01_exploration

January 20, 2026

1 01_exploration.ipynb - Exploratory Data Analysis

Objective: Predict medical charges (`charges`) using regression models.

This notebook follows the following required workflow: 1. Load and inspect data 2. Check for missing values 3. Visualize distributions 4. Correlation analysis 5. Identify potential issues

2 1. Load and Inspect Data

2.1 1.1 Import libraries

Import core python libraries for data analysis.

[ ]:
```
# Load libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

2.2 1.2 Load dataset

The dataset is loaded from `data/` folder to ensure reproducibility.

[ ]:
```
# Load dataset from local path (repo data folder)
from pathlib import Path
DATA_PATH = Path("../data/insurance.csv")
df = pd.read_csv(DATA_PATH)


print("Insurance Dataset:")
print("Samples:", df.shape[0])
print("Features:", df.shape[1])
print("List of features:", df.columns.to_list())
df.head()
```

```
Insurance Dataset:
Samples: 1338
Features: 7
List of features: ['age', 'sex', 'bmi', 'children', 'smoker', 'region',
'charges']
```

[ ]:

	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

2.3 1.3 Handle missing values

We check the dataset for any missing values.

[ ]:
```
# Check for missing values
missing_values = df.isnull().sum()
print("Missing values:")
print(missing_values.to_string())
print("% of dataset with missing values:", missing_values.sum() / df.shape[0] *␣
↪100)
```

```
Missing values:
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
% of dataset with missing values: 0.0
```

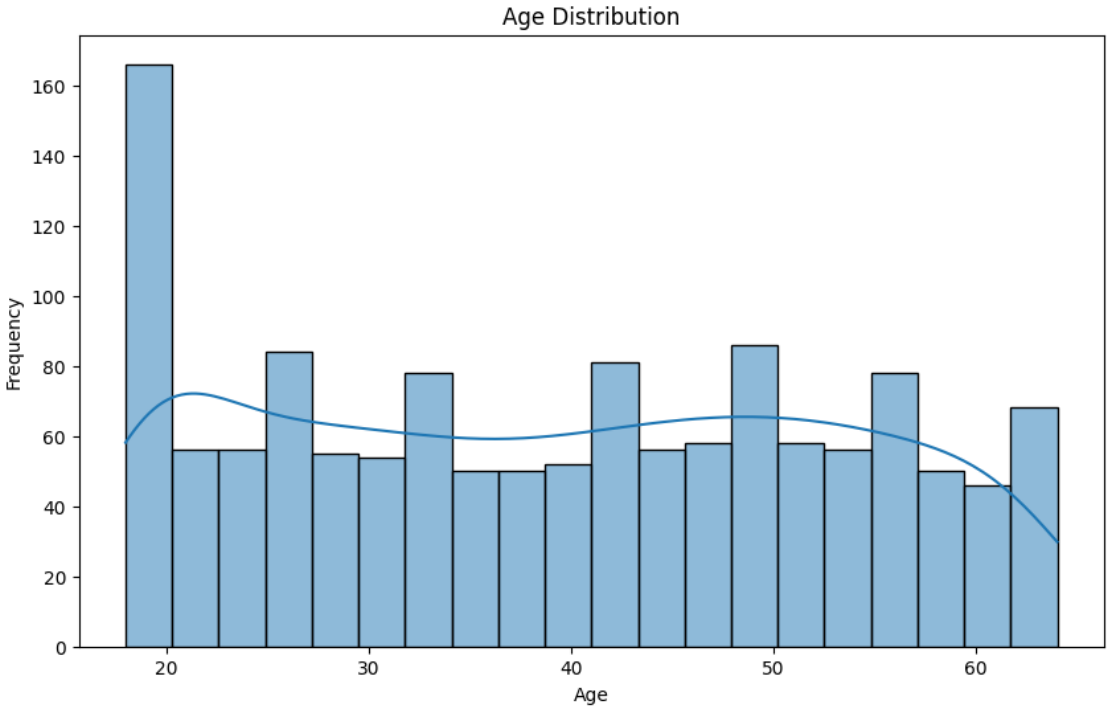
3 2. Distribution Analysis

3.1 2.1. Age Distribution

We group samples by age groups to get raw values and percentages.

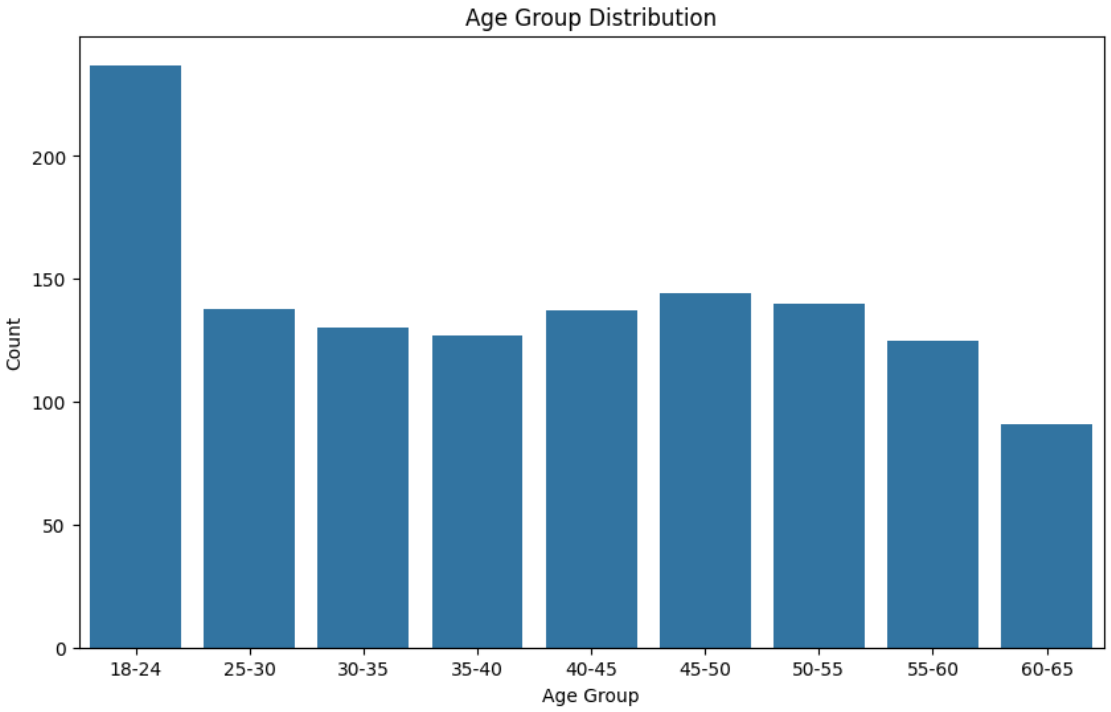
[ ]:
```
# Visualize age distribution
plt.figure(figsize=(10, 6))
sns.histplot(df['age'], bins=20, kde=True)
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()

# Group samples by age groups - necessary for readability
age_groups = ['18-24', '25-30', '30-35', '35-40', '40-45', '45-50', '50-55',␣
↪'55-60', '60-65']
df['age_group'] = pd.cut(df['age'], bins=[18, 25, 30, 35, 40, 45, 50, 55, 60,␣
↪65], labels=age_groups)
```

```python
# Print size of age groups in both raw values and percentages
print("Age groups:")
print(df['age_group'].value_counts().to_string())

print("Age group percentages:")
print(df['age_group'].value_counts(normalize=True).to_string())

# Visualize age group distributions
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='age_group')
plt.title('Age Group Distribution')
plt.xlabel('Age Group')
plt.ylabel('Count')
plt.show()
```




```
Age groups:
age_group
18-24    237
45-50    144
50-55    140
25-30    138
40-45    137
30-35    130
35-40    127
```

```
55-60    125
60-65     91
Age group percentages:
age_group
18-24    0.186761
45-50    0.113475
50-55    0.110323
25-30    0.108747
40-45    0.107959
30-35    0.102443
35-40    0.100079
55-60    0.098503
60-65    0.071710
```

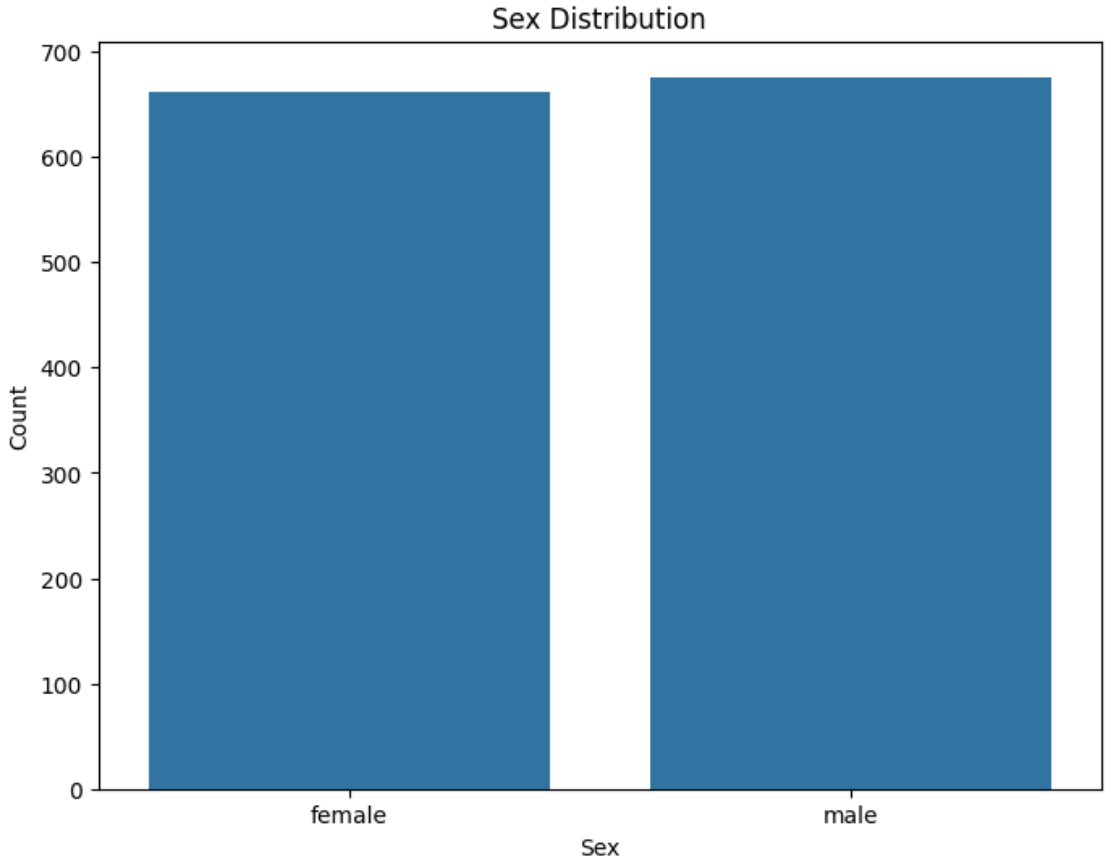


3.2 2.2 Sex Distribution

[ ]:
```python
# Print distribution of the sexes in raw values and percentages
print("Sex counts:")
print(df['sex'].value_counts().to_string())

print("Sex percentages:")
print(df['sex'].value_counts(normalize=True).to_string())

# Visualize sex distribution
```

```
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='sex')
plt.title('Sex Distribution')
plt.xlabel('Sex')
plt.ylabel('Count')
plt.show()
```

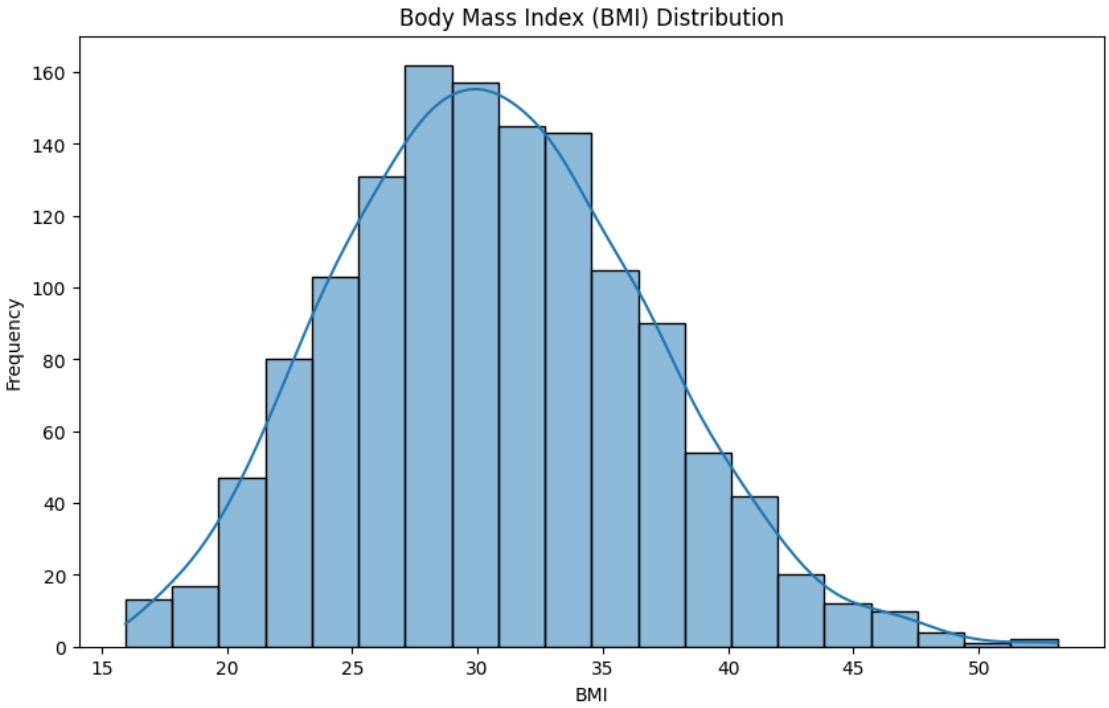
```
Sex counts:
sex
male      676
female    662
Sex percentages:
sex
male      0.505232
female    0.494768
```



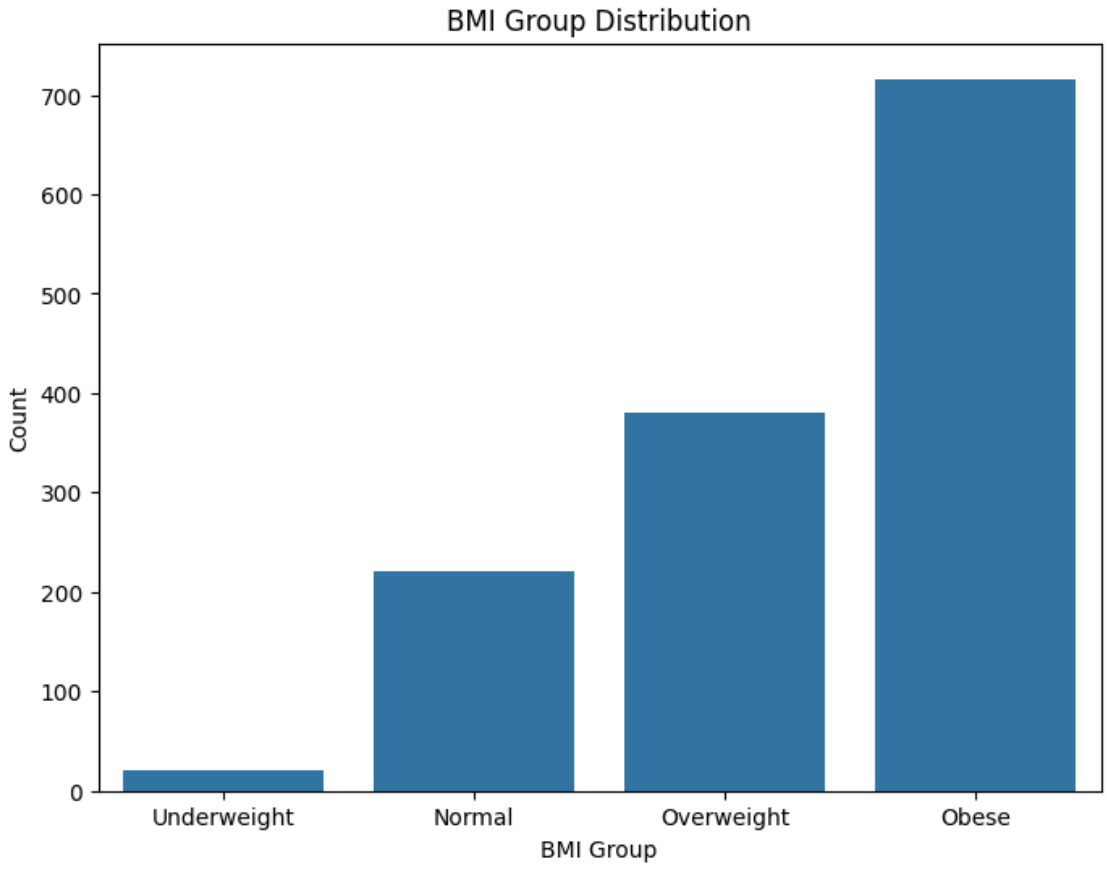
4 2.3. Body Mass Index (BMI) distribution

We group individual BMI into their respective groups to analyze how many of our patients would fall under one group.

[ ]:
```
# Group samples into body mass index (bmi) groups - necessary for consistency
# with bmi scale
bmi_groups = ['Underweight', 'Normal', 'Overweight', 'Obese']
df['bmi_group'] = pd.cut(df['bmi'], bins=[0, 18.5, 24.9, 29.9, np.inf],␣
 ↪labels=bmi_groups)

# Visualize BMI distribution
plt.figure(figsize=(10, 6))
sns.histplot(df['bmi'], bins=20, kde=True)
plt.title('Body Mass Index (BMI) Distribution')
plt.xlabel('BMI')
plt.ylabel('Frequency')
plt.show()

# Print size of BMI groups in raw values and percentages
print("BMI groups:")
print(df['bmi_group'].value_counts().to_string())

print("BMI percentages:")
print(df['bmi_group'].value_counts(normalize=True).to_string())

# Visualize BMI group distribution
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='bmi_group')
plt.title('BMI Group Distribution')
plt.xlabel('BMI Group')
plt.ylabel('Count')
plt.show()
```




```
BMI groups:
bmi_group
Obese          716
Overweight     380
Normal         221
Underweight     21
BMI percentages:
bmi_group
Obese          0.535127
Overweight     0.284006
Normal         0.165172
Underweight    0.015695
```

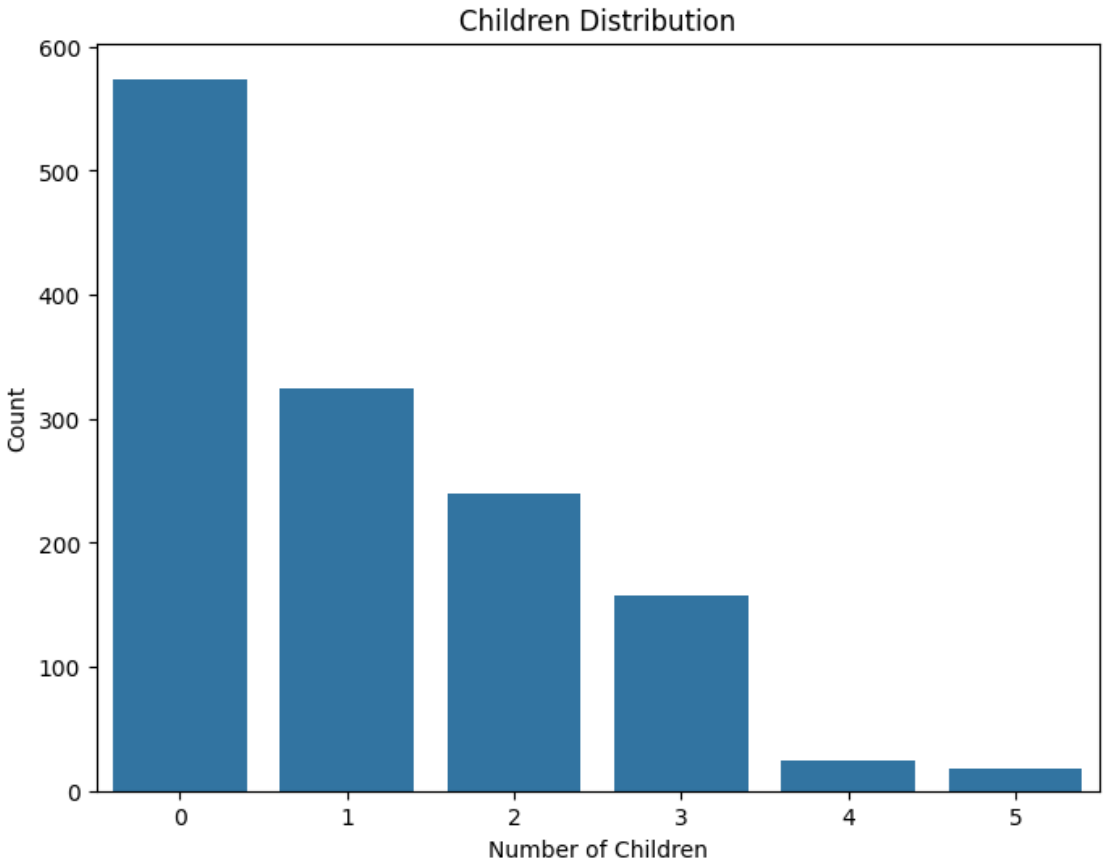


4.1 2.4. Number of Children Distribution

[ ]:
```
# Print number of children in terms of raw values and percentages
print("Number of children values:")
print(df['children'].value_counts().to_string())

print("Number of children percentages:")
print(df['children'].value_counts(normalize=True).to_string())

# Visualize children distribution
plt.figure(figsize=(8,6))
sns.countplot(data=df, x='children')
plt.title('Children Distribution')
plt.xlabel('Number of Children')
plt.ylabel('Count')
plt.show()
```

```
Number of children values:
children
0    574
```

```
1    324
2    240
3    157
4     25
5     18
Number of children percentages:
children
0    0.428999
1    0.242152
2    0.179372
3    0.117339
4    0.018685
5    0.013453
```

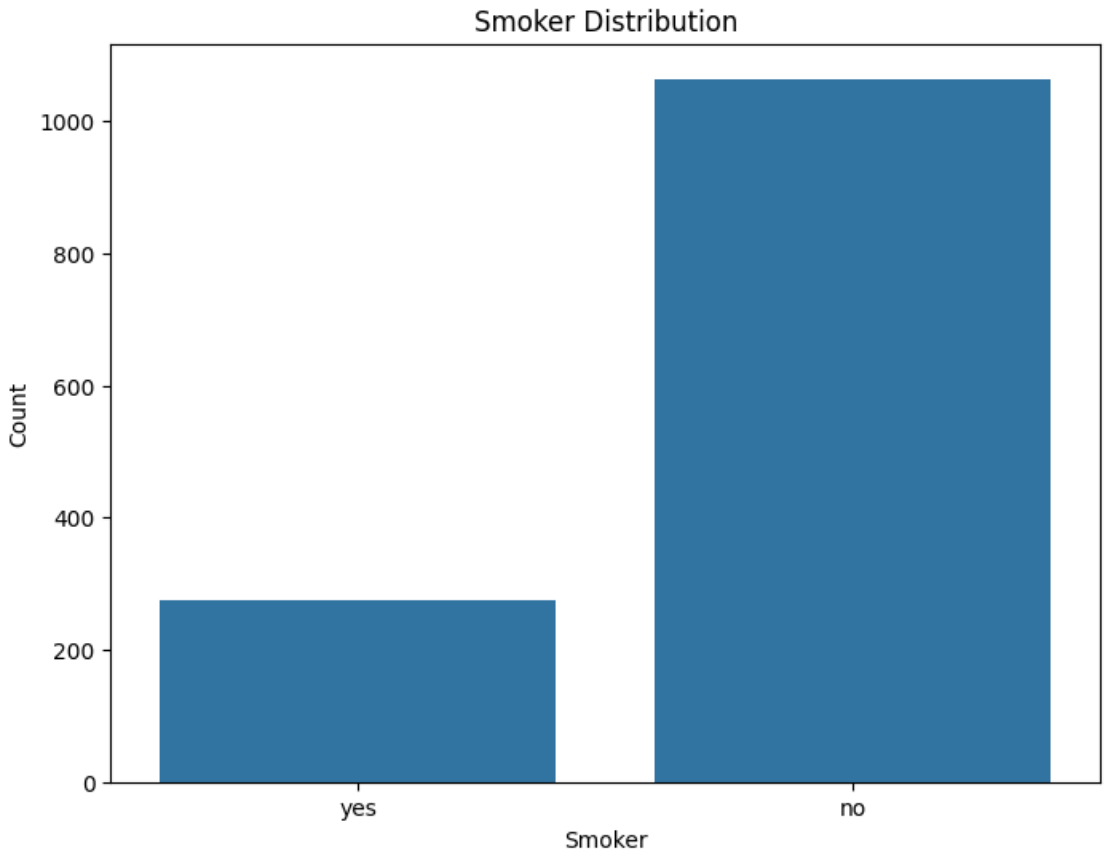


4.2 2.5. Smoker Status Distribution

[ ]:
```
# Print distribution of smoking status in raw values and percentages
print("Smoker values:")
print(df['smoker'].value_counts().to_string())
```

```
print("Smoker percentages:")
print(df['smoker'].value_counts(normalize=True).to_string())

# Visualize smoker distribution
plt.figure(figsize=(8, 6))
sns.countplot(data=df, x='smoker')
plt.title('Smoker Distribution')
plt.xlabel('Smoker')
plt.ylabel('Count')
plt.show()
```

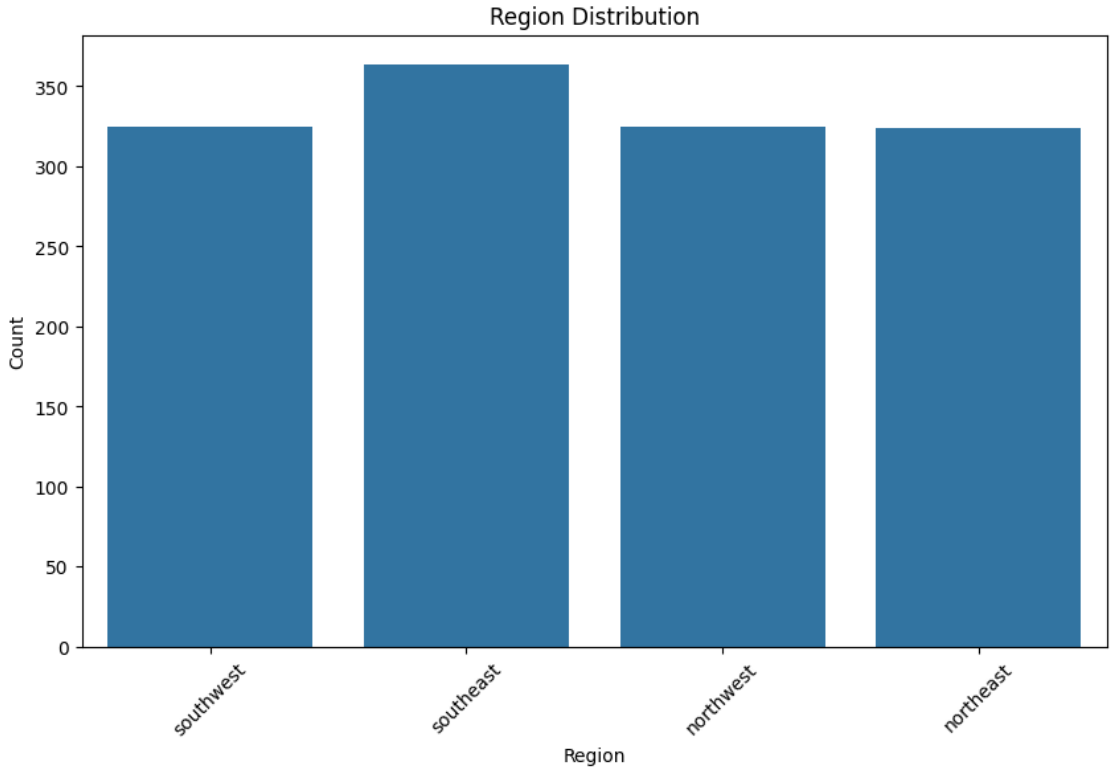
```
Smoker values:
smoker
no     1064
yes     274
Smoker percentages:
smoker
no     0.795217
yes    0.204783
```



4.3 2.6. Region Distribution

[ ]:
```
# Print region distribution in raw values and percentages
print("Region values:")
print(df['region'].value_counts().to_string())

print("Region percentages:")
print(df['region'].value_counts(normalize=True).to_string())

# Visualize region distribution
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='region')
plt.title('Region Distribution')
plt.xlabel('Region')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```

```
Region values:
region
southeast    364
southwest    325
northwest    325
northeast    324
Region percentages:
region
southeast    0.272048
southwest    0.242900
northwest    0.242900
northeast    0.242152
```

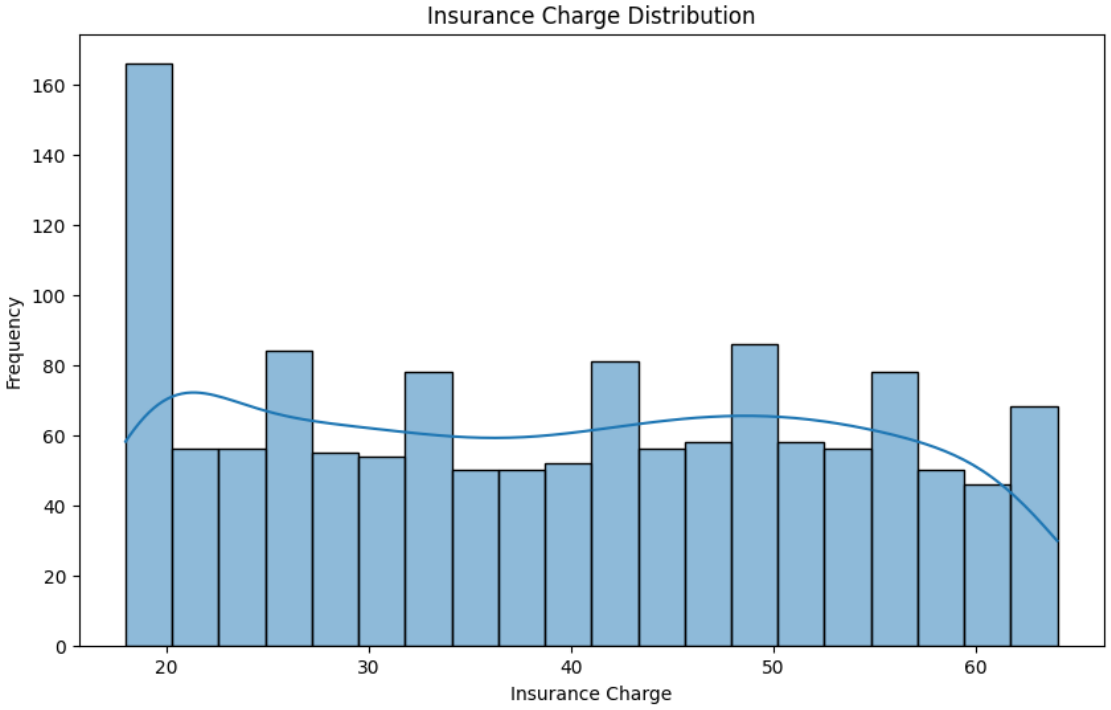


4.4 2.7 Insurance Charges Distribution

We group individual charges into insurance brackets to make analysis easier.

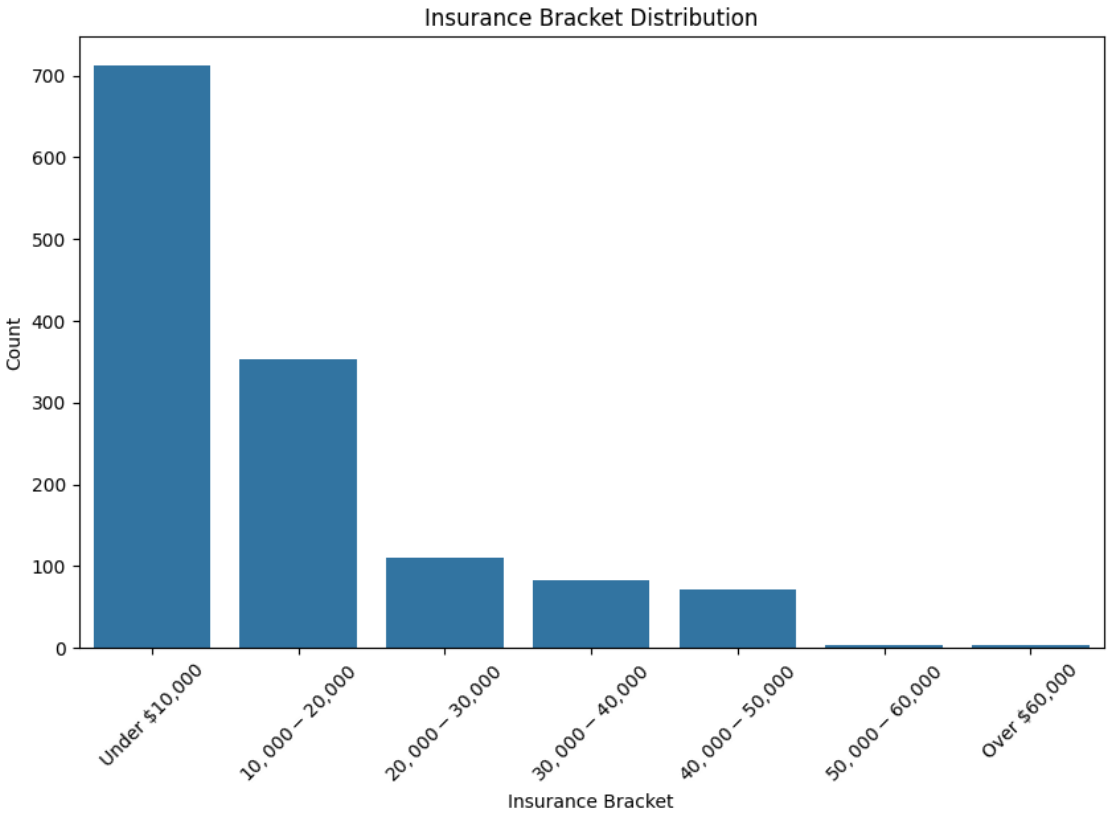
[ ]:
```
# Group dataset into insurance brackets
insurance_brackets = [
    'Under $10,000', '$10,000 - $20,000', '$20,000 - $30,000',
    '$30,000 - $40,000', '$40,000 - $50,000', '$50,000 - $60,000',
    'Over $60,000'
]

df['insurance_bracket'] = pd.cut(
    df['charges'],
    bins=[0, 10000, 20000, 30000, 40000, 50000, 60000, float("inf")],
    labels=insurance_brackets
)

# Visualize insurance charge distribution
plt.figure(figsize=(10, 6))
sns.histplot(data=df, x="age", bins=20, kde=True)

plt.title('Insurance Charge Distribution')
```

```
plt.xlabel('Insurance Charge')
plt.ylabel('Frequency')
plt.show()

# Print insurance bracket size in terms of raw values and percentages
print("Insurance bracket values:")
print(df['insurance_bracket'].value_counts().to_string())

print("Insurance bracket percentages:")
print(df['insurance_bracket'].value_counts(normalize=True).to_string())

# Print insurance bracket distributions
plt.figure(figsize=(10, 6))
sns.countplot(data=df, x='insurance_bracket')
plt.title('Insurance Bracket Distribution')
plt.xlabel('Insurance Bracket')
plt.ylabel('Count')
plt.xticks(rotation=45)
plt.show()
```




```
Insurance bracket values:
insurance_bracket
Under $10,000        712
$10,000 - $20,000    353
```

```
$20,000 - $30,000    111
$30,000 - $40,000     83
$40,000 - $50,000     72
$50,000 - $60,000      4
Over $60,000           3
Insurance bracket percentages:
insurance_bracket
Under $10,000        0.532138
$10,000 - $20,000    0.263827
$20,000 - $30,000    0.082960
$30,000 - $40,000    0.062033
$40,000 - $50,000    0.053812
$50,000 - $60,000    0.002990
Over $60,000         0.002242
```



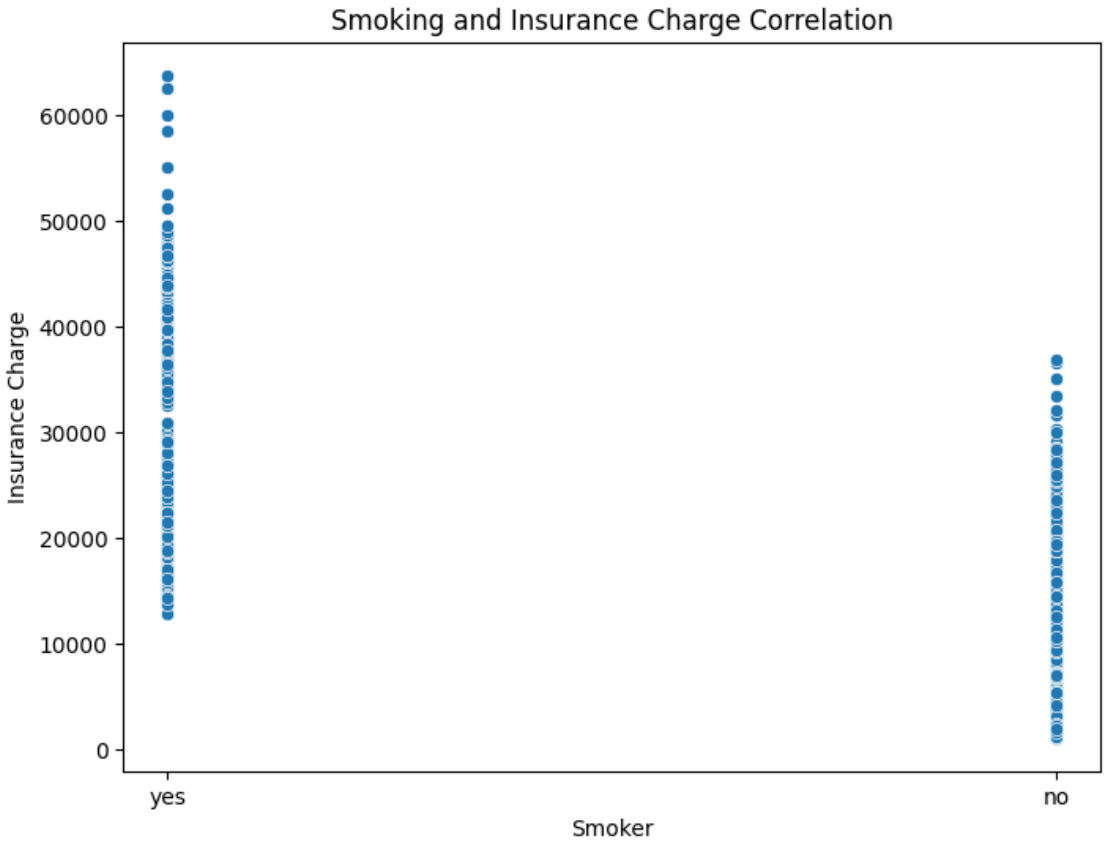
5 3. Correlation Analysis

We calculate the corrrelation between every feature and `charges`.

5.1 3.1. Smoking Correlation

[ ]:
```
correlations = {}

# Print correlation between smoking and insurance charges
print("Correlation between smoking and insurance charges:")
print(df['smoker'].map({'yes': 1, 'no': 0}).corr(df['charges']))
correlations['smoker'] = (df['smoker'].map({'yes': 1, 'no': 0}).
 ↪corr(df['charges']))

# Visualize correlation between smoking and insurance charges
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='smoker', y='charges')
plt.title('Smoking and Insurance Charge Correlation')
plt.xlabel('Smoker')
plt.ylabel('Insurance Charge')
plt.show()
```

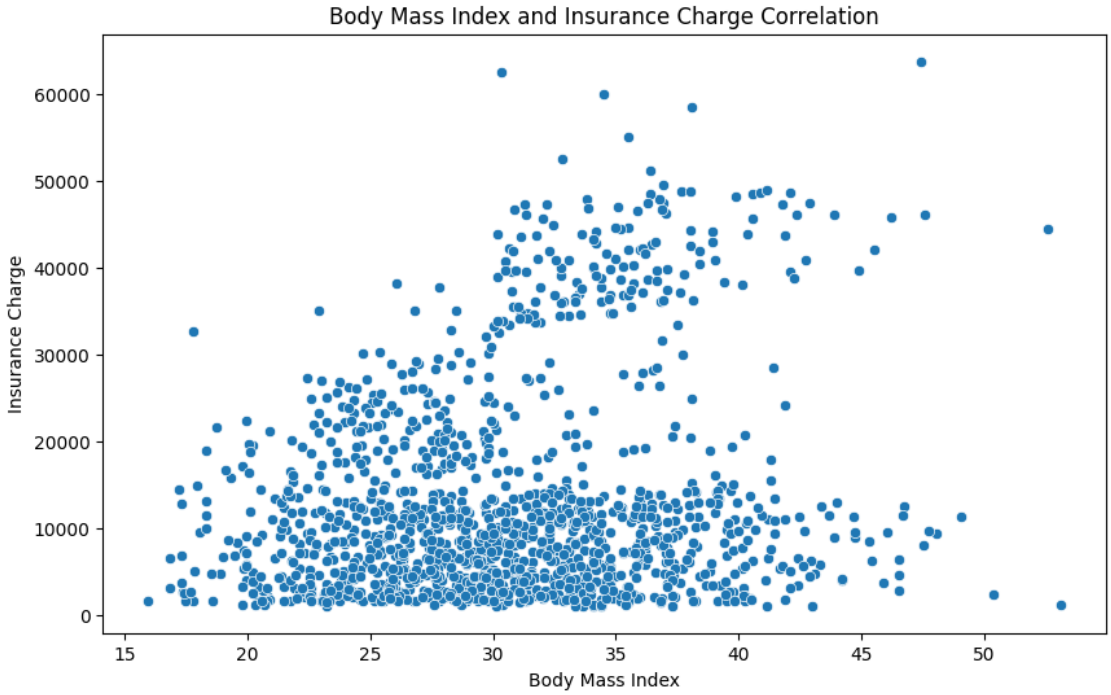
```
Correlation between smoking and insurance charges:
0.7872514304984779
```



5.2 3.2. BMI Correlation

[ ]:
```
# Print correlation between body mass index and insurance charges
print("Correlation between body mass index and insurance charges:")
print(df['bmi'].corr(df['charges']))
correlations['bmi'] = (df['bmi'].corr(df['charges']))

# Visualize correlation between body mass index and insurance charges
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='bmi', y='charges')
plt.title('Body Mass Index and Insurance Charge Correlation')
plt.xlabel('Body Mass Index')
plt.ylabel('Insurance Charge')
plt.show()
```

```
Correlation between body mass index and insurance charges:
0.19834096883362895
```

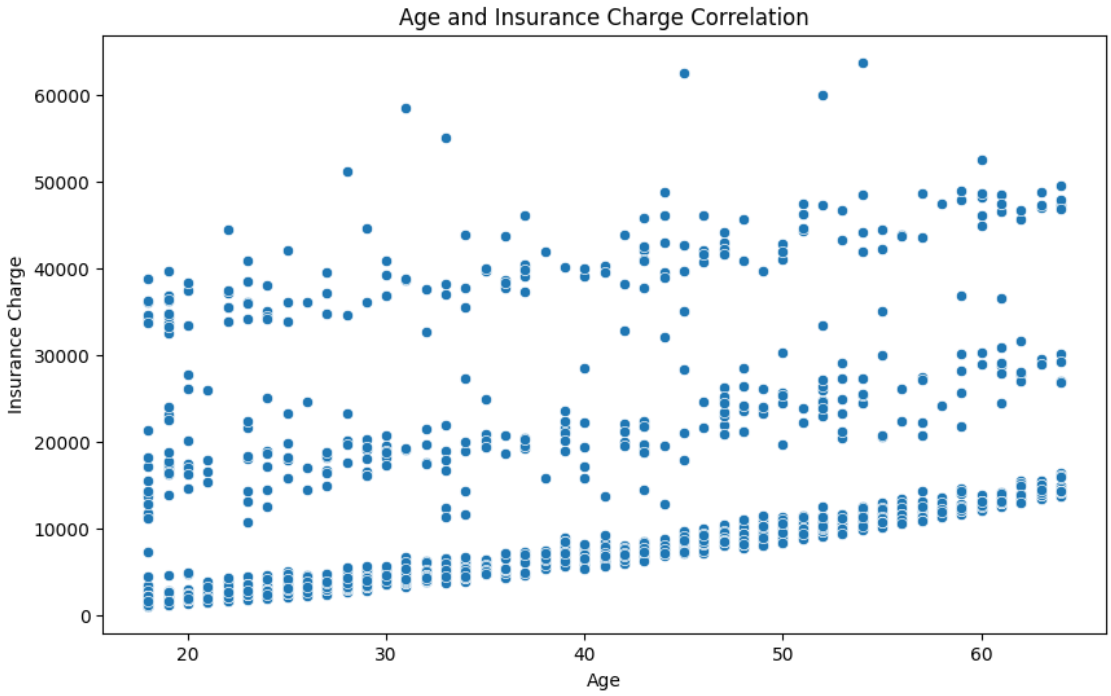


5.3 3.3 Age Correlation

[ ]:
```
# Print correlation between age and insurance charge
print("Correlation between age and insurance charge:")
print(df['age'].corr(df['charges']))
correlations['age'] = (df['age'].corr(df['charges']))
```

```
# Visualize correlation between age and insurance charge
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='age', y='charges')
plt.title('Age and Insurance Charge Correlation')
plt.xlabel('Age')
plt.ylabel('Insurance Charge')
plt.show()
```

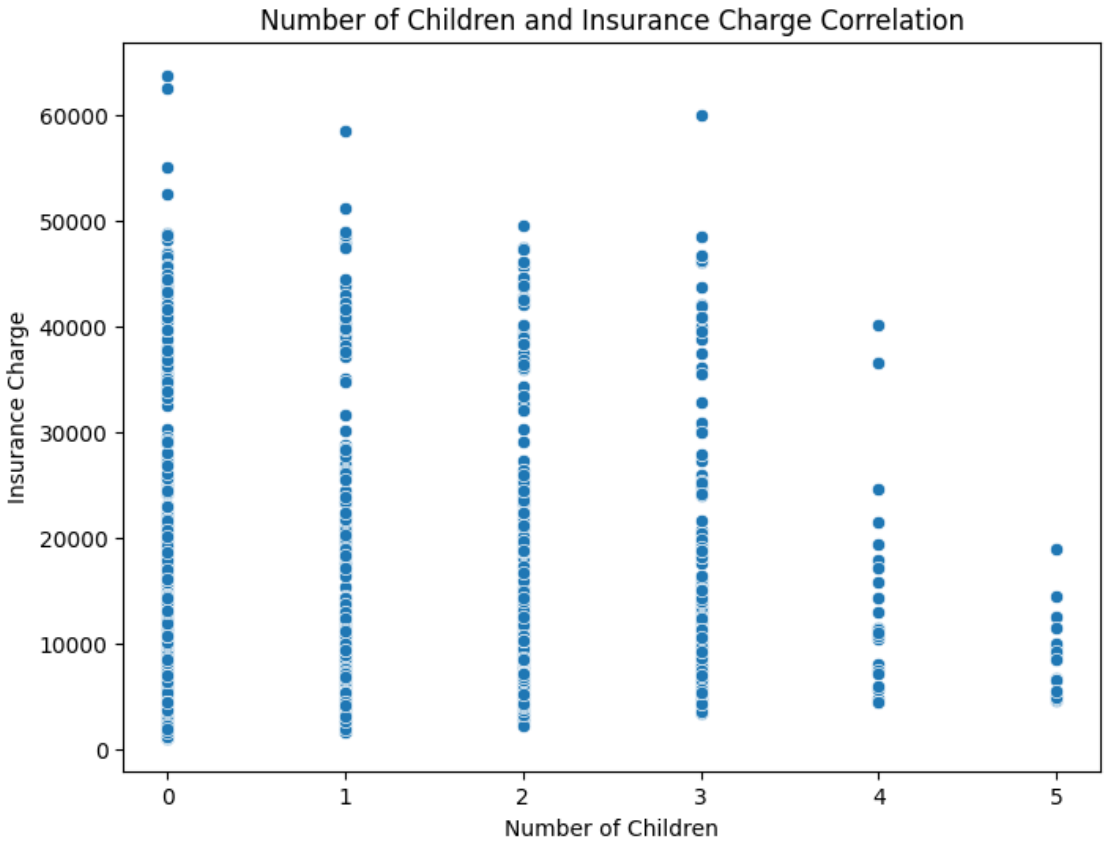
```
Correlation between age and insurance charge:
0.29900819333064754
```



5.4 3.4 Number of Children Correlation

[ ]:
```
# Print correlation between number of children and insurance charges
print("Correlation between number of children and insurance charges:")
print(df['children'].corr(df['charges']))
correlations['children'] = (df['children'].corr(df['charges']))

# Correlation between number of children and insurance charges
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='children', y='charges')
plt.title('Number of Children and Insurance Charge Correlation')
plt.xlabel('Number of Children')
plt.ylabel('Insurance Charge')
plt.show()
```

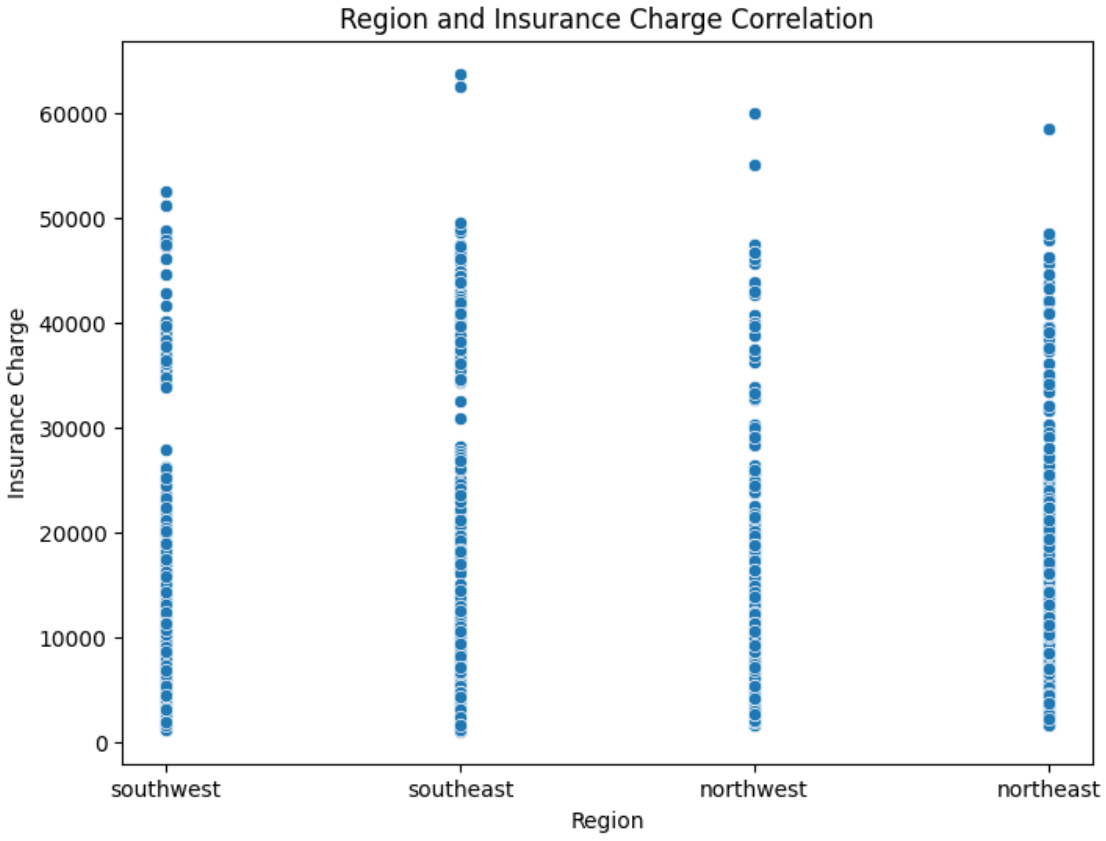
```
Correlation between number of children and insurance charges:
0.0679982268479048
```



5.5 3.5 Region Correlation

[ ]:
```
# Print correlation between region and insurance charges
print("Correlation between region and insurance charges:")
print(df['region'].map({'southwest':0, 'southeast':1, 'northwest':2,␣
 ↪'northeast':3}).corr(df['charges']))
correlations['region'] = (df['region'].map({'southwest':0, 'southeast':1,␣
 ↪'northwest':2, 'northeast':3}).corr(df['charges']))

# Visualize correlation between region and insurance charges
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x='region', y='charges')
plt.title('Region and Insurance Charge Correlation')
plt.xlabel('Region')
plt.ylabel('Insurance Charge')
plt.show()
```

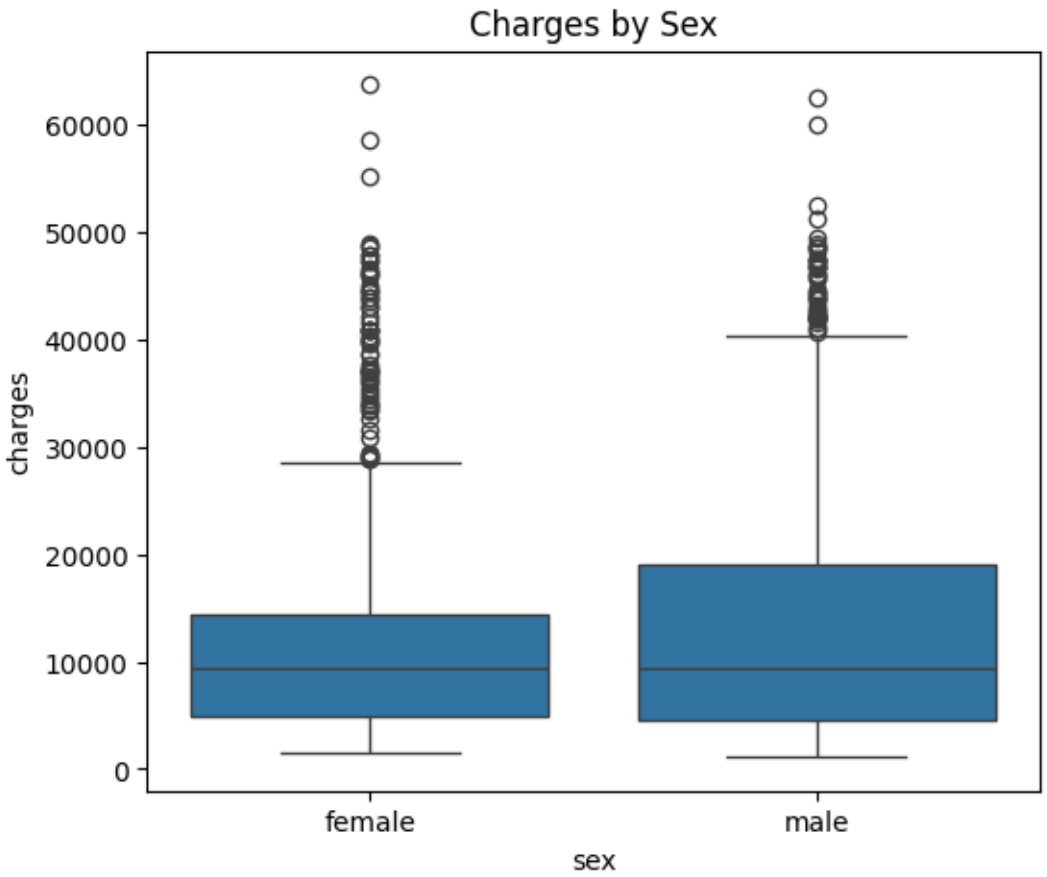
```
Correlation between region and insurance charges:
0.006208234909444512
```



6 3.6. Sex Correlation

[ ]:
```
# Print correlation between sex and insurance charges
print("Correlation between sex and insurance charges:")
print(df['sex'].map({'male':0, 'female':1}).corr(df['charges']))
correlations['sex'] = (df['sex'].map({'male':0, 'female':1}).
 ↪corr(df['charges']))

# Visualize correlation between sex and insurance charges
df.groupby("sex")["charges"].mean()

plt.figure(figsize=(6,5))
sns.boxplot(data=df, x="sex", y="charges")
plt.title("Charges by Sex")
plt.show()
```

```
Correlation between sex and insurance charges:
-0.05729206220202548
```



6.1 3.7. Most Correlated Features

A number close to 1 implies a strong correlation, a number close to 0 implies a weak correlation, and a number close to -1 implies an inverse correlation.

[ ]:
```
# Sort features by most correlated to insurance charges
correlations = {k: v for k, v in sorted(correlations.items(), key=lambda item:␣
 ↪item[1], reverse=True)}
print("Features sorted by most correlated to insurance charges:")
print(list(correlations.keys()))
print("Correlation values:")
print(list(correlations.values()))
```

```
Features sorted by most correlated to insurance charges:
['smoker', 'age', 'bmi', 'children', 'region', 'sex']
Correlation values:
[np.float64(0.7872514304984779), np.float64(0.29900819333064754),
np.float64(0.19834096883362895), np.float64(0.0679982268479048),
np.float64(0.006208234909444512), np.float64(-0.05729206220202548)]
```

7 4. Written Analysis

- No missing values

7.1 Distributions

- Age is skewed towards 18-24 age group, but only by 7% more than the second largest age group (45-50).
- Smoking is split across the sample at around an 80% no to 20% yes split.
- Body Mass Index is skewed towards a range between 28 to 35. About 54% of the sample would be considered obese.
- The number of children is skewed towards 0-1. About half of the sample do not have children covered by health insurance.
- Insurance charges are mostly under $10,000. The distribution is right-skewed, so applying a log-transform to charges may improve regression performance.
- Sex and Region are almost equally distributed between all features.

7.2 Correlations

- Smoking has the highest correlation with insurance charges (0.78).
- Region has the least correlation with insurance charges (0.006).
- Sex shows a very weak relationship with charges (correlation -0.06). This suggests sex alone is not a strong predictor compared with smoking, age, and BMI.

7.3 Issues with modelling

- Age, BMI and Charges have large ranges
- Sex, Region, and Smoker are categorical