```
#dataset->https://drive.google.com/file/d/
1cB1xI4ppWu9rFKDzDbMltSaB9oN7apMs/view?usp=sharing
```

Problem Statement

Insurance premiums are often based on various factors that in the end decide the amount that will be covered from the insurance company. As a data analyst/scientist you are given a set of historical data for an organizations, customers and the respective charges that were levied upon the insurance company.

The data gives you the information about the users including their age, sex, bmi, hospitalization history, annual income, etc. Analyze and gather insights from the data and create a linear regression model that will best predict the insurance charges for a new set of data.

Dataset Information

Column Name	Description
age	Age of the person concerned
sex	Gender of the person concerned(male, female)
bmi	The bmi of the person concerned
children	The number of children they have
smoker	If they are smokers or not
Claim_Amount	The claim amount
past_consultations	The number of past consultations
num_of_steps	The number of steps they covered
Hospital_expenditure	The hospital expenditure in question
NUmber_of_past_hospitalizations	The number of past hospitalizations for the person concerned
Anual_Salary	The annual salary of the person concerned
region	The region that they are from
charges	The final charges that were levied for the person concerned.

```
# Importing Libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
insurance=pd.read_csv('/content/new_insurance_data.csv')
insurance.head()
```


```
{"summary":"{\n  \"name\": \"insurance\",\n  \"rows\": 1338,\n
\"fields\": [\n    {\n      \"column\": \"age\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
14.03481785145205,\n        \"min\": 18.0,\n        \"max\": 64.0,\n
\"num_unique_values\": 47,\n        \"samples\": [\n          46.0,\n
57.0,\n          44.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"sex\",\n      \"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n
\"female\",\n          \"male\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"bmi\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 6.101689761114468,\n
\"min\": 15.96,\n        \"max\": 53.13,\n
\"num_unique_values\": 547,\n        \"samples\": [\n
35.435,\n          25.8\n        ],\n        \"semantic_type\": \"\",\
n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"children\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 1.2018555798969655,\n        \"min\":
0.0,\n        \"max\": 5.0,\n        \"num_unique_values\": 6,\n
\"samples\": [\n          0.0,\n          1.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"smoker\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
2,\n        \"samples\": [\n          \"yes\",\n          \"no\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Claim_Amount\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
15617.28833698884,\n        \"min\": 1920.136268,\n        \"max\":
77277.98848,\n        \"num_unique_values\": 1324,\n
\"samples\": [\n          43002.74626,\n          36170.21613\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"past_consultations\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
7.467723170663613,\n        \"min\": 1.0,\n        \"max\": 40.0,\n
\"num_unique_values\": 39,\n        \"samples\": [\n          35.0,\n
37.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"num_of_steps\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 91886.11959215358,\n        \"min\":
695430.0,\n        \"max\": 1107872.0,\n        \"num_unique_values\":
1335,\n        \"samples\": [\n          939384.0,\n
990274.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Hospital_expenditure\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 26693047.737618804,\n        \"min\":
29452.53296,\n        \"max\": 261631699.3,\n
```

```
\"num_unique_values\": 1334,\n        \"samples\": [\n
5550481.835,\n          14471108.95\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\":
\"NUmber_of_past_hospitalizations\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0.5335830912314956,\n
\"min\": 0.0,\n        \"max\": 3.0,\n        \"num_unique_values\":
4,\n        \"samples\": [\n          1.0,\n          3.0\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Anual_Salary\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
566884292.1106706,\n        \"min\": 2747071.908,\n        \"max\":
4117196637.0,\n        \"num_unique_values\": 1332,\n
\"samples\": [\n          1874538842.0,\n          470062608.8\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"region\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
4,\n        \"samples\": [\n          \"southwest\",\n
\"northeast\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"charges\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 12110.011236694003,\n        \"min\": 1121.8739,\n
\"max\": 63770.42801,\n        \"num_unique_values\": 1337,\n
\"samples\": [\n          12979.358,\n          19964.7463\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"insurance"}
```

```
insurance.columns
```

```
Index(['age', 'sex', 'bmi', 'children', 'smoker', 'Claim_Amount',
       'past_consultations', 'num_of_steps', 'Hospital_expenditure',
       'NUmber_of_past_hospitalizations', 'Anual_Salary', 'region',
'charges'],
      dtype='object')
```

```
# Size and Dimensions of the data
insurance.shape
```

```
(1338, 13)
```

```
# Dataset details
insurance.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 13 columns):
 #   Column                            Non-Null Count  Dtype
---  ------                            --------------  -----
 0   age                               1329 non-null   float64
 1   sex                               1338 non-null   object
```

```
 2   bmi                                 1335 non-null   float64
 3   children                            1333 non-null   float64
 4   smoker                              1338 non-null   object
 5   Claim_Amount                        1324 non-null   float64
 6   past_consultations                  1332 non-null   float64
 7   num_of_steps                        1335 non-null   float64
 8   Hospital_expenditure                1334 non-null   float64
 9   NUmber_of_past_hospitalizations     1336 non-null   float64
 10  Anual_Salary                        1332 non-null   float64
 11  region                              1338 non-null   object
 12  charges                             1338 non-null   float64
dtypes: float64(10), object(3)
memory usage: 136.0+ KB
```

The null values are present in the data. We try to manipulate it by filling average values.

```
insurance.isna().sum()
```

```
age                                   9
sex                                   0
bmi                                   3
children                              5
smoker                                0
Claim_Amount                         14
past_consultations                    6
num_of_steps                          3
Hospital_expenditure                  4
NUmber_of_past_hospitalizations       2
Anual_Salary                          6
region                                0
charges                               0
dtype: int64
```

```
# Checking for duplicates
insurance.duplicated().sum()
```

```
np.int64(0)
```

There are no duplicate records in the data.

```
# Filling null values by Median
insurance['age'].fillna(insurance['age'].median(),inplace=True)
insurance['bmi'].fillna(insurance['bmi'].median(),inplace=True)
insurance['children'].fillna(insurance['children'].median(),inplace=Tr
ue)
insurance['Claim_Amount'].fillna(insurance['Claim_Amount'].median(),in
place=True)
insurance['past_consultations'].fillna(insurance['past_consultations']
.median(),inplace=True)
```

```
insurance['num_of_steps'].fillna(insurance['num_of_steps'].median(),in
place=True)
insurance['Hospital_expenditure'].fillna(insurance['Hospital_expenditu
re'].median(),inplace=True)
insurance['NUmber_of_past_hospitalizations'].fillna(insurance['NUmber_
of_past_hospitalizations'].median(),inplace=True)
insurance['Anual_Salary'].fillna(insurance['Anual_Salary'].median(),in
place=True)
```

```
/tmp/ipython-input-2509978150.py:2: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
  insurance['age'].fillna(insurance['age'].median(),inplace=True)
```

```
/tmp/ipython-input-2509978150.py:3: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
  insurance['bmi'].fillna(insurance['bmi'].median(),inplace=True)
```

```
/tmp/ipython-input-2509978150.py:4: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
insurance['children'].fillna(insurance['children'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:5: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.



insurance['Claim_Amount'].fillna(insurance['Claim_Amount'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:6: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.



insurance['past_consultations'].fillna(insurance['past_consultations'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:7: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform the operation inplace on the original object.



insurance['num_of_steps'].fillna(insurance['num_of_steps'].median(),in
```

```
place=True)
/tmp/ipython-input-2509978150.py:8: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



insurance['Hospital_expenditure'].fillna(insurance['Hospital_expenditu
re'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:9: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



insurance['NUmber_of_past_hospitalizations'].fillna(insurance['NUmber_
of_past_hospitalizations'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:10: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



insurance['Anual_Salary'].fillna(insurance['Anual_Salary'].median(),in
place=True)

insurance.isna().sum()
```

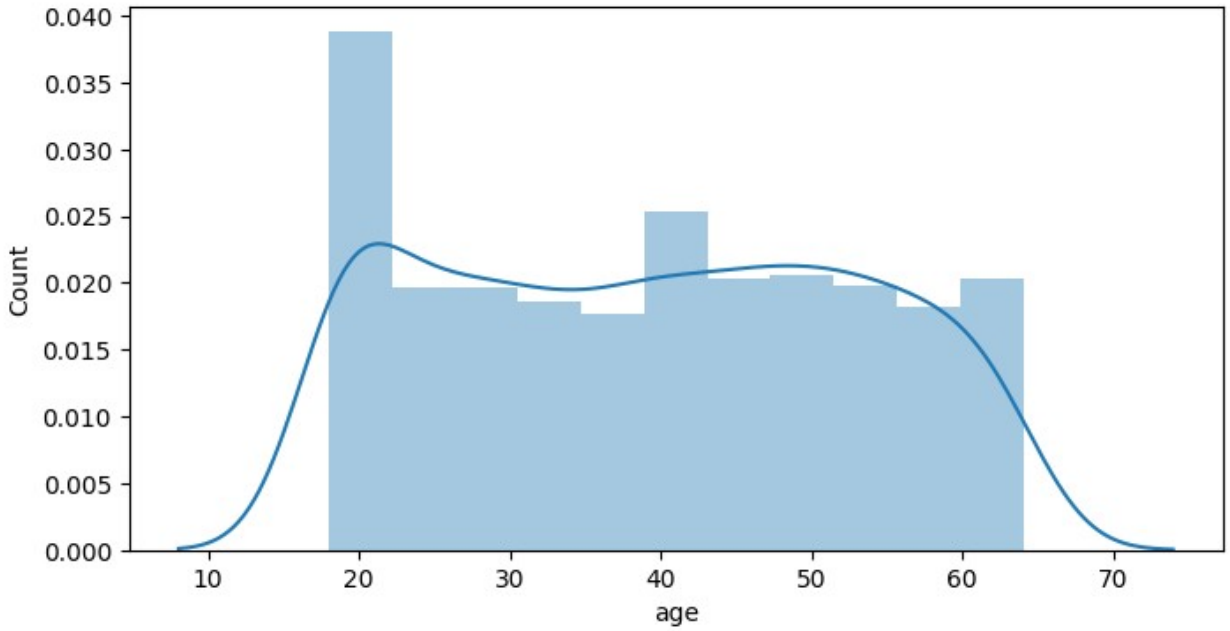
```
age                                0
sex                                0
bmi                                0
children                           0
smoker                             0
Claim_Amount                       0
past_consultations                 0
num_of_steps                       0
Hospital_expenditure               0
NUmber_of_past_hospitalizations    0
Anual_Salary                       0
region                             0
charges                            0
dtype: int64
```

All the null values are removed from the data.

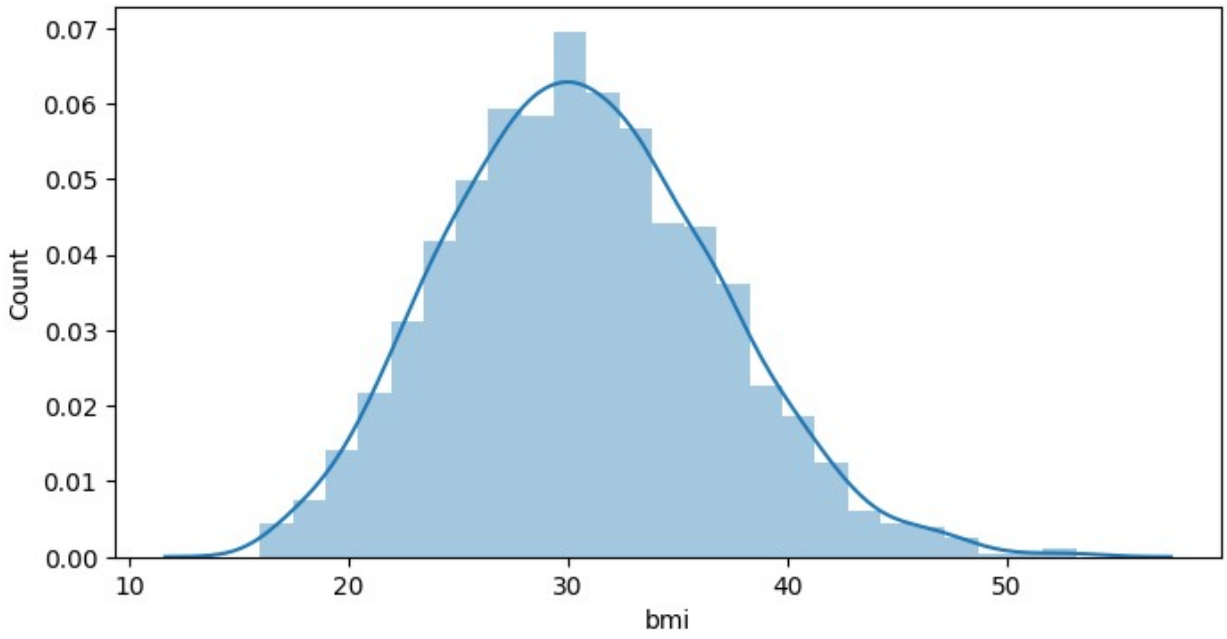
Checking distribution of each variable in the dataset.

```
numeric_cols=insurance.select_dtypes(include=['int64','float64']).colu
mns
categorical_cols=insurance.select_dtypes(include=['object']).columns

import matplotlib.pyplot as plt
import seaborn as sns

for col_name in numeric_cols:
  plt.figure(figsize=(8,4))
  sns.distplot(insurance[col_name])
  plt.xlabel(col_name)
  plt.ylabel('Count')
  plt.show()
```

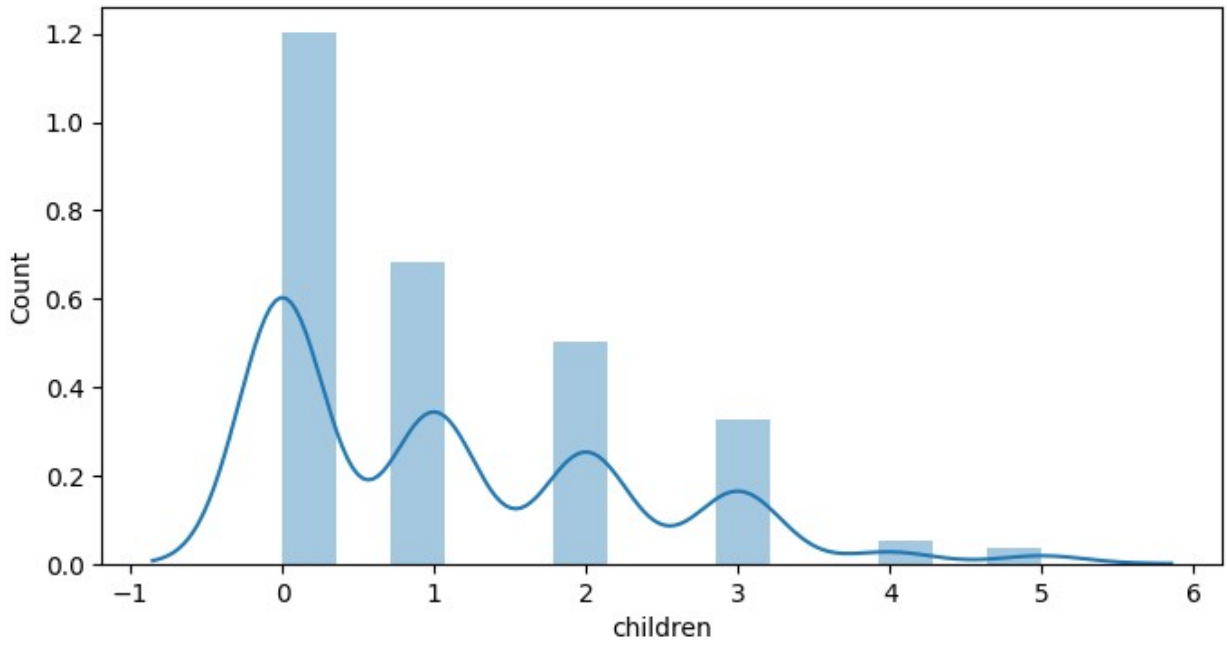
```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```



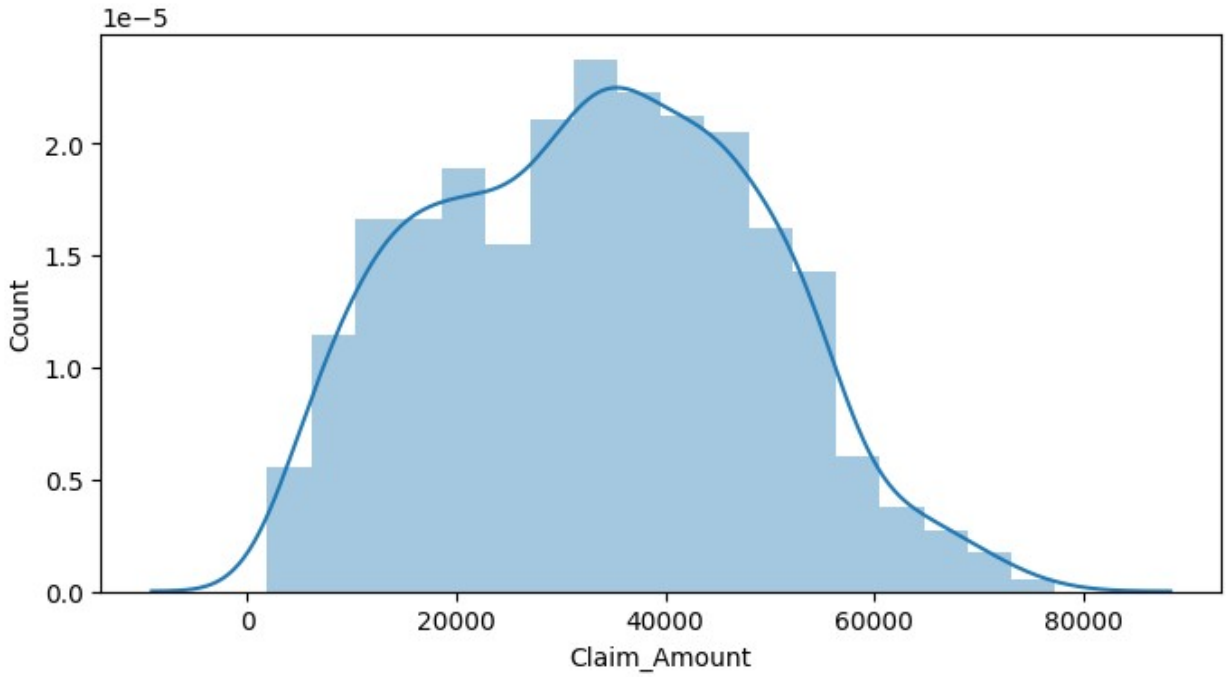

```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```



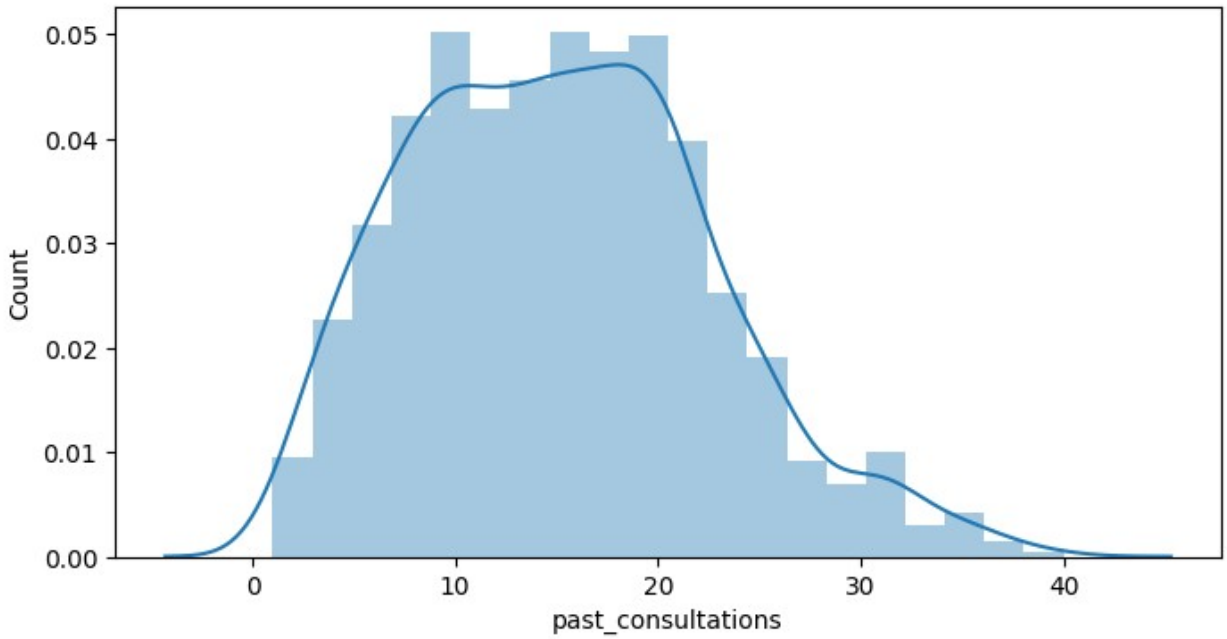

```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```



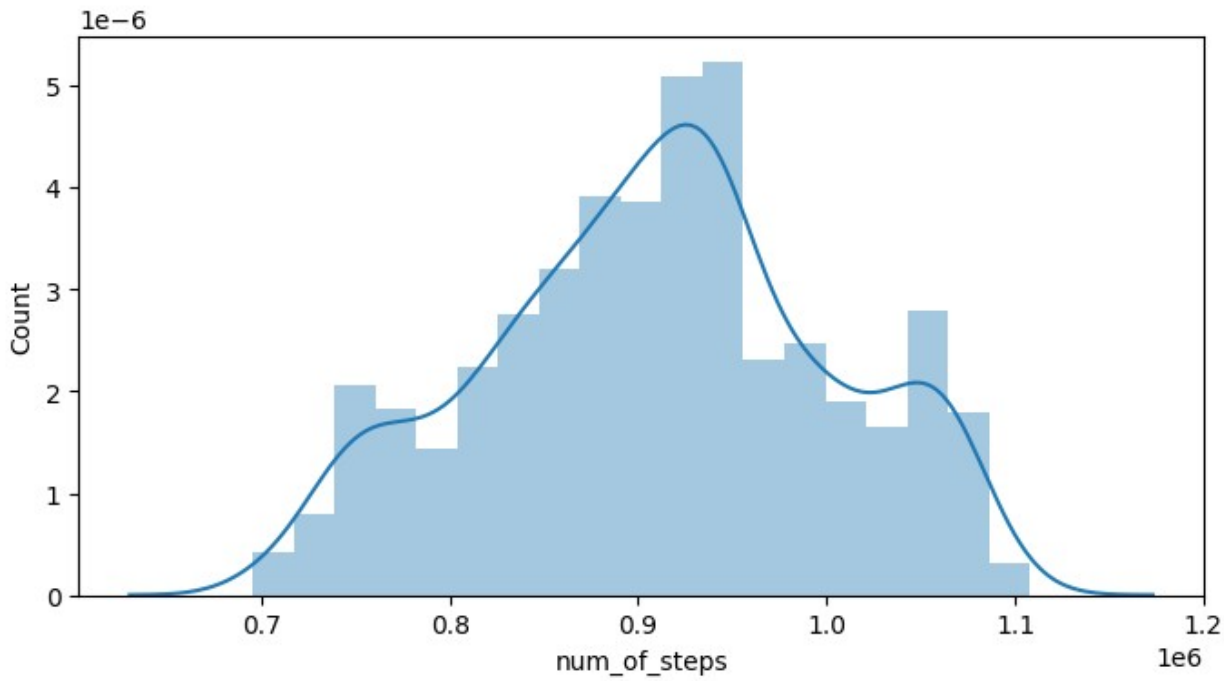

```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```



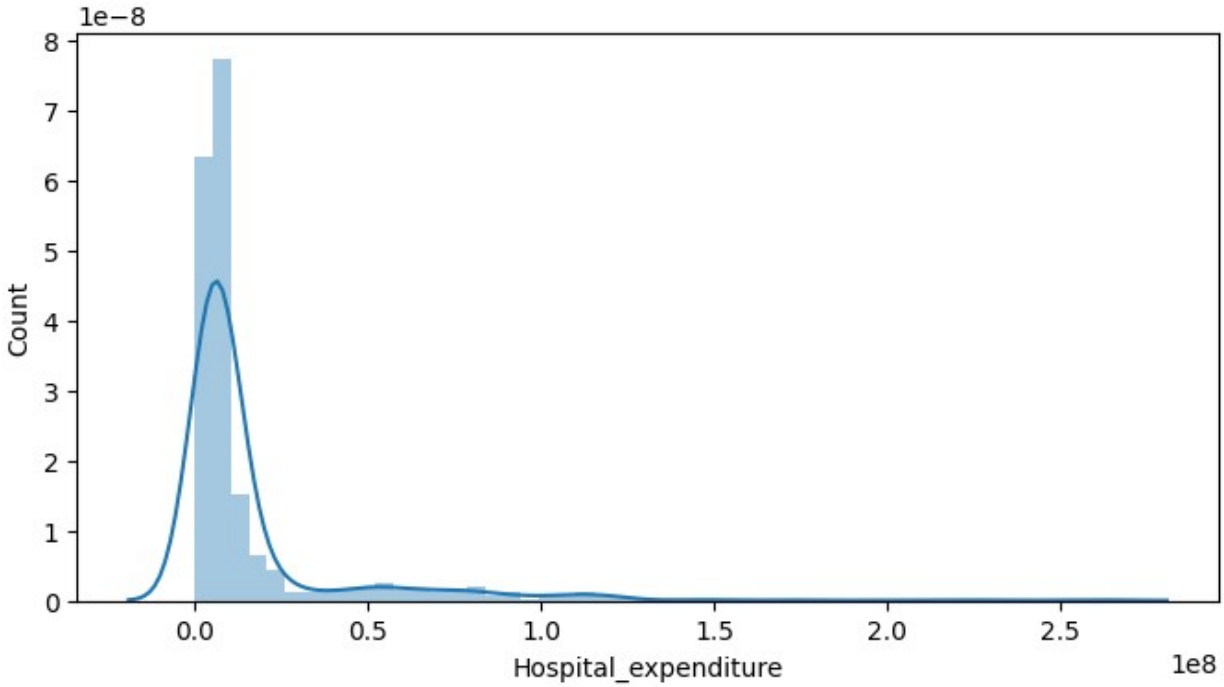

```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```



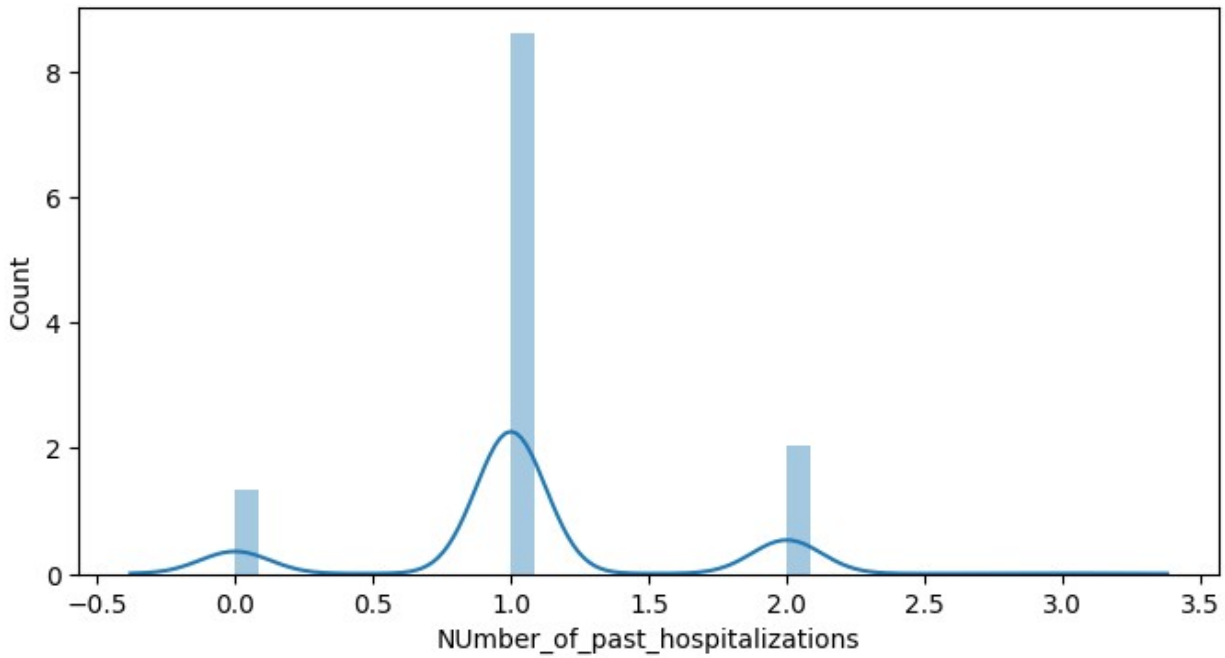

```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```



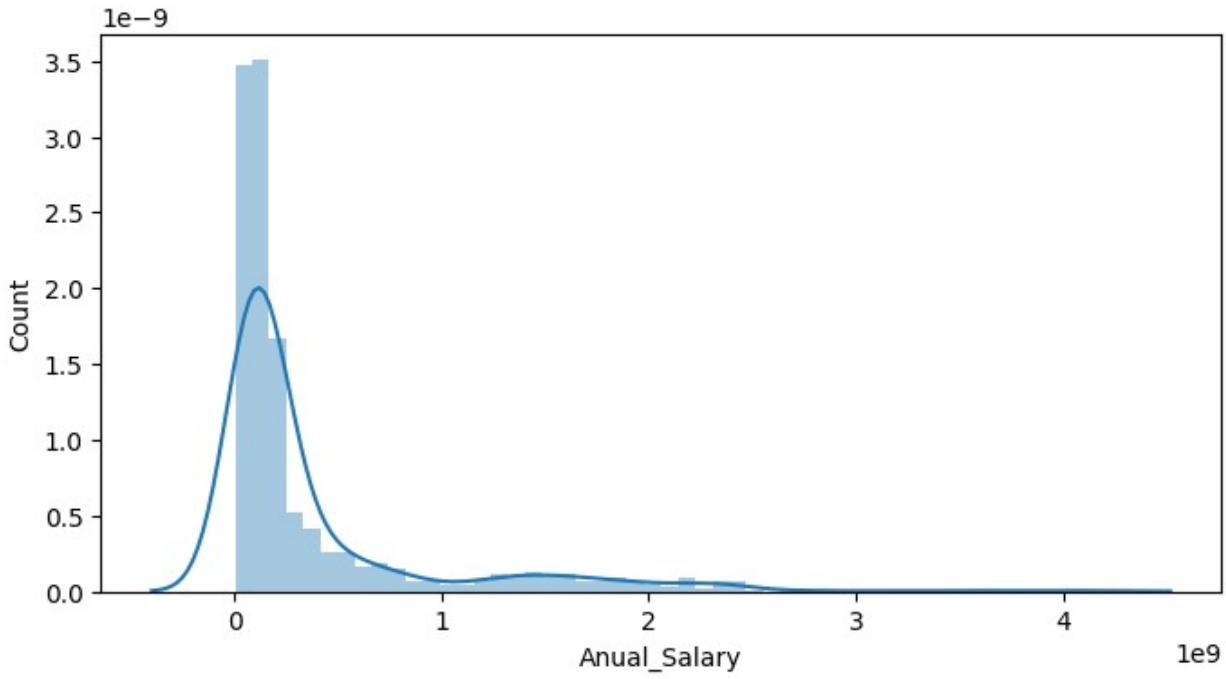

```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```



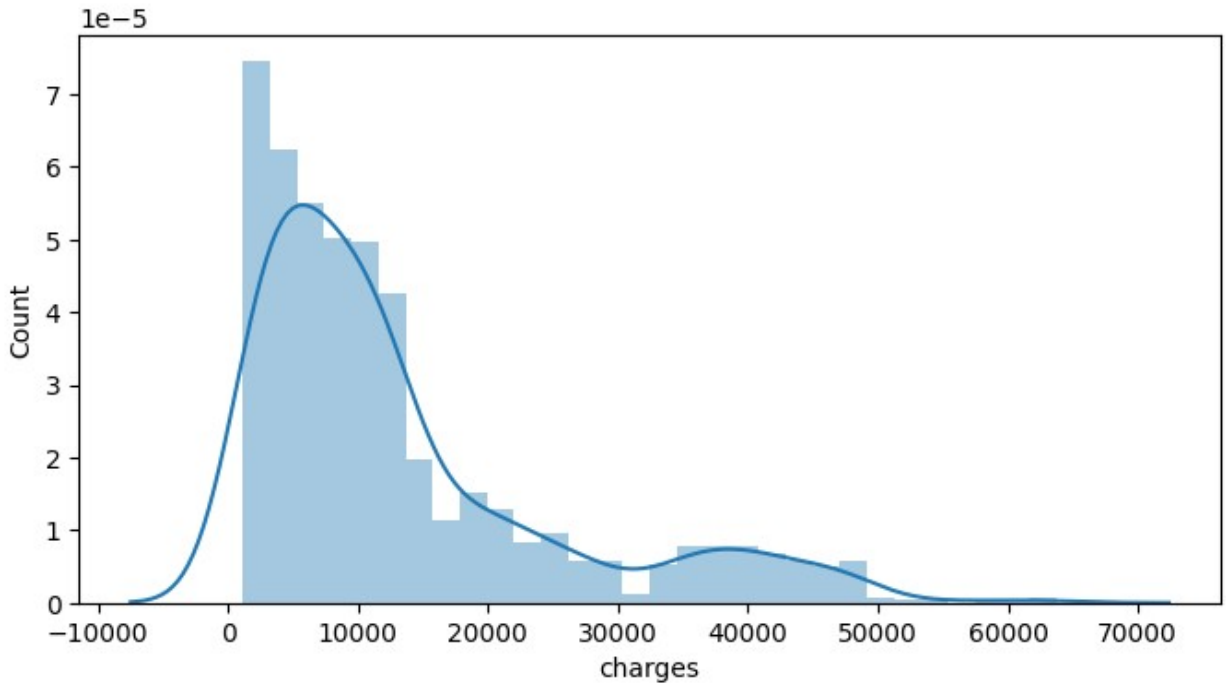

```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```




```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```




```
/tmp/ipython-input-186704904.py:6: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn
v0.14.0.

Please adapt your code to use either `displot` (a figure-level
function with
similar flexibility) or `histplot` (an axes-level function for
histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(insurance[col_name])
```



From the above distribution plots we can observe that,

i) The 'age' column is slightly right skewed. Most of the people having age from 20 to 65. But some outliers are preent in the data(more aged people).

ii) The 'bmi' curve looks roughly normal but slightly right-skewed — meaning there are some higher-than-typical BMI values (possible outliers) on the right tail.

iii) The 'Children' plot shows right skewed distribution. Most of the Policyholders have 0 childrens and the childrens count increases number of policyholders decreases.

iv) The 'claim_amount' plot shows bell shaped like structure but it is right skewed. Most of the claim values are between range 20000 to 50000. No extreme outliers present- few policyholders claim about 80000.

v) The 'past_consultations' plot is slightly right skewed.Most customers have between 8–20 consultations. A smaller number have very high consultations (30+), which might be due to chronic illnesses or frequent hospital visits.

vi) The 'num_of_steps' distribution is approximately normal, centered around 900,000–950,000 steps. Slight right skew, meaning a few people have exceptionally high step counts (>1,050,000). Very few low step counts (<750,000), which might indicate low activity.

vii) The 'Hospital_Expenditure' distribution is strongly right skewed. Most customers have low expenditures (clustered near zero). A few have extremely high expenditures (up to 250M), which are clear outliers.

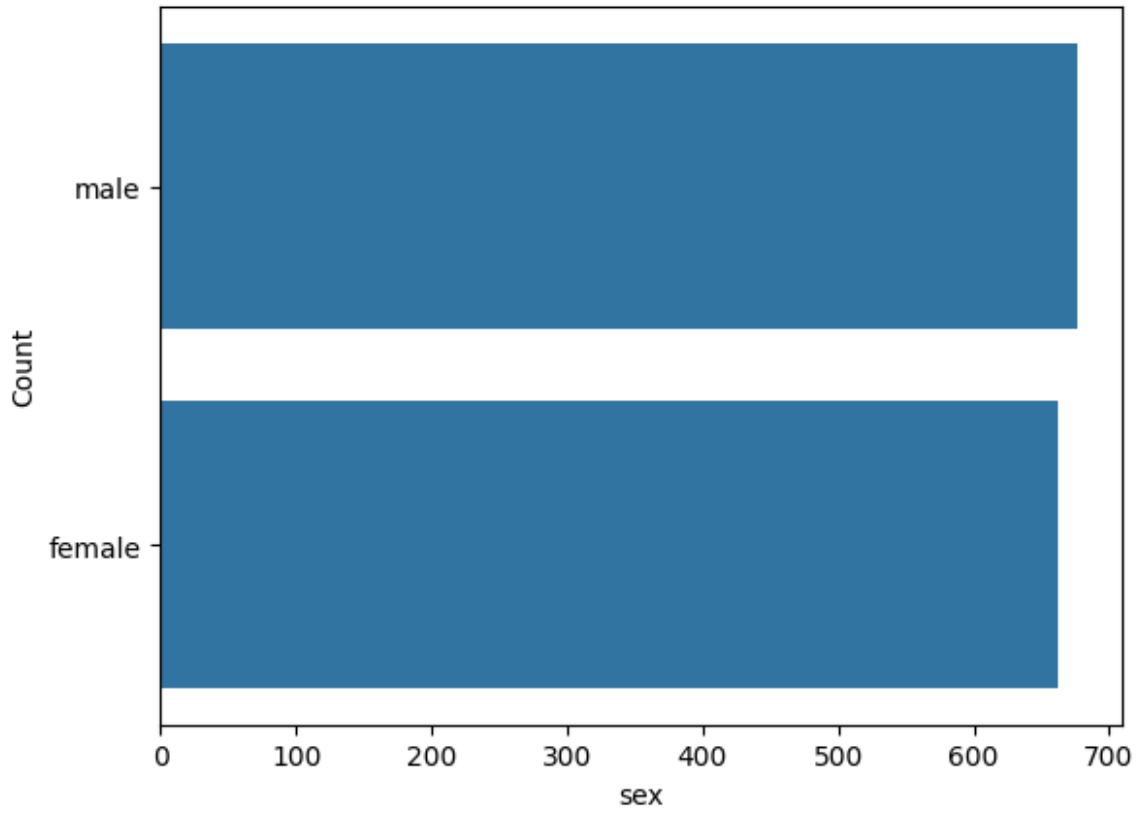
viii) For 'The number_of_past_hospitalizations', Most customers have 1 hospitalization in the past. Some have 0 or 2, and very few have more than 2. The distribution is heavily concentrated,

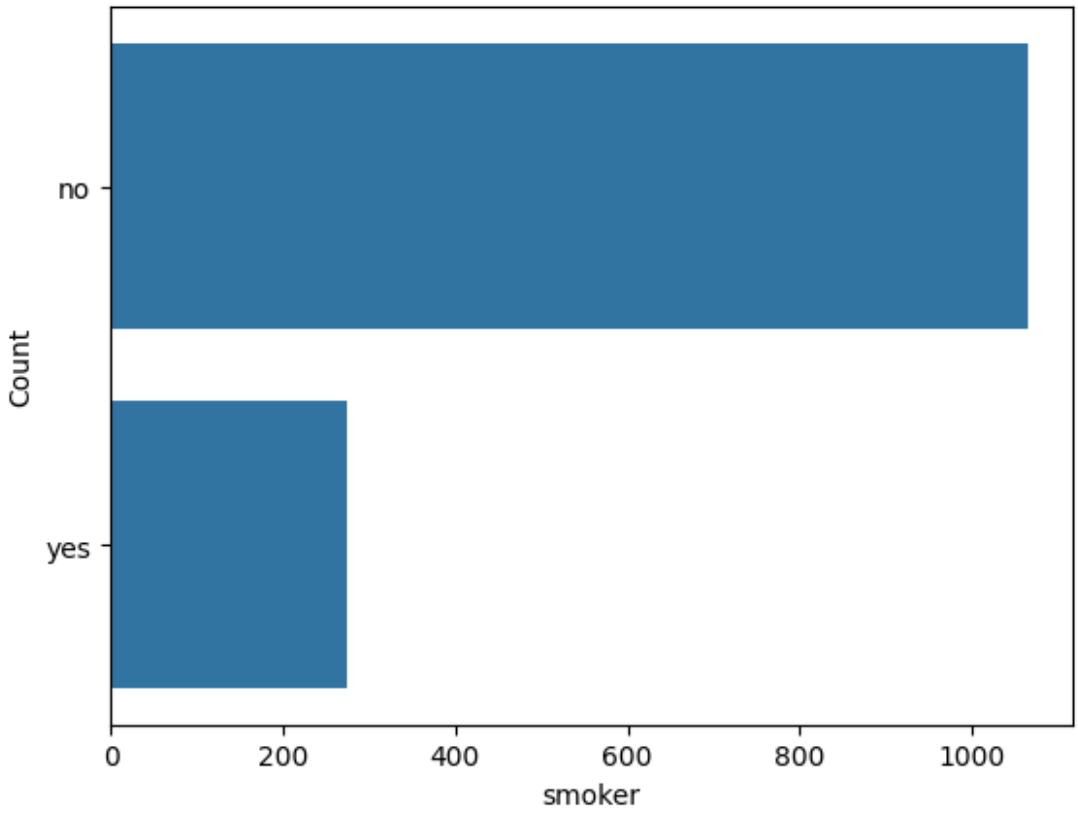
which suggests: Hospitalization is not very frequent for most people. This could be a strong predictor of claim amounts — higher past hospitalizations might indicate higher risk.

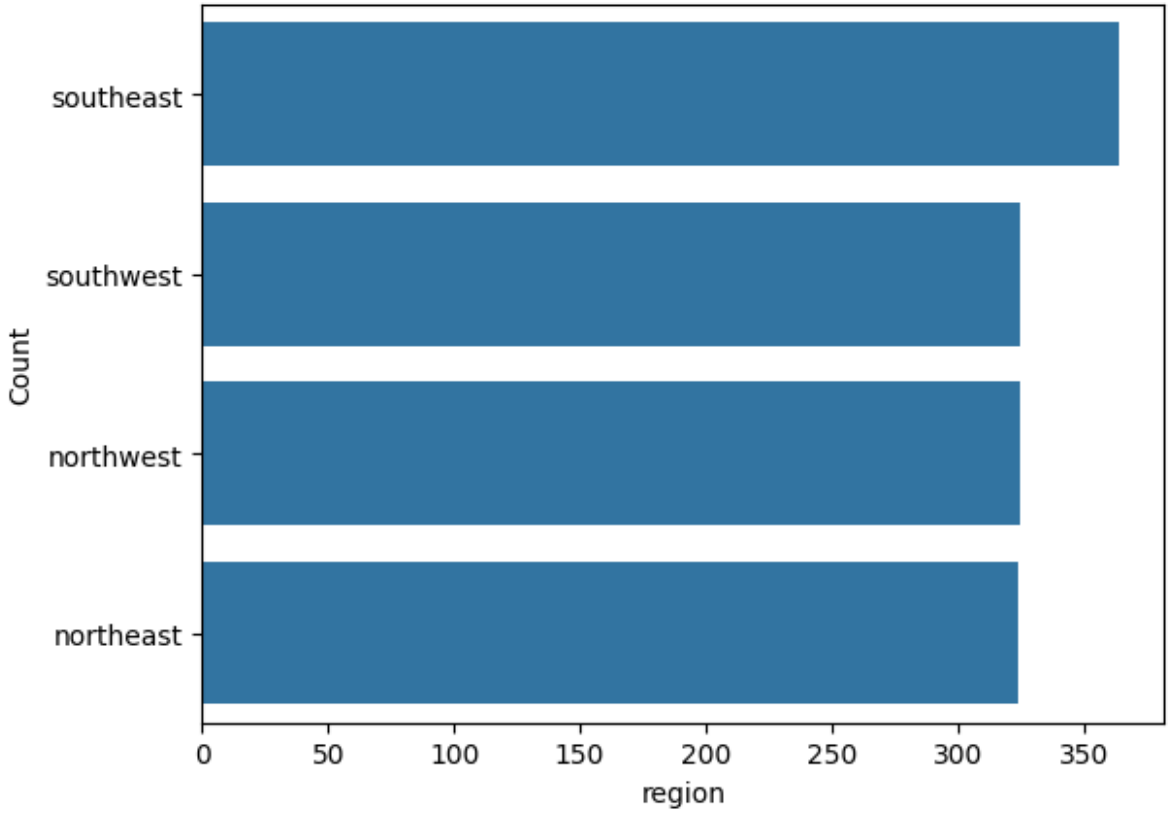
ix) For the Annual Salary distribution: Highly right-skewed distribution. Most people have lower annual salaries, concentrated near the left side of the plot. A few extremely high salaries act as outliers.

x) For the charges distribution: Right-skewed distribution. Majority of the values are on the lower side, with a long tail towards higher charges. This suggests a small group of people incur very high charges. The outliers present on the right side.

```
for col_name in categorical_cols:
  sns.countplot(insurance[col_name])
  plt.xlabel(col_name)
  plt.ylabel("Count")
  plt.show()
```




no
yes
Count
0
200
400
600
800
1000
smoker


southeast
southwest
northwest
northeast
Count
0
50
100
150
200
250
300
350
region

```
categorical_cols
```

```
Index(['sex', 'smoker', 'region'], dtype='object')
```

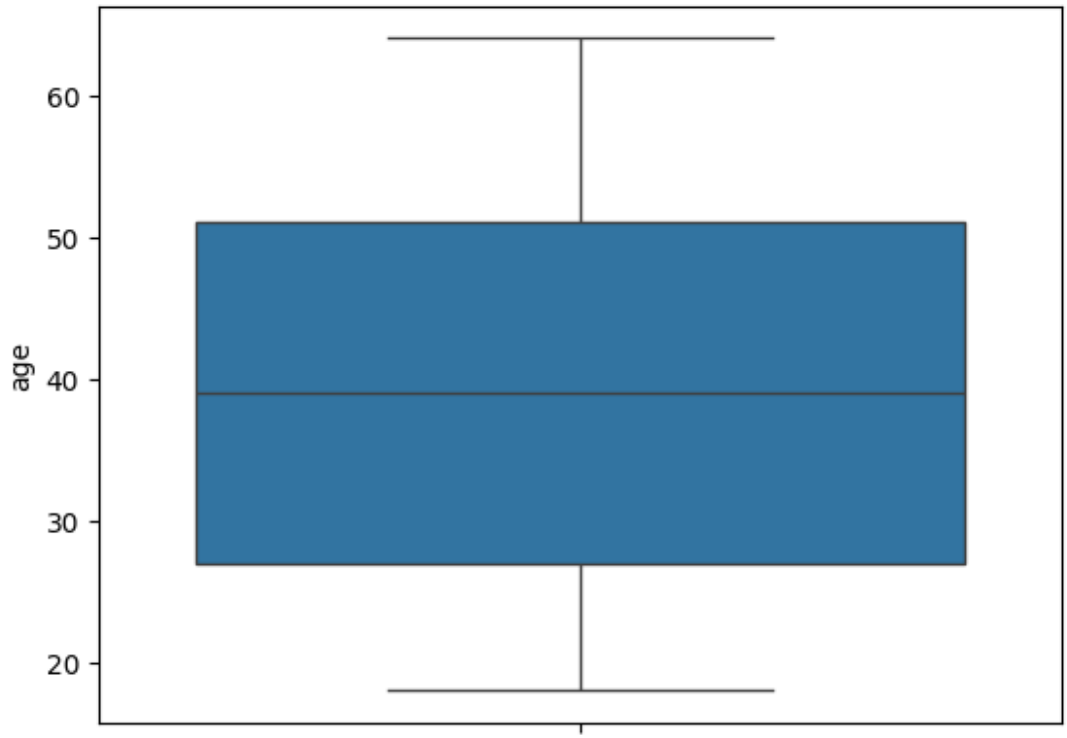
Encoding the categorical attributes.

```
# Reload dataset
from sklearn.preprocessing import OneHotEncoder
insurance = pd.read_csv("/content/new_insurance_data.csv")

categorical_cols = ['sex', 'smoker', 'region']
ohe = OneHotEncoder(drop='first', sparse_output=False)  # drop='first'
to avoid dummy variable trap

encoded = ohe.fit_transform(insurance[categorical_cols])

# Convert to DataFrame
encoded_df = pd.DataFrame(encoded,

columns=ohe.get_feature_names_out(categorical_cols),
                          index=insurance.index)

# Concatenate with original DF (and drop original column)
insurance = pd.concat([insurance.drop(columns=categorical_cols),
encoded_df], axis=1)
```

Measure of Peakedness and Outlier detection using Boxplot

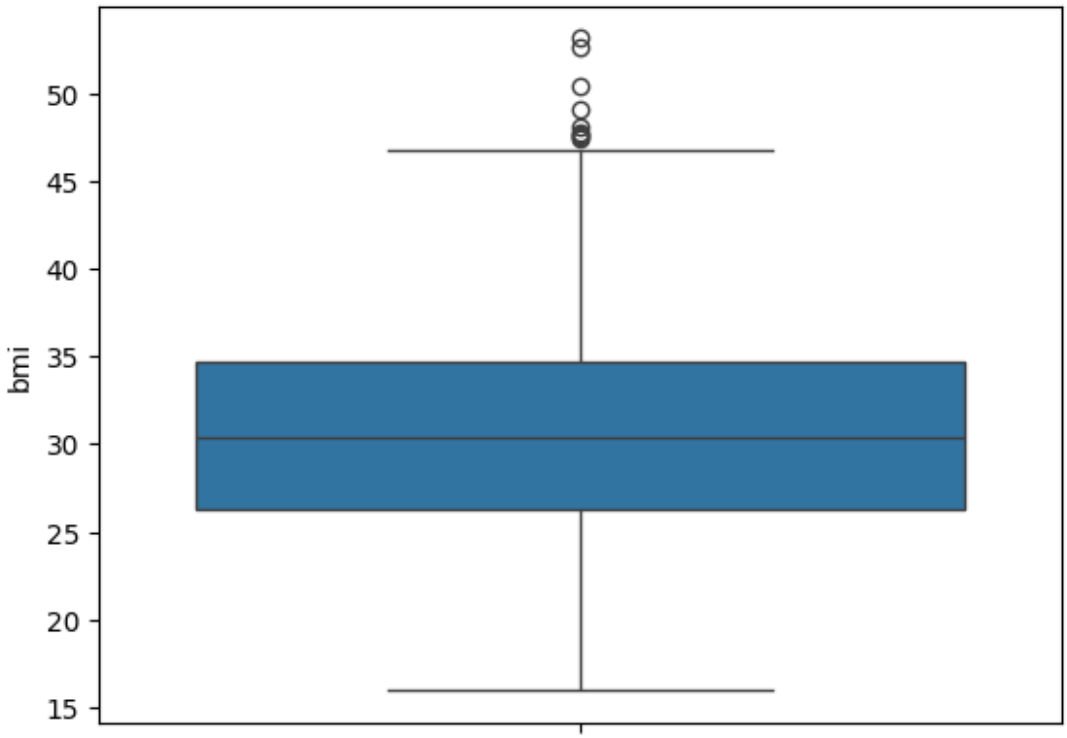
```
import seaborn as sns

sns.boxplot(insurance['age'])
```

```
<Axes: ylabel='age'>
```



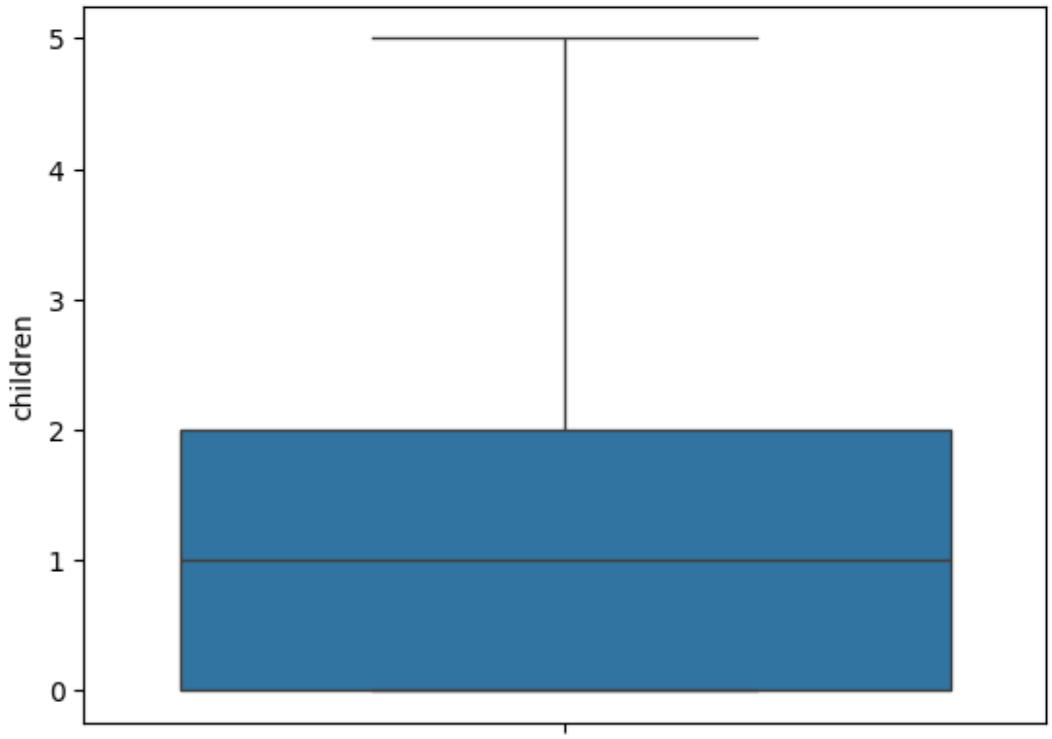

```
sns.boxplot(insurance['bmi'])
```

```
<Axes: ylabel='bmi'>
```



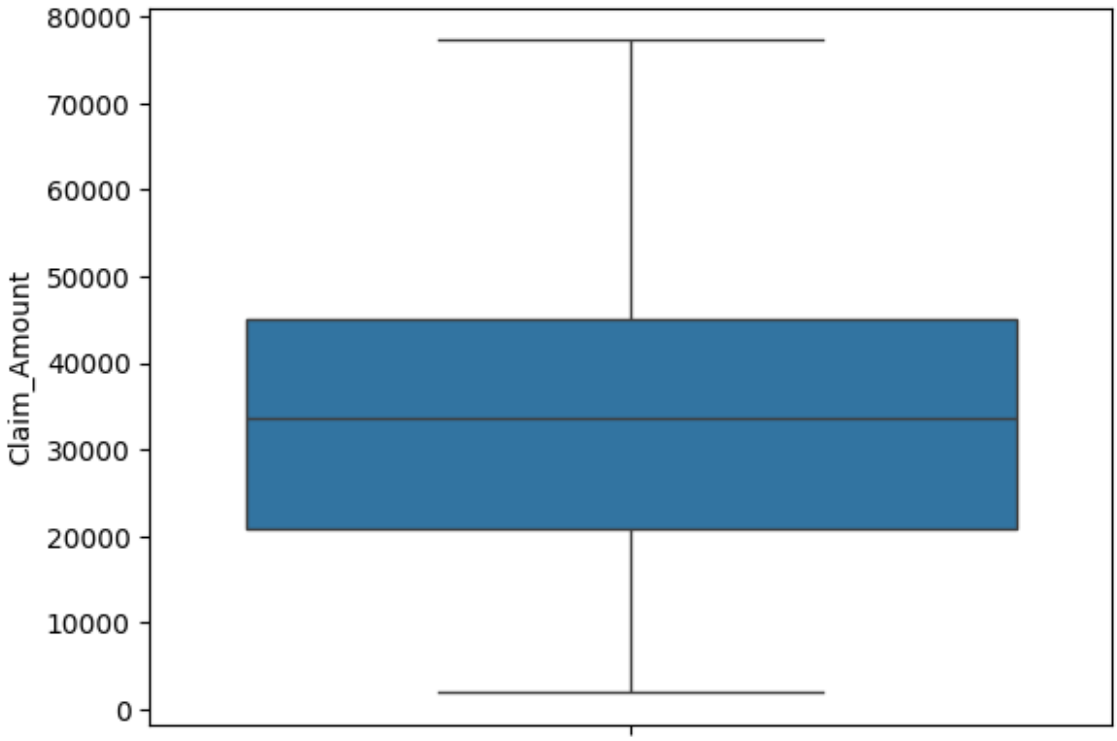
```
sns.boxplot(insurance['children'])
```

```
<Axes: ylabel='children'>
```



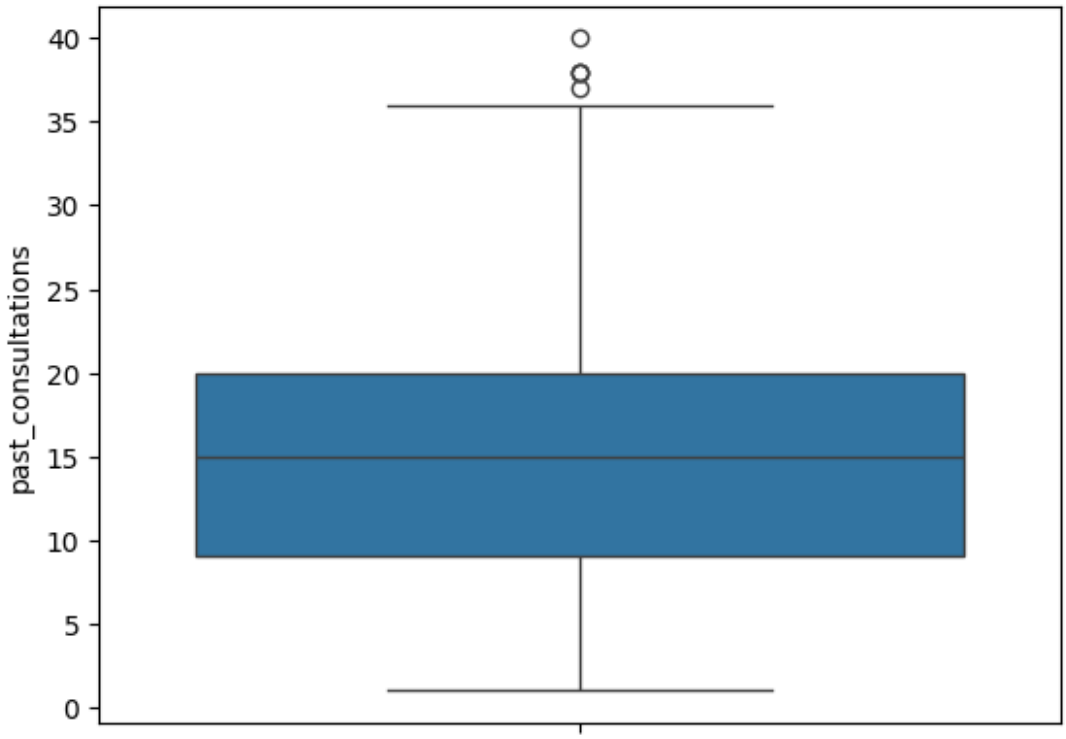

```
sns.boxplot(insurance['Claim_Amount'])
```

```
<Axes: ylabel='Claim_Amount'>
```



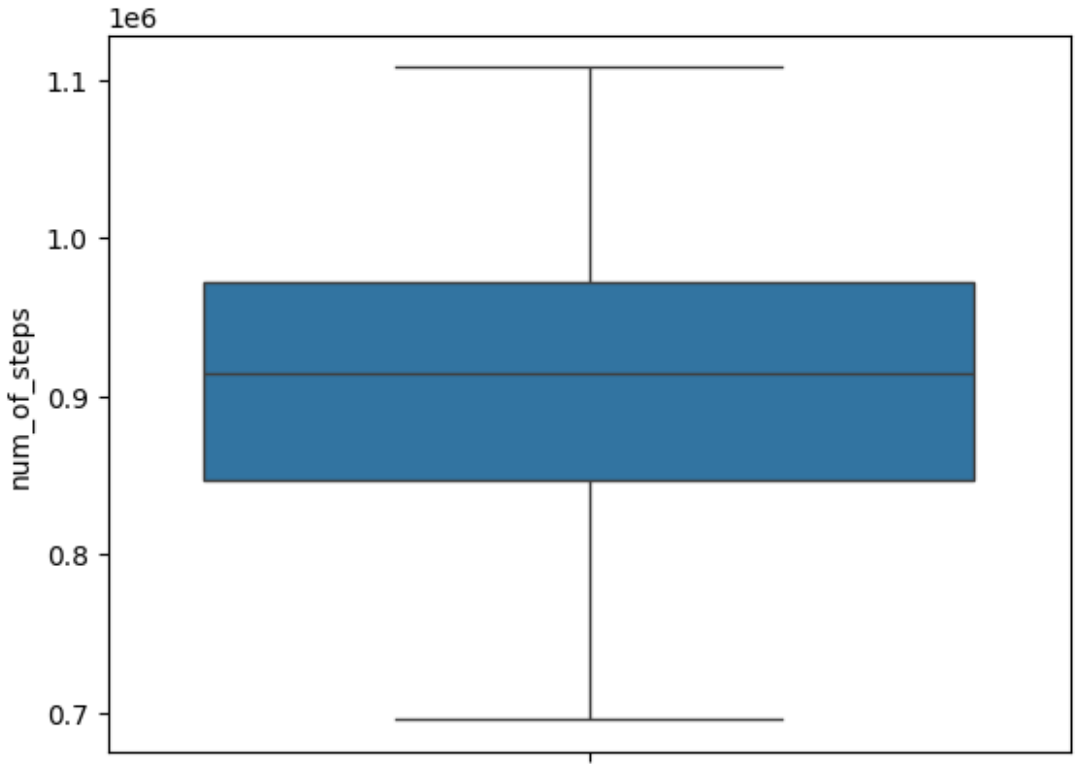

```
sns.boxplot(insurance['past_consultations'])
```

```
<Axes: ylabel='past_consultations'>
```



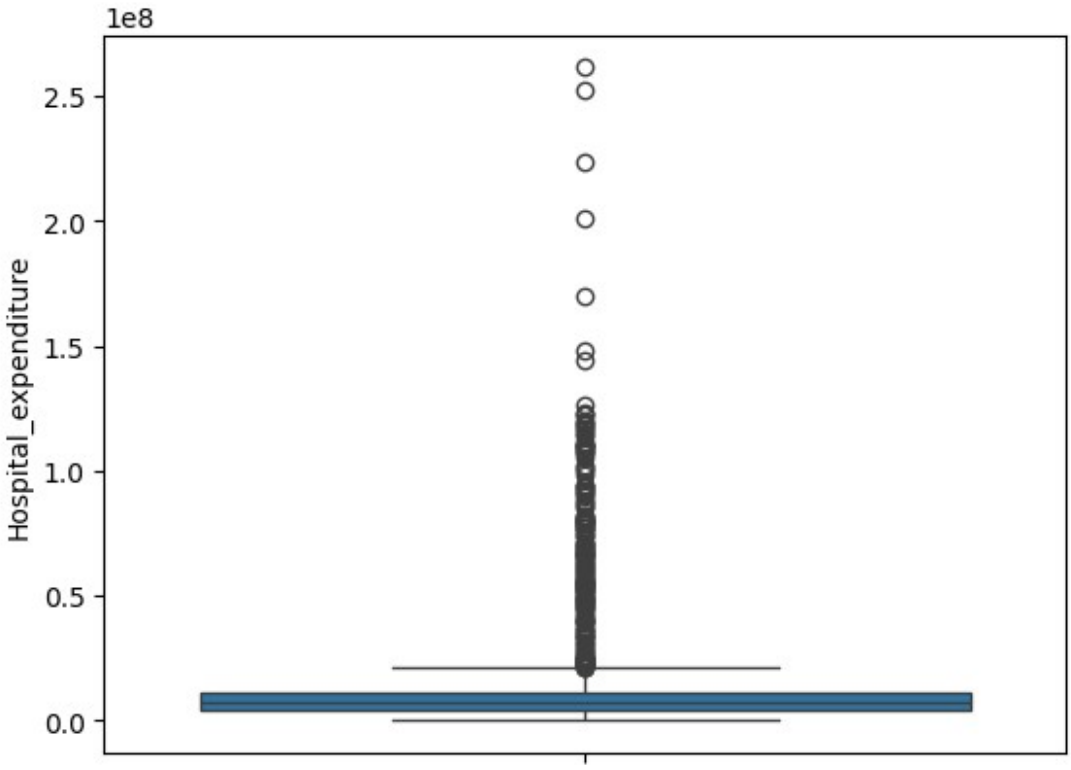
```
sns.boxplot(insurance['num_of_steps'])
```

```
<Axes: ylabel='num_of_steps'>
```



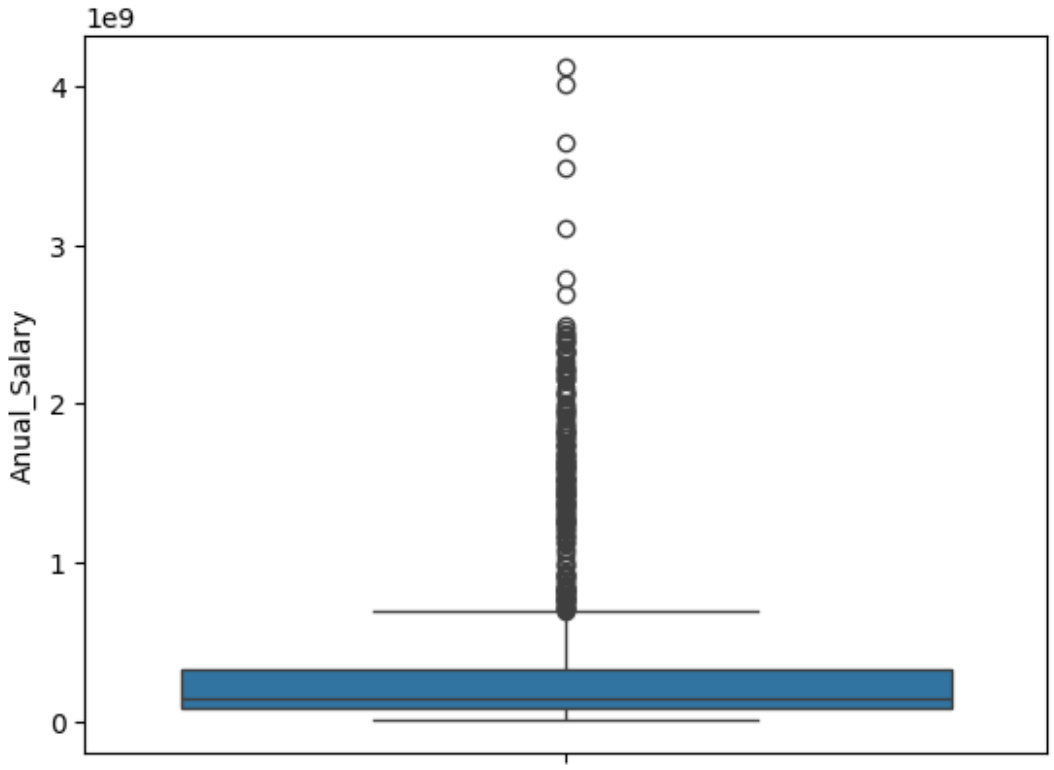

```
sns.boxplot(insurance['Hospital_expenditure'])
```

```
<Axes: ylabel='Hospital_expenditure'>
```



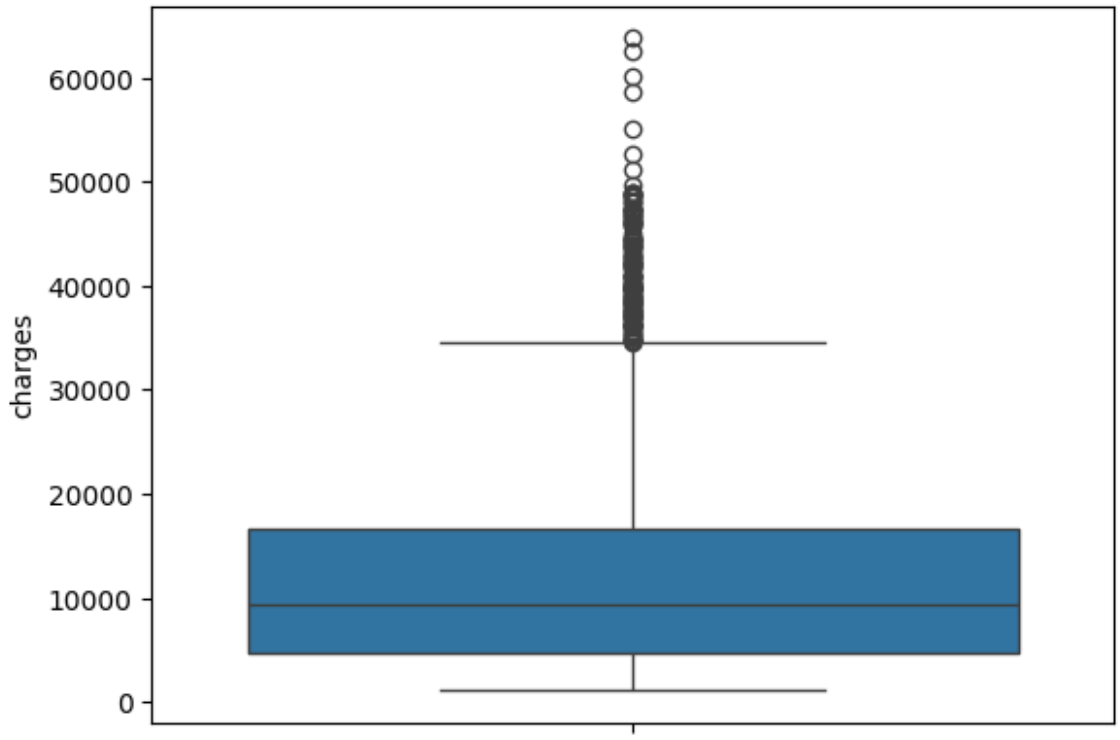

```
sns.boxplot(insurance['Anual_Salary'])
```

```
<Axes: ylabel='Anual_Salary'>
```




```
sns.boxplot(insurance['charges'])
```

```
<Axes: ylabel='charges'>
```



There are presence of outliers in the columns 'charges', 'annual_salary', 'hospital_expenditure', 'past_consultations', 'bmi', etc.

We will not treat the outliers, since the target variable also consists of outliers which is driven from the other independent variables.

Feature Selection for Data Modeling.

```
correlation = insurance.corr()
correlation
```


```
{"summary":"{\n  \"name\": \"correlation\",\n  \"rows\": 15,\n
\"fields\": [\n    {\n      \"column\": \"age\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.27257445614081327,\n        \"min\": -0.0288932380738003,\n
\"max\": 1.0,\n        \"num_unique_values\": 15,\n
\"samples\": [\n          0.2943895277694444,\n          -
0.0288932380738003,\n          1.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"bmi\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0.2564406297632759,\n
\"min\": -0.137597332861898,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.19879386809840113,\n          0.0036106423779739168,\n
0.11284919510677308\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

```
\"children\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.25183656017320133,\n        \"min\": -
0.021080568821585325,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.0707469188846681,\n          0.009469480702483928,\n
0.04155766149411748\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Claim_Amount\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.2686010846762314,\n        \"min\": -
0.02083345325476421,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.4391609746847491,\n          0.3362304606678937,\n
0.12342970446770292\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"past_consultations\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.3069044235294517,\n        \"min\": -
0.029293702614170563,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.6298364993933864,\n          0.5044977969363836,\n
0.16927454475783757\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"num_of_steps\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.3640140715016467,\n        \"min\": -
0.04416676101142949,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.8906422133147405,\n          0.6659034060641287,\n
0.5179300766003345\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Hospital_expenditure\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.3710682596874046,\n        \"min\": -
0.049229323107799726,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.87407851338074,\n          0.6640564736826944,\n
0.1369299265818389\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"NUmber_of_past_hospitalizations\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0.35281582586900034,\n
\"min\": -0.0413973356603584,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.8235807190028873,\n          0.5897255368504415,\n
0.36304051994693687\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Anual_Salary\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.3919045789909955,\n        \"min\": -
0.043900985137417456,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.9542546697464123,\n          0.7465117044030166,\n
0.1643279210200574\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
```

```
\"charges\",\n      \"properties\": {\n        \"dtype\": \"number\",\
n        \"std\": 0.3983920611572902,\n        \"min\": -
0.04321002899168422,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n          1.0,\n
0.7872514304984779,\n          0.2943895277694444\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"sex_male\",\n      \"properties\":
{\n        \"dtype\": \"number\",\n        \"std\":
0.2532836335965133,\n        \"min\": -0.019169614005645024,\n
\"max\": 1.0,\n        \"num_unique_values\": 15,\n
\"samples\": [\n          0.05729206220202525,\n
0.07618481692109515,\n          -0.019169614005645024\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"smoker_yes\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.36678010389241916,\n        \"min\": -0.036945474017607054,\n
\"max\": 1.0,\n        \"num_unique_values\": 15,\n
\"samples\": [\n          0.7872514304984779,\n          1.0,\n
-0.0288932380738003\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"region_northwest\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.2988524866348525,\n        \"min\": -
0.34626466140733164,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n          -
0.039904864040437485,\n          -0.036945474017607005,\n
0.0019406963239010768\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"region_southeast\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.30211241454209914,\n        \"min\": -
0.34626466140733164,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n
0.07398155156575986,\n          0.06849841031175373,\n          -
0.01047969676048724\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"region_southwest\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.2967605422449048,\n        \"min\": -
0.3462646614073316,\n        \"max\": 1.0,\n
\"num_unique_values\": 15,\n        \"samples\": [\n          -
0.04321002899168422,\n          -0.036945474017607054,\n
0.010612023085228302\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"correlation"}
```


Inferences

There is a strong to moderate correlation of the charges column with the following columns:

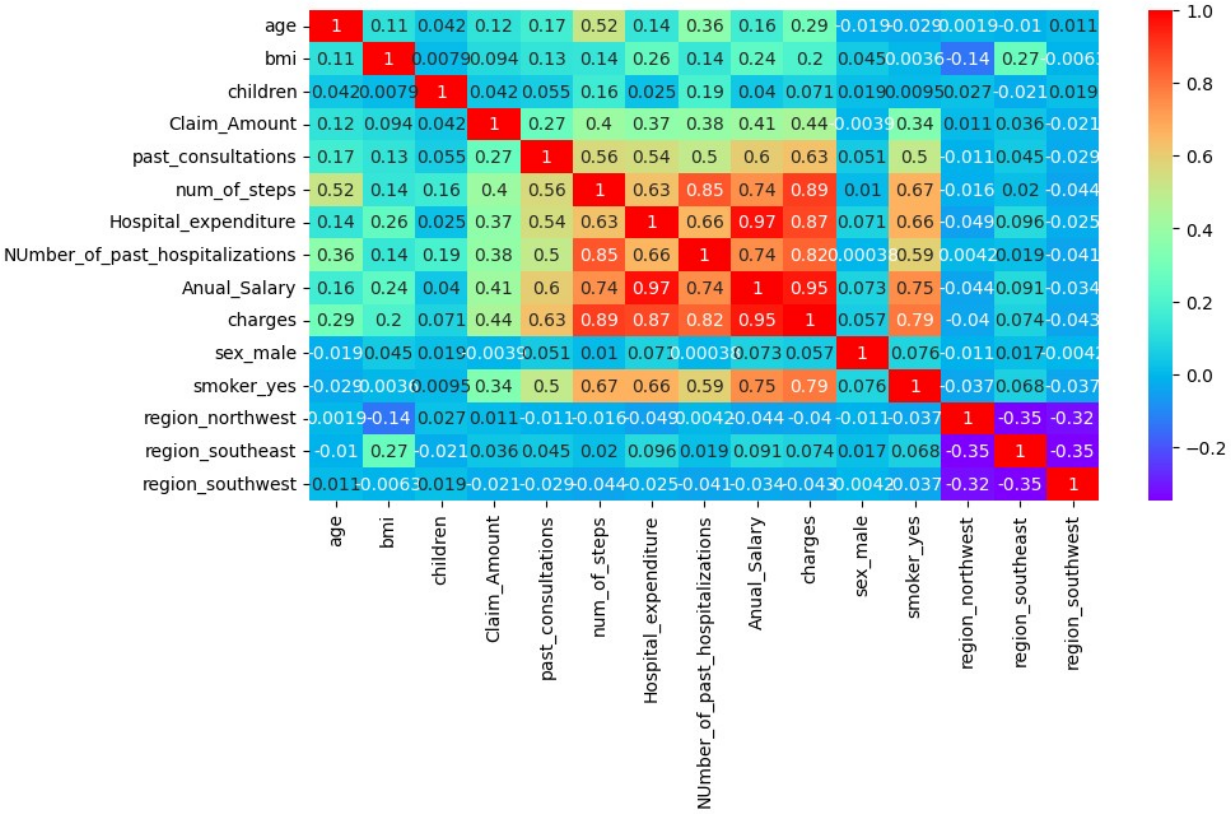
1. smoker
2. Claim_Amount
3. past_consultations

4. num_of_steps
5. Hospital_expenditure
6. NUmber_of_past_hospitalizations
7. Anual_Salary

The column 'age', 'sex', 'region', 'children', 'sex' shows no considerable correlation with the charges column, so we will not be considering these features for the initial model

```
plt.figure(figsize=(10,5))
sns.heatmap(correlation,annot=True,cmap='rainbow')
```

```
<Axes: >
```



The correlation heatmap shows the columns that will be most useful for the modeling. The sex, children, and region column shows no considerable correlation.

Data Preprocessing before model training

```
from sklearn.model_selection import train_test_split

# Since 'sex' and 'region' are already encoded and removed, we drop
only the columns that still exist
X = insurance.drop(['charges', 'age', 'bmi', 'children'], axis=1)  #
```

```
Independent variables
y = insurance['charges']  # Target variable

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

print("Shape of X_train:", X_train.shape)
print("Shape of X_test:", X_test.shape)
```

```
Shape of X_train: (1070, 11)
Shape of X_test: (268, 11)
```

```
X_train
```


```
{"summary":"{\n  \"name\": \"X_train\",\n  \"rows\": 1070,\n
\"fields\": [\n    {\n      \"column\": \"Claim_Amount\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
15583.297247253708,\n        \"min\": 1920.136268,\n        \"max\":
77277.98848,\n        \"num_unique_values\": 1060,\n
\"samples\": [\n          24815.79062,\n          26469.28602,\n
55704.82162\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"past_consultations\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 7.349681741699724,\n        \"min\":
1.0,\n        \"max\": 40.0,\n        \"num_unique_values\": 38,\n
\"samples\": [\n          36.0,\n          33.0,\n          10.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"num_of_steps\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
91851.18606647194,\n        \"min\": 695430.0,\n        \"max\":
1107872.0,\n        \"num_unique_values\": 1068,\n        \"samples\":
[\n          945540.0,\n          898991.0,\n          919946.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Hospital_expenditure\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
26549490.150321122,\n        \"min\": 35822.43757,\n        \"max\":
261631699.3,\n        \"num_unique_values\": 1068,\n
\"samples\": [\n          19977178.6,\n          4909391.306,\n
21805186.36\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"NUmber_of_past_hospitalizations\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0.5357701456653742,\n
\"min\": 0.0,\n        \"max\": 3.0,\n        \"num_unique_values\":
4,\n        \"samples\": [\n          2.0,\n          3.0,\n
1.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Anual_Salary\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 559608118.3285589,\n        \"min\":
2747071.908,\n        \"max\": 4117196637.0,\n
```

\"num_unique_values\": 1066,\n   \"samples\": [\n
120543051.8,\n   521058167.9,\n   374322826.7\
n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"sex_male\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 0.5001280620733015,\n   \"min\":
0.0,\n   \"max\": 1.0,\n   \"num_unique_values\": 2,\n
\"samples\": [\n   1.0,\n   0.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"smoker_yes\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0.40433381343257524,\n   \"min\": 0.0,\n   \"max\": 1.0,\n
\"num_unique_values\": 2,\n   \"samples\": [\n   1.0,\n
0.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"region_northwest\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 0.4290900480356549,\n   \"min\":
0.0,\n   \"max\": 1.0,\n   \"num_unique_values\": 2,\n
\"samples\": [\n   0.0,\n   1.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"region_southeast\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
0.4507526148518467,\n   \"min\": 0.0,\n   \"max\": 1.0,\n
\"num_unique_values\": 2,\n   \"samples\": [\n   1.0,\n
0.0\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"region_southwest\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 0.4273966295702688,\n   \"min\":
0.0,\n   \"max\": 1.0,\n   \"num_unique_values\": 2,\n
\"samples\": [\n   1.0,\n   0.0\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"X_train"}

X_test

{"summary":"{\n \"name\": \"X_test\",\n \"rows\": 268,\n
\"fields\": [\n   {\n   \"column\": \"Claim_Amount\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":
15366.59800126633,\n   \"min\": 3830.10039,\n   \"max\":
65906.21254,\n   \"num_unique_values\": 266,\n
\"samples\": [\n   19676.81836,\n   40228.76482,\n
39832.06667\n   ],\n   \"semantic_type\": \"\",\n
\"description\": \"\"\n   }\n   },\n   {\n   \"column\":
\"past_consultations\",\n   \"properties\": {\n   \"dtype\":
\"number\",\n   \"std\": 7.826164708488113,\n   \"min\":
2.0,\n   \"max\": 38.0,\n   \"num_unique_values\": 34,\n
\"samples\": [\n   8.0,\n   17.0,\n   38.0\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   },\n   {\n   \"column\": \"num_of_steps\",\n
\"properties\": {\n   \"dtype\": \"number\",\n   \"std\":

```
91642.84371897571,\n        \"min\": 729237.0,\n        \"max\":
1086594.0,\n        \"num_unique_values\": 268,\n        \"samples\":
[\n          817473.0,\n          1074883.0,\n          875244.0\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Hospital_expenditure\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
27088705.343827937,\n        \"min\": 29452.53296,\n        \"max\":
122180043.2,\n        \"num_unique_values\": 268,\n
\"samples\": [\n          7452952.984,\n          78892373.24,\n
7225505.199\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"NUmber_of_past_hospitalizations\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0.5229780733259461,\n
\"min\": 0.0,\n        \"max\": 2.0,\n        \"num_unique_values\":
3,\n        \"samples\": [\n          1.0,\n          2.0,\n
0.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Anual_Salary\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 590558144.544042,\n        \"min\":
7109737.472,\n        \"max\": 2446348418.0,\n
\"num_unique_values\": 268,\n        \"samples\": [\n
18842264.78,\n          1852637040.0,\n          119741374.4\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"sex_male\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.5007122212332838,\n        \"min\":
0.0,\n        \"max\": 1.0,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          1.0,\n          0.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"smoker_yes\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.40186551398430176,\n        \"min\": 0.0,\n        \"max\": 1.0,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1.0,\n
0.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"region_northwest\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.4294194086962543,\n        \"min\":
0.0,\n        \"max\": 1.0,\n        \"num_unique_values\": 2,\n
\"samples\": [\n          1.0,\n          0.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"region_southeast\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.4200752190615738,\n        \"min\": 0.0,\n        \"max\": 1.0,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          0.0,\n
1.0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"region_southwest\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0.4359597817443302,\n        \"min\":
0.0,\n        \"max\": 1.0,\n        \"num_unique_values\": 2,\n
```

```
\"samples\": [\n          1.0,\n          0.0\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"X_test"}
```

```
y_train
```

```
560      7731.85785
1285    42983.45850
1142    25656.57526
969     14449.85440
486      6753.03800
           ...
1095    21797.00040
1130    24603.04837
1294    44260.74990
860     12235.83920
1126    24476.47851
Name: charges, Length: 1070, dtype: float64
```

```
y_test
```

```
764     10928.84900
887     12648.70340
890     12797.20962
1293    44202.65360
259      3925.75820
           ...
109      2154.36100
575      8062.76400
535      7371.77200
543      7448.40395
846     12029.28670
Name: charges, Length: 268, dtype: float64
```

Standardizing the Features

```
from sklearn.preprocessing import StandardScaler

# Initialize the scaler
sc = StandardScaler()

# Fit on training data and transform
X_train = sc.fit_transform(X_train)

# Transform test data
X_test = sc.transform(X_test)

X_train
```

```
array([[-0.2457309 , -1.23338465, -0.25442042, ...,  1.76504522,
        -0.62852656, -0.56223942],
       [ 2.14271388,  1.08072229,  1.6774429 , ...,  1.76504522,
        -0.62852656, -0.56223942],
       [ 0.18432091,  0.12785473,  1.00435528, ..., -0.56655772,
        -0.62852656, -0.56223942],
       ...,
       [ 0.58607603,  1.21684623,  1.65298976, ..., -0.56655772,
         1.59102267, -0.56223942],
       [-0.39039156,  0.94459835,  0.36594864, ...,  1.76504522,
        -0.62852656, -0.56223942],
       [ 0.25267611,  1.4890941 ,  1.30298856, ...,  1.76504522,
        -0.62852656, -0.56223942]])
```

```
y_train_arr=y_train.values
y_train=y_train_arr.reshape((-1,1))
```

```
y_train
```

```
array([[ 7731.85785],
       [42983.4585 ],
       [25656.57526],
       ...,
       [44260.7499 ],
       [12235.8392 ],
       [24476.47851]])
```

```
y_train=sc.fit_transform(y_train)
```

Model Building

```
# Filling null values by Median
insurance['age'].fillna(insurance['age'].median(),inplace=True)
insurance['bmi'].fillna(insurance['bmi'].median(),inplace=True)
insurance['children'].fillna(insurance['children'].median(),inplace=Tr
ue)
insurance['Claim_Amount'].fillna(insurance['Claim_Amount'].median(),in
place=True)
insurance['past_consultations'].fillna(insurance['past_consultations']
.median(),inplace=True)
insurance['num_of_steps'].fillna(insurance['num_of_steps'].median(),in
place=True)
insurance['Hospital_expenditure'].fillna(insurance['Hospital_expenditu
re'].median(),inplace=True)
insurance['NUmber_of_past_hospitalizations'].fillna(insurance['NUmber_
of_past_hospitalizations'].median(),inplace=True)
insurance['Anual_Salary'].fillna(insurance['Anual_Salary'].median(),in
place=True)
```

```
/tmp/ipython-input-2509978150.py:2: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  insurance['age'].fillna(insurance['age'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:3: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.


  insurance['bmi'].fillna(insurance['bmi'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:4: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



insurance['children'].fillna(insurance['children'].median(),inplace=Tr
ue)
/tmp/ipython-input-2509978150.py:5: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.
```

```
For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



insurance['Claim_Amount'].fillna(insurance['Claim_Amount'].median(),in
place=True)
/tmp/ipython-input-2509978150.py:6: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



insurance['past_consultations'].fillna(insurance['past_consultations']
.median(),inplace=True)
/tmp/ipython-input-2509978150.py:7: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.



insurance['num_of_steps'].fillna(insurance['num_of_steps'].median(),in
place=True)
/tmp/ipython-input-2509978150.py:8: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
```

```
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
insurance['Hospital_expenditure'].fillna(insurance['Hospital_expenditu
re'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:9: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
insurance['NUmber_of_past_hospitalizations'].fillna(insurance['NUmber_
of_past_hospitalizations'].median(),inplace=True)
/tmp/ipython-input-2509978150.py:10: FutureWarning: A value is trying
to be set on a copy of a DataFrame or Series through chained
assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never
work because the intermediate object on which we are setting values
always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try
using 'df.method({col: value}, inplace=True)' or df[col] =
df[col].method(value) instead, to perform the operation inplace on the
original object.
```

```
insurance['Anual_Salary'].fillna(insurance['Anual_Salary'].median(),in
place=True)
```

```
insurance.isna().sum()
```

```
age                                0
bmi                                0
children                           0
Claim_Amount                       0
past_consultations                 0
num_of_steps                       0
Hospital_expenditure               0
NUmber_of_past_hospitalizations    0
Anual_Salary                       0
```

```
charges                             0
sex_male                            0
smoker_yes                          0
region_northwest                    0
region_southeast                    0
region_southwest                    0
dtype: int64
```

```
from sklearn.linear_model import LinearRegression

model_uno = LinearRegression()
#fitting the model
model_uno.fit(X_train, y_train)
```

```
LinearRegression()
```

```
import numpy as np
print(np.any(np.isnan(X_train)), np.any(np.isinf(X_train)))
print(np.any(np.isnan(y_train)), np.any(np.isinf(y_train)))
```

```
False False
False False
```

```
insurance.isna().sum()
```

```
age                                 0
bmi                                 0
children                            0
Claim_Amount                        0
past_consultations                  0
num_of_steps                        0
Hospital_expenditure                0
NUmber_of_past_hospitalizations     0
Anual_Salary                        0
charges                             0
sex_male                            0
smoker_yes                          0
region_northwest                    0
region_southeast                    0
region_southwest                    0
dtype: int64
```

```
#predictions - Model
predictions = model_uno.predict(X_test)
predictions
```

```
array([[-1.94381147e-01],
       [-1.41032881e-01],
       [-1.02141547e-01],
       [ 2.53273849e+00],
       [-7.28147817e-01],
```

```
       [ 2.94835985e-01],
       [-1.07170791e-01],
       [-1.65297250e-01],
       [ 2.44939729e+00],
       [-2.55673611e-01],
       [-2.24672177e-01],
       [-3.35770857e-01],
       [-1.11525773e+00],
       [-6.57309714e-01],
       [-1.04469710e-01],
       [-6.36902802e-02],
       [-3.28233481e-01],
       [-3.15348956e-01],
       [-4.18931372e-01],
       [-4.50802964e-02],
       [-4.89459461e-01],
       [-3.76221937e-01],
       [-4.51347183e-01],
       [-9.30913678e-01],
       [ 8.43518403e-01],
       [-3.48927748e-01],
       [-7.99515166e-02],
       [-4.88511016e-01],
       [-6.96036200e-01],
       [-9.08607803e-02],
       [ 2.20648480e+00],
       [-5.67212043e-02],
       [-2.03674428e-01],
       [ 9.60611797e-01],
       [ 6.19043414e-01],
       [-2.89420872e-01],
       [-5.83919860e-01],
       [ 2.30921182e+00],
       [ 1.14354096e+00],
       [-1.00017593e+00],
       [-1.04133953e+00],
       [-4.75894679e-01],
       [ 3.57789813e-02],
       [ 3.68267645e-01],
       [-6.47539308e-01],
       [-8.47956933e-01],
       [-1.04866032e+00],
       [-1.25426758e-01],
       [-6.72202532e-01],
       [-7.54097951e-01],
       [-3.59726028e-01],
       [-7.72803355e-01],
       [-6.06230963e-02],
       [ 2.69167664e+00],
```

```
       [ 1.91982958e+00],
       [-3.66655599e-02],
       [-9.86037781e-01],
       [-6.22190047e-01],
       [-1.22484835e-01],
       [-8.63958538e-01],
       [-7.87261609e-01],
       [-3.65162676e-01],
       [-1.62998742e-01],
       [-5.89781915e-01],
       [ 5.88651474e-01],
       [-1.07841822e+00],
       [-3.91336852e-01],
       [-7.17769603e-01],
       [ 5.42963844e-01],
       [-1.01465583e-01],
       [-1.14580335e+00],
       [-5.74969709e-01],
       [ 2.37664239e+00],
       [ 1.04198405e+00],
       [-7.37997889e-01],
       [ 1.35649415e-01],
       [-9.21857692e-02],
       [-4.20488641e-01],
       [-6.82955169e-01],
       [ 2.15263376e+00],
       [-6.39045277e-01],
       [ 2.98800512e+00],
       [-8.92688788e-01],
       [-7.43066986e-01],
       [ 3.97411587e-01],
       [-7.27240674e-01],
       [-2.76885806e-01],
       [-4.64124567e-02],
       [-2.30413103e-01],
       [-2.25591705e-01],
       [-4.38388921e-01],
       [-2.75553439e-01],
       [-8.60662931e-02],
       [-2.08981617e-01],
       [ 9.29257452e-01],
       [ 2.78320792e+00],
       [-1.27318415e-01],
       [-2.52805810e-01],
       [ 3.03539301e+00],
       [-9.67165510e-02],
       [-1.55465764e-03],
       [-1.13318325e+00],
       [-3.28274373e-01],
```

```
       [-3.05955261e-01],
       [-1.22656753e-01],
       [-2.88708990e-01],
       [-8.36469428e-01],
       [-1.06249053e-01],
       [ 1.89934262e+00],
       [-1.12973699e+00],
       [-6.15145922e-01],
       [ 3.94011909e-02],
       [ 1.61904684e+00],
       [-1.07935709e+00],
       [-5.98187470e-01],
       [-8.06212774e-01],
       [-1.01454012e+00],
       [ 1.17942967e+00],
       [-1.11955403e+00],
       [-2.27785592e-02],
       [-3.80604786e-01],
       [-1.57448133e-01],
       [-5.75242705e-01],
       [-7.58371176e-01],
       [-3.21513746e-02],
       [-6.33819032e-01],
       [-3.95407777e-01],
       [ 4.81831723e-01],
       [-3.00447466e-01],
       [-6.37025779e-01],
       [-3.25563457e-01],
       [-6.50908463e-02],
       [ 7.37914144e-01],
       [-9.88193358e-01],
       [-8.43696067e-01],
       [ 1.84877827e+00],
       [-6.04335585e-01],
       [-4.70264184e-01],
       [-3.16404514e-01],
       [-1.48045723e-01],
       [ 2.63890114e-01],
       [-4.60961444e-01],
       [ 1.79981664e-01],
       [-3.20333976e-01],
       [-8.54025369e-01],
       [-2.47381047e-01],
       [ 2.71933002e+00],
       [-3.84545826e-01],
       [ 2.17800187e+00],
       [-5.81724591e-01],
       [ 8.48268344e-01],
       [-1.61321126e-01],
```

```
       [-1.46165186e-01],
       [ 2.24911082e+00],
       [-3.86807094e-01],
       [ 1.57624360e-01],
       [-2.42369742e-02],
       [-2.59618566e-01],
       [-3.15828923e-01],
       [-5.30971438e-01],
       [-1.00724762e+00],
       [ 2.67950966e+00],
       [-6.37727323e-01],
       [ 8.19356770e-01],
       [ 5.77896628e-01],
       [ 1.91606940e+00],
       [-9.68734030e-01],
       [-9.92973906e-01],
       [-1.06373948e+00],
       [ 1.23599424e+00],
       [-2.93537265e-01],
       [-1.71911725e-02],
       [ 6.98528953e-01],
       [ 3.07158558e+00],
       [ 1.23958097e+00],
       [-7.16701492e-01],
       [-6.04952901e-01],
       [-3.95496748e-01],
       [-8.95306354e-01],
       [-5.66912632e-01],
       [-7.08825521e-01],
       [-5.17542870e-01],
       [-3.00927912e-01],
       [-8.03920094e-01],
       [-1.79634144e-01],
       [-9.22510807e-01],
       [-6.43810282e-01],
       [-6.26528630e-01],
       [ 3.91043506e-01],
       [ 6.30958325e-01],
       [-2.00272426e-01],
       [-5.40712841e-01],
       [-1.22056551e-01],
       [-1.05723972e+00],
       [ 3.10103137e+00],
       [-2.72105398e-01],
       [-2.68166856e-01],
       [ 4.71329960e-01],
       [-1.20236790e+00],
       [-5.52218449e-01],
       [ 6.96732369e-01],
```

```
       [-7.02759063e-01],
       [-1.39299235e-01],
       [ 3.42509906e-01],
       [-7.06024071e-01],
       [-9.82697490e-01],
       [-1.49241879e-01],
       [ 2.08628658e+00],
       [ 7.45194370e-02],
       [ 2.67113889e+00],
       [-4.30636720e-01],
       [-1.73818725e-01],
       [-6.98512662e-01],
       [ 2.50333523e+00],
       [ 1.91539366e+00],
       [ 4.43652121e-01],
       [-3.47254868e-01],
       [-7.05260392e-01],
       [-1.07085323e+00],
       [-2.34393343e-02],
       [-1.04788454e+00],
       [-9.07323394e-01],
       [-2.98491374e-01],
       [-3.30871711e-01],
       [ 1.70865573e+00],
       [ 4.49173334e-01],
       [ 2.82134389e+00],
       [-2.95352029e-01],
       [ 3.13210357e+00],
       [-6.54056320e-01],
       [-4.65347983e-01],
       [-8.94453840e-01],
       [ 8.86067646e-01],
       [ 2.77589987e+00],
       [ 1.00131239e+00],
       [-8.30965835e-02],
       [-9.73272964e-01],
       [ 9.31698782e-01],
       [ 3.06043429e+00],
       [ 7.51540487e-01],
       [ 5.53791681e-01],
       [-1.88144549e-01],
       [ 2.37220124e+00],
       [-7.12182158e-01],
       [-5.30553057e-01],
       [ 2.14748233e+00],
       [-7.50053516e-01],
       [ 8.28428047e-01],
       [-4.64842817e-01],
       [-5.30441871e-01],
```

```
       [-1.47442894e-01],
       [-8.36226952e-01],
       [ 5.60855882e-01],
       [-6.13043936e-01],
       [-8.21923057e-01],
       [-4.05937484e-02],
       [-2.55518538e-01],
       [ 2.35277439e+00],
       [-9.84605578e-03],
       [-2.34870923e-01],
       [-1.05186160e+00],
       [-5.47123912e-01],
       [-2.85226469e-01],
       [-9.37522251e-01],
       [-4.09752368e-01],
       [-4.06972901e-01],
       [-4.31504445e-01],
       [-6.47290675e-02]])
```

```
predictions=sc.inverse_transform(predictions)
predictions
```

```
array([[10838.90034367],
       [11477.18675879],
       [11942.50290349],
       [43467.57823658],
       [ 4452.63825688],
       [16692.14859622],
       [11882.33041396],
       [11186.87522267],
       [42470.44077729],
       [10105.56542285],
       [10476.48270765],
       [ 9147.24030589],
       [ -178.94614156],
       [ 5300.18216738],
       [11914.6475484 ],
       [12402.55386141],
       [ 9237.42140111],
       [ 9391.57856341],
       [ 8152.26464644],
       [12625.21338884],
       [ 7308.42990906],
       [ 8663.26254358],
       [ 7764.42502746],
       [ 2026.64202066],
       [23256.87044931],
       [ 8989.82442043],
       [12207.99597084],
       [ 7319.77759774],
```

```
       [ 4836.83834698],
       [12077.47186481],
       [39564.10945183],
       [12485.93551072],
       [10727.71069729],
       [24657.83673167],
       [20571.13492557],
       [ 9701.79564061],
       [ 6178.25656104],
       [40793.1888721 ],
       [26846.4964101 ],
       [ 1197.9524316 ],
       [  705.44974156],
       [ 7470.7260214 ],
       [13592.65584772],
       [17570.72317763],
       [ 5417.08039403],
       [ 3019.17967685],
       [  617.85995619],
       [11663.90653064],
       [ 5121.99674387],
       [ 4142.15735902],
       [ 8860.62818908],
       [ 3918.35616939],
       [12439.25124535],
       [45369.19699239],
       [36134.41725017],
       [12725.89167279],
       [ 1367.10857577],
       [ 5720.37213689],
       [11699.10522308],
       [ 2827.72814212],
       [ 3745.37009961],
       [ 8795.581303  ],
       [11214.37576445],
       [ 6108.11988831],
       [20207.50994667],
       [  261.82101139],
       [ 8482.41983901],
       [ 4576.8086117 ],
       [19660.87937134],
       [11950.59048562],
       [ -544.40983512],
       [ 6285.34082379],
       [41599.96331843],
       [25631.41669084],
       [ 4334.78687369],
       [14787.55766723],
       [12061.61900826],
```

```
       [ 8133.63267459],
       [ 4993.34660027],
       [38919.80754283],
       [ 5518.70739893],
       [48914.62510341],
       [ 2483.98448921],
       [ 4274.13755906],
       [17919.41645032],
       [ 4463.49179208],
       [ 9851.77169478],
       [12609.27472997],
       [10407.79528895],
       [10465.48099328],
       [ 7919.46440817],
       [ 9867.71282599],
       [12134.83560333],
       [10664.21273267],
       [24282.69703475],
       [46464.3248064 ],
       [11641.2737575 ],
       [10139.87729265],
       [49481.5984451 ],
       [12007.41037843],
       [13145.9767713 ],
       [ -393.41639134],
       [ 9236.93214321],
       [ 9503.96961368],
       [11697.04830331],
       [ 9710.31297046],
       [ 3156.62215791],
       [11893.35856597],
       [35889.30058543],
       [ -352.18356543],
       [ 5804.65170354],
       [13635.99384762],
       [32535.69613588],
       [  250.58785311],
       [ 6007.55145189],
       [ 3518.62850302],
       [ 1026.0917541 ],
       [27275.88760085],
       [ -230.34923044],
       [12892.04297318],
       [ 8610.82385906],
       [11280.78614291],
       [ 6282.07455088],
       [ 4091.03026524],
       [12779.90173411],
       [ 5581.23691038],
```

```
       [ 8433.71318281],
       [18929.46287143],
       [ 9569.86774033],
       [ 5542.86971771],
       [ 9269.36694991],
       [12385.79676056],
       [21993.36616866],
       [ 1341.31813835],
       [ 3070.15888737],
       [35284.32246367],
       [ 5933.99220258],
       [ 7538.09218925],
       [ 9378.94931122],
       [11393.28146262],
       [16321.89609933],
       [ 7649.3950028 ],
       [15317.97175927],
       [ 9331.9351912 ],
       [ 2946.5737485 ],
       [10204.78197809],
       [45700.05635336],
       [ 8563.67120667],
       [39223.32480402],
       [ 6204.5218979 ],
       [23313.70122251],
       [11234.4476378 ],
       [11415.7811865 ],
       [40074.10933173],
       [ 8536.61622099],
       [15050.47733687],
       [12874.59373722],
       [10058.36592796],
       [ 9385.83598767],
       [ 6811.75901988],
       [ 1113.34307149],
       [45223.62484769],
       [ 5534.47608536],
       [22967.78810554],
       [20078.83336858],
       [36089.42851407],
       [ 1574.13965324],
       [ 1284.12117147],
       [  437.44498539],
       [27952.65554221],
       [ 9652.54497819],
       [12958.89337396],
       [21522.14125496],
       [49914.62519688],
       [27995.56908281],
```

```
       [ 4589.58804521],
       [ 5926.6063139 ],
       [ 8432.64868873],
       [ 2452.66657319],
       [ 6381.73988356],
       [ 4683.82025274],
       [ 6972.42539645],
       [ 9564.11943248],
       [ 3546.05931828],
       [11015.34116023],
       [ 2127.17833653],
       [ 5461.69639573],
       [ 5668.46307469],
       [17843.2254259 ],
       [20713.69111486],
       [10768.41402473],
       [ 6695.20780543],
       [11704.22943732],
       [  515.21154517],
       [50266.92984265],
       [ 9908.96697646],
       [ 9956.08975024],
       [18803.81432554],
       [-1221.1774366 ],
       [ 6557.54873225],
       [21500.64598997],
       [ 4756.40251512],
       [11497.9289997 ],
       [17262.54418406],
       [ 4717.33826098],
       [ 1407.07356098],
       [11378.97003707],
       [38125.99533772],
       [14056.16681185],
       [45123.47262674],
       [ 8012.21577501],
       [11084.91986463],
       [ 4807.20866966],
       [43115.78238204],
       [36081.34354572],
       [18472.6622605 ],
       [ 9009.83963234],
       [ 4726.47531139],
       [  352.33240874],
       [12884.13711647],
       [  627.14180862],
       [ 2308.8884491 ],
       [ 9593.27144009],
       [ 9205.85623816],
```

```
       [33607.8234353 ],
       [18538.72091982],
       [46920.60334   ],
       [ 9630.8321887 ],
       [50638.69383936],
       [ 5339.10746261],
       [ 7596.91217656],
       [ 2462.86649284],
       [23765.95173613],
       [46376.88753126],
       [25144.79977076],
       [12170.36675421],
       [ 1519.83349036],
       [24311.90638106],
       [49781.20534917],
       [22156.39882761],
       [19790.42924382],
       [10913.51825346],
       [41546.82709898],
       [ 4643.65970036],
       [ 6816.76474857],
       [38858.17322578],
       [ 4190.54708344],
       [23076.32158029],
       [ 7602.95624228],
       [ 6818.09504243],
       [11400.4940233 ],
       [ 3159.52327148],
       [19874.94902553],
       [ 5829.80096438],
       [ 3330.66250305],
       [12678.89278229],
       [10107.42080129],
       [41314.39413087],
       [13046.77417104],
       [10354.45961148],
       [  579.55820678],
       [ 6618.50241733],
       [ 9751.97965968],
       [ 1947.57361608],
       [ 8262.08703217],
       [ 8295.34202385],
       [ 8001.83386334],
       [12390.12526961]])
```

```
y_test
```

```
764     10928.84900
887     12648.70340
890     12797.20962
```

```
1293    44202.65360
259      3925.75820
           ...
109      2154.36100
575      8062.76400
535      7371.77200
543      7448.40395
846     12029.28670
Name: charges, Length: 268, dtype: float64
```

Model Evaluation

```
# R2 score

from sklearn.metrics import *
r2_score(y_test,predictions)
```

```
0.9711828012265369
```

R2 Score is 0.9712 which is close to 1. Our model is Good Fitted. It explains about 97.12% of variation in target variable on the effect of Independent variables.

```
!pip install ydata-profiling
```

```
Requirement already satisfied: ydata-profiling in
/usr/local/lib/python3.12/dist-packages (4.17.0)
Requirement already satisfied: scipy<1.16,>=1.4.1 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(1.15.3)
Requirement already satisfied: pandas!=1.4.0,<3.0,>1.1 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (2.2.2)
Requirement already satisfied: matplotlib<=3.10,>=3.5 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(3.10.0)
Requirement already satisfied: pydantic>=2 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(2.11.10)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (6.0.3)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (3.1.6)
Requirement already satisfied: visions<0.8.2,>=0.7.5 in
/usr/local/lib/python3.12/dist-packages (from
visions[type_image_path]<0.8.2,>=0.7.5->ydata-profiling) (0.8.1)
Requirement already satisfied: numpy<2.2,>=1.16.0 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (2.0.2)
Requirement already satisfied: minify-html>=0.15.0 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
```

```
(0.18.1)
Requirement already satisfied: filetype>=1.0.0 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (1.2.0)
Requirement already satisfied: phik<0.13,>=0.11.1 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(0.12.5)
Requirement already satisfied: requests<3,>=2.24.0 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(2.32.4)
Requirement already satisfied: tqdm<5,>=4.48.2 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(4.67.1)
Requirement already satisfied: seaborn<0.14,>=0.10.1 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(0.13.2)
Requirement already satisfied: multimethod<2,>=1.4 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (1.12)
Requirement already satisfied: statsmodels<1,>=0.13.2 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(0.14.5)
Requirement already satisfied: typeguard<5,>=3 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (4.4.4)
Requirement already satisfied: imagehash==4.3.1 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (4.3.1)
Requirement already satisfied: wordcloud>=1.9.3 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (1.9.4)
Requirement already satisfied: dacite>=1.8 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling) (1.9.2)
Requirement already satisfied: numba<=0.61,>=0.56.0 in
/usr/local/lib/python3.12/dist-packages (from ydata-profiling)
(0.60.0)
Requirement already satisfied: PyWavelets in
/usr/local/lib/python3.12/dist-packages (from imagehash==4.3.1->ydata-
profiling) (1.9.0)
Requirement already satisfied: pillow in
/usr/local/lib/python3.12/dist-packages (from imagehash==4.3.1->ydata-
profiling) (11.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.12/dist-packages (from jinja2<3.2,>=2.11.1-
>ydata-profiling) (3.0.3)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.12/dist-packages (from matplotlib<=3.10,>=3.5-
>ydata-profiling) (1.3.3)
Requirement already satisfied: cycler>=0.10 in
/usr/local/lib/python3.12/dist-packages (from matplotlib<=3.10,>=3.5-
>ydata-profiling) (0.12.1)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.12/dist-packages (from matplotlib<=3.10,>=3.5-
>ydata-profiling) (4.60.1)
```

```
Requirement already satisfied: kiwisolver>=1.3.1 in
/usr/local/lib/python3.12/dist-packages (from matplotlib<=3.10,>=3.5-
>ydata-profiling) (1.4.9)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.12/dist-packages (from matplotlib<=3.10,>=3.5-
>ydata-profiling) (25.0)
Requirement already satisfied: pyparsing>=2.3.1 in
/usr/local/lib/python3.12/dist-packages (from matplotlib<=3.10,>=3.5-
>ydata-profiling) (3.2.5)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.12/dist-packages (from matplotlib<=3.10,>=3.5-
>ydata-profiling) (2.9.0.post0)
Requirement already satisfied: llvmlite<0.44,>=0.43.0dev0 in
/usr/local/lib/python3.12/dist-packages (from numba<=0.61,>=0.56.0-
>ydata-profiling) (0.43.0)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.12/dist-packages (from pandas!=1.4.0,<3.0,>1.1-
>ydata-profiling) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in
/usr/local/lib/python3.12/dist-packages (from pandas!=1.4.0,<3.0,>1.1-
>ydata-profiling) (2025.2)
Requirement already satisfied: joblib>=0.14.1 in
/usr/local/lib/python3.12/dist-packages (from phik<0.13,>=0.11.1-
>ydata-profiling) (1.5.2)
Requirement already satisfied: annotated-types>=0.6.0 in
/usr/local/lib/python3.12/dist-packages (from pydantic>=2->ydata-
profiling) (0.7.0)
Requirement already satisfied: pydantic-core==2.33.2 in
/usr/local/lib/python3.12/dist-packages (from pydantic>=2->ydata-
profiling) (2.33.2)
Requirement already satisfied: typing-extensions>=4.12.2 in
/usr/local/lib/python3.12/dist-packages (from pydantic>=2->ydata-
profiling) (4.15.0)
Requirement already satisfied: typing-inspection>=0.4.0 in
/usr/local/lib/python3.12/dist-packages (from pydantic>=2->ydata-
profiling) (0.4.2)
Requirement already satisfied: charset_normalizer<4,>=2 in
/usr/local/lib/python3.12/dist-packages (from requests<3,>=2.24.0-
>ydata-profiling) (3.4.4)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.12/dist-packages (from requests<3,>=2.24.0-
>ydata-profiling) (3.11)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.12/dist-packages (from requests<3,>=2.24.0-
>ydata-profiling) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.12/dist-packages (from requests<3,>=2.24.0-
>ydata-profiling) (2025.10.5)
Requirement already satisfied: patsy>=0.5.6 in
```

```
/usr/local/lib/python3.12/dist-packages (from statsmodels<1,>=0.13.2->ydata-profiling) (1.0.2)
Requirement already satisfied: attrs>=19.3.0 in /usr/local/lib/python3.12/dist-packages (from visions<0.8.2,>=0.7.5->visions[type_image_path]<0.8.2,>=0.7.5->ydata-profiling) (25.4.0)
Requirement already satisfied: networkx>=2.4 in /usr/local/lib/python3.12/dist-packages (from visions<0.8.2,>=0.7.5->visions[type_image_path]<0.8.2,>=0.7.5->ydata-profiling) (3.5)
Requirement already satisfied: puremagic in /usr/local/lib/python3.12/dist-packages (from visions<0.8.2,>=0.7.5->visions[type_image_path]<0.8.2,>=0.7.5->ydata-profiling) (1.30)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.7->matplotlib<=3.10,>=3.5->ydata-profiling) (1.17.0)
```

```
from ydata_profiling import ProfileReport

profile=ProfileReport(insurance,title="ML Profile Report",explorative=True)

profile
```

{"model_id":"93f49e315d4949d5bef219197ab9caff","version_major":2,"version_minor":0}

```
  0%|          | 0/15 [00:00<?, ?it/s]
```

{"model_id":"e28024411535453c9b79ed016b63b8c2","version_major":2,"version_minor":0}

{"model_id":"c7b94a083108433187c85bcaf47c2fd0","version_major":2,"version_minor":0}

```
<IPython.core.display.HTML object>
```

```
profile.to_file("dataset_profile_report.html")
```

{"model_id":"97f459f35a1d417199b23d05efef30ad","version_major":2,"version_minor":0}