

Mata Kuliah Aplikasi Data Scientist

Laporan Tugas P8: Linear Regresi

Dosen Pengampu: Ledy Elsera Astrianty, S.Kom., M.Kom.



Disusun oleh:

- Lathif Ramadhan (5231811022)
- Andini Angel M. (5231811029)
- Rama Panji N. (5231811033)
- Giffari Riyanda P. (5231811036)

PROGRAM STUDI SAINS DATA

FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS TEKNOLOGI YOGYAKARTA

YOGYAKARTA

2025

Daftar Isi

Daftar Isi.....	2
Bab 1: Pendahuluan.....	3
Bab 2: Pembahasan	5
2.1. Import Libraries.....	5
2.2. Load Dataset	7
2.3. Preprocessing (Penanganan Missing Value dan Duplikasi Data).....	8
2.3.1. Cek Duplikasi Data.....	8
2.3.2. Cek Missing Value	9
2.4. Exploratory Data Analysis (EDA)	10
2.4.1. Deskripsi/Ringkasan Data	10
2.4.2. Informasi Data (kolom)	10
2.4.3. Distribusi Variabel Kategori	12
2.4.4. Hubungan Variabel Kategorik dengan Target (charges).....	14
2.4.5. Distribusi Variabel Numerik	16
2.4.6. Hubungan Variabel Numerik dengan Target (charges).....	19
2.4.6. Outlier Detection (Opsional)	20
2.4.7. Analisis Korelasi Numerik.....	21
2.5. Preprocessing (Persiapan untuk model Regresi)	22
2.5.2. Konfirmasi Hasil Encoding	23
2.5.3. Memisahkan fitur (X) dan target (y)	24
2.5.4. Split data training-testing (80:20).....	25
2.5.5. Penskalaan Fitur Numerik	25
2.5.6. Verifikasi Bentuk Data Setelah Split	27
2.6. Pemodelan dan Simulasi	27
2.6.1. Inisialisasi dan training model.....	27
2.6.2. Prediksi data testing	28
2.6.4. Analisis Residual.....	30
2.7. Simulasi Prediksi	35
2.7.1. Contoh 1: Non-perokok, laki-laki, 30 tahun, bmi 25, 1 anak, di southeast.....	35
2.7.2. Contoh 2: Sama seperti contoh 1, tetapi perokok	35
2.7.3. Perbedaan biaya karena merokok (untuk contoh ini).....	36
Bab 3: Kesimpulan	37
Bab 4: Lampiran	39

Bab 1: Pendahuluan

Dataset yang digunakan dalam proyek analisis ini merupakan data biaya asuransi kesehatan individu yang bersumber dari platform Kaggle (*Insurance Dataset*). Data ini awalnya dikumpulkan untuk mendukung pembelajaran dalam buku "*Machine Learning with R*" karya Brett Lantz. Meskipun dataset ini merupakan data publik, data tersebut telah melalui proses pembersihan dan penyesuaian format agar sesuai dengan kebutuhan analisis dalam buku tersebut. Proyek ini bertujuan untuk memanfaatkan dataset tersebut guna membangun model prediktif yang dapat memperkirakan biaya medis individu berdasarkan faktor-faktor demografis dan kebiasaan hidup.

Link Sumber Dataset: <https://www.kaggle.com/datasets/mirichoi0218/insurance>

Deskripsi Variabel

Dataset terdiri dari **1.337 entri** (baris) dengan **7 variabel** (kolom) yang mencakup:

1. **Usia Tertanggung (age):**

Menunjukkan usia penerima manfaat utama asuransi. Rentang usia dalam dataset ini berkisar antara **18 hingga 64 tahun**, dengan rata-rata usia **39.2 tahun** dan distribusi yang relatif merata (standar deviasi ± 14 tahun).

2. **Jenis Kelamin (sex):**

Kategori jenis kelamin tertanggung, yaitu **perempuan (female)** dan **laki-laki (male)**. Variabel ini termasuk dalam tipe data kategorik (*object*).

3. **Indeks Massa Tubuh (bmi):**

Nilai BMI (*Body Mass Index*) yang dihitung berdasarkan rasio berat badan (kg) terhadap kuadrat tinggi badan (m^2). Nilai BMI ideal umumnya berada di kisaran **18.5–24.9**, namun rata-rata BMI dalam dataset ini adalah **30.66**, mengindikasikan bahwa sebagian besar tertanggung memiliki berat badan di atas normal (*overweight* atau *obese*). Rentang BMI cukup lebar, mulai dari **15.96** (sangat kurus) hingga **53.13** (obesitas ekstrem).

4. **Jumlah Tanggungan (children):**

Menunjukkan jumlah anak atau dependen yang tercakup dalam polis asuransi. Sebagian besar tertanggung memiliki **0–2 anak** (nilai median = 1), dengan maksimal **5 anak**.

5. **Status Merokok (smoker):**

Variabel kategorik yang mengidentifikasi apakah tertanggung merupakan perokok

(*yes*) atau bukan (*no*). Kebiasaan merokok diduga kuat berpengaruh signifikan terhadap biaya asuransi.

6. Wilayah Tempat Tinggal (region):

Wilayah geografis tempat tinggal tertanggung di AS, terbagi menjadi empat kategori: *northeast*, *southeast*, *southwest*, dan *northwest*.

7. Biaya Asuransi (charges):

Variabel target dalam proyek ini, yaitu biaya medis individu yang ditagihkan kepada perusahaan asuransi. Biaya ini memiliki variasi yang sangat besar, mulai dari **\$1,121.87** hingga **\$63,770.43**, dengan rata-rata **\$13,279.12** dan standar deviasi tinggi ($\pm \$12,110.36$). Sebanyak 25% tertanggung memiliki biaya di bawah **\$4,746**, sementara 75% di bawah **\$16,657**.

Kualitas Data

Berdasarkan pemeriksaan menggunakan `df.info()`, seluruh kolom memiliki **1.337 entri** tanpa nilai kosong (*non-null*), yang mengonfirmasi bahwa data sudah bersih dari *missing values*.

Tipe data untuk setiap kolom juga sudah sesuai:

- Numerik (*float64* dan *int64*): age, bmi, children, charges
- Kategorik (*object*): sex, smoker, region

Relevansi dengan Proyek

Dataset ini dipilih karena memuat variabel-variabel kunci yang secara logis berkaitan dengan biaya medis, seperti usia, kebiasaan merokok, dan kondisi kesehatan (BMI). Variasi yang tinggi pada biaya asuransi (charges) menunjukkan kompleksitas hubungan antar variabel, sehingga analisis ini akan fokus pada identifikasi pola dan faktor dominan yang memengaruhi biaya. Hasil akhir dari proyek ini diharapkan dapat memberikan insight bagi perusahaan asuransi dalam memperkirakan risiko klaim dan menyusun strategi penetapan premi yang lebih akurat.

Berikut link file .ipynb pengerjaan proyek untuk kasus ini:

Bab 2: Pembahasan

2.1. Import Libraries

1. Import Libraries

```
[ ] import kagglehub
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score, mean_absolute_error,
mean_absolute_percentage_error, mean_squared_error
import scipy.stats as stats
```

Bagian awal dari skrip analisis ini adalah serangkaian perintah `import`. Dalam konteks pemrograman Python, `import` berfungsi untuk memuat (atau '*mengimpor*') pustaka dan modul eksternal ke dalam lingkungan kerja saat ini. Pustaka dan modul ini berisi kumpulan fungsi, kelas, dan alat yang telah dibuat sebelumnya, yang sangat membantu dalam melakukan tugas-tugas analisis data dan pemodelan tanpa harus membangun semuanya dari awal.

Berikut adalah penjelasan setiap baris impor, dikaitkan dengan perannya dalam proyek prediksi biaya asuransi ini:

- `import kagglehub`: Pustaka ini diimpor untuk memfasilitasi pengambilan data. Dalam proyek ini, `kagglehub` digunakan untuk mengunduh dataset '*insurance.csv*' yang bersumber dari Kaggle secara langsung ke lingkungan komputasi. Hal ini memungkinkan akses mudah ke data yang akan dianalisis.
- `import pandas as pd`: `pandas` adalah pustaka fundamental untuk manipulasi dan analisis data dalam bentuk tabel (*DataFrame*). Setelah dataset berhasil diunduh, `pandas` digunakan untuk membacanya menjadi *DataFrame*, memungkinkan operasi seperti melihat struktur data, memeriksa baris duplikat, menangani nilai yang hilang (walaupun pada dataset ini tidak ada), serta melakukan transformasi data dan *encoding* variabel kategorik. Alias `pd` diberikan untuk efisiensi dalam penulisan kode.
- `import numpy as np`: `numpy` menyediakan dukungan untuk *array* dan matriks multi-dimensi, serta berbagai fungsi matematika tingkat tinggi yang beroperasi pada struktur data tersebut. Dalam proyek ini, `numpy` diperlukan untuk perhitungan numerik, seperti menghitung akar kuadrat saat menghitung

Root Mean Squared Error (RMSE), dan juga menjadi basis bagi banyak operasi komputasi dalam pustaka lain. Alias `np` diberikan.

- `import matplotlib.pyplot as plt`: `matplotlib` adalah pustaka dasar untuk membuat visualisasi statis dan interaktif di Python. Sub-modul `pyplot` menyediakan antarmuka yang sederhana untuk membuat berbagai jenis plot, seperti histogram, boxplot, dan scatter plot. Dalam tahap *Exploratory Data Analysis* (EDA) proyek ini, `matplotlib.pyplot` digunakan untuk memvisualisasikan distribusi variabel dan hubungan antar variabel untuk mendapatkan pemahaman awal tentang data asuransi. Alias `plt` diberikan.
- `import seaborn as sns`: `seaborn` adalah pustaka visualisasi data yang dibangun di atas `matplotlib`. `seaborn` menyediakan fungsi-fungsi untuk membuat grafik statistik yang lebih kompleks dan menarik dengan sedikit kode. Di proyek ini, `seaborn` digunakan untuk menghasilkan visualisasi seperti distribusi dengan kurva densitas, boxplot untuk membandingkan grup, scatter plot dengan pemisah kategori, dan heatmap korelasi, yang semuanya memperkaya analisis visual data asuransi. Alias `sns` diberikan.
- `from sklearn.linear_model import LinearRegression`: `Scikit-learn` (`sklearn`) adalah pustaka terdepan untuk *machine learning* di Python. Dari modul `linear_model`, kelas `LinearRegression` diimpor. Kelas ini mengimplementasikan model regresi linier, yang merupakan algoritma utama yang dipilih dalam proyek ini untuk memprediksi biaya asuransi (`charges`) sebagai fungsi linier dari fitur-fitur input.
- `from sklearn.model_selection import train_test_split`: Dari modul `model_selection` `Scikit-learn`, fungsi `train_test_split` diimpor. Fungsi ini krusial untuk membagi dataset secara acak menjadi dua subset: data pelatihan (*training set*) dan data pengujian (*test set*). Data pelatihan digunakan untuk melatih model, sementara data pengujian digunakan untuk mengevaluasi kinerja model pada data yang belum pernah dilihat sebelumnya, memastikan evaluasi yang objektif.
- `from sklearn.preprocessing import StandardScaler`: Dari modul `preprocessing` `Scikit-learn`, kelas `StandardScaler` diimpor. Kelas ini digunakan untuk menerapkan teknik penskalaan standar (*standardization*) pada fitur-fitur numerik (seperti usia dan BMI). Penskalaan ini mentransformasi data sehingga memiliki rata-rata nol dan deviasi standar satu, sebuah langkah penting dalam pra-pemrosesan data untuk model regresi linier agar semua fitur memberikan kontribusi yang setara tanpa dipengaruhi oleh skala aslinya.
- `from sklearn.metrics import r2_score, mean_absolute_error, mean_absolute_percentage_error, mean_squared_error`: Dari modul `metrics` `Scikit-learn`, berbagai fungsi untuk evaluasi performa model diimpor. Fungsi-fungsi ini meliputi:

- `r2_score`: Untuk menghitung koefisien determinasi (R-squared), yang mengukur proporsi variabilitas variabel target yang dapat dijelaskan oleh model.
- `mean_absolute_error` (MAE): Untuk menghitung rata-rata selisih absolut antara nilai prediksi dan nilai aktual.
- `mean_absolute_percentage_error` (MAPE): Untuk menghitung rata-rata persentase selisih absolut, memberikan interpretasi kesalahan dalam persentase.
- `mean_squared_error` (MSE): Untuk menghitung rata-rata kuadrat selisih, memberikan penalti yang lebih besar untuk kesalahan yang besar. Metrik-metrik ini digunakan untuk mengukur seberapa baik model regresi linier yang dibangun dalam memprediksi biaya asuransi pada data pengujian.
- `import scipy.stats as stats`: Dari pustaka `scipy` (SciPy, pustaka untuk komputasi ilmiah dan teknis), modul `stats` diimpor. Modul ini berisi berbagai alat statistik dan fungsi distribusi probabilitas. Dalam analisis residual proyek ini, `scipy.stats` digunakan untuk membuat Q-Q plot (*Quantile-Quantile plot*) dari residual, yang merupakan metode visual untuk memeriksa apakah residual mengikuti distribusi normal, salah satu asumsi kunci dalam regresi linier. Alias `stats` diberikan.

Dengan mengimpor pustaka-pustaka ini di awal skrip, semua alat dan fungsi yang diperlukan untuk setiap tahapan proyek — mulai dari pengambilan data, pra-pemrosesan, analisis eksplorasi, pembangunan model, hingga evaluasi kinerja model — menjadi tersedia dan siap digunakan.

2.2. Load Dataset

```
try:
    path = kagglehub.dataset_download("mirichoi0218/insurance")
    csv_file_path = f'{path}/insurance.csv'
    print(f'Dataset loaded from Kaggle Hub: {csv_file_path}')
except Exception as e:
    print(f'Failed to load from Kaggle Hub: {e}')
    csv_file_path = 'insurance.csv'
    print(f'Attempting to load from local path: {csv_file_path}')

Dataset loaded from Kaggle Hub: /kaggle/input/insurance/insurance.csv

[ ] df = pd.read_csv(csv_file_path)
df
```

```
[ ] df = pd.read_csv(csv_file_path)
df
```

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.92400
1	18	male	33.770	1	no	southeast	1725.55230
2	28	male	33.000	3	no	southeast	4449.46200
3	33	male	22.705	0	no	northwest	21984.47061
4	32	male	28.880	0	no	northwest	3866.85520
...
1333	50	male	30.970	3	no	northwest	10600.54830
1334	18	female	31.920	0	no	northeast	2205.98080
1335	18	female	36.850	0	no	southeast	1629.83350
1336	21	female	25.800	0	no	southwest	2007.94500
1337	61	female	29.070	0	yes	northwest	29141.36030

1338 rows x 7 columns

2.3. Preprocessing (Penanganan Missing Value dan Duplikasi Data)

2.3.1. Cek Duplikasi Data

```

3. Preprocessing (Penanganan Missing Value dan Duplikasi Data)

3.1. Cek Duplikasi Data

[ ] df.duplicated().sum()
np.int64(1)

Terdapat 1 baris data yang duplikat, maka kita bisa menghapus baris data yang duplikat itu.

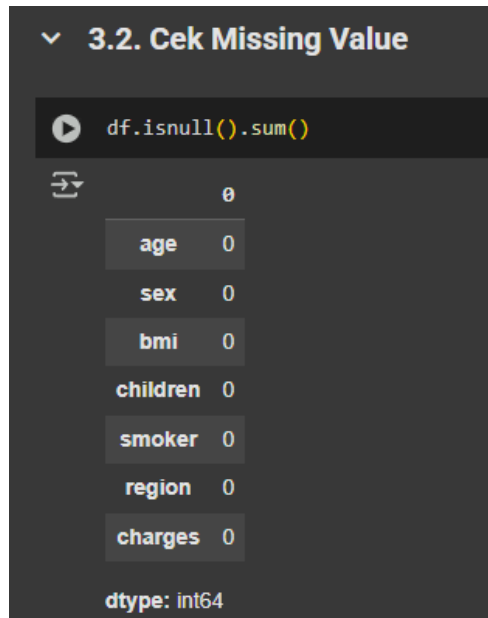
df.drop_duplicates(inplace=True)

Marimkita cek ulang untuk memastikan bahwa sudah tidak ada lagi data yang duplikat.

[ ] print(f'Jumlah data yang duplikat: {df.duplicated().sum()}')
Jumlah data yang duplikat: 0

```


2.3.2. Cek Missing Value



```
3.2. Cek Missing Value

df.isnull().sum()

0
age      0
sex      0
bmi      0
children 0
smoker   0
region   0
charges  0
dtype: int64
```

Setiap kolom menampilkan angka 0: Artinya tidak ada missing value di seluruh kolom (age, sex, bmi, dll.).

Mengapa harus mengecek missing value pada data ini?

1. Modeling/Regresi

Algoritma machine learning seperti Linear Regression tidak bisa bekerja dengan missing value.

Jika ada NaN, perlu di-handle sebelum training model.

2. Analisis Statistik

Missing value dapat mengganggu perhitungan mean, median, atau korelasi.

2.4. Exploratory Data Analysis (EDA)

2.4.1. Deskripsi/Ringkasan Data

4. Exploratory Data Analysis (EDA)

4.1. Deskripsi/Ringkasan Data

```
df.describe()
```

	age	bmi	children	charges
count	1337.000000	1337.000000	1337.000000	1337.000000
mean	39.222139	30.663452	1.095737	13279.121487
std	14.044333	6.100468	1.205571	12110.359656
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.290000	0.000000	4746.344000
50%	39.000000	30.400000	1.000000	9386.161300
75%	51.000000	34.700000	2.000000	16657.717450
max	64.000000	53.130000	5.000000	63770.428010

Pernyataan `df.describe()` adalah metode dalam pustaka `pandas` yang secara otomatis menghitung dan menampilkan ringkasan statistik deskriptif untuk semua kolom yang berisi data numerik dalam `DataFrame` `df`. Output dari perintah ini memberikan gambaran cepat mengenai distribusi nilai dalam setiap kolom numerik, yang sangat penting di awal tahap Exploratory Data Analysis (EDA).

2.4.2. Informasi Data (kolom)

4.2. Informasi Data(kolom)

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>  
Index: 1337 entries, 0 to 1337  
Data columns (total 7 columns):  
#   Column      Non-Null Count  Dtype  
---  ---  
0   age         1337 non-null   int64  
1   sex         1337 non-null   object  
2   bmi         1337 non-null   float64  
3   children    1337 non-null   int64  
4   smoker      1337 non-null   object  
5   region      1337 non-null   object  
6   charges     1337 non-null   float64  
dtypes: float64(2), int64(2), object(3)  
memory usage: 83.6+ KB
```

Output `df.info()` dan Relevansinya dalam Proyek Prediksi Biaya Asuransi

Output dari `df.info()` memberikan informasi penting yang langsung relevan dengan langkah-langkah selanjutnya dalam proyek prediksi biaya asuransi ini:

- `<class 'pandas.core.frame.DataFrame'>`: Ini mengkonfirmasi bahwa objek df memang adalah sebuah DataFrame pandas.
- Index: 1337 entries, 0 to 1337: Menunjukkan jumlah baris data dalam DataFrame ini setelah duplikasi dihapus (Kita sebelumnya menghapus 1 duplikat dari 1338 baris asli, sehingga tersisa 1337 baris). Ini adalah jumlah sampel individu dalam dataset yang akan kita gunakan. Rentang indeksnya dari 0 hingga 1337.
- Data columns (total 7 columns):: Ini memberi tahu kita bahwa DataFrame ini memiliki total 7 kolom (atau variabel).

Bagian tabel di bawahnya memberikan detail per kolom:

1. #: Nomor indeks kolom.
2. Column: Nama kolom (misalnya, age, sex, bmi, dll.). Ini adalah nama-nama fitur dan target yang akan kita gunakan.
3. Non-Null Count: Menunjukkan jumlah nilai yang tidak kosong (non-missing) di setiap kolom. Untuk semua kolom (age hingga charges), nilainya adalah 1337, yang sama dengan jumlah total baris. Ini sangat penting karena mengkonfirmasi temuan kita sebelumnya bahwa tidak ada nilai yang hilang (missing values) dalam dataset ini. Ini berarti kita tidak perlu melakukan imputasi atau penghapusan baris/kolom karena missing value.
4. Dtype: Menunjukkan tipe data dari nilai-nilai dalam kolom tersebut.
 - int64: Bilangan bulat (seperti age dan children). Ini adalah tipe data numerik diskrit.
 - float64: Bilangan riil/desimal (seperti bmi dan charges). Ini adalah tipe data numerik kontinu. Variabel target kita, charges, bertipe float, sesuai untuk masalah regresi.
 - object: Umumnya menandakan string atau campuran tipe data. Dalam kasus ini, ini menunjukkan kolom-kolom kategorik (sex, smoker, region) yang berisi teks.
- dtypes: float64(2), int64(2), object(3): Ini adalah ringkasan jumlah kolom berdasarkan tipe datanya. Ada 2 kolom float, 2 kolom integer, dan 3 kolom bertipe object. Informasi tipe data ini krusial karena memengaruhi cara kita memproses kolom-kolom ini. Kolom numerik (int64, float64) bisa langsung digunakan dalam perhitungan atau diskalakan (StandardScaler), sementara kolom object (kategorik) perlu di-encoding (seperti yang kita lakukan dengan `pd.get_dummies`) sebelum dimasukkan ke dalam model regresi yang membutuhkan input numerik.
- memory usage: 83.6+ KB: Memberikan perkiraan penggunaan memori oleh DataFrame.

Singkatnya, output `df.info()` adalah `"kartu identitas"` dataset ini. Dalam proyek ini, ini memberikan konfirmasi vital bahwa data ini:

1. Bersih dari missing values
2. Dengan jelas mengidentifikasi mana kolom numerik (siap untuk perhitungan/penskalaan) dan mana kolom kategorik (membutuhkan encoding)
3. Membimbing langkah-langkah pra-pemrosesan data ini selanjutnya

2.4.3. Distribusi Variabel Kategorik

4.3. Distribusi Variabel Kategorik

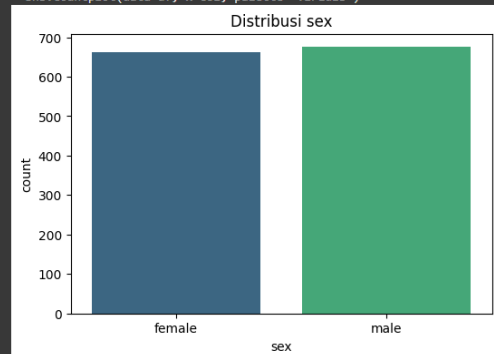
```
print("\nDistribusi variabel kategorik:")
for col in ['sex', 'smoker', 'region', 'children']:
    print(f"\n--- {col} ---")
    print(df[col].value_counts(normalize=True) * 100)
    plt.figure(figsize=(6, 4))
    sns.countplot(data=df, x=col, palette='viridis')
    plt.title(f'Distribusi {col}')
    plt.show()
```

Distribusi variabel kategorik:

```
--- sex ---
sex
male    50.486163
female  49.513837
Name: proportion, dtype: float64
<ipython-input-10-25baba5d9d8f>:6: FutureWarning:
```

Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

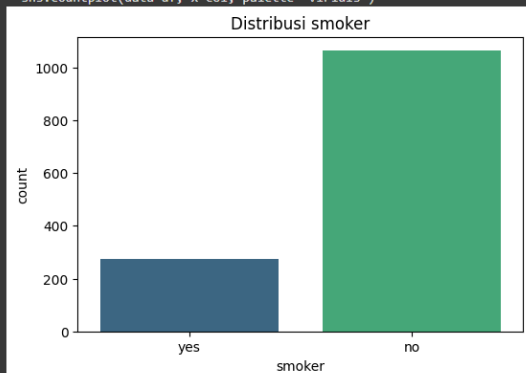
```
sns.countplot(data=df, x=col, palette='viridis')
```



```
--- smoker ---
smoker
no    79.506358
yes   20.493642
Name: proportion, dtype: float64
<ipython-input-10-25baba5d9d8f>:6: FutureWarning:
```

Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

```
sns.countplot(data=df, x=col, palette='viridis')
```



```

--- region ---
region
southeast    27.225131
southwest    24.308153
northwest    24.233358
northeast    24.233358
Name: proportion, dtype: float64
<ipython-input-10-25baba5d9d8f>:6: FutureWarning:

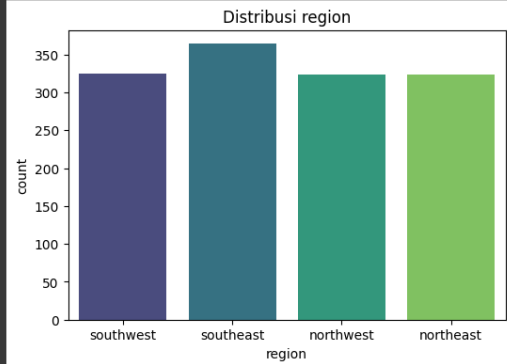
```

Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

```

sns.countplot(data=df, x=col, palette='viridis')

```



```

--- children ---

```

```

--- children ---
children
0    42.857143
1    24.233358
2    17.950636
3    11.742708
4     1.869858
5     1.346298
Name: proportion, dtype: float64
<ipython-input-10-25baba5d9d8f>:6: FutureWarning:

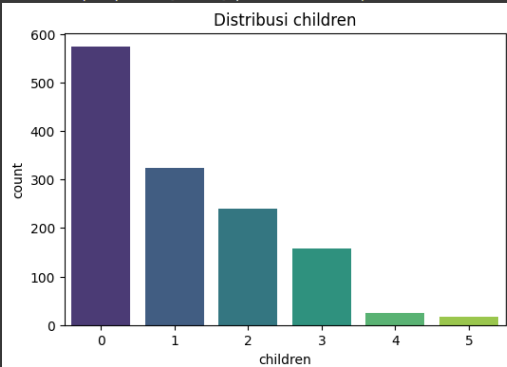
```

Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.

```

sns.countplot(data=df, x=col, palette='viridis')

```



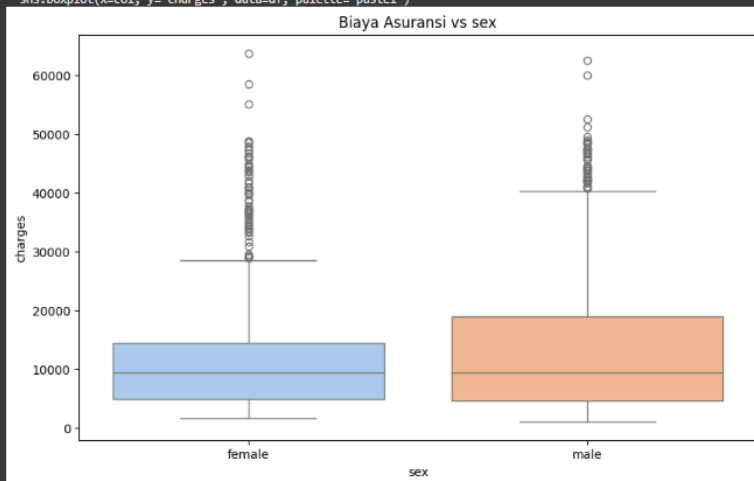
2.4.4. Hubungan Variabel Kategorik dengan Target (charges)

4.4. Hubungan Variabel Kategorik dengan Target (charges)

```
category_cols = ['sex', 'smoker', 'region', 'children']
for col in category_cols:
    plt.figure(figsize=(10, 6))
    sns.boxplot(x=col, y='charges', data=df, palette='pastel')
    plt.title(f'Biaya Asuransi vs {col}')
    plt.show()
```

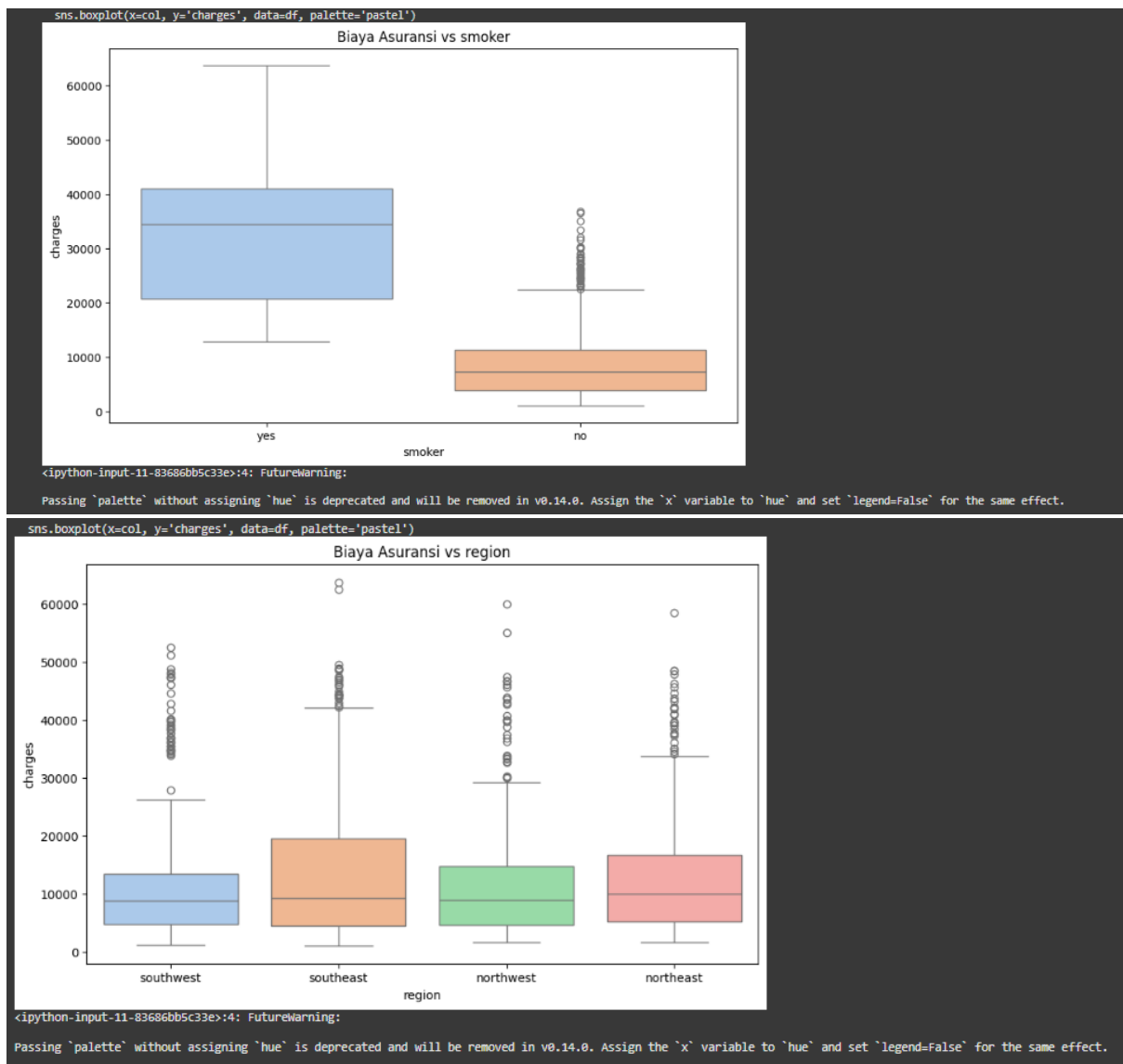
<ipython-input-11-83686bb5c33e>:4: FutureWarning:

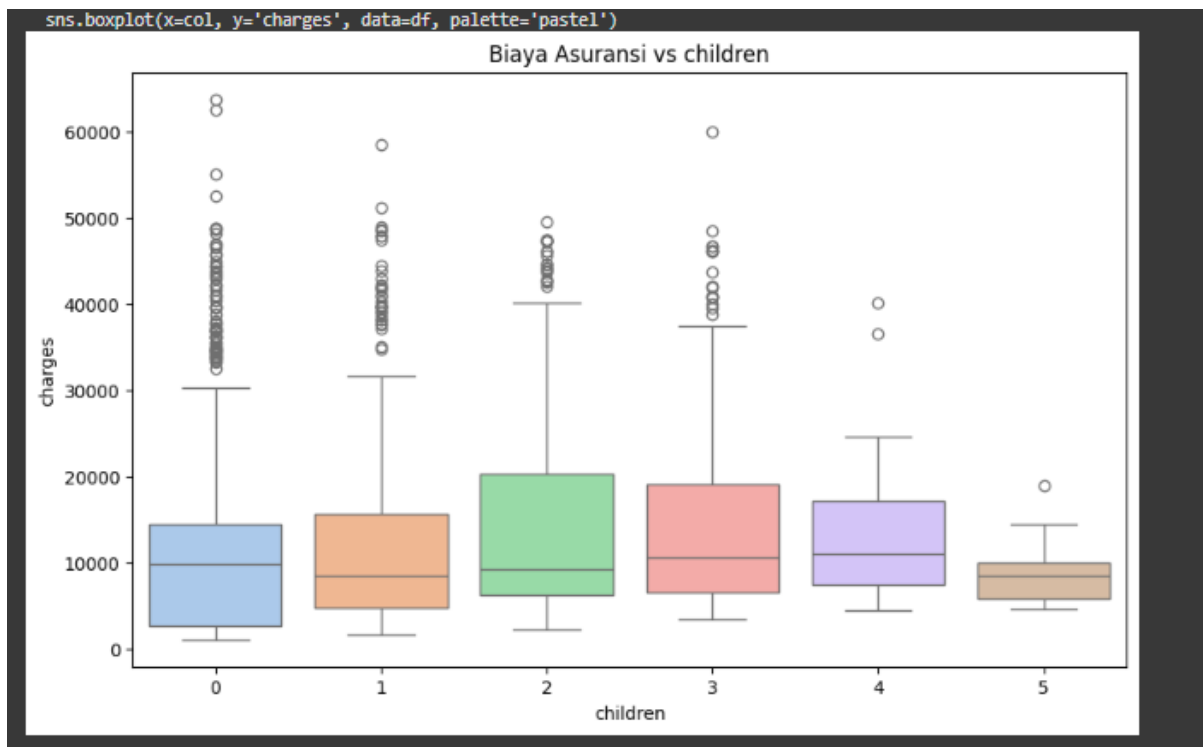
Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.
sns.boxplot(x=col, y='charges', data=df, palette='pastel')



<ipython-input-11-83686bb5c33e>:4: FutureWarning:

Passing 'palette' without assigning 'hue' is deprecated and will be removed in v0.14.0. Assign the 'x' variable to 'hue' and set 'legend=False' for the same effect.





2.4.5. Distribusi Variabel Numerik

4.5. Distribusi Variabel Numerik

```
from IPython.display import display, HTML

# List kolom numerik dan judulnya
numerical_cols = ['charges', 'bmi', 'age']
titles = ['Distribusi Biaya Asuransi', 'Distribusi Body Mass Index (BMI)', 'Distribusi Umur']

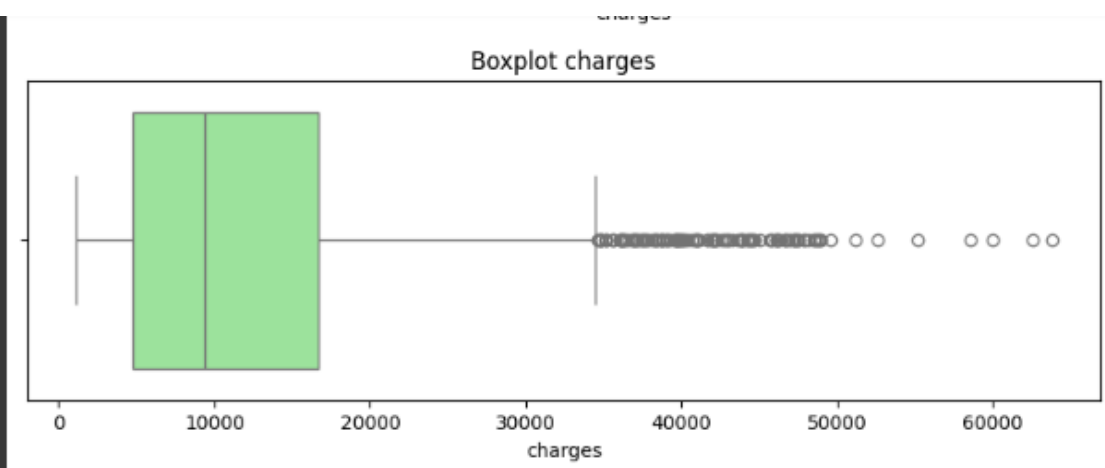
# Membuat container untuk output scrollable
display(HTML("<div style='height: 400px; overflow-y: scroll; border: 1px solid #ccc; padding: 10px;'>"))

# Loop melalui setiap kolom numerik
for col, title in zip(numerical_cols, titles):
    print(f"\n=== {title} ===")
    print(f"Statistik deskriptif {col}:")
    print(df[col].describe())

    # Visualisasi
    plt.figure(figsize=(10, 5))
    sns.histplot(df[col], kde=True, color='skyblue', bins=20)
    plt.title(title)
    plt.xlabel(col)
    plt.ylabel('Frekuensi')
    plt.grid(axis='y', alpha=0.3)
    plt.show()

    # Boxplot tambahan untuk melihat outliers
    plt.figure(figsize=(10, 3))
    sns.boxplot(x=df[col], color='lightgreen')
    plt.title(f'Boxplot {col}')
    plt.show()

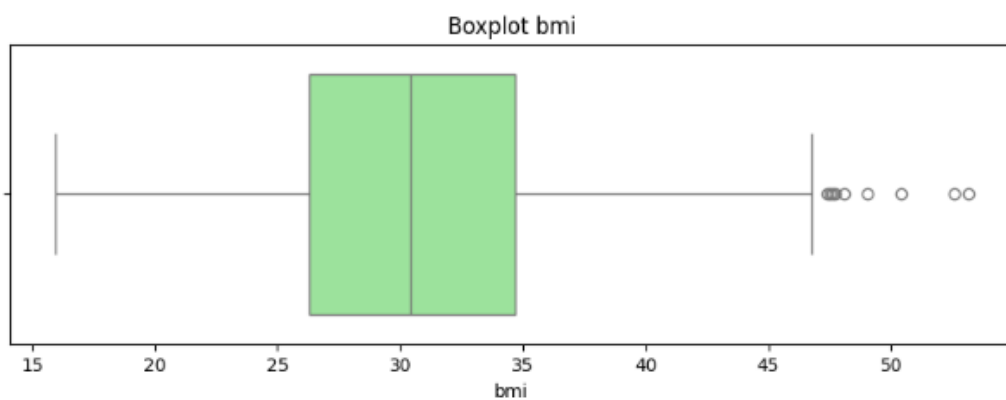
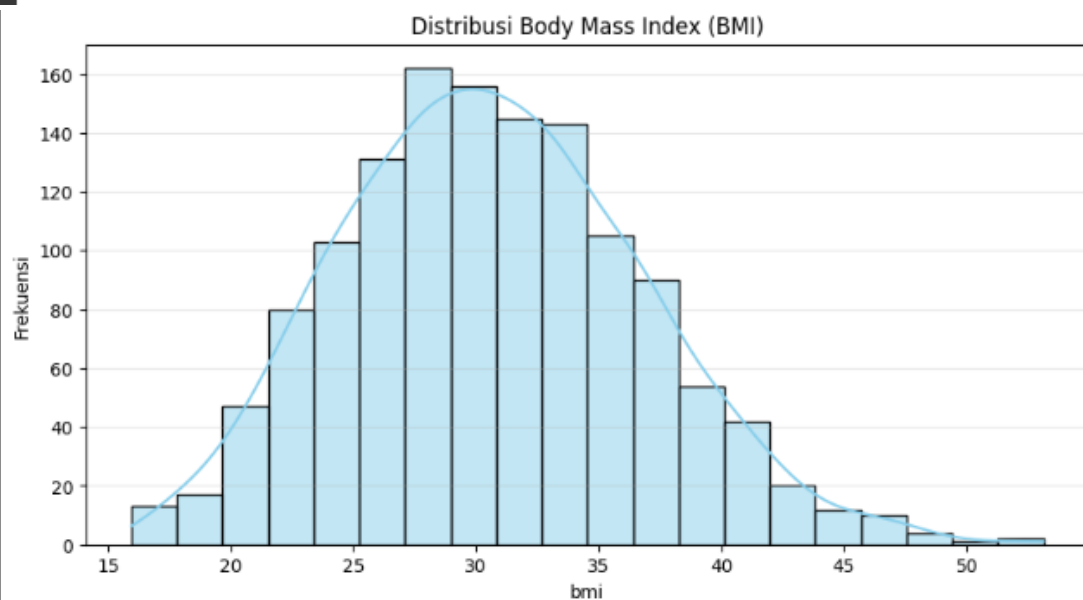
display(HTML("</div>"))
```

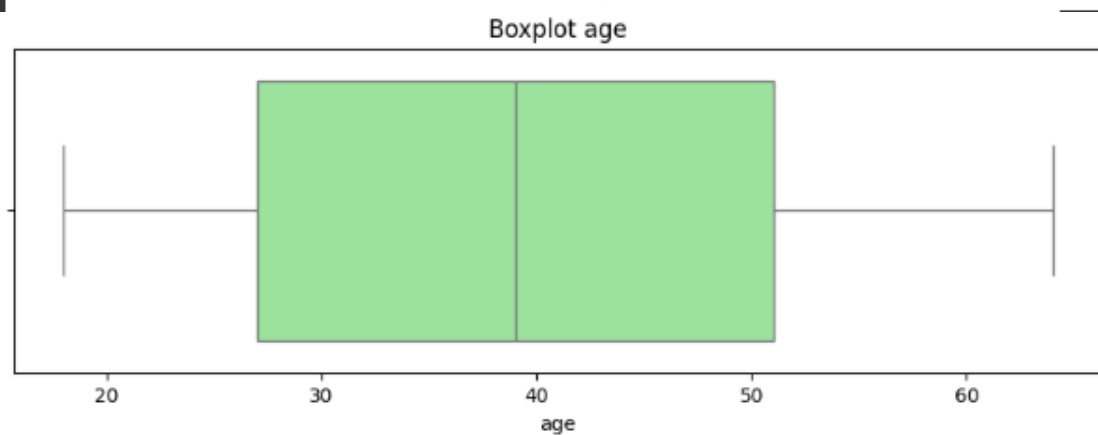
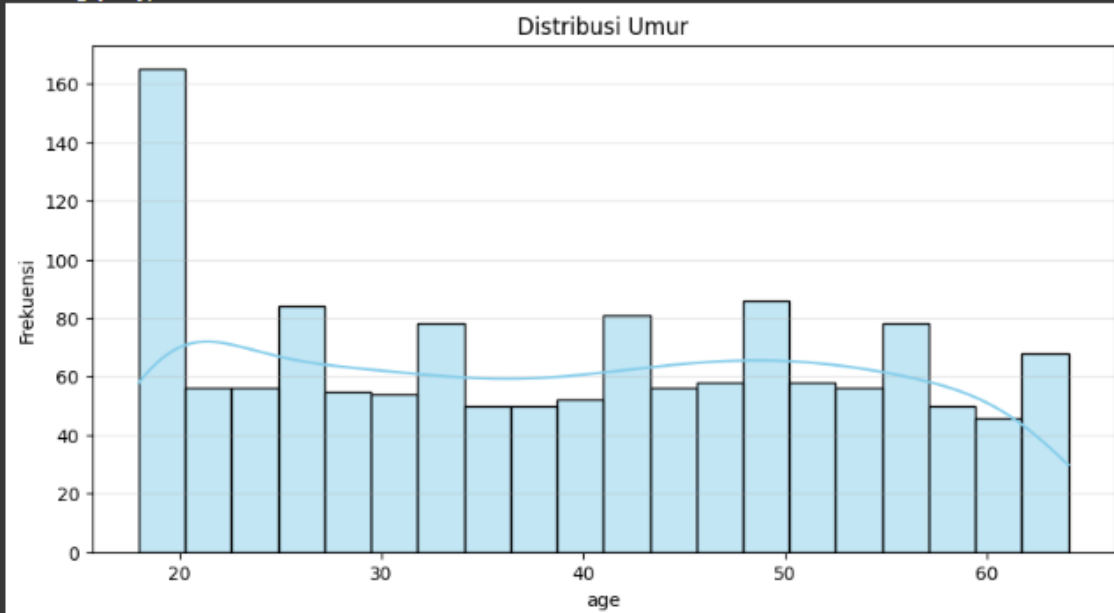
```

=== Distribusi Body Mass Index (BMI) ===
Statistik deskriptif bmi:
count    1337.000000
mean      30.663452
std        6.100468
min       15.960000
25%       26.290000
50%       30.400000
75%       34.700000
max       53.130000
Name: bmi, dtype: float64

```



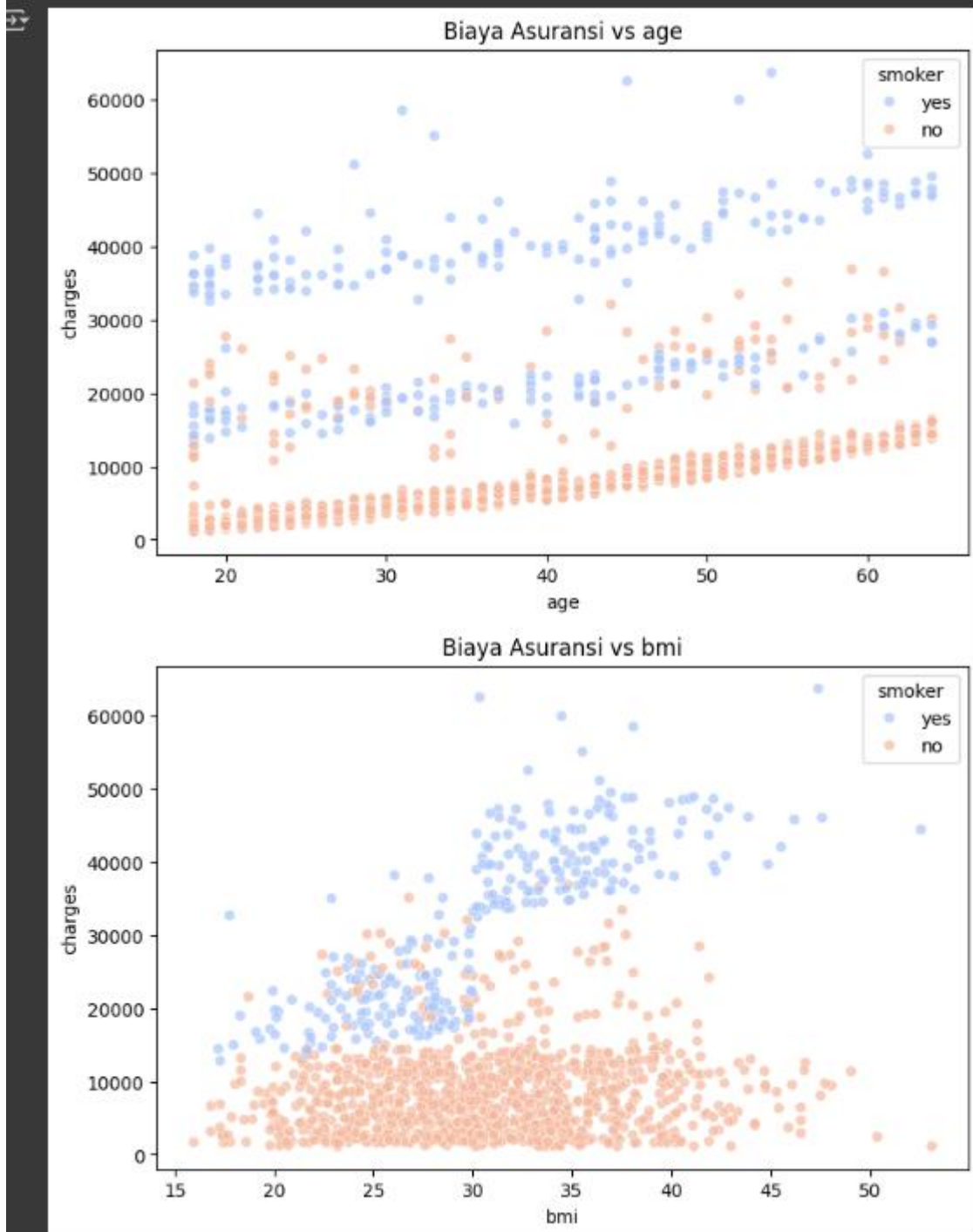
```
=== Distribusi Umur ===  
Statistik deskriptif age:  
count    1337.000000  
mean      39.222139  
std       14.044333  
min       18.000000  
25%       27.000000  
50%       39.000000  
75%       51.000000  
max       64.000000  
Name: age, dtype: float64
```



2.4.6. Hubungan Variabel Numerik dengan Target (charges)

▼ f. Hubungan Variabel Numerik dengan Target (charges)

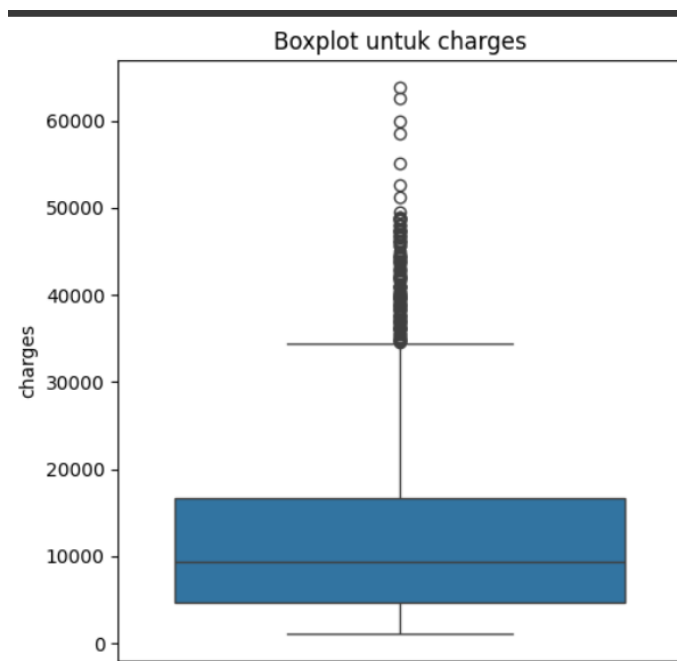
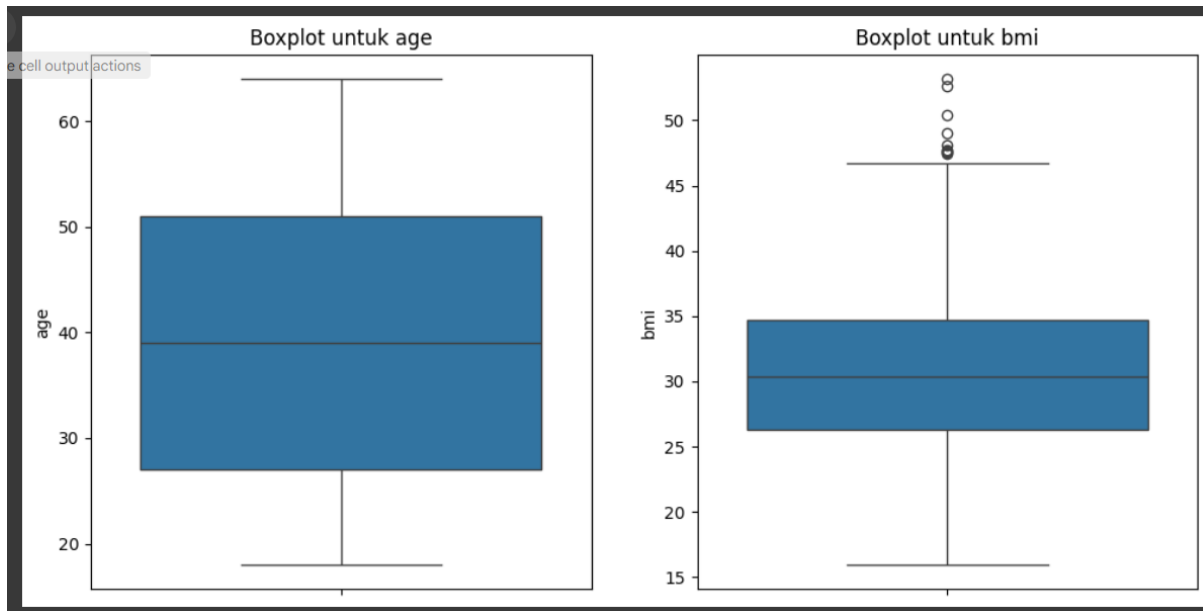
```
numerical_cols_for_scatter = ['age', 'bmi'] # children bisa dianggap kategorik/ordinal di sini
for col in numerical_cols_for_scatter:
    plt.figure(figsize=(8, 5))
    sns.scatterplot(x=col, y='charges', data=df, hue='smoker', palette='coolwarm', alpha=0.7)
    plt.title(f'Biaya Asuransi vs {col}')
    plt.show()
```



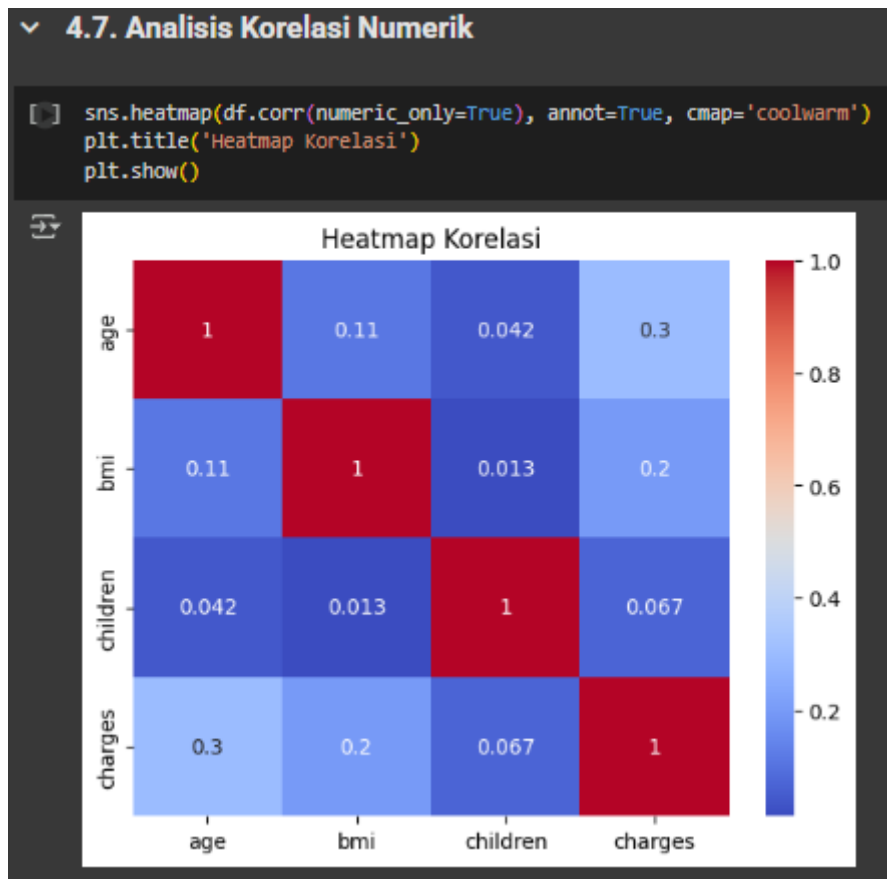
2.4.6. Outlier Detection (Optional)



```
numerical_cols = ['age', 'bmi', 'charges']  
plt.figure(figsize=(15, 5))  
for i, col in enumerate(numerical_cols):  
    plt.subplot(1, 3, i+1)  
    sns.boxplot(y=df[col])  
    plt.title(f'Boxplot untuk {col}')  
plt.tight_layout()  
plt.show()
```



2.4.7. Analisis Korelasi Numerik



Berikut penjelasan hasil analisis korelasi antara variabel-variabel numerik dalam dataset asuransi kesehatan:

1. Korelasi Umum Antar Variabel

- Usia (age) dengan Biaya (charges):
Menunjukkan korelasi positif sedang (0.3). Artinya, semakin tua usia tertanggung, cenderung memiliki biaya asuransi yang lebih tinggi. Hal ini masuk akal karena risiko kesehatan umumnya meningkat seiring usia.
- BMI dengan Biaya (charges):
Korelasi positif lemah (0.2). Nilai BMI yang lebih tinggi sedikit berkaitan dengan biaya asuransi yang lebih besar, meskipun pengaruhnya tidak terlalu kuat.
- Jumlah Anak (children) dengan Variabel Lain:
 - Korelasi sangat lemah dengan usia (0.042) dan BMI (0.013), menunjukkan hampir tidak ada hubungan.
 - Korelasi negatif lemah dengan biaya (-0.2), artinya jumlah anak tidak terlalu memengaruhi biaya asuransi dalam dataset ini.

2. Pola Menarik

- Korelasi Terkuat:
Usia dengan biaya asuransi (0.3) adalah korelasi tertinggi, menjadikan usia sebagai prediktor potensial yang penting dalam model prediksi biaya.
- BMI:

Meski memiliki korelasi lemah dengan biaya (0.2), kombinasi dengan faktor lain (seperti status merokok) mungkin berpengaruh lebih signifikan.

- Anak (children):
Korelasi negatif dengan biaya (-0.2) mungkin mengejutkan, tetapi nilai yang kecil menunjukkan pengaruhnya minimal. Perlu investigasi lebih lanjut apakah ada faktor lain yang berperan.

3. Implikasi untuk Proyek Ini

- Variabel Prediktor Utama:
Usia layak dipertimbangkan sebagai fitur utama dalam model regresi karena korelasinya yang signifikan dengan biaya.
- Transformasi Data:
Kolom seperti BMI mungkin perlu ditransformasi (contoh: dikelompokkan dalam kategori) untuk meningkatkan korelasi jika digunakan dalam model.
- Variabel Kategorikal:
Variabel seperti smoker (tidak termasuk dalam heatmap karena non-numerik) mungkin memiliki pengaruh besar. Pastikan untuk meng-encode variabel ini dalam pra-pemrosesan.
- Multikolinearitas:
Tidak ada korelasi tinggi antar variabel independen (misal: age dan BMI hanya 0.11), sehingga risiko multikolinearitas rendah.

Catatan Penting

Nilai korelasi di atas berdasarkan data mentah. Setelah pra-pemrosesan (encoding variabel kategorikal, penskalaan), pola korelasi mungkin berubah.

Korelasi tidak berarti sebab-akibat. Faktor lain (seperti status merokok atau kondisi medis) mungkin menjadi penyebab sebenarnya dari biaya tinggi.

2.5. Preprocessing (Persiapan untuk model Regresi)

```
5.1. Encoding variabel kategorik

[ ] df_encoded = pd.get_dummies(
    df,
    columns=['sex', 'smoker', 'region'],
    drop_first=True # Mengurangi dimensi dengan menghilangkan kolom pertama
)
```

Proses One-Hot Encoding ini sangat penting dalam proyek prediksi biaya asuransi ini karena:

- Menyiapkan Data untuk Model: Model LinearRegression di Scikit-learn memerlukan semua input fitur dalam format numerik. One-Hot Encoding mengubah variabel kategorikal menjadi format numerik (biner 0/1) yang dapat diproses oleh model.
- Mempertahankan Informasi: Teknik ini memastikan bahwa informasi yang terkandung dalam variabel kategorikal (yaitu, individu termasuk dalam kategori tertentu atau tidak) tetap ada dan dapat digunakan oleh model untuk membuat prediksi.

2.5.2. Konfirmasi Hasil Encoding

Bagian ini terdiri dari tiga baris kode terpisah yang masing-masing berfungsi untuk menampilkan informasi berbeda mengenai DataFrame `df_encoded` yang baru saja dibuat melalui One-Hot Encoding. Tujuannya adalah untuk memastikan bahwa kolom-kolom kategorik asli telah diganti dengan kolom-kolom biner yang dihasilkan oleh `pd.get_dummies()`, dan bahwa struktur DataFrame sudah sesuai.

Kolom setelah encoding:

```
[ ] df_encoded.columns
```

```
Index(['age', 'bmi', 'children', 'charges', 'sex_male', 'smoker_yes',  
      'region_northwest', 'region_southeast', 'region_southwest'],  
      dtype='object')
```

DataFrame setelah encoding (5 baris pertama):

```
[ ] df_encoded.head()
```

```
age    bmi  children    charges  sex_male  smoker_yes  region_northwest  region_southeast  region_southwest
0    19  27.900        0  16884.92400    False         True             False             False              True
1    18  33.770        1   1725.55230     True         False             False             True              False
2    28  33.000        3   4449.46200     True         False             False             True              False
3    33  22.705        0  21984.47061     True         False             True              False              False
4    32  28.880        0   3866.85520     True         False             True              False              False
```

Info DataFrame setelah encoding:

```
[ ] df_encoded.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 1337 entries, 0 to 1337
Data columns (total 9 columns):
#   Column                Non-Null Count  Dtype
---  -
0   age                   1337 non-null   int64
1   bmi                   1337 non-null   float64
2   children              1337 non-null   int64
3   charges               1337 non-null   float64
4   sex_male              1337 non-null   bool
5   smoker_yes            1337 non-null   bool
6   region_northwest      1337 non-null   bool
7   region_southeast      1337 non-null   bool
8   region_southwest      1337 non-null   bool
dtypes: bool(5), float64(2), int64(2)
memory usage: 58.8 KB
```

Secara keseluruhan, ketiga baris kode ini bersama-sama berfungsi sebagai pemeriksaan kualitas setelah One-Hot Encoding, memastikan bahwa data kita telah berhasil ditransformasikan ke format yang siap untuk digunakan dalam pembangunan model prediksi biaya asuransi.

2.5.3. Memisahkan fitur (X) dan target (y)

Dalam pembangunan model machine learning, terutama untuk masalah regresi seperti prediksi biaya asuransi, dataset perlu dibagi menjadi dua bagian utama:

1. Fitur (Features), sering dilambangkan dengan X: Ini adalah kolom-kolom input yang akan digunakan model untuk "belajar" dan membuat prediksi. Fitur-fitur ini adalah karakteristik dari setiap individu (usia, BMI, jenis kelamin, status perokok, wilayah, jumlah anak, dll.).
2. Target (Target Variable atau Label), sering dilambangkan dengan y: Ini adalah kolom output yang ingin diprediksi oleh model. Dalam proyek ini, targetnya adalah biaya asuransi (charges).

Kode yang kita gunakan untuk melakukan pemisahan ini adalah:

```
[ ] X = df_encoded.drop('charges', axis=1)
    y = df_encoded['charges']
```

Fitur (X) - 5 baris pertama:

Kita akan melihat tabel yang menampilkan 5 baris pertama dari DataFrame X. Tabel ini akan berisi semua kolom numerik dan biner yang sebelumnya ada di df_encoded, kecuali kolom charges. Anda akan melihat kolom seperti age, bmi, children, sex_male, smoker_yes, dan kolom region_....

Ini mengkonfirmasi bahwa X berhasil dibuat dan hanya berisi fitur-fitur input yang akan digunakan model untuk memprediksi.

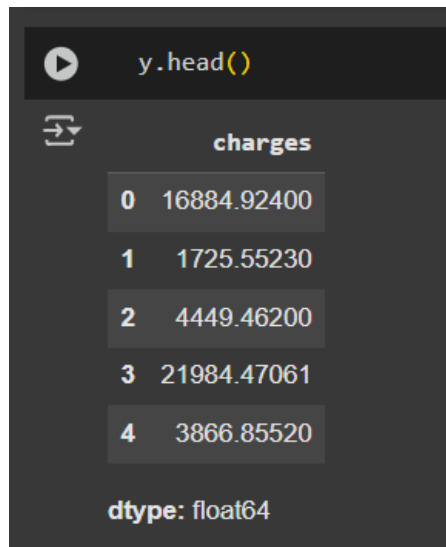
```
[ ] X.head()
```

	age	bmi	children	sex_male	smoker_yes	region_northwest	region_southeast	region_southwest
0	19	27.900	0	False	True	False	False	True
1	18	33.770	1	True	False	False	True	False
2	28	33.000	3	True	False	False	True	False
3	33	22.705	0	True	False	True	False	False
4	32	28.880	0	True	False	True	False	False

Target (y) - 5 baris pertama:

Kita akan melihat output berupa Series (bukan tabel) yang menampilkan 5 nilai pertama dari variabel target charges. Output ini hanya akan berisi angka-angka yang merepresentasikan biaya asuransi.

Ini mengkonfirmasi bahwa y berhasil dibuat dan hanya berisi variabel yang akan menjadi target prediksi model.



	charges
0	16884.92400
1	1725.55230
2	4449.46200
3	21984.47061
4	3866.85520

dtype: float64

Dengan melakukan pemisahan ini, kita secara eksplisit mendefinisikan mana data yang akan menjadi masukan (fitur) dan mana data yang akan menjadi keluaran (target) untuk model regresi linear kita. Ini adalah langkah fundamental sebelum membagi data menjadi set pelatihan dan pengujian, serta melatih model.

2.5.4. Split data training-testing (80:20)

Setelah data pra-pemrosesan (seperti One-Hot Encoding dan pemisahan fitur/target) selesai, langkah krusial sebelum melatih model adalah membagi dataset yang ada menjadi dua bagian terpisah: satu bagian untuk melatih model (training set) dan satu bagian lainnya untuk menguji seberapa baik model tersebut bekerja pada data yang belum pernah dilihat sebelumnya (test set). Tujuan pembagian ini adalah untuk mendapatkan evaluasi model yang objektif dan menghindari overfitting (di mana model terlalu "menghafal" data pelatihan tetapi buruk dalam memprediksi data baru).

```
[ ] X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.2,
    random_state=42
)
```

Dengan membagi data seperti ini, kita memitigasi risiko overfitting dan memastikan bahwa metrik evaluasi performa model ini benar-benar mencerminkan kemampuan generalisasi model pada data yang tidak digunakan selama pelatihan.

2.5.5. Penskalaan Fitur Numerik

Sebelum melatih model regresi linier, seringkali disarankan untuk melakukan penskalaan (scaling) pada fitur-fitur numerik. Fitur-fitur seperti usia (age), BMI (bmi), dan jumlah anak (children) memiliki rentang nilai yang berbeda-beda. Misalnya, usia bisa berkisar dari

belasan hingga puluhan, sementara BMI mungkin berkisar dari 15 hingga 50, dan jumlah anak dari 0 hingga 5. Model regresi linier, meskipun tidak sesensitif model berbasis jarak, terkadang dapat sedikit ditingkatkan performanya jika fitur-fitur numerik berada dalam skala yang serupa. Selain itu, standarisasi ini merupakan praktik umum dan penting untuk model lain di Scikit-learn.

Metode penskalaan yang umum digunakan adalah Standardization (Standard Scaler). Proses ini mengubah nilai setiap fitur sehingga rata-ratanya menjadi 0 dan standar deviasinya menjadi 1.

```
[ ] numerical_features = ['age', 'bmi', 'children']
```

Pastikan fitur-fitur ini ada di X_train.columns

```
[ ] existing_numerical_features = [col for col in numerical_features if col in
X_train.columns]

if existing_numerical_features:
    scaler = StandardScaler()

    # Fit scaler HANYA pada X_train dan transform X_train & X_test
    X_train[existing_numerical_features] = scaler.fit_transform(X_train
[existing_numerical_features])
    X_test[existing_numerical_features] = scaler.transform(X_test
[existing_numerical_features])

    print("\nX_train setelah penskalaan (5 baris pertama):")
    print(X_train.head())
else:
    print("\nTidak ada fitur numerik yang dipilih untuk penskalaan atau fitur
tidak ditemukan.")
```



```
X_train setelah penskalaan (5 baris pertama):
      age      bmi  children  sex_male  smoker_yes  region_northwest \
1114 -1.157680 -0.996928 -0.907908      True      False             False
968  -1.300619 -0.792762  0.766904      True      False             False
599   0.914926  1.154664  0.766904     False      False             True
170   1.701087  1.806837 -0.907908      True      False             False
275   0.557580 -0.651417  0.766904     False      False             False

      region_southeast  region_southwest
1114              False              False
968              False              False
599              False              False
170              True              False
275              False              False
```

2.5.6. Verifikasi Bentuk Data Setelah Split

Setelah membagi dataset fitur (X) dan target (y) menjadi set pelatihan (X_train, y_train) dan set pengujian (X_test, y_test) menggunakan fungsi `train_test_split()`, langkah penting berikutnya adalah memverifikasi apakah pembagian tersebut menghasilkan jumlah baris dan kolom yang sesuai dengan proporsi yang diinginkan. Memastikan dimensi data (bentuk atau shape) ini sudah benar sangat penting sebelum melatih model, karena dimensi yang salah dapat menyebabkan error saat proses pelatihan.

```
[ ] print("\nBentuk data training dan testing:")
    print(f"X_train shape: {X_train.shape}")
    print(f"X_test shape: {X_test.shape}")
    print(f"y_train shape: {y_train.shape}")
    print(f"y_test shape: {y_test.shape}")
```



```
Bentuk data training dan testing:
X_train shape: (1069, 8)
X_test shape: (268, 8)
y_train shape: (1069,)
y_test shape: (268,)
```

2.6. Pemodelan dan Simulasi

2.6.1. Inisialisasi dan training model

Setelah dataset siap — sudah dibersihkan, di-encode, diskalakan (fitur numerik), dan dibagi menjadi set pelatihan (X_train, y_train) serta set pengujian (X_test, y_test) — langkah

berikutnya adalah membangun model prediksi. Dalam proyek ini, kita memilih model Regresi Linier untuk memprediksi biaya asuransi (charges).

```
[ ] y_pred = model.predict(X_test)
```

Setelah baris kode ini dieksekusi, objek model kini berisi semua informasi yang dibutuhkan untuk melakukan prediksi pada data baru. Model ini sekarang memiliki koefisien yang spesifik untuk setiap fitur (age, bmi, children, sex_male, smoker_yes, region...) dan sebuah nilai intercept yang ditemukan selama proses pelatihan. Model ini siap untuk digunakan pada tahap selanjutnya, yaitu membuat prediksi pada data pengujian (X_test).

2.6.2. Prediksi data testing

Setelah model Regresi Linier kita berhasil dilatih menggunakan set data pelatihan (X_train dan y_train), langkah berikutnya adalah menggunakan model yang sudah "belajar" tersebut untuk membuat prediksi. Proses prediksi ini dilakukan pada set data pengujian (X_test), yaitu data yang belum pernah dilihat oleh model selama proses pelatihan.

```

r2 = r2_score(y_test, y_pred)
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
try:
    mape = mean_absolute_percentage_error(y_test, y_pred)
    print(f"MAPE: {mape:.2%}")
except ValueError: # Handle jika ada y_test = 0
    print("MAPE tidak dapat dihitung karena ada nilai target nol.")

print(f"R-squared\t\t: {r2:.3f}")
print(f"MAE\t\t\t: ${mae:.2f}")
print(f"MSE\t\t\t: {mse:.2f}")
print(f"RMSE\t\t\t: ${rmse:.2f}")

```

MAPE	: 41.40%
R-squared	: 0.807
MAE	: \$4177.05
MSE	: 35478020.68
RMSE	: \$5956.34

Hasil Evaluasi Model Regresi Linier

1. MAPE (Mean Absolute Percentage Error): 41.40%

- Secara rata-rata, prediksi biaya asuransi meleset sekitar **41.40%** dari nilai sebenarnya pada data pengujian.
- Angka ini menunjukkan bahwa meskipun model sudah belajar dari data, masih ada variabilitas signifikan dalam biaya asuransi yang belum sepenuhnya dapat dijelaskan oleh fitur-fitur dalam model. Hal ini mengindikasikan ruang untuk perbaikan, mungkin dengan:
 - Menambahkan fitur lain
 - Mencoba model yang lebih kompleks

- Memproses outlier atau nilai ekstrem
2. **R-squared (Koefisien Determinasi): 0.807**
 - Nilai **0.807** berarti sekitar **80.7%** variasi dalam biaya asuransi dapat dijelaskan oleh fitur-fitur yang digunakan.
 - Nilai ini dianggap cukup baik untuk data dunia nyata, menunjukkan bahwa fitur-fitur yang dipilih (*age*, *bmi*, *children*, dan variabel hasil encoding) merupakan prediktor yang relevan.
 3. **MAE (Mean Absolute Error): \$4,177.05**
 - Rata-rata selisih absolut antara prediksi dan nilai aktual adalah **\$4,177.05**.
 - Memberikan ukuran kesalahan dalam satuan dolar yang mudah dipahami, menunjukkan akurasi model dari sudut pandang rata-rata kesalahan.
 4. **MSE (Mean Squared Error): 35,478,020.68**
 - Rata-rata kuadrat selisih antara nilai aktual dan prediksi.
 - MSE lebih sulit diinterpretasikan langsung karena satuannya kuadrat (*dolar kuadrat*). Metrik ini memberikan bobot yang lebih besar pada kesalahan prediksi yang besar (nilai outlier pada residual). Angka yang besar ini mencerminkan adanya beberapa prediksi dengan selisih yang cukup signifikan.
 5. **RMSE (Root Mean Squared Error): \$5,956.34**
 - RMSE sebesar **\$5,956.34** adalah akar kuadrat dari MSE dan memiliki satuan yang sama dengan variabel target (dolar).
 - RMSE sering menjadi metrik evaluasi utama karena satuannya yang sama dengan target dan sensitivitasnya terhadap kesalahan besar.

Nilai **\$5,956.34** menunjukkan bahwa, rata-rata, model ini membuat kesalahan prediksi sekitar **\$5,956.34**, dengan mempertimbangkan bahwa kesalahan yang lebih besar mendapat penalti yang lebih berat (dibandingkan MAE **\$4,177.05**).

Kesimpulan Evaluasi

Model Regresi Linier ini menunjukkan performa yang cukup baik dengan **R-squared 0.807**, namun masih memiliki beberapa keterbatasan:

- **Akurasi Prediksi:**
 - MAPE **41.40%** menunjukkan rata-rata kesalahan prediksi yang cukup besar.
 - Perbedaan signifikan antara MAE (**\$4,177.05**) dan RMSE (**\$5,956.34**) mengindikasikan adanya outlier.
- **Rekomendasi Perbaikan:**
 - Eksplorasi fitur tambahan yang mungkin berpengaruh.
 - Penanganan outlier yang lebih baik.
 - Pertimbangan untuk menggunakan model alternatif yang lebih kompleks.
- **Kelayakan Model:**
 - Meskipun ada kesalahan prediksi, model ini sudah dapat memberikan perkiraan biaya asuransi dengan tingkat keakuratan yang layak untuk kasus penggunaan tertentu.
 - Cocok sebagai *baseline model* sebelum pengembangan lebih lanjut.

2.6.4. Analisis Residual

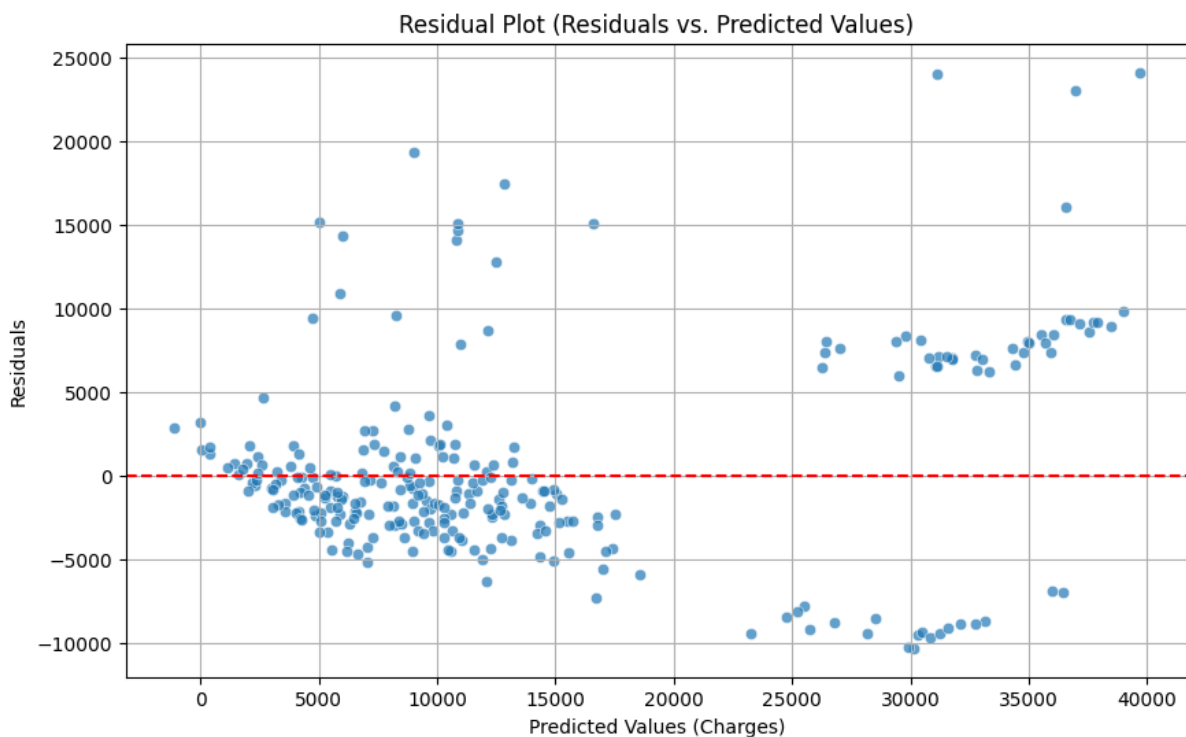
Setelah melatih model Regresi Linier dan mengevaluasi performanya menggunakan metrik seperti R-squared dan MAE/RMSE, langkah penting selanjutnya adalah memeriksa residual. Residual adalah selisih antara nilai biaya asuransi yang sebenarnya (y_{test}) dengan nilai biaya asuransi yang diprediksi oleh model (y_{pred}) untuk setiap individu dalam data pengujian.

Analisis residual sangat penting karena residual seharusnya tidak menunjukkan pola yang jelas jika model regresi linier sudah sesuai. Jika ada pola dalam residual, ini bisa mengindikasikan bahwa asumsi model regresi linier tidak terpenuhi atau ada informasi penting dalam data yang belum berhasil ditangkap oleh model.

```
[ ] residuals = y_test - y_pred
```

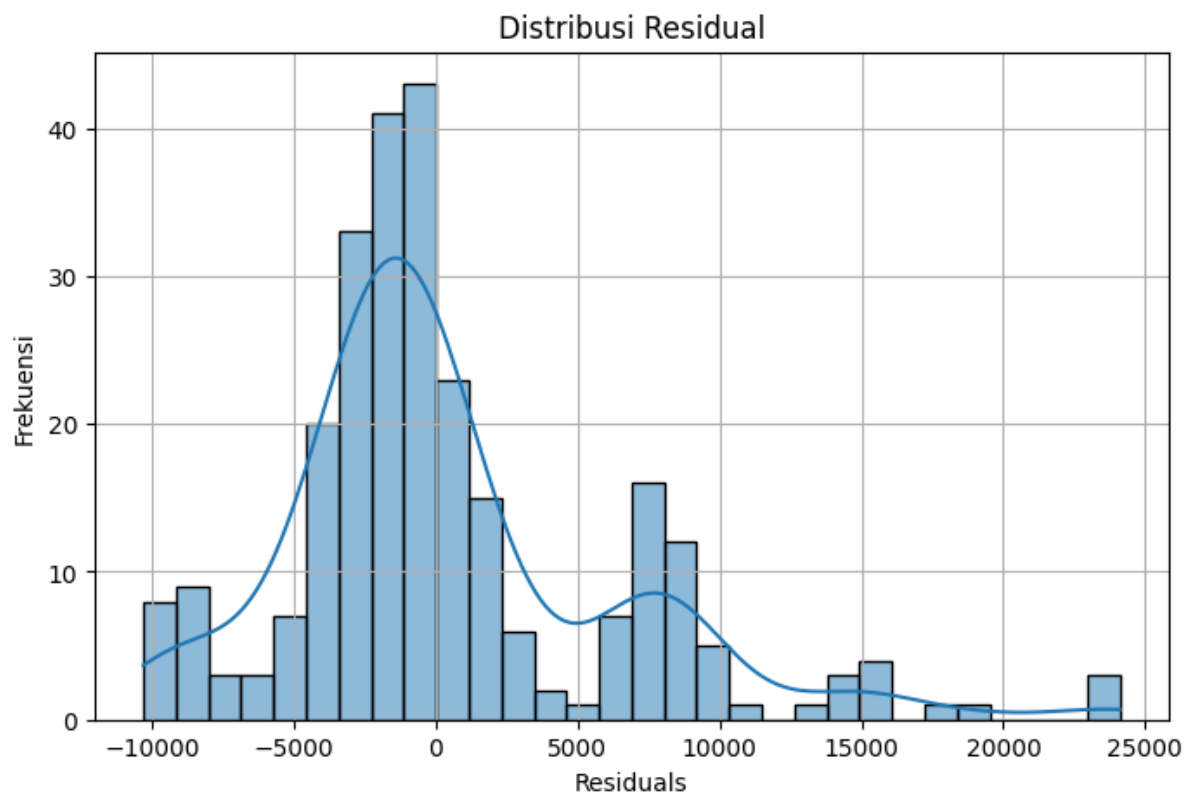
2.6.4.1. Plot Residual vs. Predicted Values

```
[ ] plt.figure(figsize=(10, 6))
    sns.scatterplot(x=y_pred, y=residuals, alpha=0.7)
    plt.axhline(y=0, color='r', linestyle='--')
    plt.xlabel('Predicted Values (Charges)')
    plt.ylabel('Residuals')
    plt.title('Residual Plot (Residuals vs. Predicted Values)')
    plt.grid(True)
    plt.show()
```



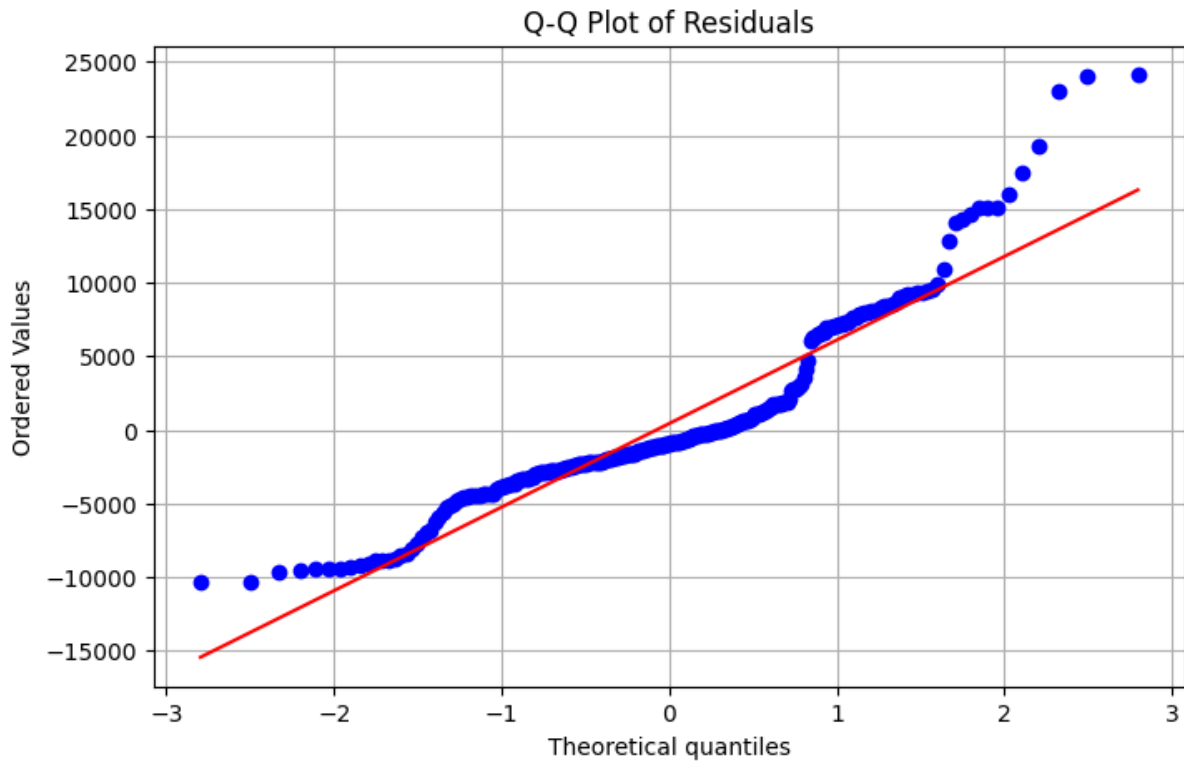
2.6.4.2. Histogram Residual

```
plt.figure(figsize=(8, 5))
sns.histplot(residuals, kde=True, bins=30)
plt.title('Distribusi Residual')
plt.xlabel('Residuals')
plt.ylabel('Frekuensi')
plt.grid(True)
plt.show()
```



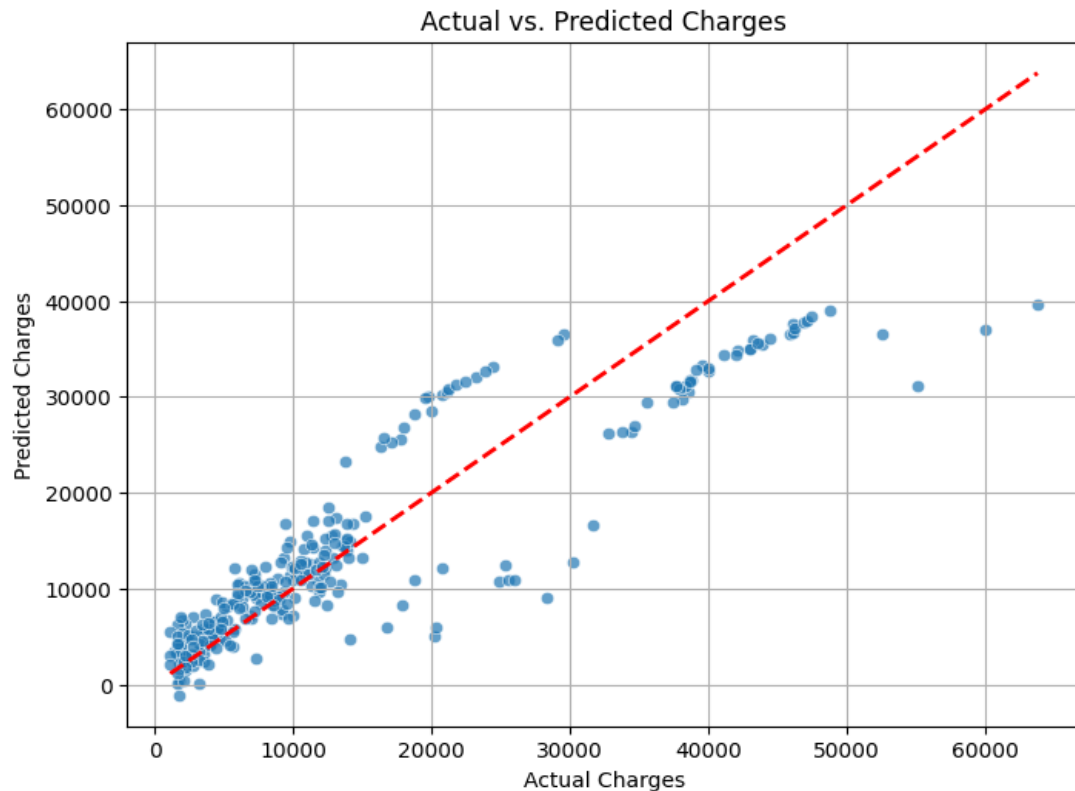
2.6.4.3. Q-Q Plot Residual

```
plt.figure(figsize=(8, 5))
stats.probplot(residuals, dist="norm", plot=plt)
plt.title('Q-Q Plot of Residuals')
plt.grid(True)
plt.show()
```



2.6.4.4. Plot Actual vs. Predicted Values

```
[ ] plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, alpha=0.7, edgecolors='w', linewidth=0.5)
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--',
lw=2) # Garis y=x
plt.xlabel('Actual Charges')
plt.ylabel('Predicted Charges')
plt.title('Actual vs. Predicted Charges')
plt.grid(True)
plt.show()
```

2.6.7. Tampilkan Koefisien dan Intercept

```
[ ] print(f"Intercept (Konstanta): {model.intercept_:.2f}")

coefficients = pd.DataFrame({
    'Feature': X_train.columns, # Menggunakan X_train.columns karena X mungkin
    'Coefficient': model.coef_
})

print("\nKoefisien Regresi:")
# Sortir berdasarkan nilai absolut koefisien untuk melihat pengaruh terbesar,
# atau biarkan seperti sebelumnya
# sorted_coefficients = coefficients.reindex(coefficients.Coefficient.abs().
# sort_values(ascending=False).index)
sorted_coefficients = coefficients.sort_values(by='Coefficient',
ascending=False)
print(sorted_coefficients)
```

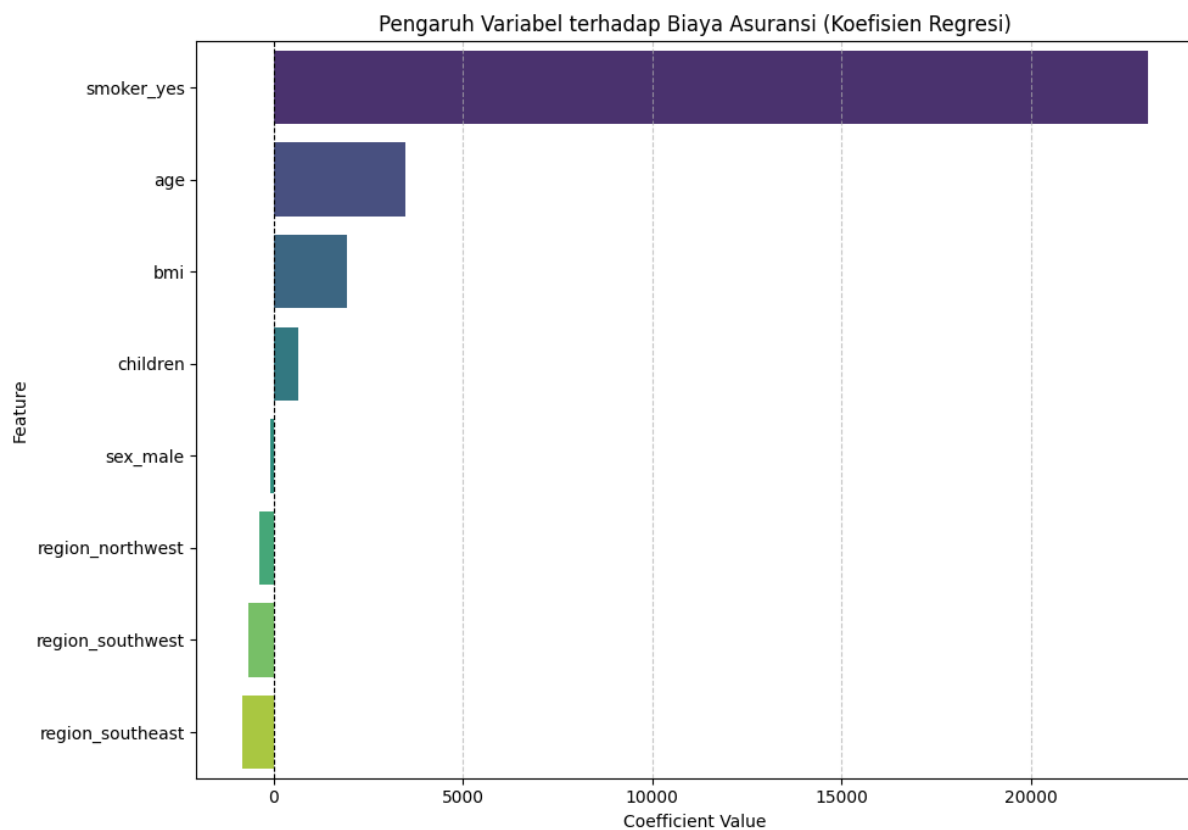
Intercept (Konstanta): 8947.95

Koefisien Regresi:

	Feature	Coefficient
4	smoker_yes	23077.764593
0	age	3472.975553
1	bmi	1927.828251
2	children	636.501185
3	sex_male	-101.542054
5	region_northwest	-391.761455
7	region_southwest	-659.139752
6	region_southeast	-838.919616

Visualisasi Koefisien yang Ditingkatkan

```
[ ] plt.figure(figsize=(10, 7)) # Sedikit lebih besar untuk label
sns.barplot(
    x='Coefficient',
    y='Feature',
    data=sorted_coefficients,
    palette='viridis'
)
plt.axvline(x=0, color='black', linewidth=0.8, linestyle='--') # Garis nol
plt.title('Pengaruh Variabel terhadap Biaya Asuransi (Koefisien Regresi)')
plt.xlabel('Coefficient Value')
plt.ylabel('Feature')
plt.grid(axis='x', linestyle='--', alpha=0.7)
plt.tight_layout() # Untuk memastikan label tidak terpotong
plt.show()
```



2.7. Simulasi Prediksi

```
[ ] # Fungsi untuk membuat DataFrame input baru
def buat_data_input(age, bmi, children, sex_male, smoker_yes, region_northwest,
region_southeast, region_southwest):
    data_baru = pd.DataFrame({
        'age': [age],
        'bmi': [bmi],
        'children': [children],
        'sex_male': [sex_male],
        'smoker_yes': [smoker_yes],
        'region_northwest': [region_northwest],
        'region_southeast': [region_southeast],
        'region_southwest': [region_southwest]
    })
    # Pastikan urutan kolom sesuai dengan X_train saat training
    return data_baru[X_train.columns]
```

2.7.1. Contoh 1: Non-perokok, laki-laki, 30 tahun, bmi 25, 1 anak, di southeast

```
[ ] data_contoh1 = buat_data_input(age=30, bmi=25, children=1, sex_male=1,
smoker_yes=0,
region_northwest=0, region_southeast=1,
region_southwest=0)
prediksi1 = model.predict(data_contoh1)
print(f"Prediksi biaya untuk Contoh 1: ${prediksi1[0]:.2f}")
```

➡ Prediksi biaya untuk Contoh 1: \$161028.96

2.7.2. Contoh 2: Sama seperti contoh 1, tetapi perokok

```
[ ] data_contoh2 = buat_data_input(age=30, bmi=25, children=1, sex_male=1,
smoker_yes=1,
region_northwest=0, region_southeast=1,
region_southwest=0)
# Jika scaling:
# data_contoh2[numerical_features] = scaler.transform(data_contoh2
[numerical_features])

prediksi2 = model.predict(data_contoh2)
print(f"Prediksi biaya untuk Contoh 2 (Perokok): ${prediksi2[0]:.2f}")
```

➡ Prediksi biaya untuk Contoh 2 (Perokok): \$184106.73

2.7.3. Perbedaan biaya karena merokok (untuk contoh ini)

```
[ ] print(f"Perbedaan biaya karena merokok (untuk contoh ini): ${prediksi2[0] - prediksi1[0]:.2f}")
```

```
⇒ Perbedaan biaya karena merokok (untuk contoh ini): $23077.76
```

Bab 3: Kesimpulan

Berdasarkan analisis yang telah dilakukan dalam bab sebelumnya, dapat ditarik beberapa kesimpulan penting terkait model prediksi biaya asuransi kesehatan:

1. Pengembangan Model dan Kinerja

Telah berhasil dibangun sebuah model regresi linier yang mampu memprediksi biaya asuransi kesehatan individu. Model ini menunjukkan kinerja yang cukup baik dalam menjelaskan variasi biaya asuransi, sebagaimana tercermin dari nilai R-squared sebesar 0.807. Artinya, sekitar 80.7% variabilitas dalam biaya asuransi dapat dijelaskan oleh variabel-variabel yang digunakan dalam model. Meskipun demikian, nilai MAPE (Mean Absolute Percentage Error) sebesar 41.40% mengindikasikan bahwa rata-rata persentase kesalahan prediksi masih cukup signifikan.

2. Faktor-Faktor Penting

Analisis koefisien regresi menunjukkan bahwa status merokok merupakan faktor yang paling signifikan dalam mempengaruhi biaya asuransi, dengan koefisien positif yang besar. Variabel lain seperti usia dan indeks massa tubuh (BMI) juga menunjukkan pengaruh positif terhadap peningkatan biaya asuransi. Sebaliknya, beberapa kategori wilayah tempat tinggal menunjukkan korelasi negatif dengan biaya, mengindikasikan potensi biaya yang lebih rendah dibandingkan dengan wilayah referensi.

3. Proses Data

Keberhasilan model ini didukung oleh serangkaian tahapan pra-pemrosesan data yang cermat. Ini termasuk penanganan data duplikat, pengkodean (encoding) variabel-variabel kategorikal seperti jenis kelamin, status perokok, dan wilayah menjadi format numerik. Selain itu, dilakukan penskalaan (scaling) pada fitur-fitur numerik untuk memastikan semua variabel memberikan kontribusi yang setara dalam model.

4. Validasi Model

Evaluasi model dilakukan menggunakan metrik standar seperti MAE (Mean Absolute Error) sebesar \$4,177.05 dan RMSE (Root Mean Squared Error) sebesar \$5,956.34.

Perbedaan antara MAE dan RMSE menyiratkan adanya beberapa prediksi dengan kesalahan yang cukup besar, yang juga terlihat dari analisis residual. Plot residual menunjukkan adanya beberapa pola, seperti heteroskedastisitas, yang mengindikasikan bahwa varians kesalahan tidak konstan di semua tingkat nilai prediksi.

5. Implikasi dan Rekomendasi

Model yang dikembangkan dapat memberikan estimasi awal biaya asuransi dan menjadi dasar untuk analisis lebih lanjut. Meskipun memiliki kemampuan prediktif yang baik, terdapat ruang untuk perbaikan. Upaya di masa mendatang dapat difokuskan pada eksplorasi fitur tambahan, penanganan outlier yang lebih cermat, atau pertimbangan untuk menggunakan model machine learning alternatif yang mungkin lebih kompleks guna meningkatkan akurasi prediksi dan mengatasi pola yang teramati dalam residual.

Secara keseluruhan, proyek ini berhasil menunjukkan penerapan regresi linier dalam memprediksi biaya asuransi dan mengidentifikasi faktor-faktor kunci yang mempengaruhinya, seraya memberikan landasan untuk pengembangan model yang lebih lanjut.

Bab 4: Lampiran

Pembagian tugas:

- Lathif Ramadhan (5231811022) : Pembuatan kode dan penyusunan serta merapikan laporan
- Andini Angel M. (5231811029) : Pembuatan kode
- Rama Panji N. (5231811033) : Penyusunan laporan
- Giffari Riyanda P. (5231811036) : Pembuatan laporan