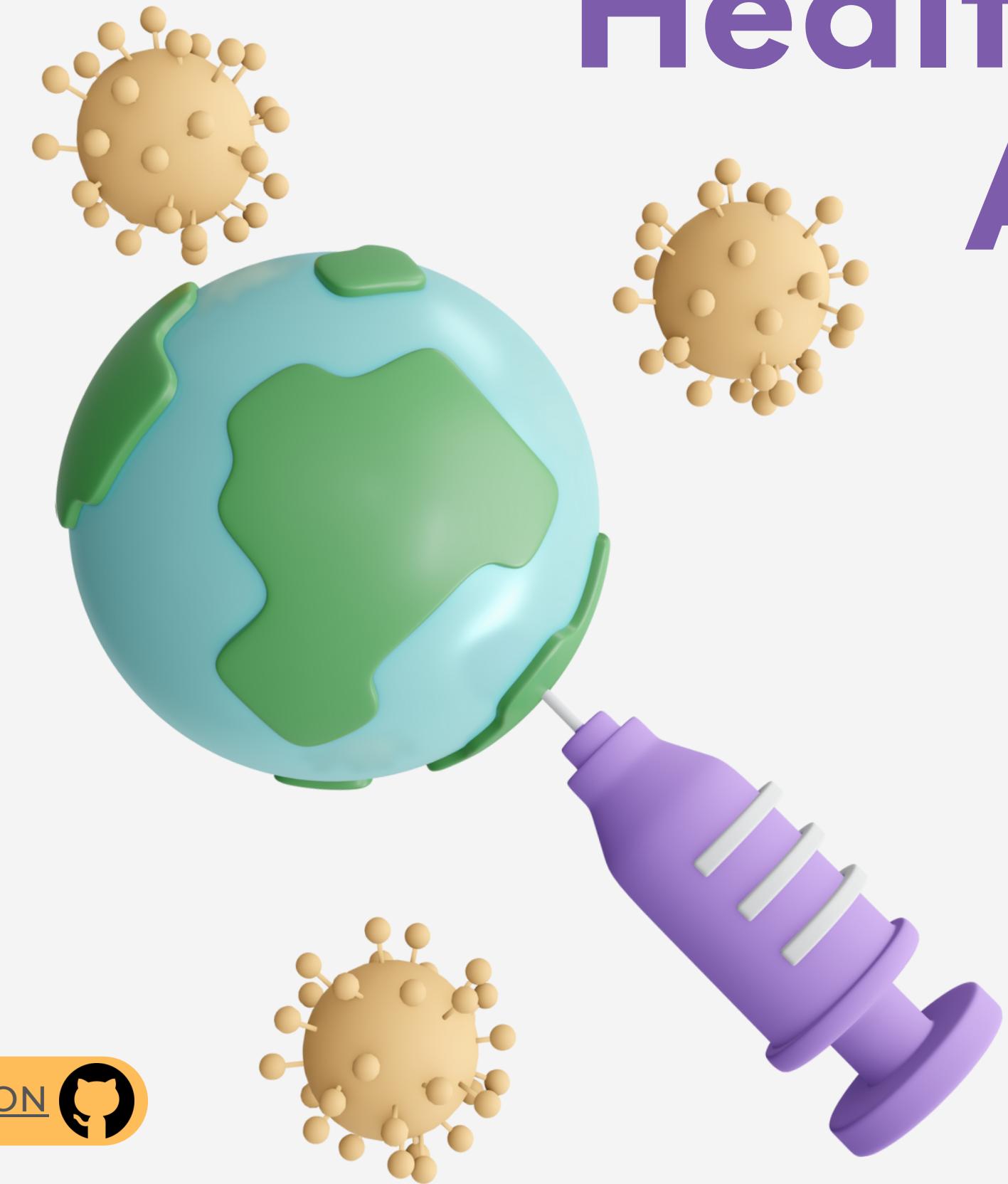


Healthcare Fraud Analysis and Prediction



SOURCE CODE ON 

Created by



Garry Louis Giancarlo
Jr Data Scientist

Content Outline

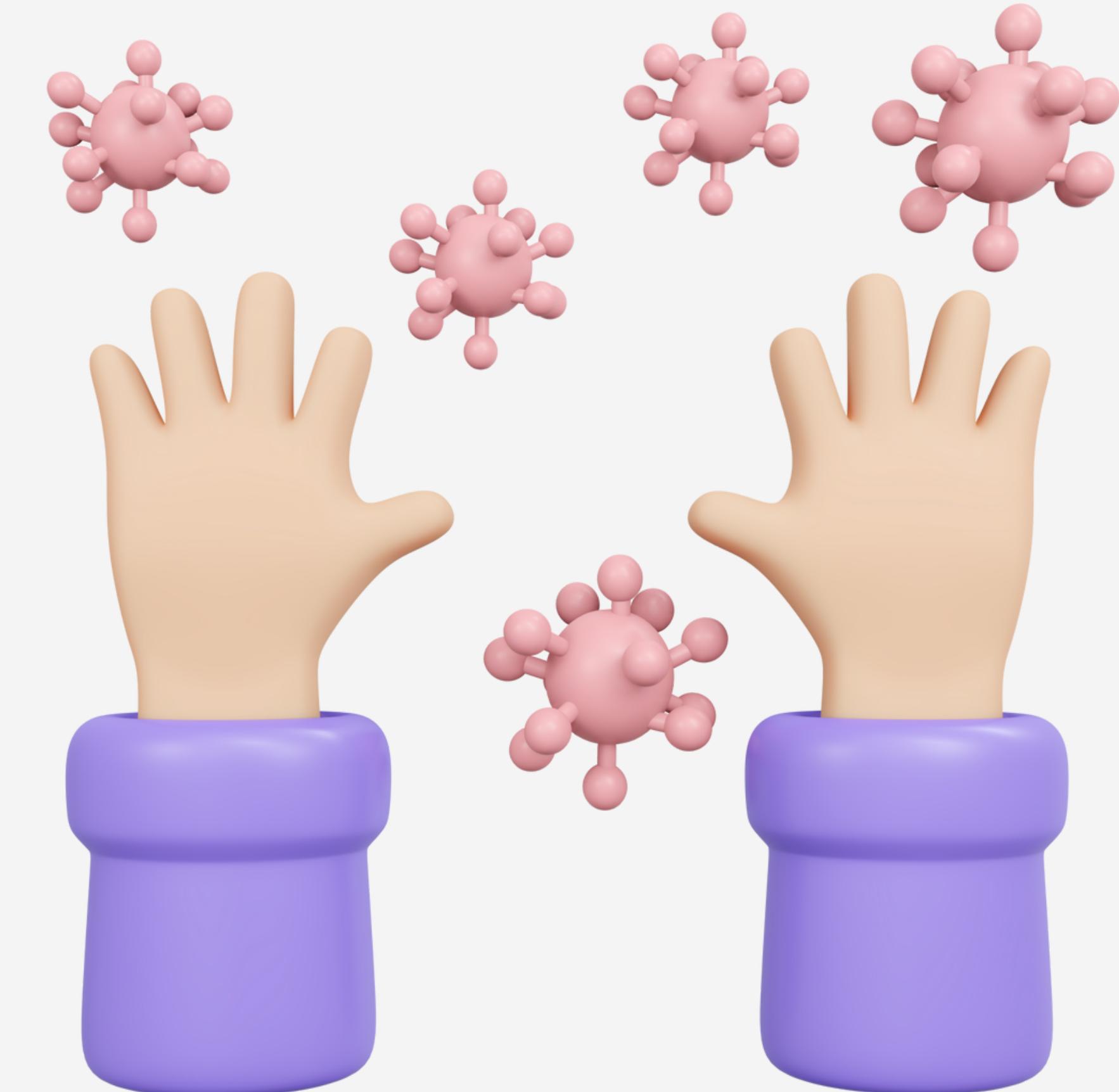
Topics for discussion

01 Business and Data Understanding

02 Preprocessing

03 Modelling and Evaluation

04 Recommendation





Business Understanding

Fraud Claim pada instansi kesehatan
adalah masalah yang sangat serius

Analisis dan Prediksi Penipuan Layanan Kesehatan:
Menerapkan algoritme **machine learning** untuk mengurangi
kerugian finansial dan menjaga integritas perawatan pasien.
Melalui analisis data perawatan kesehatan yang kompleks,
tujuannya adalah mengidentifikasi pola mencurigakan guna
deteksi penipuan. Dengan model deteksi fraud, pelaku
kesehatan, asuransi, dan badan pengawas dapat mencegah
penipuan dan memperkuat kepercayaan dalam ekosistem
layanan kesehatan.

KERUGIAN AKIBAT DARI FRAUD CLAIM PADA HEALTHCARE

periode 2008/2009

Rp2,4M

DENGAN LOSS RATIO SEBESAR

43.5%

GOALS



Mengurangi Loss
Ratios dari 43.5%
menjadi 15%

OBJECTIVES



Membuat machine learning
model yang dapat
memprediksi potensi fraud
berdasarkan pola-pola
yang tercatat pada record
claim

METRICS



Loss Ratio

Data Understanding

Empat Tabel base data awal:

1. Tabel Inpatients: Data Pasien Rawat Inap
2. Tabel Outpatients: Data Pasien Rawat Jalan
3. Tabel Beneficiary: Data Penerima Manfaat
4. Tabel Target: Data Provider dan label potensi Fraud

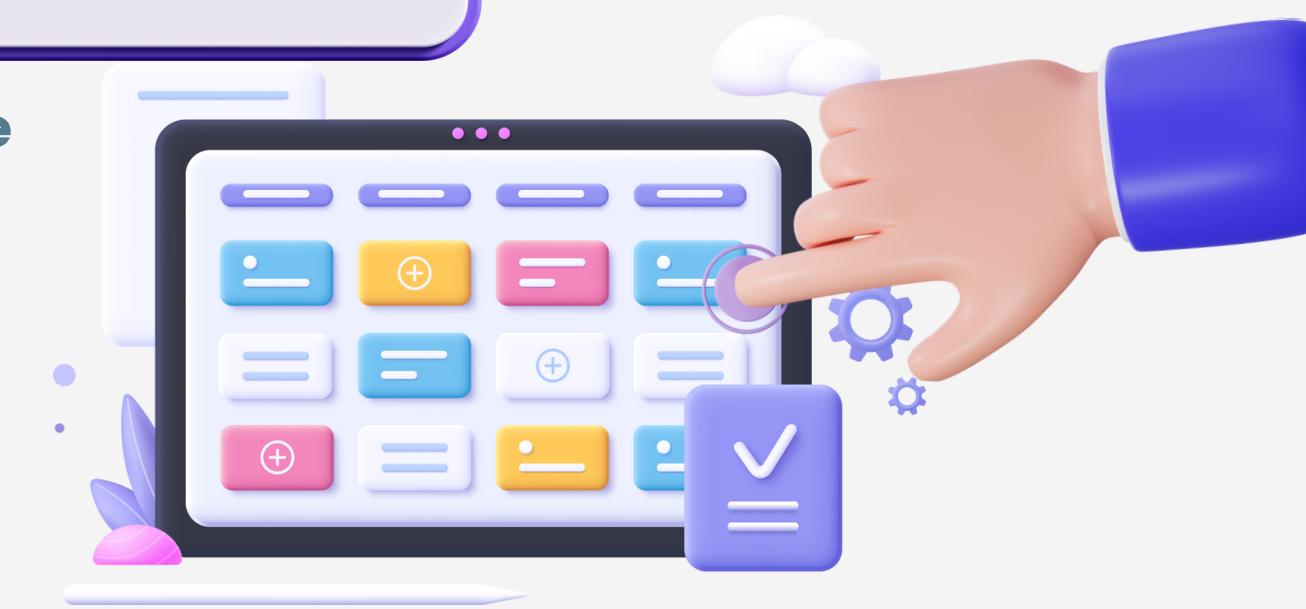
```
# Feature Engineering: Membuat kolom baru "admitted"  
data_inp[ 'admitted' ] = 1  
data_outp[ 'admitted' ] = 0  
# Menggabungkan semua dataset  
data_raw = pd.concat([data_outp, data_inp], ignore_index=True)  
data_raw = pd.merge(data_raw, data_benef, on='BeneID')  
data_raw = pd.merge(data_raw, data_label, on='Provider')
```

Shape(Row, Feature):
1. Tabel Inpatients(40474, 30)
2. Tabel Outpatients(517737, 30)
3. Tabel Beneficiary(138556, 25)
4. Tabel Target(5410, 2)



Shape(Row, Feature):
data_raw(558211,56)

Merge keempat tabel agar mendapatkan keseluruhan profile data dan mempermudah melakukan analisis dan data mining



Data Understanding

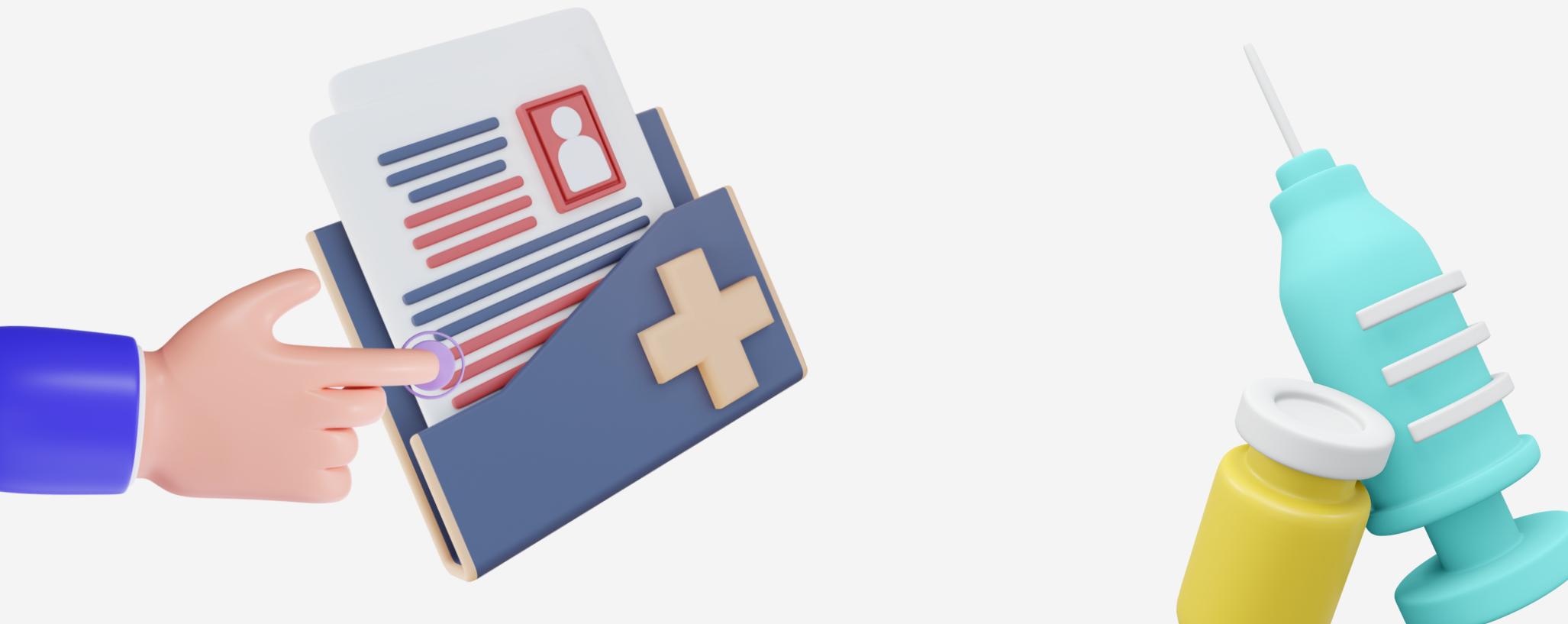
Sebelum melanjutkan tahap Exploratory untuk mendapatkan data understanding, perlu dilakukan feature engineering

Data type changes

Feature **ClaimStartDt**, **ClaimEndDt**, **AdmissionDt**, **DischargeDt**, **DOB**, **DOD** diubah menjadi format datetime agar bisa mengekstrak/ membuat feature baru yang berduga bagi analisis

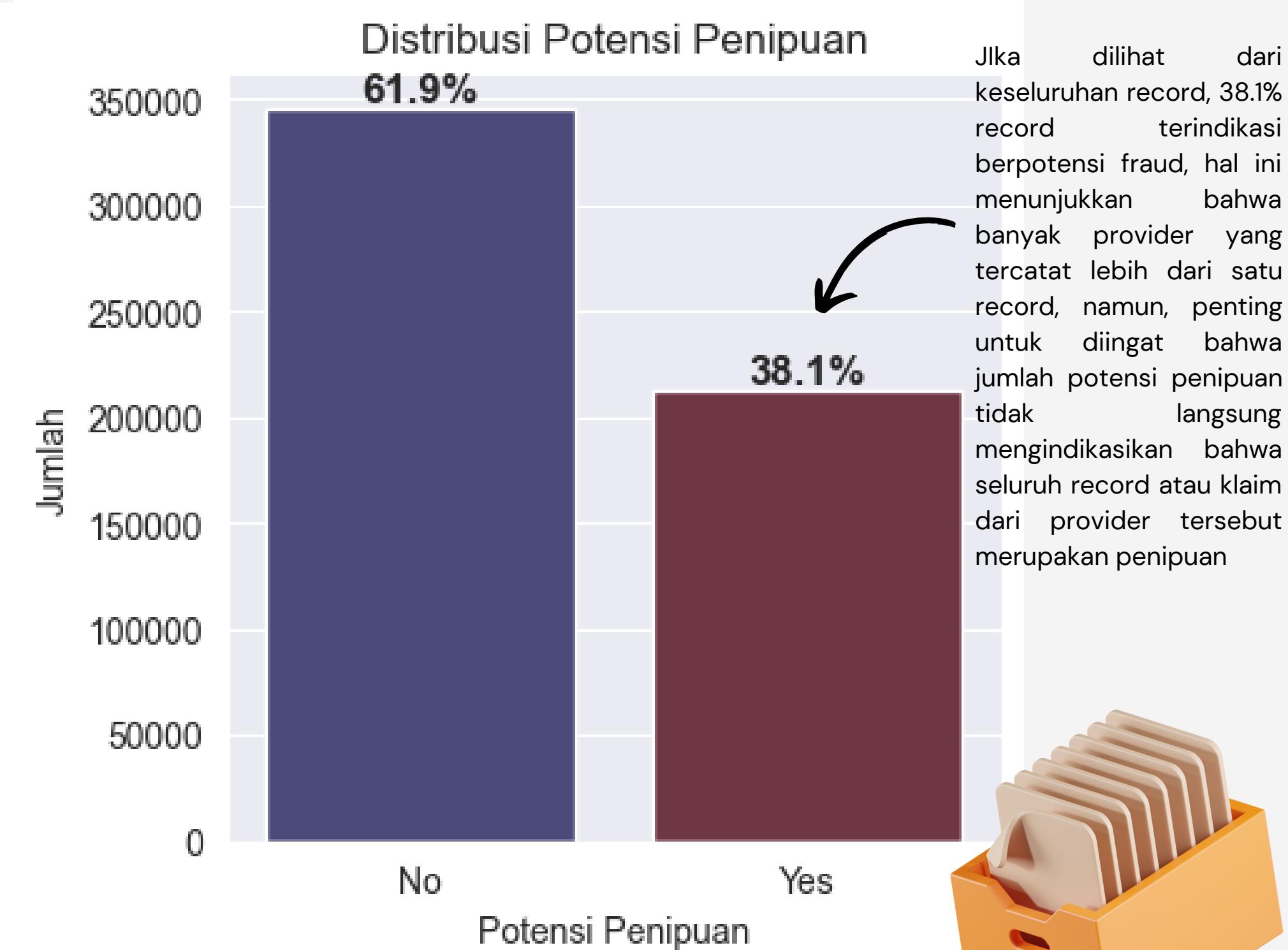
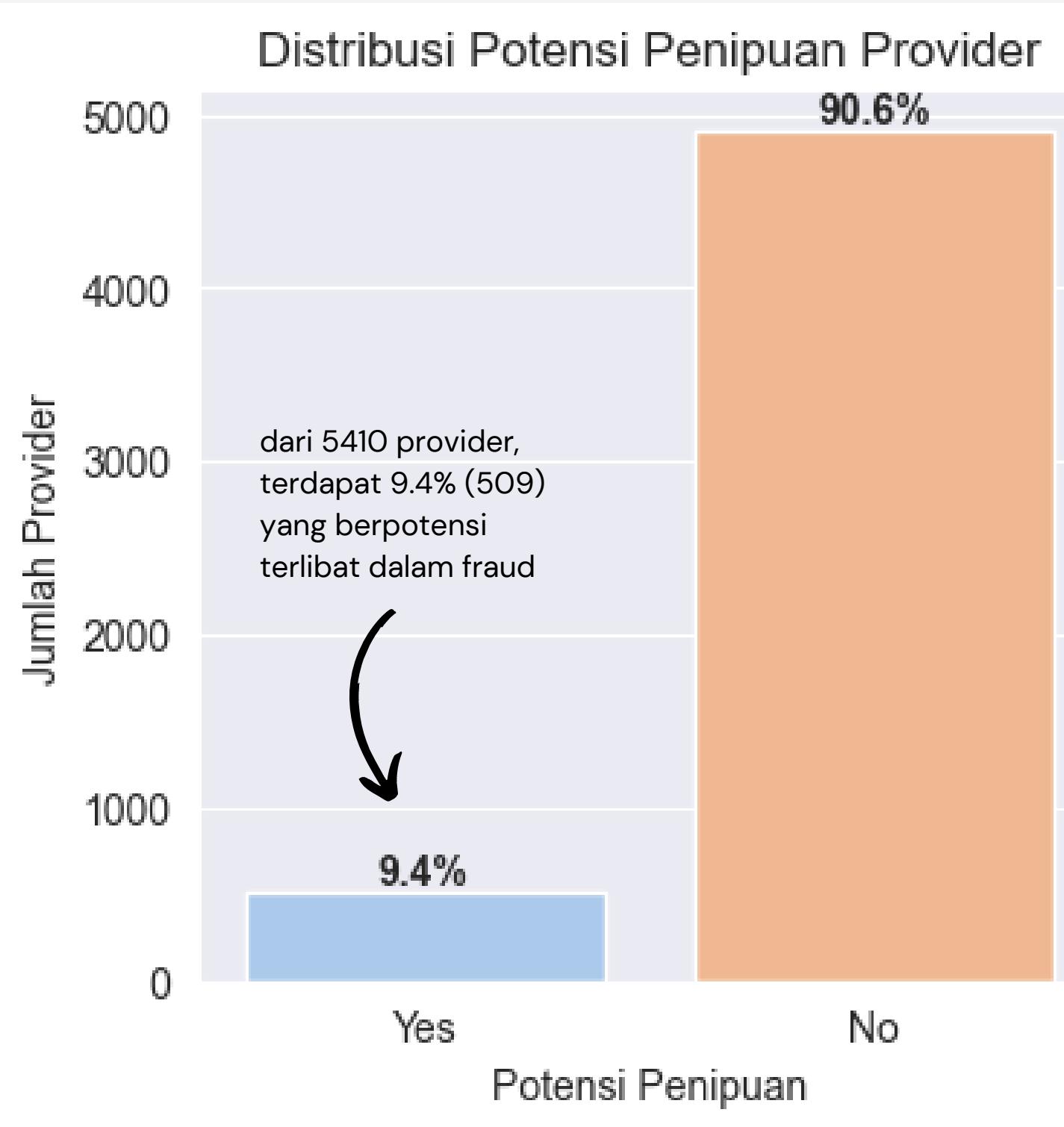
Membuat Feature baru

- Age
- IsAlive
- AgeGroup
- ClaimDuration
- AdmmissionDuration
- TotalDiagnoses
- TotalProcedures
- TotalReimbursement
- TotalCharge
- AvgChargePerDiagnosis

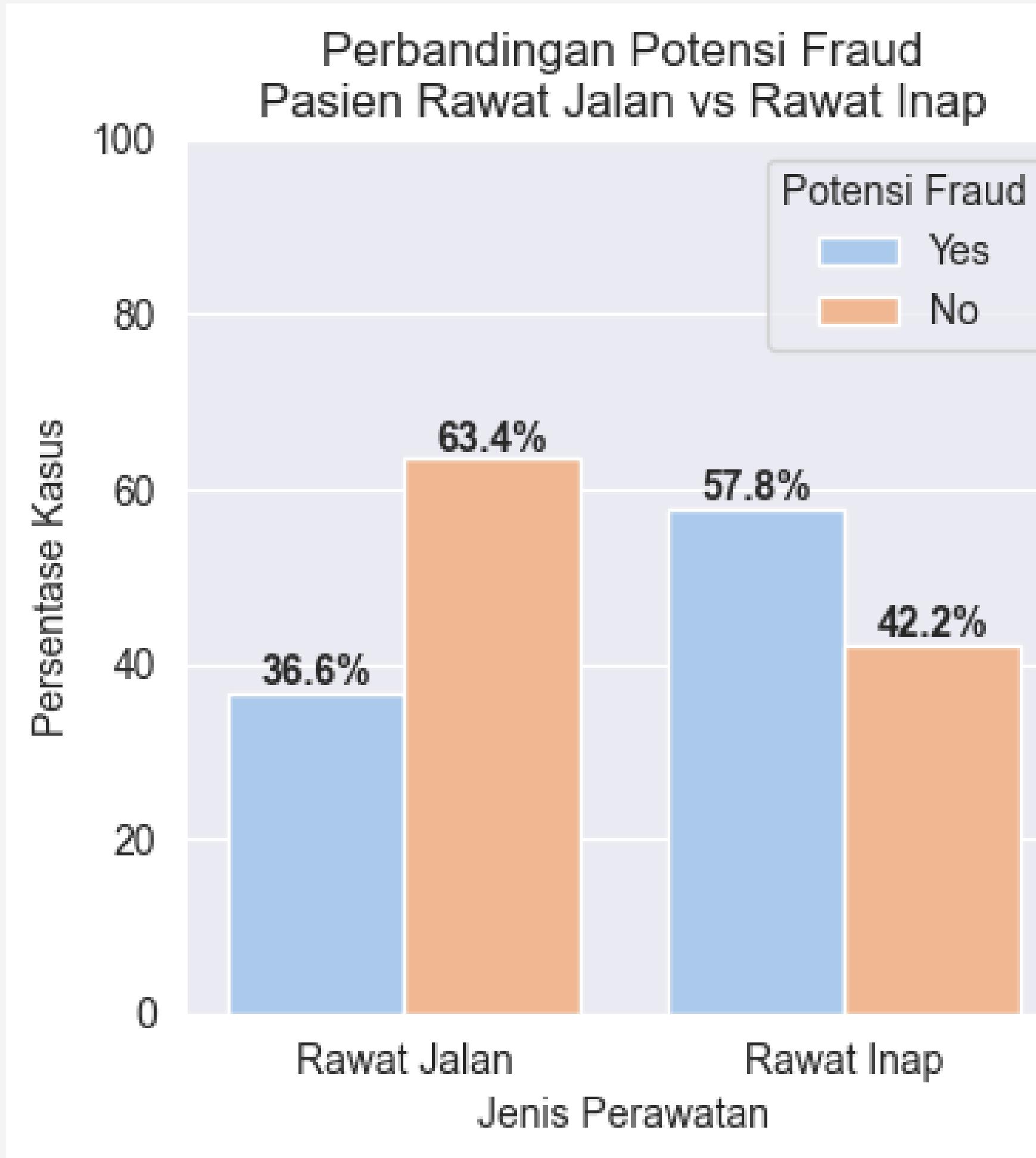


Exploratory Data

Beberapa insight utama yang bisa diperoleh untuk memberikan gambaran kondisi bisnis



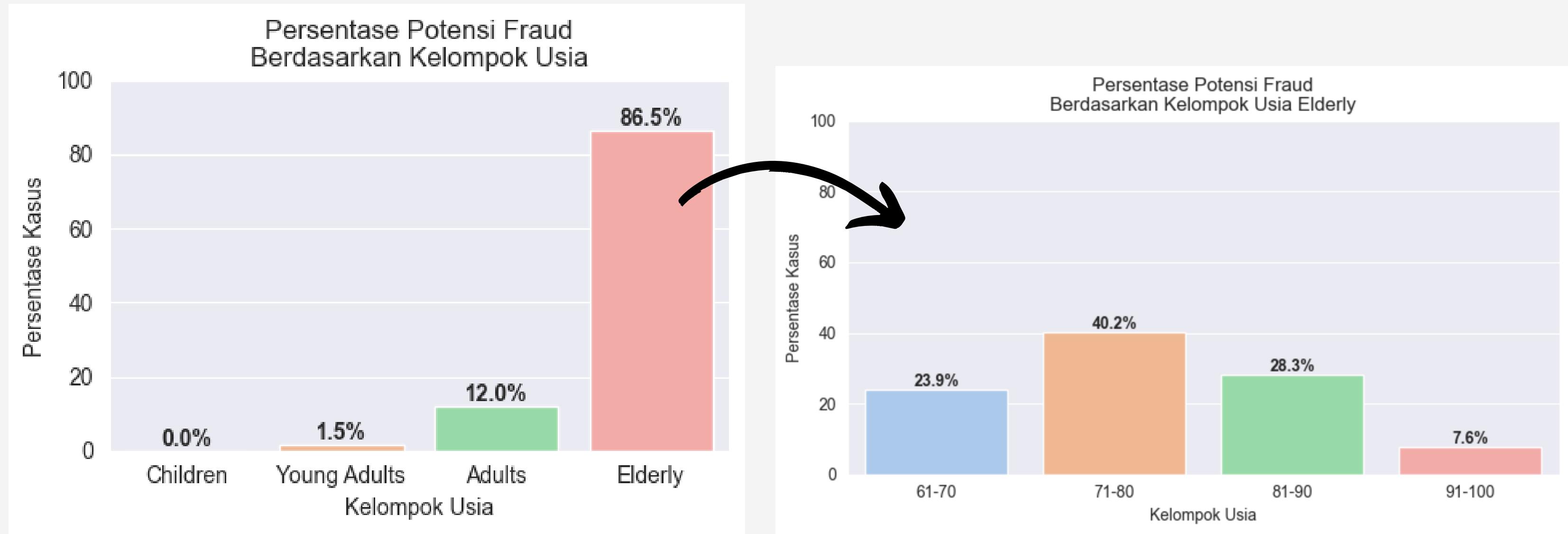
Exploratory Data



Terlihat bahwa persentase penipuan pada **pasien rawat inap** (**57.8% Yes**) cenderung lebih tinggi dibandingkan dengan pasien **rawat jalan** (**36.6% Yes**). Hal ini menunjukkan adanya potensi fraud yang lebih tinggi pada pasien yang menjalani rawat inap. Pasien rawat inap cenderung memiliki **biaya pengobatan yang lebih tinggi** dan kompleks, sehingga menarik perhatian pelaku fraud untuk melakukan penipuan yang mengakibatkan persentase penipuan yang lebih tinggi pada pasien rawat inap.



Exploratory Data



Insight yang ditemukan adalah potensi fraud cenderung lebih tinggi pada kelompok usia "Elderly" (>60 tahun) dengan persentase 86.5%. Kelompok usia "Adults" memiliki persentase 12.0% potensi fraud, sedangkan kelompok usia "Young-Adults" memiliki persentase 1.5% potensi fraud. Dalam dataset ini, tidak ada pasien di bawah 18 tahun yang terdeteksi memiliki potensi fraud. Hal ini menunjukkan bahwa pengawasan dan pemantauan yang ketat diperlukan terutama pada kelompok usia "Elderly" untuk mencegah potensi fraud.



Data Cleaning and Preprocessing

Missing Value & Feature Selection

```
# Drop Kolom Code...
code_columns = [col for col in data_pre2.columns if 'Code' in col]
data_pre2.drop(columns=code_columns, inplace=True)
```

```
cols_to_drop = [
    # identifikasi unik
    'BeneID',
    'ClaimID',
    # datetime
    'ClaimStartDt',
    'ClaimEndDt',
    'DOB',
    'DOD',
    'AdmissionDt',
    'DischargeDt',
    # Tidak Relevan
    'OperatingPhysician',
    'OtherPhysician',
    'County',
    'State',
    'AvgChargePerDiagnosis'
]
```

Drop features yang mengandung 'code' karena punya banyak missing value

Drop juga features yang **tidak relevan** untuk machine learning seperti **feature ID, feature datetime** karena sudah di ekstrak menjadi feature lain, feature seperti **OperatingPhysician, County** karena **high cardinality**

Feature	Percentage Missing Value
ClmProcedureCode_6	100
ClmProcedureCode_5	99.9
ClmProcedureCode_4	99.9
ClmProcedureCode_3	99.8
DOD	99.3
ClmDiagnosisCode_10	99.1
ClmProcedureCode_2	99
ClmProcedureCode_1	95.8
AdmissionDuration	92.7
AdmissionDt	92.7
DischargeDt	92.7
DiagnosisGroupCode	92.7
ClmDiagnosisCode_9	92.5

Feature	Percentage Missing Value
ClmDiagnosisCode_8	90.4
ClmDiagnosisCode_7	88.1
ClmDiagnosisCode_6	84.9
ClmDiagnosisCode_5	79.9
OperatingPhysician	79.4
ClmAdmitDiagnosisCode	73.9
ClmDiagnosisCode_4	70.5
OtherPhysician	64.2
ClmDiagnosisCode_3	56.5
ClmDiagnosisCode_2	35
ClmDiagnosisCode_1	1.9
AttendingPhysician	0.3
DeductibleAmtPaid	0.2
AvgChargePerDiagnosis	0.05

Feature lainnya yang memiliki mising value, diisi dengan nilai 0, hal tersebut agar model bisa mengenali pola lebih baik jika dibandingkan diisi dengan nilai statistika(mean,median)

```
data_pre2['AdmissionDuration'].fillna(0, inplace=True)
data_pre2['AttendingPhysician'].fillna(0, inplace=True)
data_pre2['DeductibleAmtPaid'].fillna(0, inplace=True)
```

Data Cleaning and Preprocessing

Mapping

Agar value feature berupa value binary (0 dan 1)



```
# Mapping 'PotentialFraud' menjadi 'Yes': 1, 'No': 0
data_pre3['PotentialFraud'] = data_pre3['PotentialFraud'].map({'Yes': 1, 'No': 0})
# Mapping 'RenalDiseaseIndicator' menjadi 'Y': 1
data_pre3['RenalDiseaseIndicator'] = data_pre3['RenalDiseaseIndicator'].replace('Y', 1)
# Daftar kolom 'ChronicCond_...'
chronic_conditions = ['ChronicCond_Alzheimer', 'ChronicCond_Heartfailure', 'ChronicCond_KidneyDisease',
                      'ChronicCond_Cancer', 'ChronicCond_ObstrPulmonary', 'ChronicCond_Depression',
                      'ChronicCond_Diabetes', 'ChronicCond_IschemicHeart', 'ChronicCond_Osteoporasis',
                      'ChronicCond_rheumatoidarthritis', 'ChronicCond_stroke']
data_pre3[chronic_conditions] = data_pre3[chronic_conditions].replace(2, 0)
```

Feature yang di mapping:

'ChronicCond_Alzheimer', 'ChronicCond_Heartfailure', 'ChronicCond_KidneyDisease', 'ChronicCond_Cancer',
"ChronicCond_ObstrPulmonary", 'ChronicCond_Depression', 'ChronicCond_Diabetes', 'ChronicCond_IschemicHeart',
'ChronicCond_Osteoporasis', 'ChronicCond_rheumatoidarthritis', 'ChronicCond_stroke', 'PotentialFraud', 'RenalDiseaseIndicator'

Data Cleaning and Preprocessing Encoding

One Hot Encoding

Feature Gender dan Race dilakukan one-hot encoding agar dapat direpresentasikan sebagai feature binary.

```
● ● ●  
# One-Hot Encoding  
to_onehot = ['Race', 'Gender']  
for onehot in to_onehot:  
    onehots = pd.get_dummies(data_pre3[onehot], prefix=onehot)  
    data_pre3 = data_pre3.join(onehots)  
# Drop feature yang sudah di encode  
data_pre3 = data_pre3.drop(columns=to_onehot).copy()
```



Target Encoding

Feature "Provider" dan "AttendingPhysician" dipertahankan karena keduanya memberikan **informasi relevan dalam analisis fraud**. Dengan menggunakan target encoding, model dapat mempelajari **hubungan antara dokter yang merawat dengan kasus fraud**, meningkatkan kemampuan model dalam mendeteksi pola dan karakteristik fraud yang melibatkan dokter-dokter tertentu. Misalnya, jika seorang dokter terlibat dalam **10% kasus fraud**, nilai pada kolom "AttendingPhysician" untuk dokter tersebut akan diganti dengan 0.1.

```
● ● ●  
# Menghitung rasio kasus fraud berdasarkan Provider  
provider_fraud_ratio = data_pre3.groupby('Provider')['PotentialFraud'].mean()  
# Menghitung rasio kasus fraud berdasarkan AttendingPhysician  
physician_fraud_ratio = data_pre3.groupby('AttendingPhysician')['PotentialFraud'].mean()  
# Mengganti nilai pada kolom Provider dengan rasio kasus fraud  
data_pre3['Provider_TargetEncoded'] = data_pre3['Provider'].map(provider_fraud_ratio)  
# Mengganti nilai pada kolom AttendingPhysician dengan rasio kasus fraud  
data_pre3['AttendingPhysician_TargetEncoded'] =  
    data_pre3['AttendingPhysician'].map(physician_fraud_ratio)  
# Drop kolom yang sudah di encode  
to_drop_2 = ['Provider', 'AttendingPhysician']  
data_pre4 = data_pre3.drop(to_drop_2, axis=1)
```

Data Cleaning and Preprocessing

Scaling, Train-Test Split & Oversampling

Train-Test Split

Train-Test Split dilakukan untuk membagi dataset menjadi dua bagian, dengan proporsi **Train 80:20 Test**. Tujuannya adalah untuk menggunakan data Train untuk melatih model ML dan data Test untuk menguji kinerja model secara independen dan mengukur kemampuan generalisasi model terhadap data baru.



```
from sklearn.model_selection import train_test_split  
X_train, X_test, y_train, y_test = train_test_split(  
    X, y, test_size=0.2,  
    random_state=42)
```

Scaling

dilakukan scaling menggunakan **MinMaxScaler** agar variabel numerik memiliki rentang nilai yang seragam antara **0 dan 1**, dapat membantu **model lebih robust**, mempermudah perbandingan dan **interpretasi** antar variabel dalam analisis fraud



```
from sklearn.preprocessing import MinMaxScaler  
scaler = MinMaxScaler()  
scaler.fit(data_pre4)  
data_scaled = pd.DataFrame(scaler.transform(data_pre4),  
                           columns= data_pre4.columns)
```

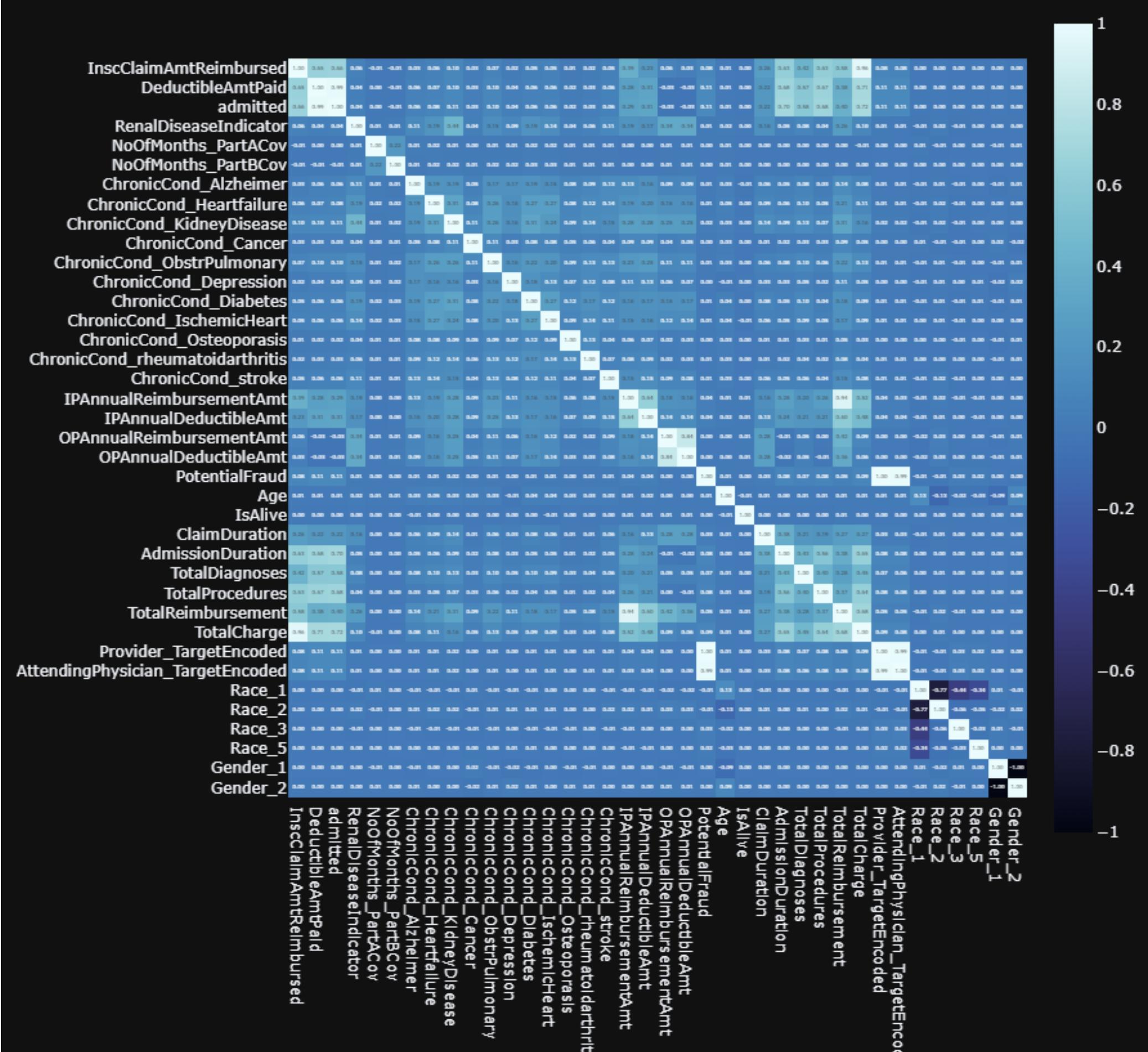
Train-Test Split

Oversampling dilakukan dengan metode **SMOTE** untuk mengatasi **class imbalance** dalam dataset. Dengan menciptakan **sampel sintetis** tambahan untuk **kelas minoritas** agar kelas tersebut memiliki **proporsi yang seimbang** dengan kelas mayoritas, agar kemampuan model untuk memprediksi kelas minoritas lebih baik.



```
from imblearn.over_sampling import SMOTE  
smote = SMOTE(random_state=42,  
              sampling_strategy='minority')  
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)
```

Multivariate Correlation Test dengan Heatmap



Pre-Model Testing

T-Test

Feature	T-statistik	p-value
Gender_1	-0.19	0.85
ChronicCond_Depression	-0.20	0.84
ChronicCond_Osteoporasis	-1.00	0.32
IsAlive	-1.07	0.28
NoOfMonths_PartBCov	-1.40	0.16
OPAnnualReimbursementAmt	-1.40	0.16
ChronicCond_Cancer	-2.78	0.01
IPAnnualDeductibleAmt	-27.24	0.00
Race_5	-13.07	0.00
Race_3	-21.43	0.00
...
ChronicCond_rheumatoidarthritis	-3.83	0.00

Uji-t dapat digunakan sebagai gambaran awal untuk **mengidentifikasi variabel-variabel yang memiliki pengaruh signifikan terhadap target atau variabel dependen**. Namun, penting untuk diingat bahwa uji-t sendiri tidak memberikan informasi tentang seberapa besar pengaruh variabel tersebut terhadap target. Hal tersebut bisa dijelaskan pada feature importance.

Pada tabel disamping, **T-statistik** adalah ukuran seberapa signifikan **perbedaan antara mean sample dan nilai yang diharapkan (biasanya 0)** dalam unit standar deviasi. **Semakin tinggi nilai T-statistik, semakin signifikan pengaruh variabel terhadap target**. Sedangkan, **p-Value** adalah probabilitas mendapatkan nilai T-statistik atau yang lebih ekstrem jika hipotesis nol benar. Nilai **p-Value yang lebih rendah menunjukkan bahwa pengaruh variabel terhadap target lebih signifikan**.

Fitur seperti **Gender_1, ChronicCond_Depression, ChronicCond_Osteoporasis, IsAlive, NoOfMonths_PartBCov, OPAnnualReimbursementAmt, ChronicCond_Cancer** yang memiliki p-value >0.05 dapat dikatakan memiliki pengaruh dengan tingkat signifikansi rendah terhadap variabel dependen, jika demikian bisa kita pilih untuk drop feature, namun untuk analisis ini, feature-feature tersebut akan dipertahankan.

Tree-Based Modeling

Decision Tree, Random Forest, Gradient Boosting, Extreme Gradient Boosting, Light Gradient Boosting

Kenapa Tree based Modeling:

Interpretasi yang Mudah

Penanganan data tidak teratur

Pengendalian overfitting

Robust terhadap outliers/missing

Fokus Metrics: Precision

Fokus pada precision karena mengutamakan tingkat akurasi dalam mengidentifikasi kasus fraud yang sebenarnya (true positives), sehingga dapat mengurangi angka false positives atau kesalahan dalam mengklasifikasikan kasus non-fraud sebagai fraud.

Modeling
and
Evaluation



Modeling

Hasil Skor dari Train dan Test 5 Tree-Based Model dengan Cross Validation Stratified K-Fold

DECISION TREE	RANDOM FOREST	GRADIENT BOOSTING
Accuracy (Train Set) : 0.563	Accuracy (Train Set) : 0.574	Accuracy (Train Set) : 0.686
Accuracy (Test Set) : 0.561	Accuracy (Test Set) : 0.574	Accuracy (Test Set) : 0.686
Precision (Train Set) : 0.737	Precision (Train Set) : 0.734	Precision (Train Set) : 0.878
Precision (Test Set) : 0.734	Precision (Test Set) : 0.733	Precision (Test Set) : 0.877
Recall (Train Set) : 0.221	Recall (Train Set) : 0.233	Recall (Train Set) : 0.432
Recall (Test Set) : 0.219	Recall (Test Set) : 0.233	Recall (Test Set) : 0.432
F1 (Train Set) : 0.321	F1 (Train Set) : 0.354	F1 (Train Set) : 0.579
F1 (Test Set) : 0.318	F1 (Test Set) : 0.353	F1 (Test Set) : 0.579
Roc_auc (Train Set) : 0.602	Roc_auc (Train Set) : 0.614	Roc_auc (Train Set) : 0.710
Roc_auc (Test Set) : 0.599	Roc_auc (Test Set) : 0.613	Roc_auc (Test Set) : 0.707
LGBBOOST	XGBOOST	Model XGBoost memiliki overall skor paling baik diantara model lainnya, dengan skor precision 88.4% dan tidak ada tanda overfitting, maka model inilah yang digunakan, untuk lebih optimal akan dilakukan tuning
Accuracy (Train Set) : 0.686	Accuracy (Train Set) : 0.696	
Accuracy (Test Set) : 0.686	Accuracy (Test Set) : 0.694	
Precision (Train Set) : 0.878	Precision (Train Set) : 0.886	
Precision (Test Set) : 0.877	Precision (Test Set) : 0.884	
Recall (Train Set) : 0.432	Recall (Train Set) : 0.449	
Recall (Test Set) : 0.432	Recall (Test Set) : 0.447	
F1 (Train Set) : 0.579	F1 (Train Set) : 0.596	
F1 (Test Set) : 0.579	F1 (Test Set) : 0.594	
Roc_auc (Train Set) : 0.710	Roc_auc (Train Set) : 0.728	
Roc_auc (Test Set) : 0.707	Roc_auc (Test Set) : 0.718	

Modeling Hyperparameter Tuning Model XGBoost

```
● ● ●  
# menentukan cross validation dengan Stratified K-Fold  
strat_k_fold = StratifiedKFold(n_splits=5,  
                               random_state=42,  
                               shuffle=True)  
  
# Menentukan Parameter  
param_dist = {  
    'max_depth': [3, 5, 7],  
    'learning_rate': [0.1, 0.3, 0.5],  
    'n_estimators': [100, 300, 500],  
    'subsample': [0.5, 0.7, 1],  
    'colsample_bytree': [0.5, 0.7, 1]}  
# membuat objek RandomizedSearchCV  
random_search = RandomizedSearchCV(xgb,  
                                    param_distributions=param_dist,  
                                    cv=strat_k_fold,  
                                    n_iter=10,  
                                    scoring='roc_auc')  
  

```

Parameter terbaik: {'subsample': 0.7, 'n_estimators': 500, 'max_depth': 7, 'learning_rate': 0.3, 'colsample_bytree': 0.7}

XGBOOST TUNED	
Accuracy (Train Set)	: 0.759
Accuracy (Test Set)	: 0.707
Precision (Train Set)	: 0.923
Precision (Test Set)	: 0.844
Recall (Train Set)	: 0.565
Recall (Test Set)	: 0.509
F1 (Train Set)	: 0.701
F1 (Test Set)	: 0.635
Roc_auc (Train Set)	: 0.851
Roc_auc (Test Set)	: 0.761

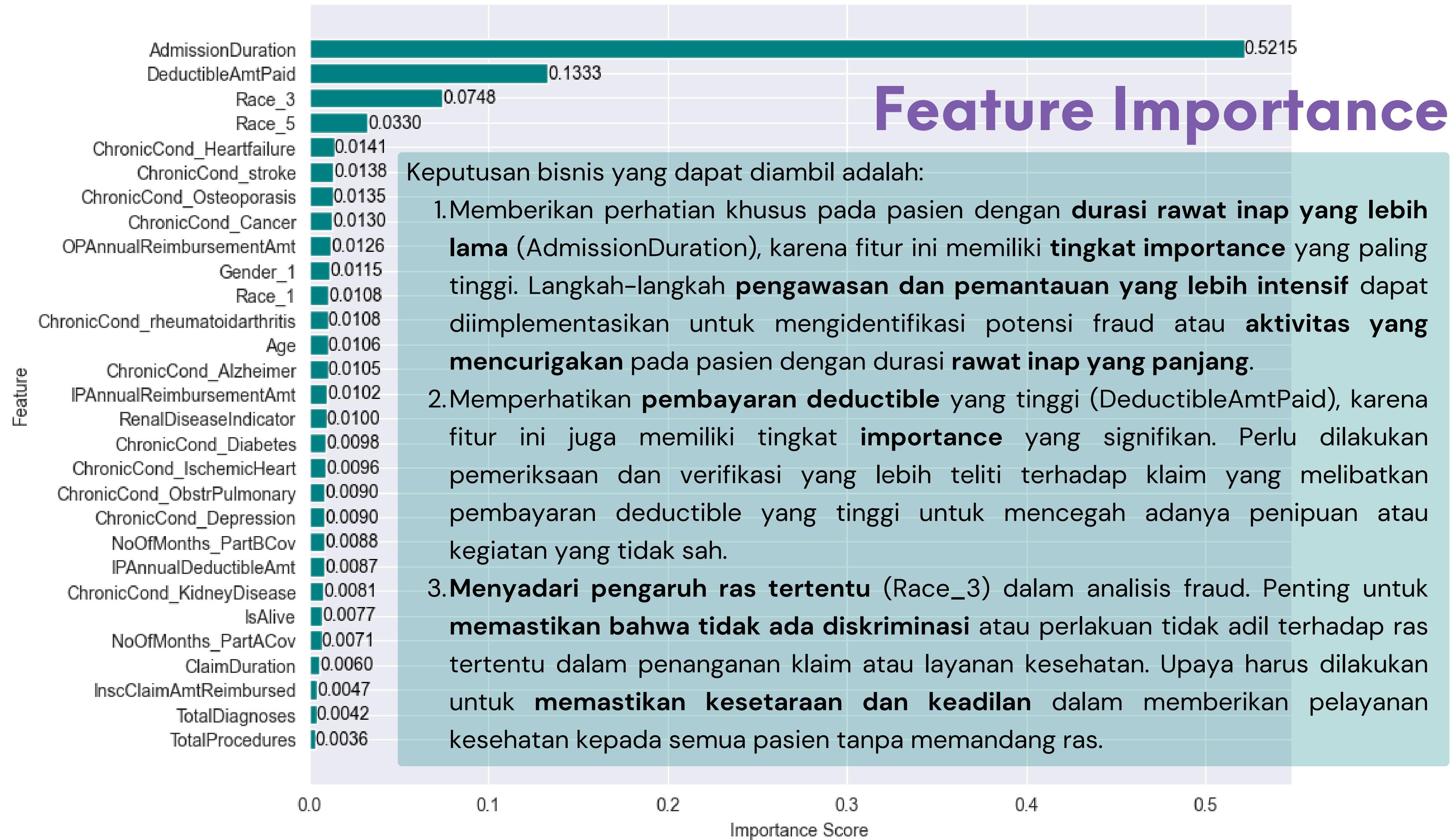


Setelah dilakukan Tuning untuk model XGBoost, yang terjadi diluar harapan yaitu skor precision yang turun ke 84.4%, dari sini bisa dilakukan beberapa cara: 1. Melakukan Tuning ulang dengan parameter yang berbeda; 2. Menggunakan model awal sebelum di tuning. Untuk analisis ini yang akan digunakan adalah kembali ke model awal sebelum di tuning. Jadi pemodelan ini menghasilkan sebuah model untuk memprediksi Fraud menggunakan algoritma XGBoost dengan skor **Precision 88.4%**

Feature Importance



Feature Importance



Impact Analysis

Menghitung Loss Ratio sebelum sesudah menggunakan Machine Learning

Loss Ratio Sebelum ML

Total Loss = Total Reimbursement Loss + Total Charge Loss

Total Loss = Rp1.989.666.550 + Rp451.851.083

Total Loss = Rp2.441.517.633

Total Claim = Total Reimbursement + Total Charge

Total Claim = Rp469.6016.800 + Rp913.301.776

Total Claim = Rp5.609.318.576

Loss Ratio = (Total Loss/Total Claim) x 100

Loss Ratio = (Rp2.441.517.633/Rp5.609.318.576) x 100

Loss Ratio = 43.52%

Loss Ratio Sesudah ML

Precision: 88.4%, artinya setiap 1000 kasus Fraud model bisa memprediksi 884 fraud, sehingga ada 116 kasus yang tidak terprediksi dengan benar. Dengan itu bisa kita rumuskan:

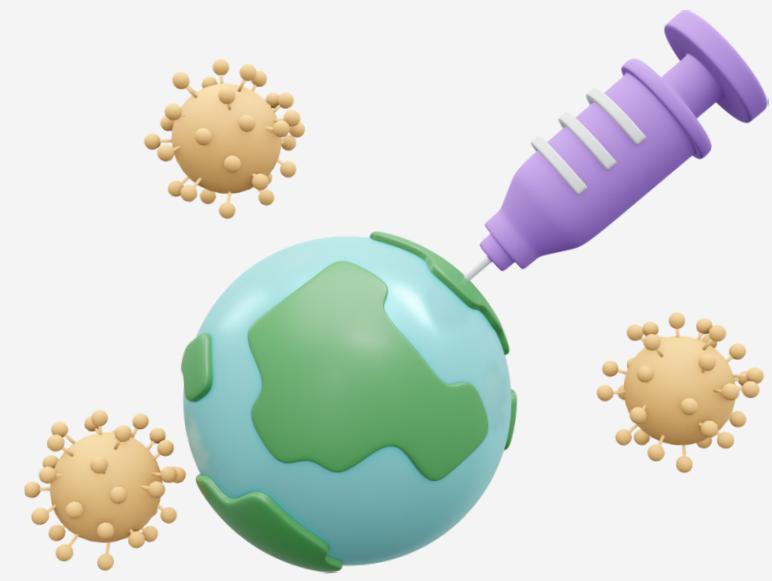
Loss Ratio Sesudah ML = (Total Loss/Total Claim) * (1 - Persentase Fraud Terdeteksi) * 100

Loss Ratio = (Rp2.441.517.633/Rp5.609.318.576) x (1-0.884) x 100

Loss Ratio = (0.4352) x (0.116) x 100

Loss Ratio = 5.04%

Loss Ratio **sebelum ML sebesar 43.52%** menunjukkan bahwa sebelum menggunakan model Machine Learning, perusahaan mengalami **kerugian sebesar 43.52% dari total klaim** yang diajukan. Namun, jika menerapkan model ML, **Loss Ratio menurun drastis menjadi 5.04%**, mengindikasikan bahwa model ML akan berhasil **mengurangi risiko kerugian yang terkait dengan klaim** yang berpotensi fraud. Hal ini menunjukkan efektivitas model ML dalam mendeteksi dan mencegah kasus fraud, yang **berdampak positif** pada keuangan perusahaan dengan mengurangi kerugian yang terjadi.



Healthcare Fraud Analysis and Prediction

Thank You



<https://www.linkedin.com/in/garrylouisg/>



<https://github.com/garrygiancarlo>



<https://linktr.ee/portfolioGarry>

Created by



Garry Louis Giancarlo
Jr Data Scientist