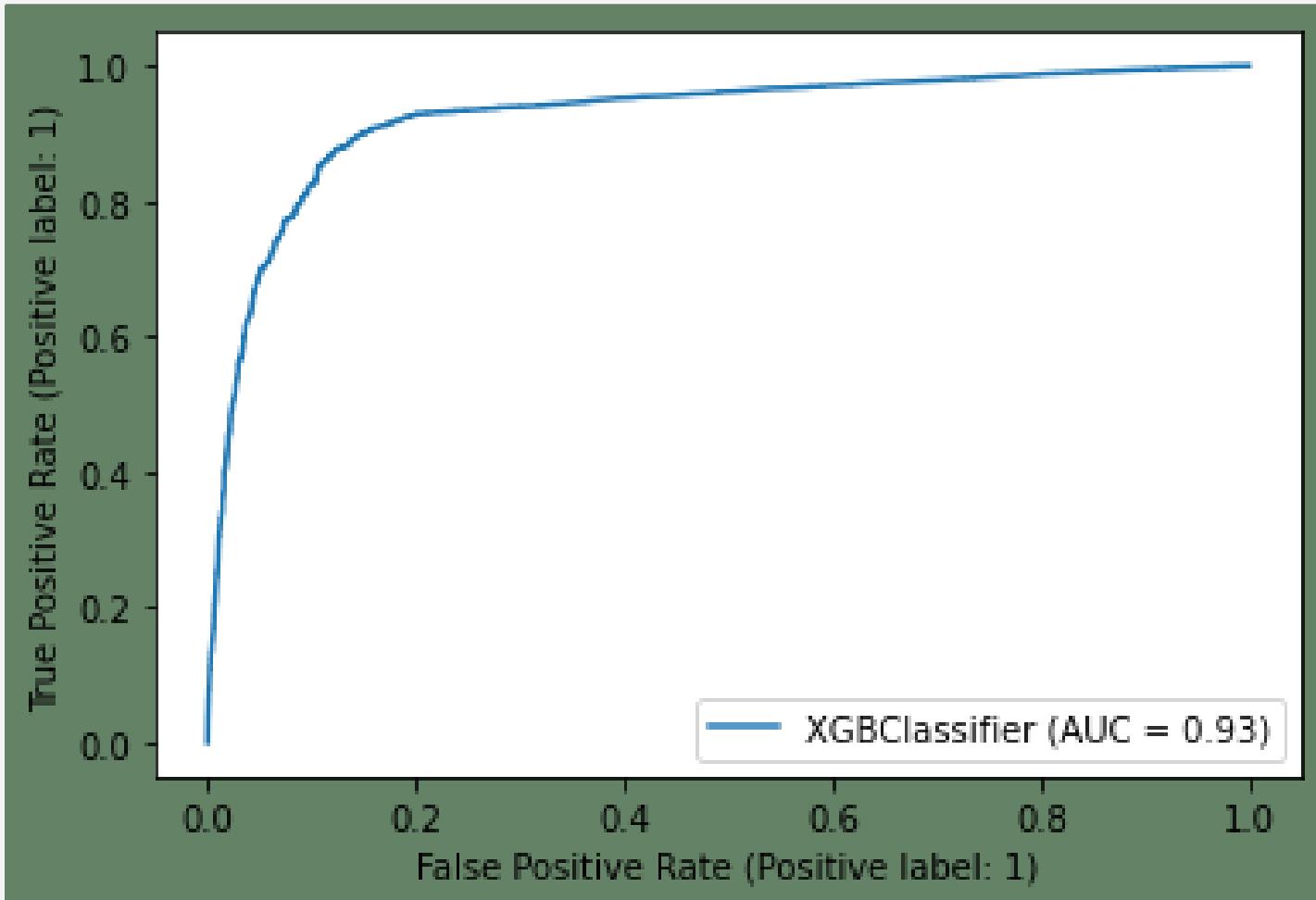
MODEL 3A: XGBOOST



tanpa Hyperparameter Tuning

```
3 from xgboost.sklearn import XGBClassifier
4
5 xgb = XGBClassifier()
6 xgb.fit(X_train,y_train)
7
8 from sklearn.metrics import classification_report
9 print(classification_report(y_test,xgb.predict(X_test)))
10
11 from sklearn.metrics import roc_auc_score
12 print(roc_auc_score(y_test,xgb.predict(X_test)))
13
14 from sklearn.metrics import plot_roc_curve
15 plot_roc_curve(xgb, X_test, y_test)
16 plt.show()
```

Rata-Rata setelah 4x Running
roc_auc_score : 0.8725195247



	PRECISION	RECALL	F1-SCORE	SUPPORT
0	0.90	0.84	0.87	10969
1	0.85	0.91	0.88	10969
ACCURACY				0.87
MACRO AVG	0.88	0.87	0.87	21938
WEIGTHED AVG	0.88	0.87	0.87	21938

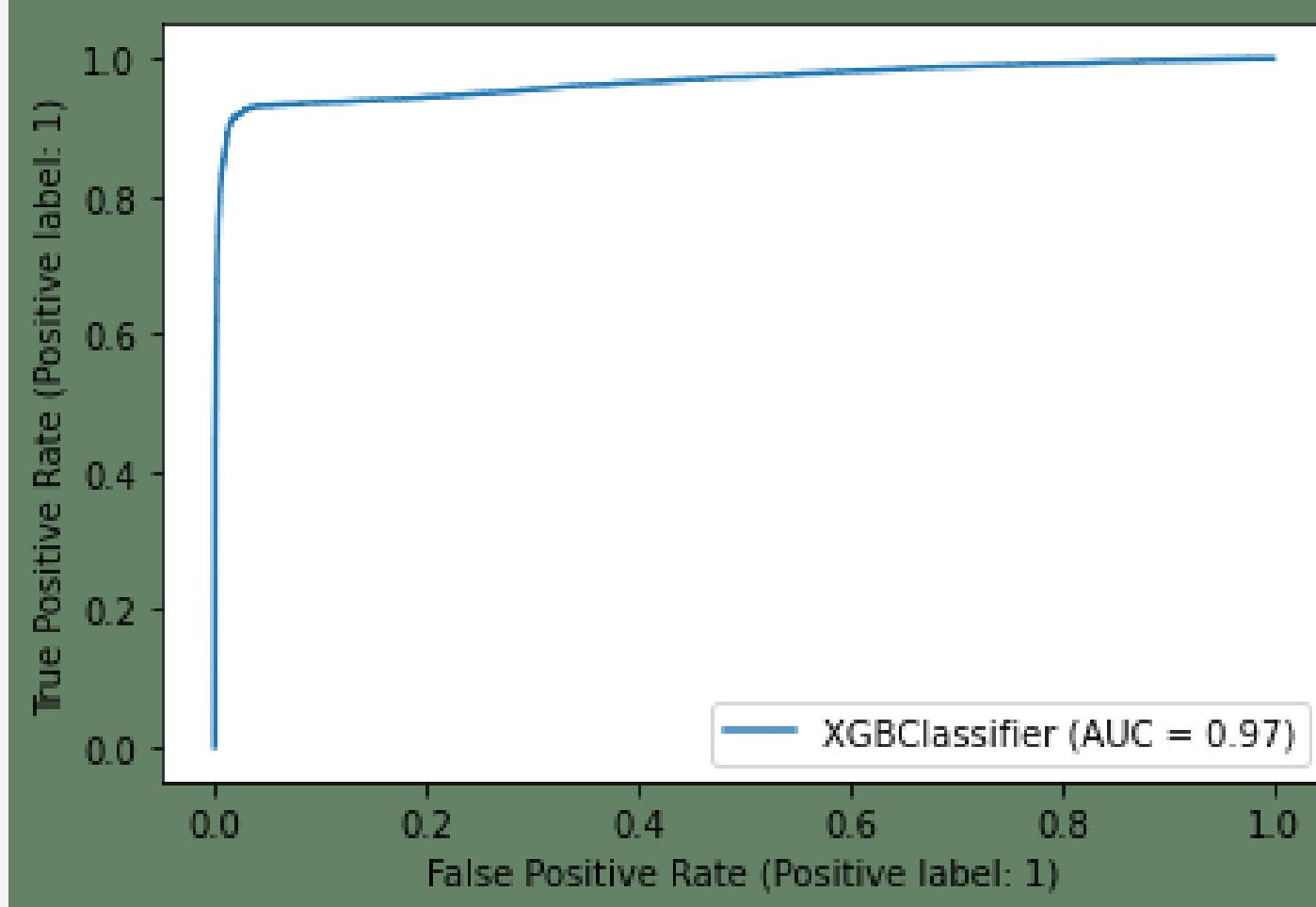
MODEL 3B : XGBOOST



dengan Hyperparameter Tuning GridSearch

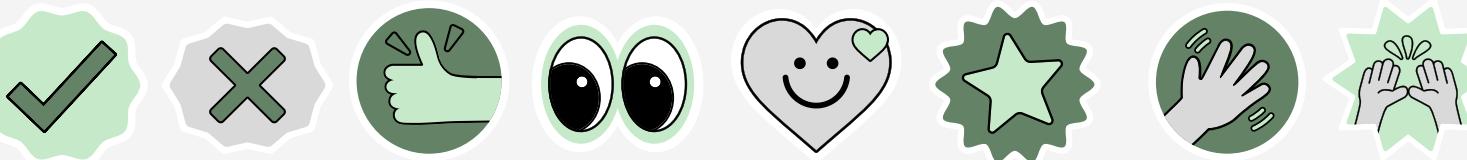
```
2 from sklearn.model_selection import GridSearchCV
3
4 ratio = float(np.sum(y_train == 0)/np.sum(y_train == 1))
5 param_grid = {
6     'max_depth': range(4,10),
7 }
8 grid = GridSearchCV(XGBClassifier(scale_pos_weight = ratio),
9                     cv = 5, scoring='roc_auc', param_grid = param_grid)
10 grid.fit(X_train, y_train)
11 xgb_grid = grid.best_estimator_
12
13 print(roc_auc_score(y_test, xgb_grid.predict(X_test)))
14
15 print("Best:%s "%(grid.best_params_))
16
17 print(classification_report(y_test,xgb_grid.predict(X_test)))
18
19 from sklearn.metrics import plot_roc_curve
20 plot_roc_curve(xgb_grid, X_test, y_test)
21 plt.show()
```

Rata-Rata setelah 4x Running
roc_auc_score : 0.9415170025



	PRECISION	RECALL	F1-SCORE	SUPPORT
0	0.93	0.95	0.94	10969
1	0.95	0.93	0.94	10969
ACCURACY				0.94
MACRO AVG	0.94	0.94	0.94	21938
WEIGTHED AVG	0.94	0.94	0.94	21938

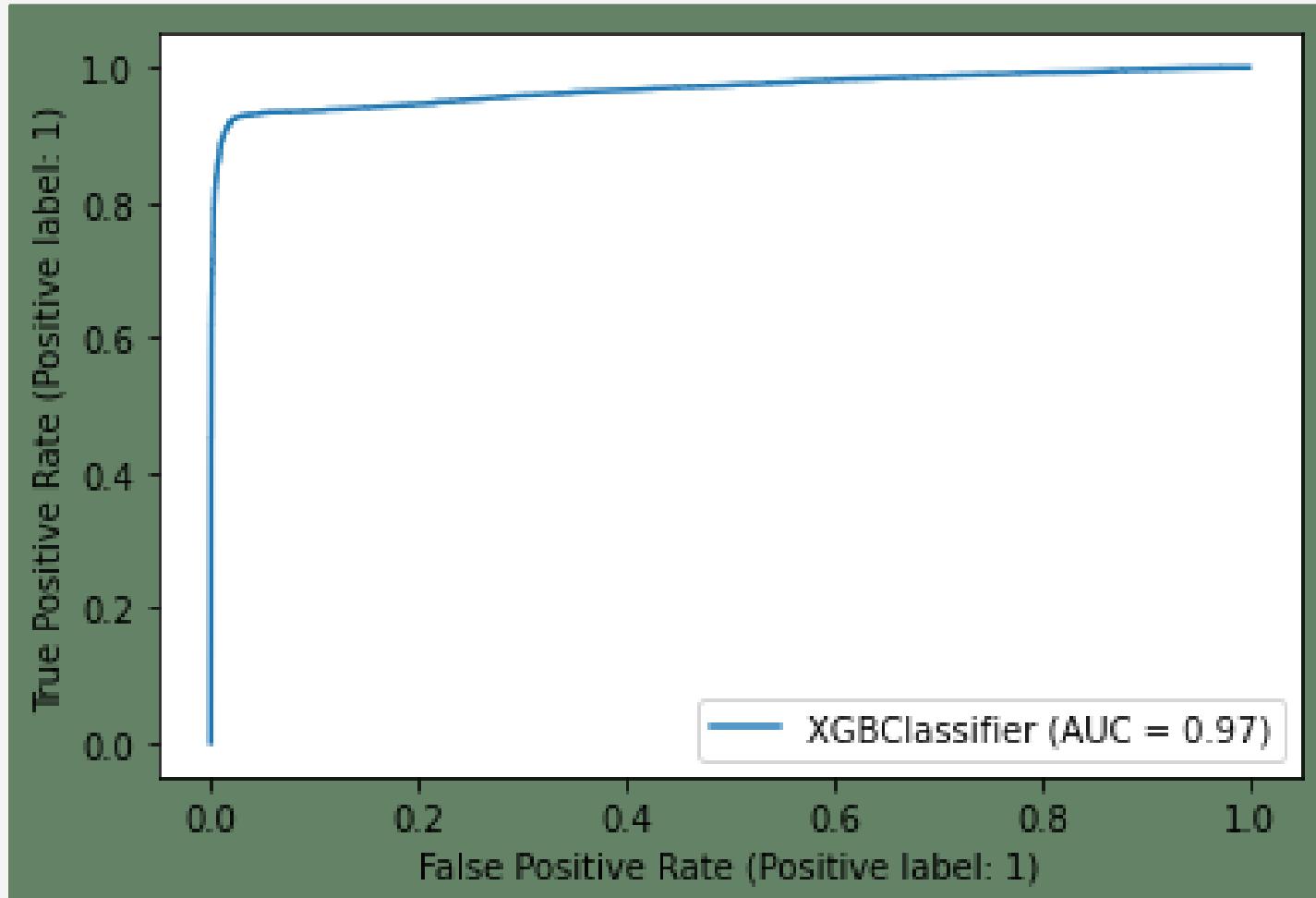
MODEL 3C : XGBOOST



dengan Hyperparameter Tuning BayesSearch

```
2 from skopt import BayesSearchCV
3
4 ratio = float(np.sum(y_train == 0)/np.sum(y_train == 1))
5 params = {
6     'max_depth':[3,4,5,6,7,8,9,10],
7     'min_child_weight': [4,5,6,7,8,9,10],
8     'gamma':[0.5,1,1.5,2,5],
9     'eta':[0.1,0.2],
10    'subsample': [0.5,0.6,0.7,0.8,0.9,1],
11    'colsample_bytree': [0.5,0.6,0.7,0.8,0.9,1],
12 }
13 bayes = BayesSearchCV[XGBClassifier(scale_pos_weight = ratio),
14                         search_spaces = params, cv = 5, n_iter = 10]
15 bayes.fit(X_train,y_train)
16 xgb_bayes = bayes.best_estimator_
17 print(roc_auc_score(y_test, xgb_bayes.predict(X_test)))
18
19 print("Best:%s "%(bayes.best_params_))
20
21 print(classification_report(y_test, xgb_bayes.predict(X_test)))
22
23 from sklearn.metrics import plot_roc_curve
24 plot_roc_curve(xgb_bayes, X_test, y_test)
25 plt.show()
```

Rata-Rata setelah 4x Running
roc_auc_score : 0.9438569301



	PRECISION	RECALL	F1-SCORE	SUPPORT
0	0.93	0.96	0.94	10969
1	0.95	0.93	0.94	10969
ACCURACY				0.94
MACRO AVG	0.94	0.94	0.94	21938
WEIGTHED AVG	0.94	0.94	0.94	21938

PERBANDINGAN MODEL

Catatan : CV(5) , n_iter(10) , GS (Grid Search) , BS(BayesSearch)

	AUC	ACCURACY	SENSITIVITY	SPECIFICITY	TIME
ADABOOST	0.90	0.84	0.77	0.80	3.419s
ADABOOST GS	0.91	0.85	0.80	0.82	208.967s
DTREE	0.93	0.93	0.91	0.91	0.034s
DTREE GS	0.92	0.88	0.86	0.86	2.169s
XGBOOST	0.93	0.87	0.84	0.85	0.012s
XGBOOST GS	0.93	0.94	0.95	0.95	269.996s
XGBOOST BS	0.97	0.94	0.96	0.95	385.14s

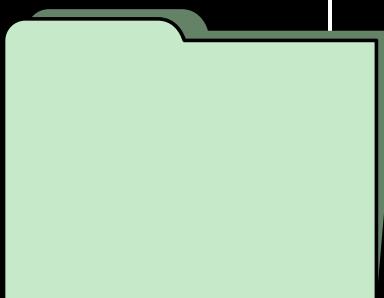
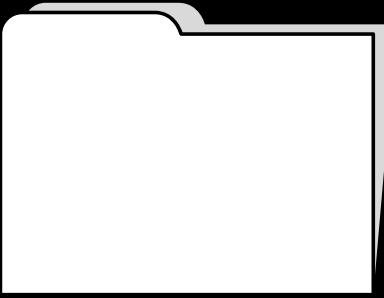
PERBANDINGAN MODEL

Hyperparameter Tuning

Catatan : CV(5) , n_iter(10) , GS (Grid Search) , BS(BayesSearch)

	AUC	ACCURACY	SENSITIVITY	SPECIFICITY	TIME
ADABOOST GS	0.91	0.85	0.80	0.82	208.967s
DTREE GS	0.92	0.88	0.86	0.86	2.169s
XGBOOST GS	0.93	0.94	0.95	0.95	269.996s
XGBOOST BS	0.97	0.94	0.96	0.95	385.14s

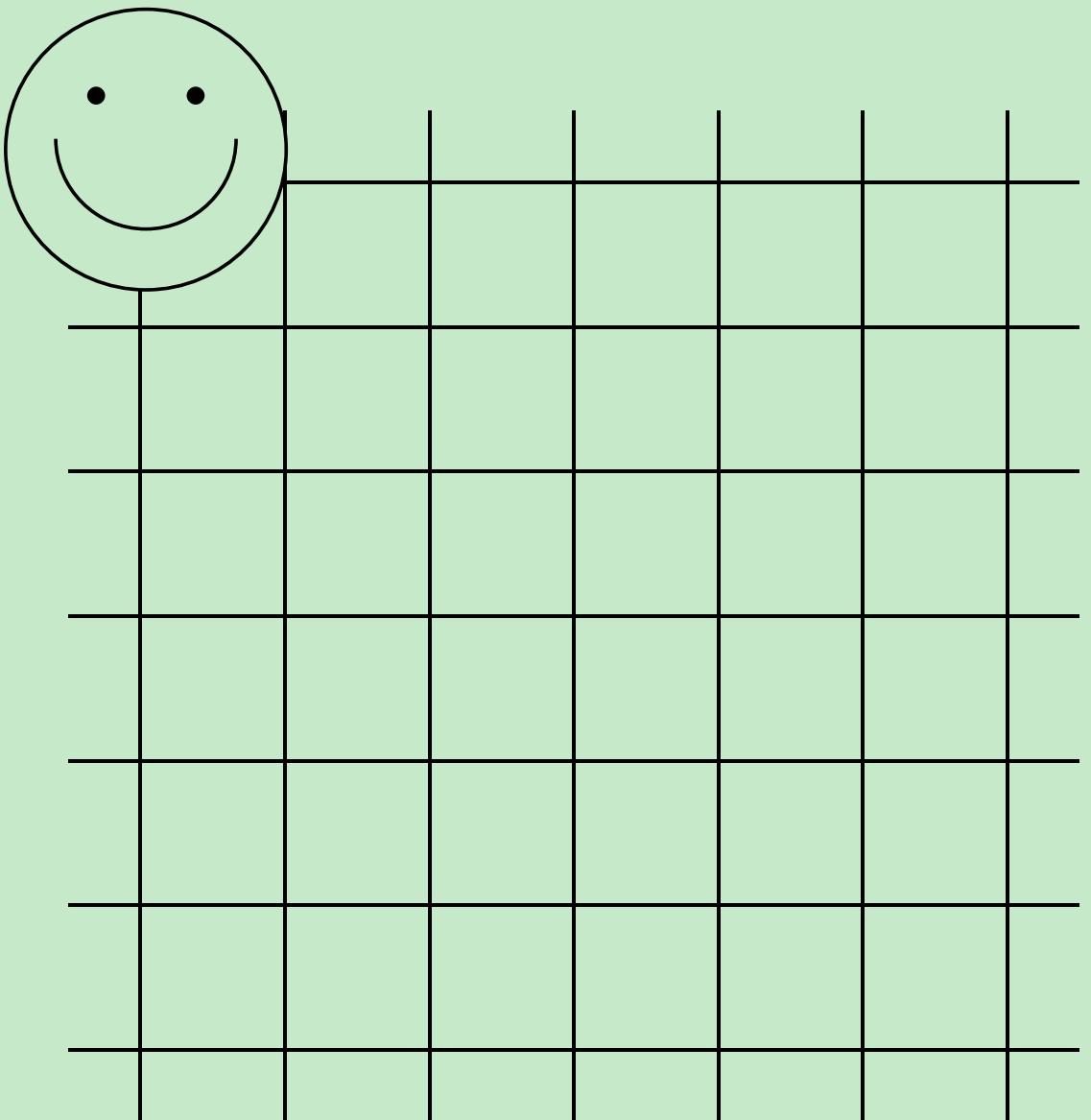
SUMMARY



Model yang terbaik dalam memprediksi klaim asuransi mobil adalah metode **XGBoost dengan Hyperparameter Tuning menggunakan Bayesian Search Cross Validation** karena memberikan hasil terbaik dibandingkan Model AdaBoost dan Decision Tree, tetapi memiliki kekurangan, yaitu Running Time yang paling lama.

SARAN

- Parameter yang di-*tuning* bisa lebih banyak dan bervariasi.
- Melakukan balancing data dengan metode selain SMOTEN.
- Melakukan uji coba dengan model selain Adaboost, XGBOOST, dan DTree.
- Mencoba kombinasi pemisahan data training dan data testing .

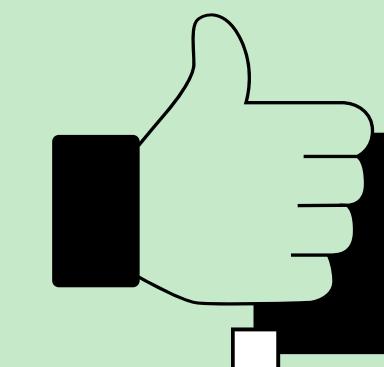


Tugas Akhir Ilmu Data

Desember 20. 2022

TERIMA

KASIH



Kelompok 12



Muhammad Iqbal AL-Fatih 2006571311
Emmanuel Justin Heumasse 2006571500
Gabriel Rico Fortino 2006571236
Muhammad Adli Rahmat Solihin 2006529184