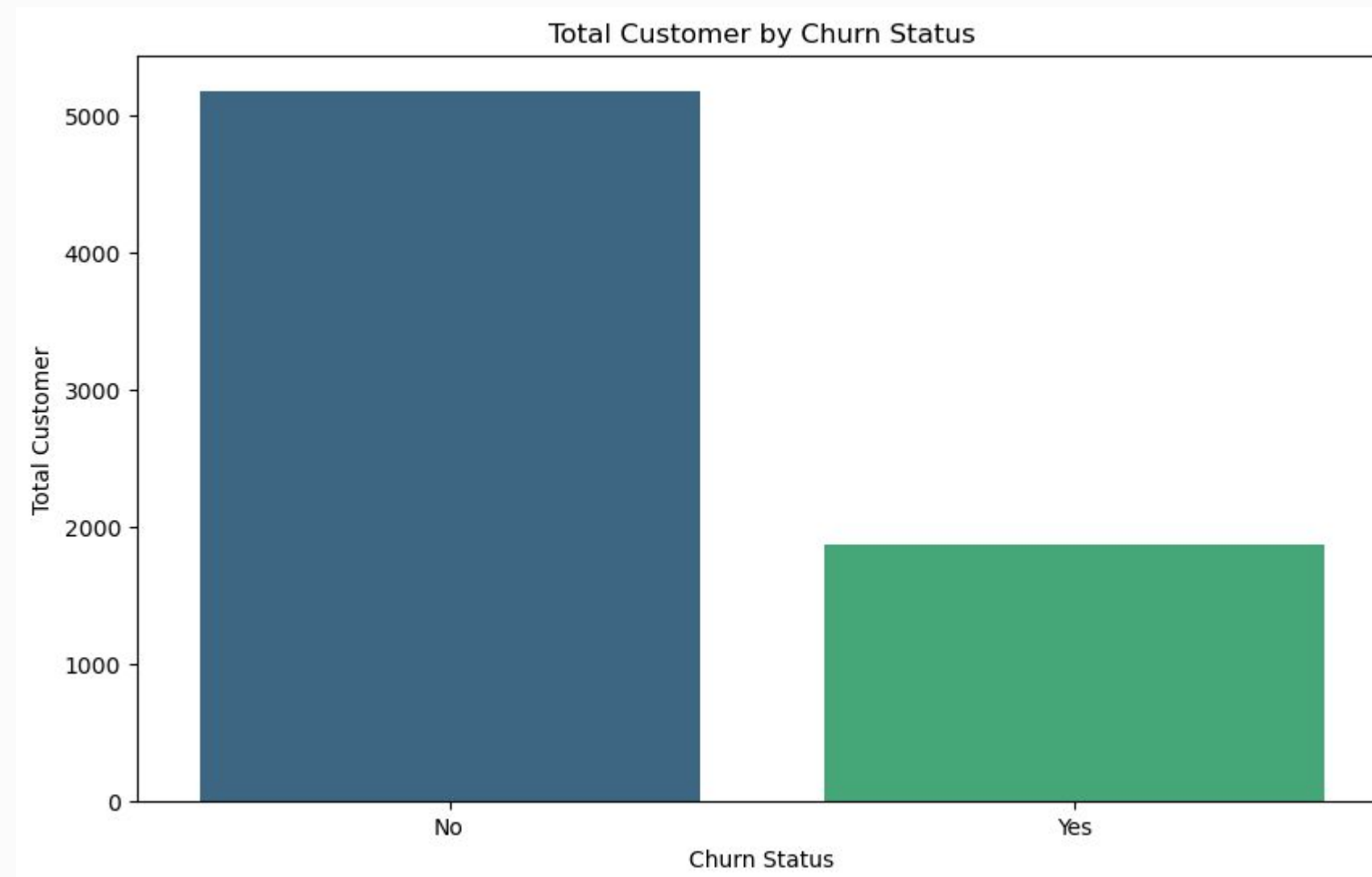


Telecommunication Customers Churn Prediction

Supervisor Learning Classification

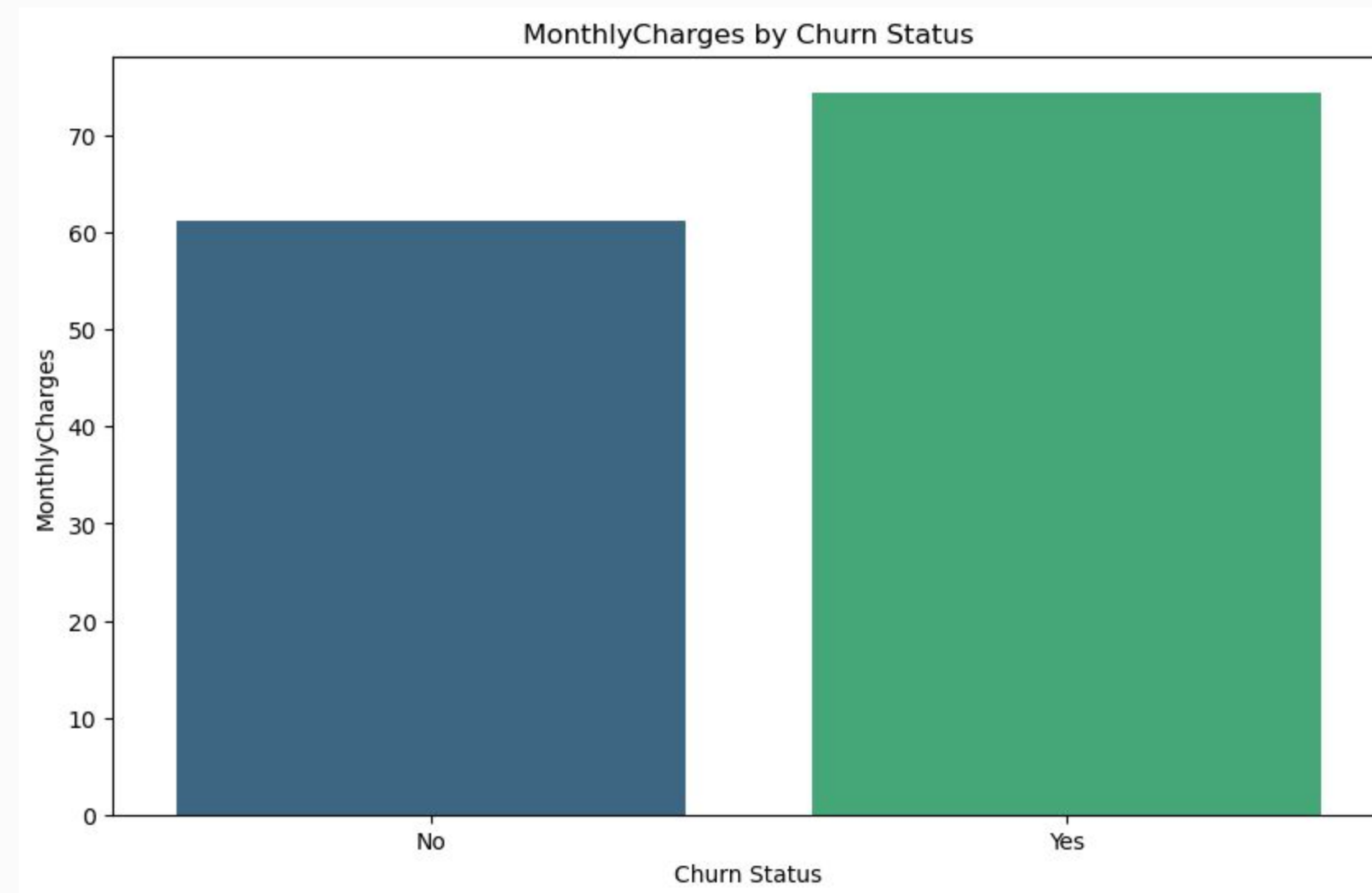
Exploratory Data Analysis

1. Berapa banyak perbandingan pelanggan yang melakukan churn dibanding tidak melakukan churn ?



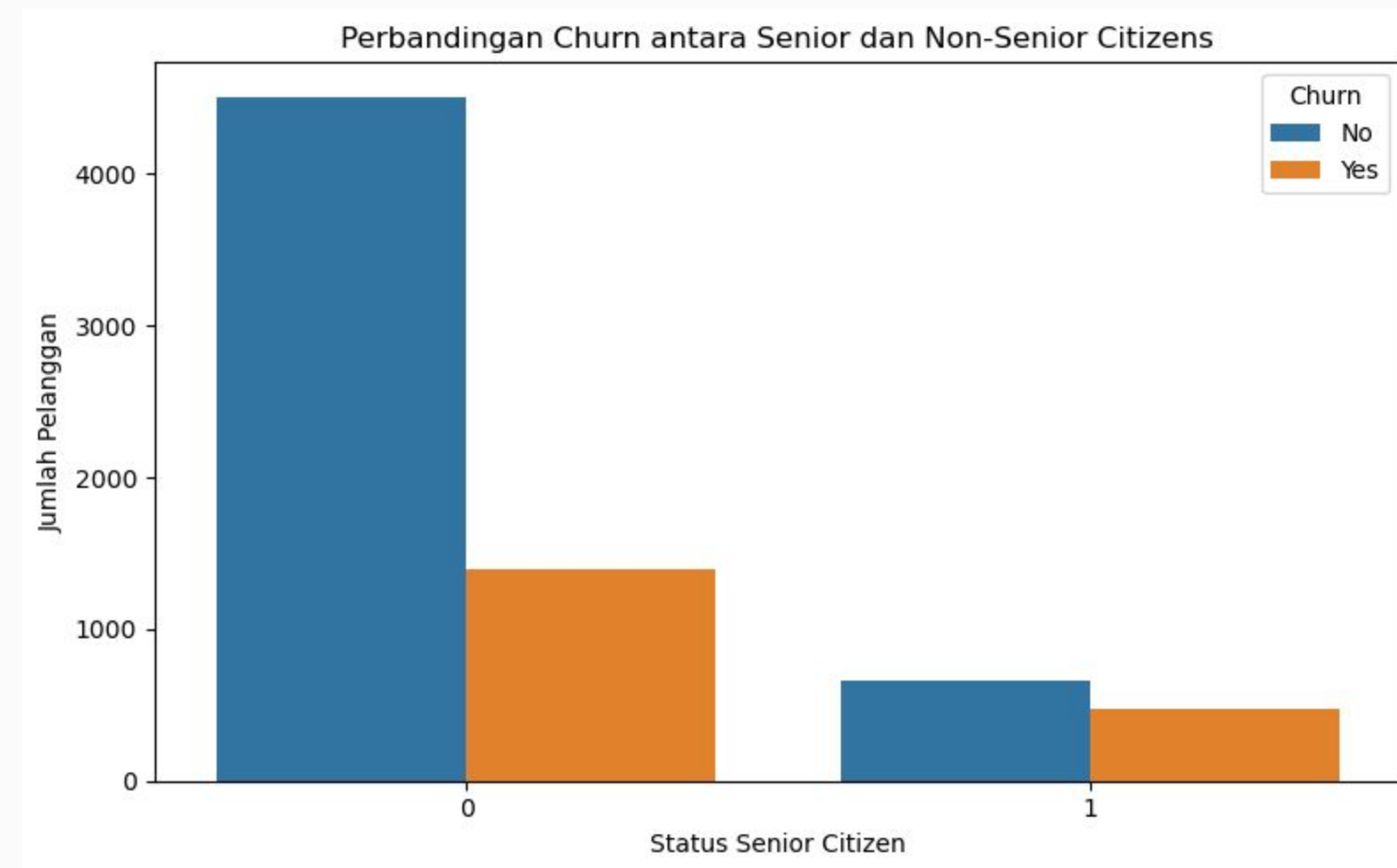
EDA

2. Apakah besarnya MonthlyCharge mempengaruhi pelanggan yang churn ?



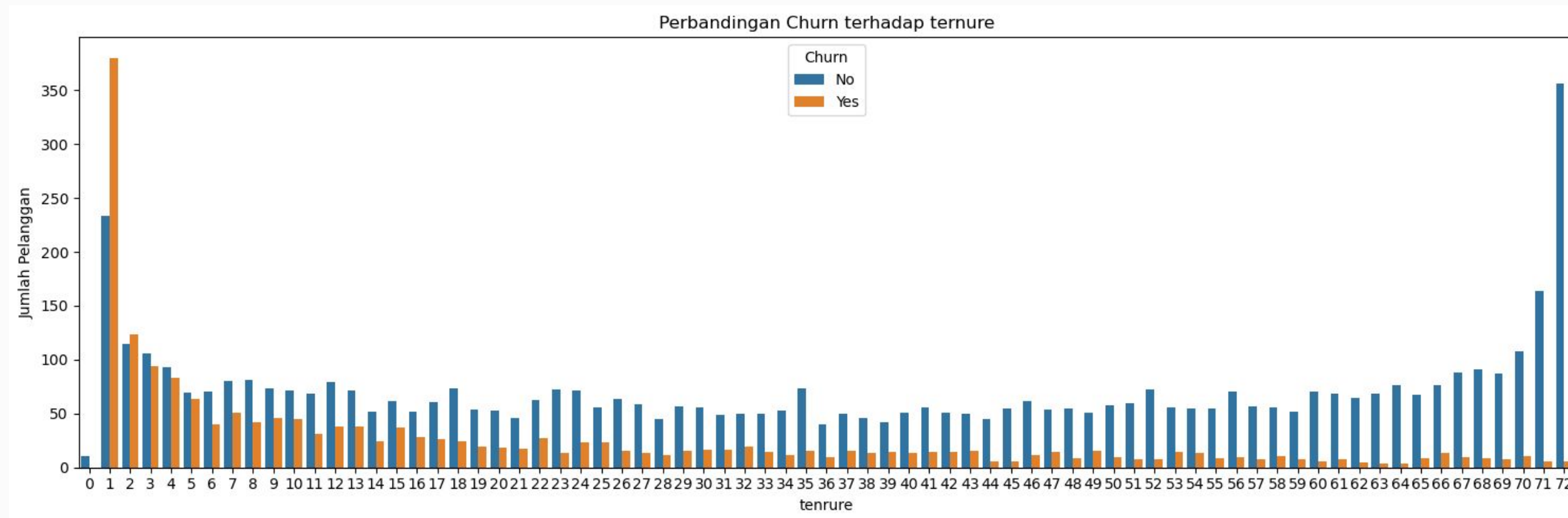
EDA

3. Apakah ada perbedaan tingkat churn antara pelanggan senior citizen dan non-senior citizen?



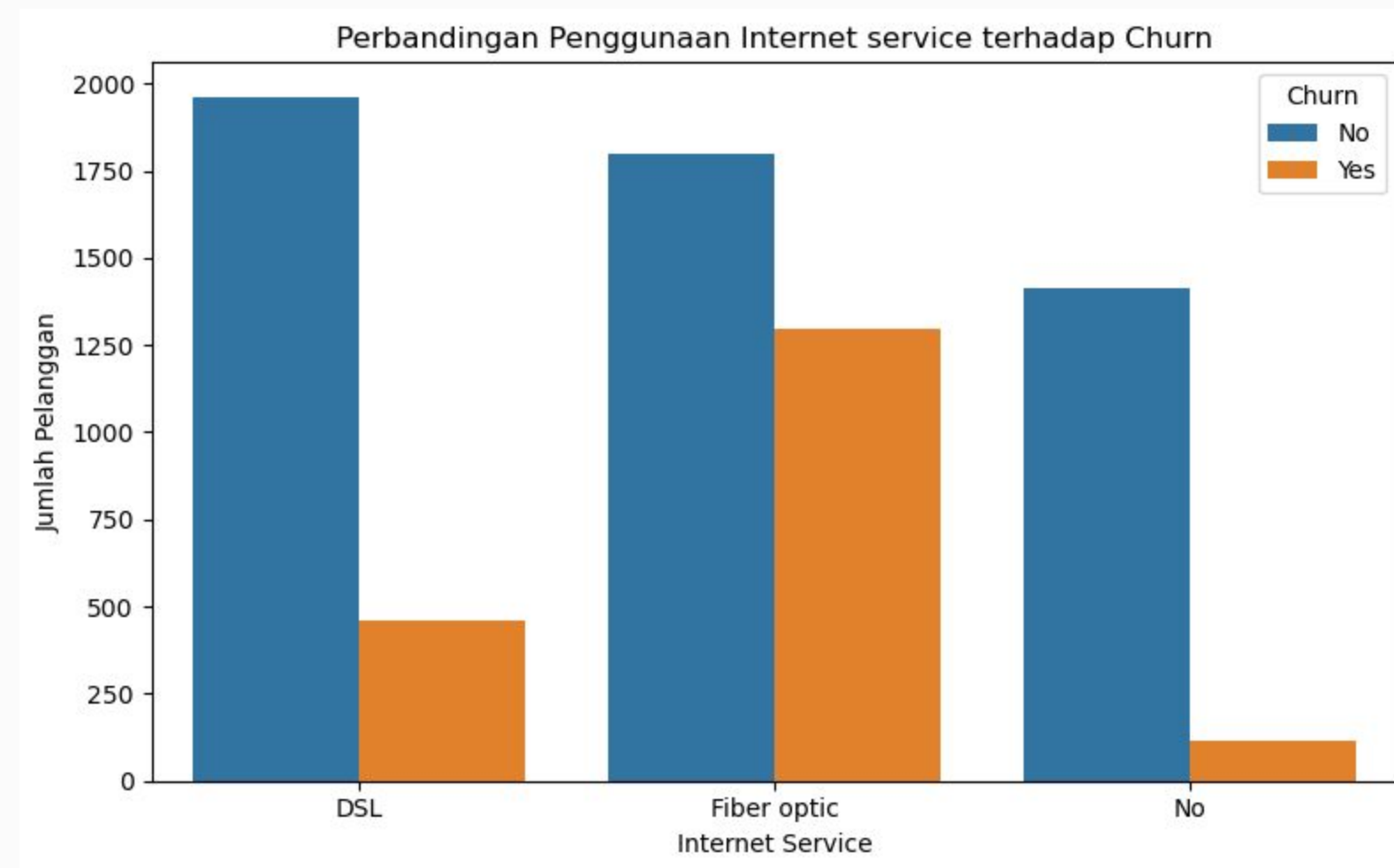
EDA

4. Apakah tingkat tenure mempengaruhi tingkat churn ?



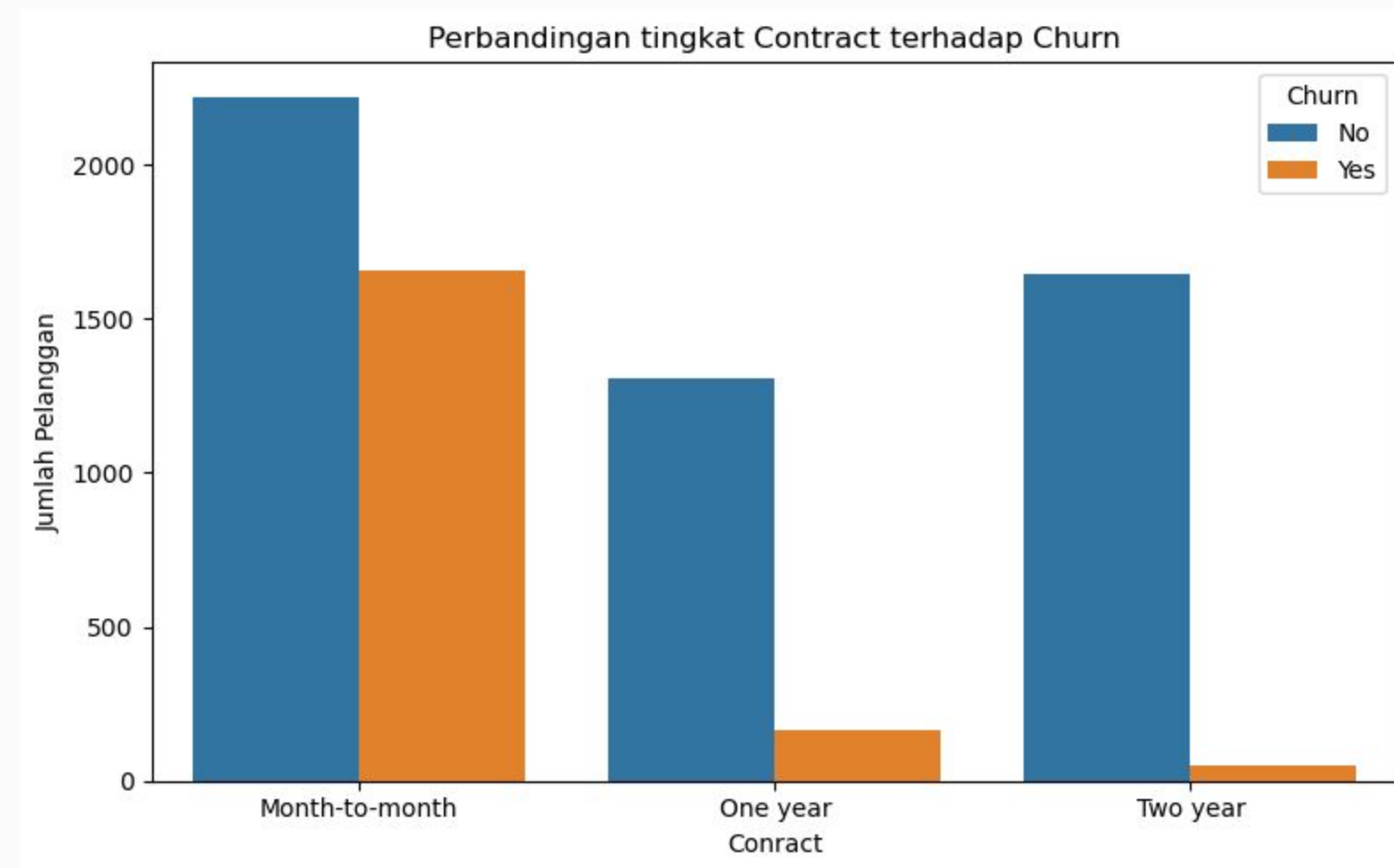
EDA

5. Apakah ada perbedaan tingkat churn antara pelanggan yang menggunakan Internet Service Fiber Optic, DSL , dan yang tidak memakan internet service?



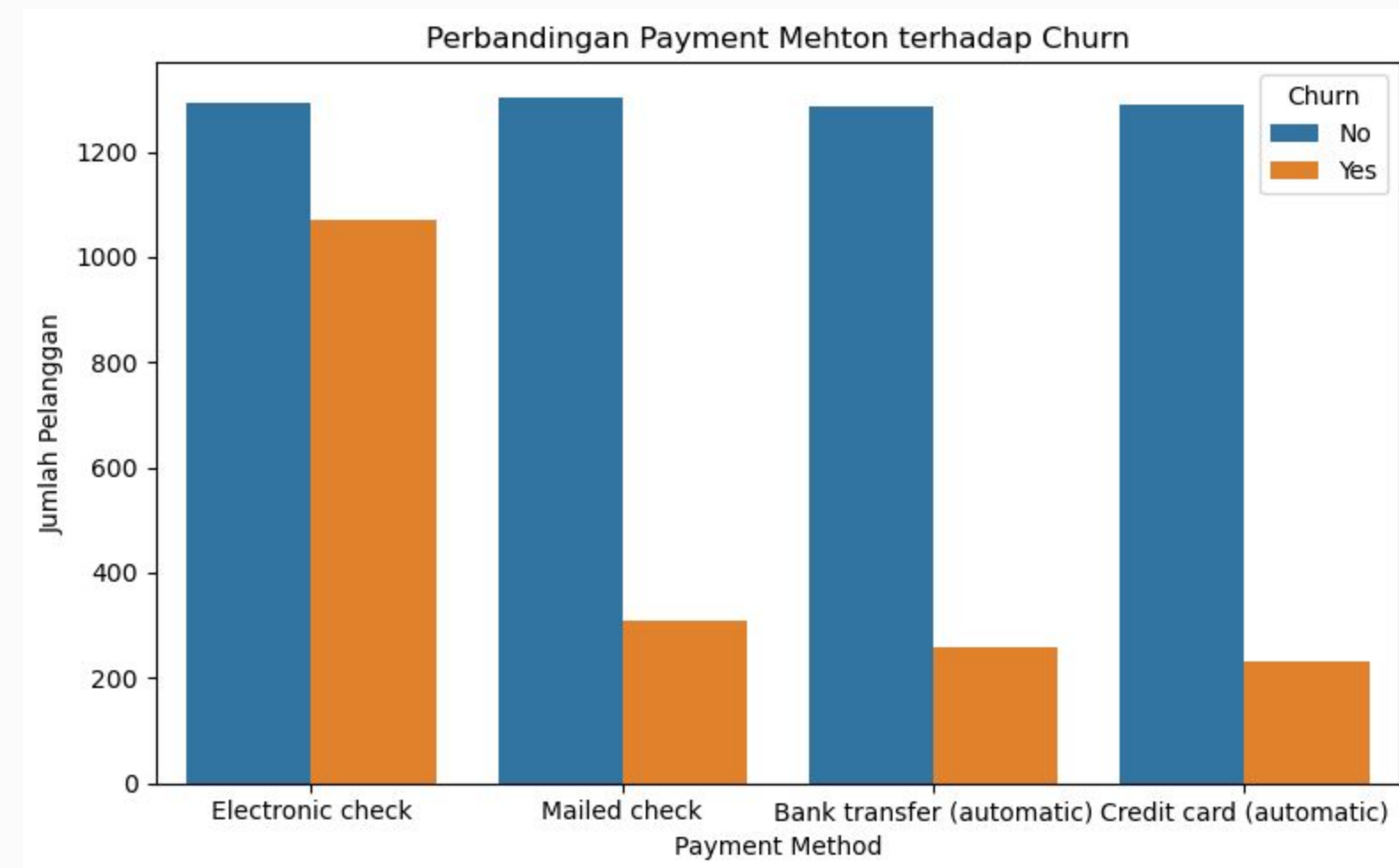
EDA

6. Apakah perbedaan tingkat kontak mempengaruhi tingkat churn ?



EDA

7. Apakah payment method mempengaruhi tingkat churn ?



EDA

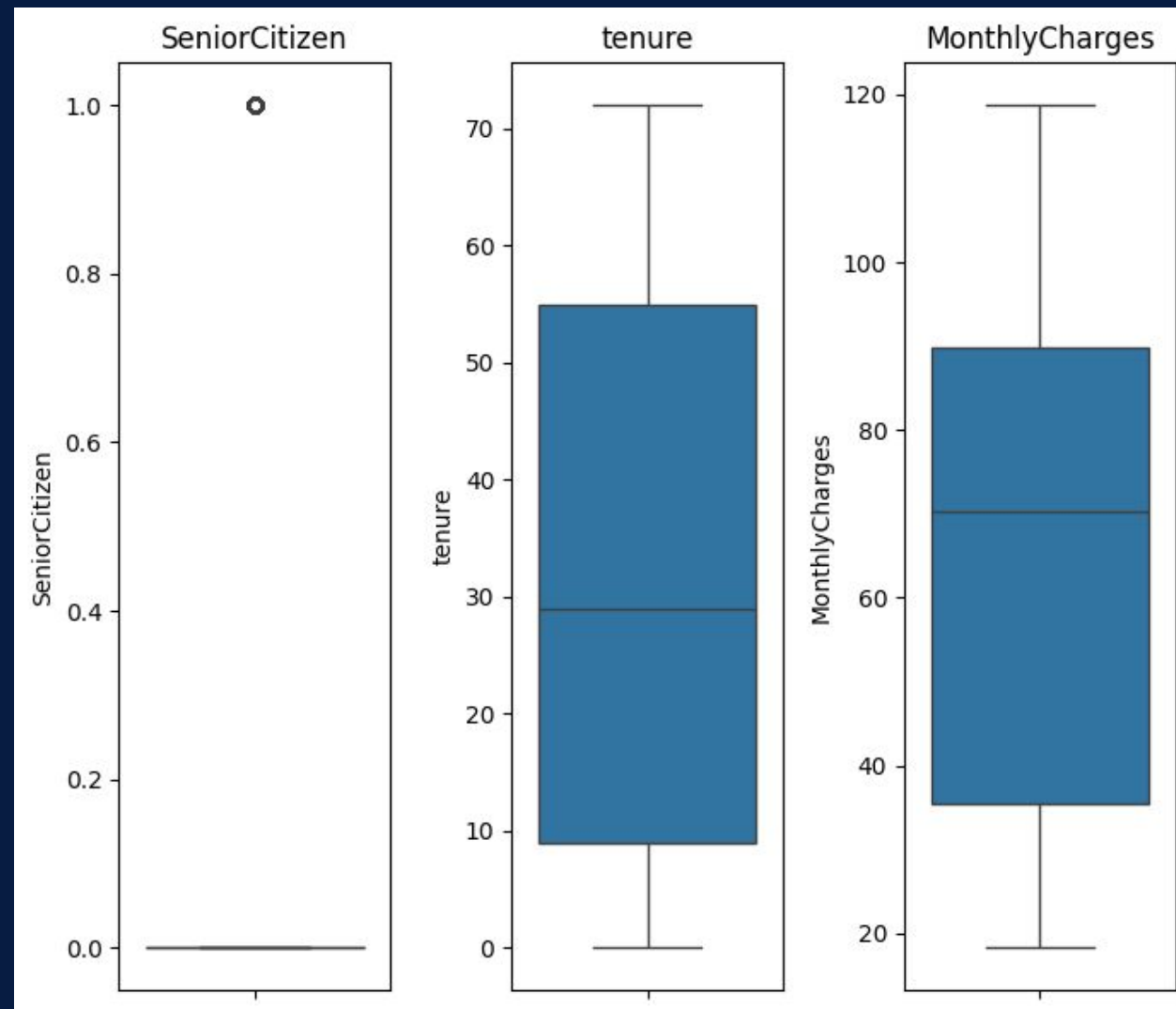
Descriptive Statistics

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
#   Column                Non-Null Count  Dtype  
---  -
0   customerID            7043 non-null  object 
1   gender                7043 non-null  object 
2   SeniorCitizen         7043 non-null  int64  
3   Partner               7043 non-null  object 
4   Dependents            7043 non-null  object 
5   tenure               7043 non-null  int64  
6   PhoneService         7043 non-null  object 
7   MultipleLines         7043 non-null  object 
8   InternetService      7043 non-null  object 
9   OnlineSecurity       7043 non-null  object 
10  OnlineBackup         7043 non-null  object 
11  DeviceProtection     7043 non-null  object 
12  TechSupport          7043 non-null  object 
13  StreamingTV          7043 non-null  object 
14  StreamingMovies      7043 non-null  object 
15  Contract             7043 non-null  object 
16  PaperlessBilling     7043 non-null  object 
17  PaymentMethod        7043 non-null  object 
18  MonthlyCharges       7043 non-null  float64 
19  TotalCharges         7043 non-null  object 
20  Churn                7043 non-null  object 
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

	count	mean	std	min	25%	50%	75%	max
SeniorCitizen	7043.0	162.147	368.612	0.00	0.0	0.00	0.00	1.00
tenure	7043.0	32.371.149	24.559.481	0.00	9.0	29.00	55.00	72.00
MonthlyCharges	7043.0	64.761.692	30.090.047	18.25	35.5	70.35	89.85	118.75

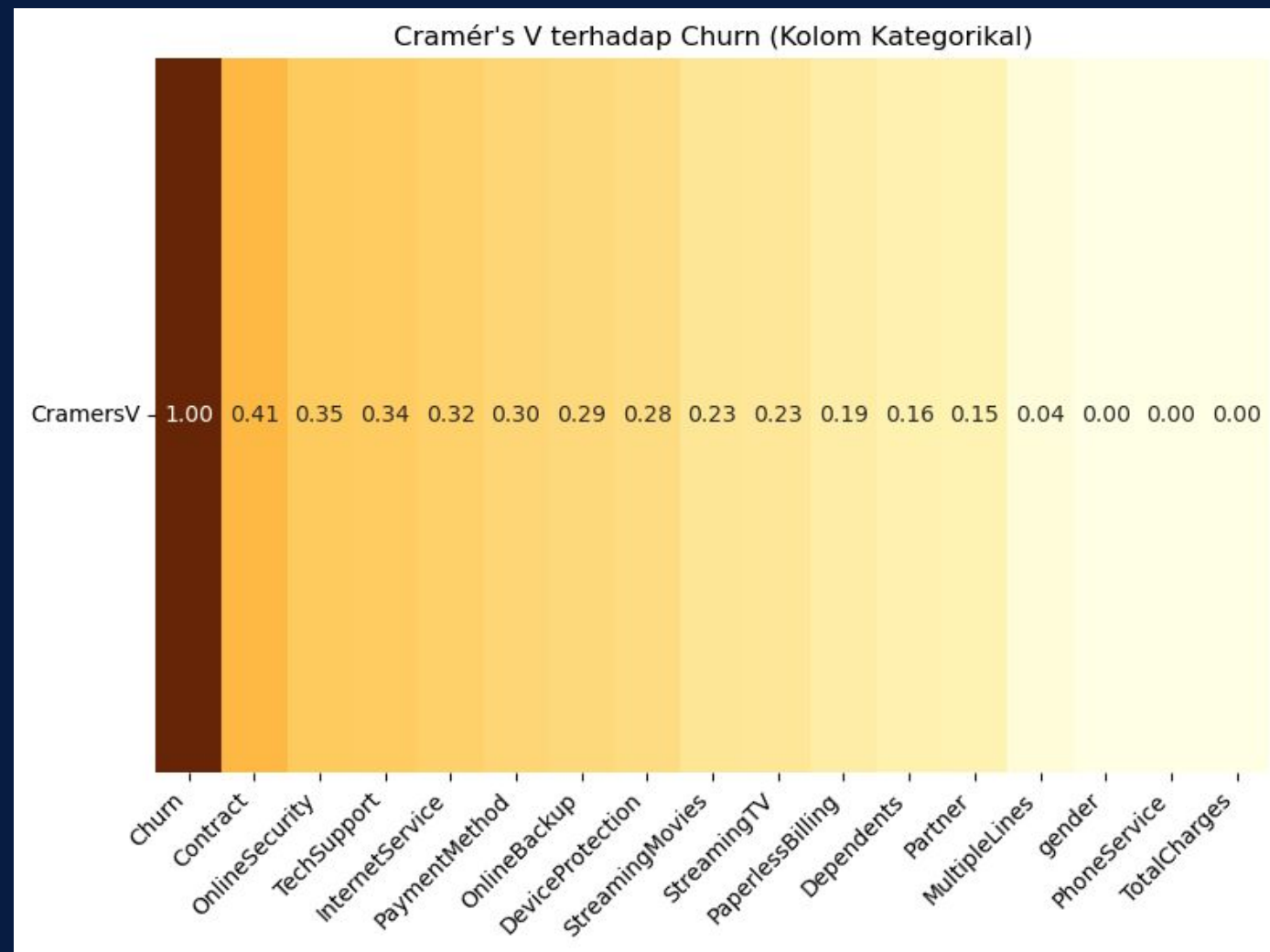
- Tidak terdapat nilai null/kosong
- Ada 3 type data yakni float, int, dan object
- terdapat kolom yang tidak sesuai type datanya
- Totalcharge seharusnya bertipe float, dan kolom bertipe object nanti saat data preprocesing akan diubah type danya.
- ada 3 kolom yang bertipe numeric, kita fokus pada tenure dan Monhtly charga saja, dimana sebaran datanya cukup beragam dan cendrung terlihat tidak ada ouliet karena jarak mean dan median jaraknya tidak terlalu besar
- untuk kolom bertipe object nanti akan diolah lagi, pada beberapa kolom kategorikal akan dilakukan one hot encoding.

Univariate Analysis



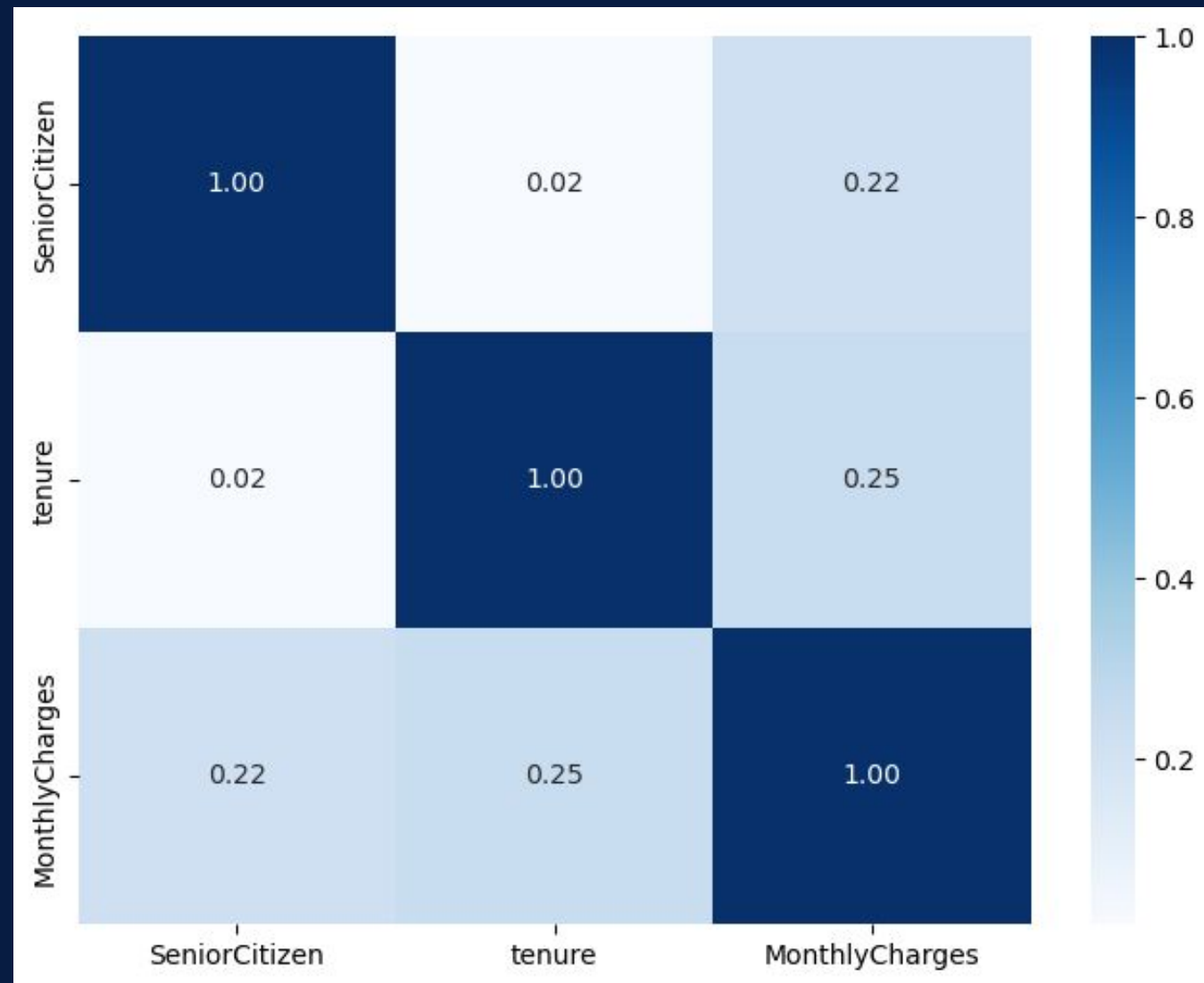
- hanya 3 kolom yang numerik
- Terlihat data tidak ada outlier dan cenderung terdistribusi secara normal

Multivariate Analysis



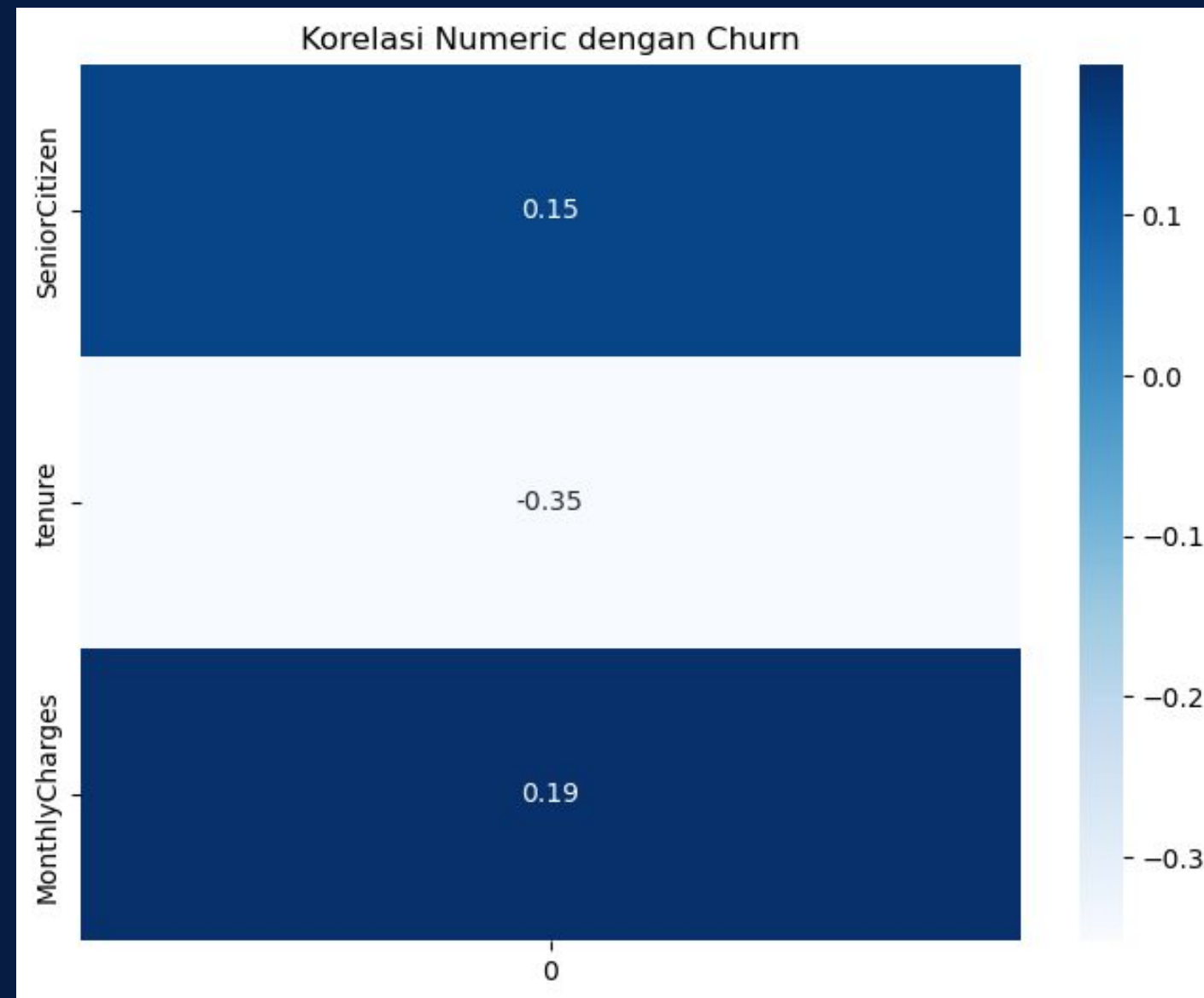
- Melihat hubungan feature kategorikal dengan menggunakan Cramer's V
- Contract, onlineSecurity, dan techSupport memiliki korelasi paling tinggi dengan target.
- Dan yang paling rendah adalah gender dan phonservice

Multivariate Analysis



- korelasi antar kolom numeric
- tidak ada nilai korelasi yang redundan, dan bisa di jadikan feature untuk modeling

Multivariate Analysis



- korelasi kolom numerik dengan target (Churn)
- nilai korelasi tertinggi adalah feature tenure,
- feature SeniorCitizen dan MonthlyCharges korelasinya cukup untuk dijadikan feature saat modeling



Data Preparation



Handling Missing Value & duplicate data

Missing Value

```
#mengecek missing value
df_cleaned.isnull().sum()

✓ 0.0s
```

SeniorCitizen	0
Partner	0
Dependents	0
tenure	0
PhoneService	0
MultipleLines	0
OnlineSecurity	0
OnlineBackup	0
DeviceProtection	0
TechSupport	0
StreamingTV	0
StreamingMovies	0
PaperlessBilling	0
MonthlyCharges	0
TotalCharges	0
Churn	0
gender_Male	0
InternetService_Fiber optic	0
InternetService_No	0
Contract_One year	0
Contract_Two year	0
PaymentMethod_Credit card (automatic)	0
PaymentMethod_Electronic check	0
PaymentMethod_Mailed check	0

dtype: int64

- Tidak ada data yang missing

Duplicate data

```
# Tampilkan semua baris yang terlibat dalam duplikasi
df_cleaned.duplicated().sum()

✓ 0.0s
```

np.int64(0)

- Tidak ada data yang duplikat

Data Preparation

Data type adjustment

```
# Bersihkan spasi di semua kolom object
for col in df_cleaned.select_dtypes(include='object').columns:
    df_cleaned[col] = df_cleaned[col].str.strip()

# Konversi TotalCharges ke numeric
df_cleaned['TotalCharges'] = pd.to_numeric(df_cleaned['TotalCharges'], errors='coerce')

# Isi NaN dengan median secara aman
df_cleaned['TotalCharges'] = df_cleaned['TotalCharges'].fillna(df_cleaned['TotalCharges'].median())

# --- Gabungkan 'No phone service' menjadi 'No' pada MultipleLines ---
df_cleaned['MultipleLines'] = df_cleaned['MultipleLines'].replace({'No phone service': 'No'})

# --- Gabungkan 'No internet service' menjadi 'No' pada kolom terkait ---
internet_cols = [
    'OnlineSecurity', 'OnlineBackup', 'DeviceProtection',
    'TechSupport', 'StreamingTV', 'StreamingMovies'
]
for col in internet_cols:
    df_cleaned[col] = df_cleaned[col].replace({'No internet service': 'No'})

# --- Mapping Yes/No menjadi 1/0 ---
binary_cols = [
    'Partner', 'Dependents', 'PhoneService', 'PaperlessBilling', 'Churn',
    'MultipleLines'
] + internet_cols

for col in binary_cols:
    df_cleaned[col] = df_cleaned[col].map({'Yes': 1, 'No': 0})
```

- Penyesuaian type data pada kolom **TotalCharge** yang sebelumnya object ke float dan mengisi nilai kosong dengan median apabila ditemukan nilai Nan/kosong
- karena ada beberapa kolom diisi dengan 3 kategori, seperti kolom MultipleLines yang berisi Yes, No, dan No phone service, Maka 'No phone service' akan diubah menjadi No, hingga sisi kolom nya hanya berisi dua kategori Yes dan No saja.
- semua kolom yang berisikan Katergori Yes dan No akan dirubah nilainya menjadi nilai 1 dan 0 saja.

Data Preparation



Feature Engineering



Feature Selection & Transformation

```
# --- Hapus kolom customerID ---  
df_cleaned.drop(['customerID'], axis=1, inplace=True)
```

```
# --- Encode kolom kategorik lainnya dengan One-Hot Encoding ---  
multi_class_cols = ['gender', 'InternetService', 'Contract', 'PaymentMethod']  
df_cleaned = pd.get_dummies(df_cleaned, columns=multi_class_cols, drop_first=True)
```

```
df_cleaned.isnull().sum()
```

```
df_cleaned.duplicated().sum()
```

✓ 0.0s

22

```
df_cleaned.drop_duplicates(inplace=True)
```

✓ 0.0s

- semua kolom akan digunakan untuk feature kecuali kolom customerID akan di drop
- melakukan encoding pada kolom kategorikal menggunakan one hot encoding
- dicek lagi apakah setelah encoding ada null value dan ada duplicate, apabila jumlahnya sedikit maka akan didrop

Feature Selection & Transformation

Code

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

#Feature scaling untuk numerik
scaler = StandardScaler()
num_cols = ['tenure', 'MonthlyCharges', 'TotalCharges']
df_cleaned[num_cols] = scaler.fit_transform(df_cleaned[num_cols])

#Split fitur dan target
X = df_cleaned.drop('Churn', axis=1)
y = df_cleaned['Churn']

#Train-test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=y
)

print("Data siap untuk modeling. Jumlah fitur:", X_train.shape[1])
```

✓ 0.2s

Data siap untuk modeling. Jumlah fitur: 23

- Melakukan standarisasi dengan standard scaler pada kolom **tenure**, **MonthlyCharges**, dan **TotalCharges**
- Memisahkan feature dan target
- split data menjadi data train dan data tes dengan proporsi 20% untuk data test
- jumlah feature keseluruhan menjadi 23 feature

Handling Imbalance

Code

```
#Handling Imbalance
from imblearn.over_sampling import SMOTE

smote = SMOTE(random_state=42)
X_train_bal, y_train_bal = smote.fit_resample(X_train, y_train)

# 8. Cek distribusi target sebelum dan sesudah
print("Sebelum SMOTE:\n", y_train.value_counts(normalize=True))
print("\nSesudah SMOTE:\n", pd.Series(y_train_bal).value_counts(normalize=True))
```

35]

```
.. Sebelum SMOTE:
   Churn
0    0.735577
1    0.264423
Name: proportion, dtype: float64

Sesudah SMOTE:
   Churn
1    0.5
0    0.5
Name: proportion, dtype: float64
```

- Melakukan balancing pada target didata train menggunakan SMOTE, sehingga didapat data target yang sebelumnya 73% dan 26%, kini menjadi sama 50%-50%



MODELING



MODELING

Code

```
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from xgboost import XGBClassifier
models = [
    ("Logistic Regression", LogisticRegression(max_iter=1000, random_state=42)),
    ("Random Forest", RandomForestClassifier(random_state=42)),
    ("Gradient Boosting", GradientBoostingClassifier(random_state=42)),
    ("XGBoost", XGBClassifier(random_state=42))
]

for name, model in models:
    model.fit(X_train_bal, y_train_bal)
    eval_classification(model, X_train_bal, y_train_bal, model_name=name)
    # plot_confusion_matrix(model, model_name=nam
```

- Melakukan modeling dengan 4 model sebagai perbandingan 1 base model **LogisticRegression**, 3 Esemble model **RandomForestClassifier**, **GradientBoostingClassifier**, **XGBClassifier**

MODEL EVALUATION

Model	Accuracy (Test)	Accuracy (Train)	Precision (Test)	Recall (Test)	Recall (Train)	F1-Score (Test)	ROC AUC (Test)	ROC AUC (Train)	Recall CV Train	Recall CV Test
Logistic Regression	0.76	0.8	0.53	0.71	0.82	0.61	0.83	0.88	0.55	0.55
Random Forest	0.77	1	0.56	0.59	1	0.57	0.81	1	1	0.49
Gradient Boosting	0.77	0.82	0.54	0.74	0.86	0.63	0.84	0.9	0.57	0.53
XGBoost	0.77	0.94	0.55	0.63	0.96	0.59	0.81	0.99	0.85	0.51

MODEL EVALUATION

- Dari ke-4 Model diatas yang akan dipilih adalah **Gradient Boosting** Karena :
 - a. LogisticRegression memperoleh nilai evaluasi yang hampir mirip, tetapi Gradient Boosting cerdrung lebih besar nilainya dibanding LogisticRegression.
 - b. Niai pada data train dan data test lebih stabil dibanding matrix lain.
 - c. Karena fokus kepada konsumen yang churn dan mengurangi False negatif sebanyak mungkin maka dipilih Recall sebagai perhatian utama, dan nilai recall paling tinggi adalah gradientboosting pada data test **0,74** dan data train **0.86**.
 - d. pada nilai ROC-AUC paling tinggi dan stabil dibanding yang lain
 - e. Nilai Recall Cross Validation menunjukan pengukuran paling stabil jika dibandingkan dengan yang lain, meski nilai nya lebih kecil daripada ketiga model yang lain

Model	Accuracy (Test)	Accuracy (Train)	Precision (Test)	Recall (Test)	Recall (Train)	F1-Score (Test)	ROC AUC (Test)	ROC AUC (Train)	Recall CV Train	Recall CV Test
Gradient Boosting	0.77	0.82	0.54	0.74	0.86	0.63	0.84	0.9	0.57	0.53

Gradient Boosting(tuning)	0.76	0.89	0.54	0.65	0.91	0.59	0.83	0.96	0.72	0.5
---------------------------	------	------	------	------	------	------	------	------	------	-----

Hyperparameter Tuning

```
# 1. Parameter grid yang lebih sempit & realistis
param_grid = {
    'n_estimators': [100, 150, 200],
    'learning_rate': [0.05, 0.1],
    'max_depth': [3, 4, 5],
    'min_samples_split': [2, 5],
    'min_samples_leaf': [1, 2],
    'subsample': [0.8, 1.0],
    'max_features': ['sqrt', None]
}

# 2. Model dasar
gb_model = GradientBoostingClassifier(random_state=42)

# 3. Cross-validation stratifikasi (penting untuk data imbalance/balance)
skf = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)

# 4. Randomized Search dengan scoring fokus ke 'recall' atau 'f1'
rs_gb = RandomizedSearchCV(
    estimator=gb_model,
    param_distributions=param_grid,
    n_iter=30, # cukup untuk awal, bisa ditambah
    scoring='f1', # atau 'recall' jika recall lebih penting
    cv=skf,
    verbose=1,
    random_state=42,
    n_jobs=-1
)

# 5. Fitting ke data yang sudah di-balance
rs_gb.fit(X_train_bal, y_train_bal)

# 6. Evaluasi hasilnya
eval_classification(
    rs_gb,
    X_train_bal,
    y_train_bal,
    model_name="Gradient Boosting (Tuned)"
)
```

```
Fitting 5 folds for each of 30 candidates, totalling 150 fits
=====
Evaluation Report for: Gradient Boosting (Tuned)
=====
Accuracy (Test Set): 0.76
Accuracy (Train Set): 0.89
Precision (Test Set): 0.54
Recall (Test Set): 0.65
Recall (Train Set): 0.91
F1-Score (Test Set): 0.59
roc_auc (test-proba): 0.83
roc_auc (train-proba): 0.96
Recall (Crossval Train): 0.72
Recall (Crossval Test): 0.51
```

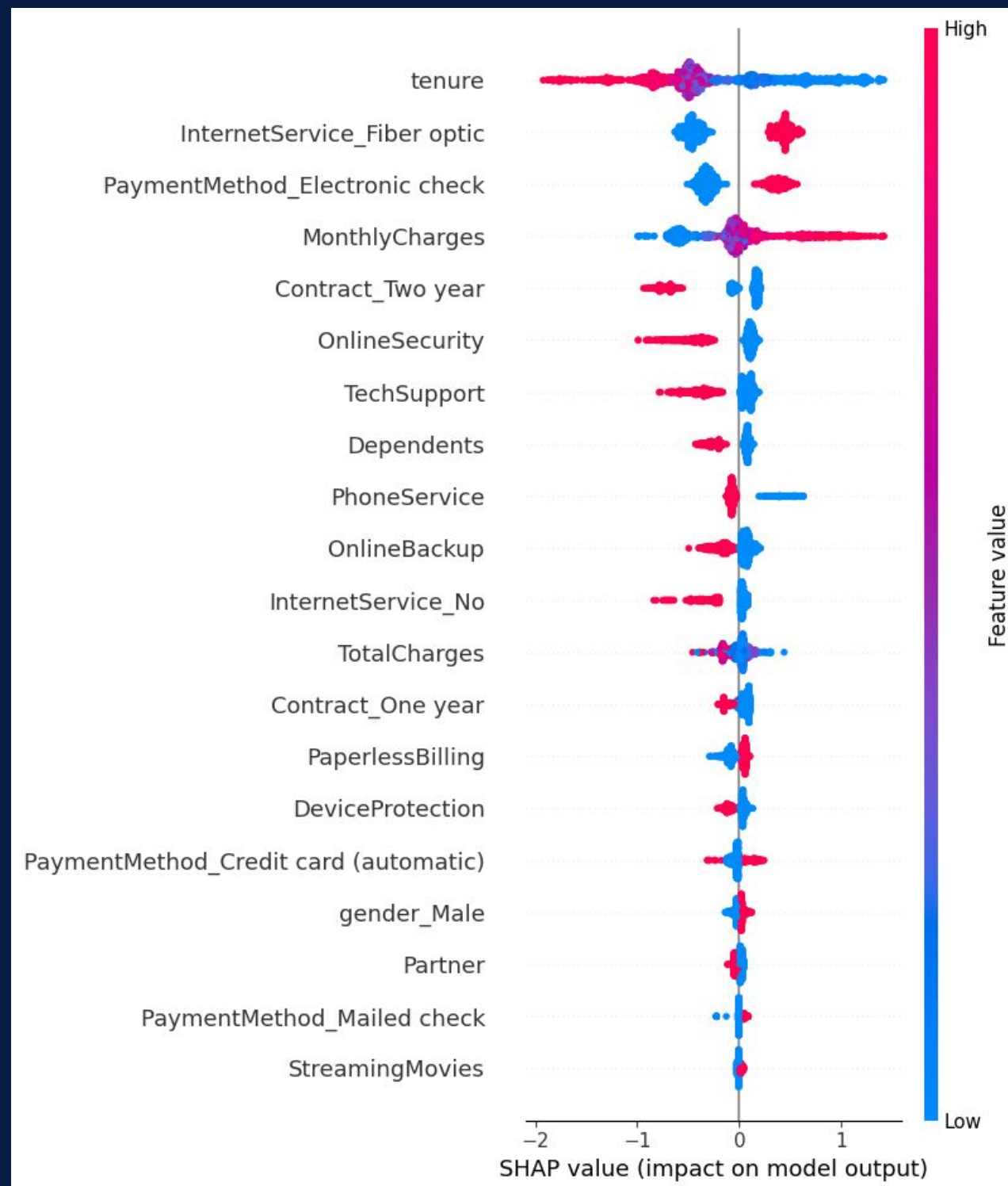
- Melakukan Tuning Hyperparameter menggunakan metode RandomizedSearchCV
-

Hyperparameter Tuning

Model	Accuracy (Test)	Accuracy (Train)	Precision (Test)	Recall (Test)	Recall (Train)	F1-Score (Test)	ROC AUC (Test)	ROC AUC (Train)	Recall CV Train	Recall CV Test
Gradient Boosting	0.77	0.82	0.54	0.74	0.86	0.63	0.84	0.9	0.57	0.53
Gradient Boosting(tuning)	0.76	0.89	0.54	0.65	0.91	0.59	0.83	0.96	0.72	0.5

- setelah dilakukan tuning banyak metrix menjadi overfitting, salah satunya pada matrix recall awalya 0,74 dan data train 0,86 dan pada hasil tuning nilai nya jadi overfitting 65 dan 91.
- karena nilai hyper lebih kecil dari model sebelumnya maka tidak dilakukan hyper tuning pada model

Shap Plot



- Feature tenure menjadi feature paling berpengaruh terhadap keputusan model
- Warna biru (tenure rendah) → SHAP value positif → meningkatkan probabilitas churn. Warna merah (tenure tinggi) → SHAP value negatif → menurunkan probabilitas churn.
- feature InternetServicefiberOptik
- Nilai merah (kontrak panjang, misalnya 1-2 tahun) → SHAP value negatif → menurunkan churn. Nilai biru (kontrak bulanan) → SHAP value positif → menaikkan churn.
- feaure MontlyCharge
- Nilai merah (biaya bulanan tinggi) → SHAP value positif → meningkatkan churn. Nilai biru (biaya rendah) → SHAP value negatif → menurunkan churn.

Business Insight

- Pelanggan yang baru bergabung (masa langganan pendek) cenderung lebih berisiko churn. Retensi harus difokuskan pada pelanggan baru di bulan-bulan awal berlangganan.
- Pelanggan dengan kontrak bulanan jauh lebih berpotensi churn. Bisa dilakukan upselling ke kontrak tahunan dengan benefit harga.
- Biaya bulanan yang tinggi bisa menjadi faktor pemicu churn. Strategi diskon atau paket hemat bisa mengurangi risiko ini.



TERIMA KASIH

