Medical Insurance Cost prediction

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn import metrics
```

In [3]:
```
insurance_dataset=pd.read_csv('insurance.csv')
```

In [4]:
```
insurance_dataset
```

Out[4]:

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

1338 rows × 7 columns

In [5]:
```
insurance_dataset.shape
```

Out[5]:
```
(1338, 7)
```

In [6]:
```
insurance_dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1338 entries, 0 to 1337
Data columns (total 7 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   age       1338 non-null   int64  
 1   sex       1338 non-null   object 
 2   bmi       1338 non-null   float64
 3   children  1338 non-null   int64  
 4   smoker    1338 non-null   object 
 5   region    1338 non-null   object 
 6   charges   1338 non-null   float64
dtypes: float64(2), int64(2), object(3)
memory usage: 73.3+ KB
```

3 categorical columns and 4 continuous columns

categorical columns

- Sex
- Smoker
- region

In [7]:
```
insurance_dataset.isnull().sum()
```

Out[7]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

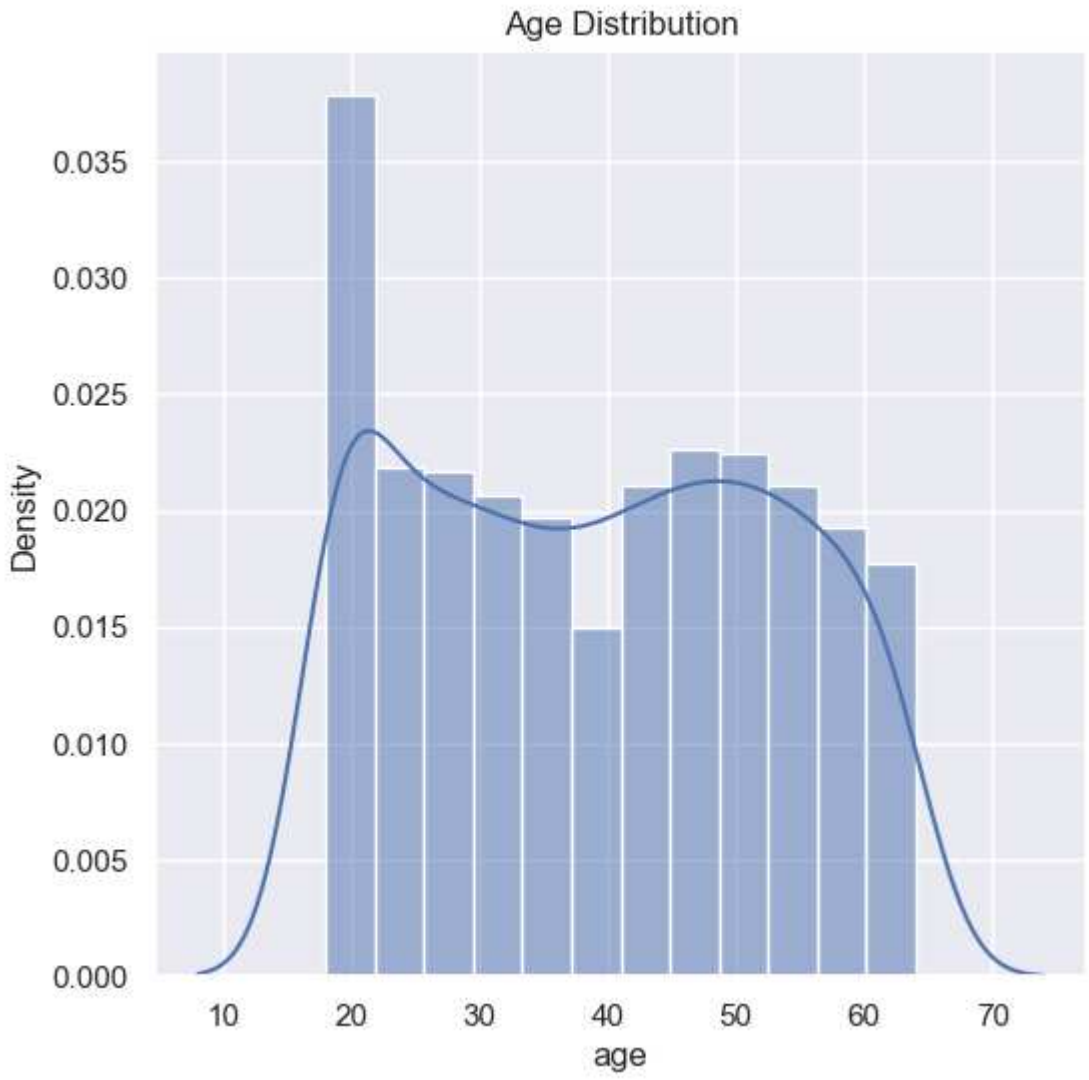
Data Analysis

In [8]:
```
insurance_dataset.describe()
```

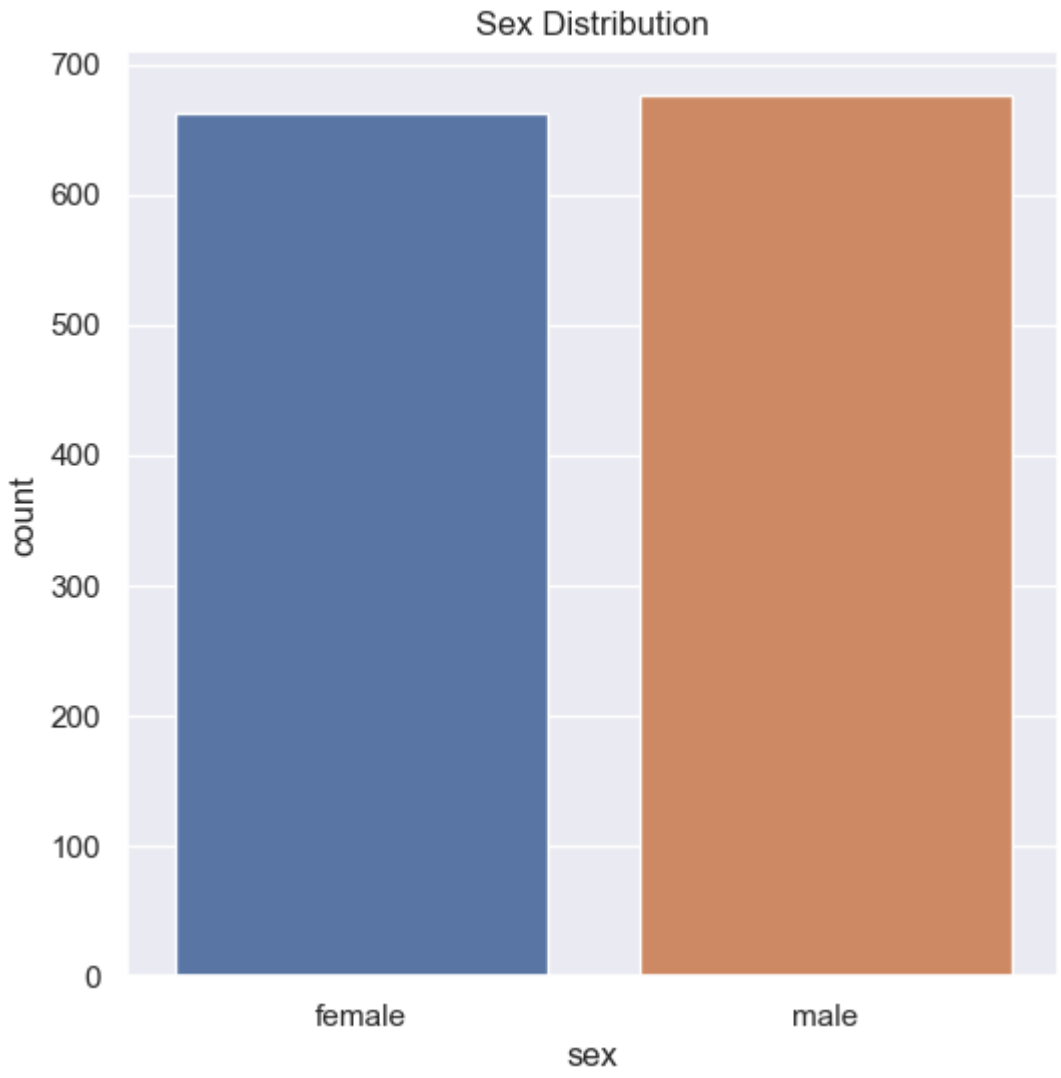
Out[8]:

	age	bmi	children	charges
count	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	30.663397	1.094918	13270.422265
std	14.049960	6.098187	1.205493	12110.011237
min	18.000000	15.960000	0.000000	1121.873900
25%	27.000000	26.296250	0.000000	4740.287150
50%	39.000000	30.400000	1.000000	9382.033000
75%	51.000000	34.693750	2.000000	16639.912515
max	64.000000	53.130000	5.000000	63770.428010

In [9]:
```
#distribution of age value
sns.set()
plt.figure(figsize=(6,6))
sns.histplot(insurance_dataset['age'],kde=True,stat="density", kde_kws=dict(cut=3))
plt.title('Age Distribution')
plt.show()
```



In [10]:
```
# distribution of gender column
plt.figure(figsize=(6,6))
sns.countplot(x='sex',data=insurance_dataset)
plt.title('Sex Distribution')
plt.show()
```

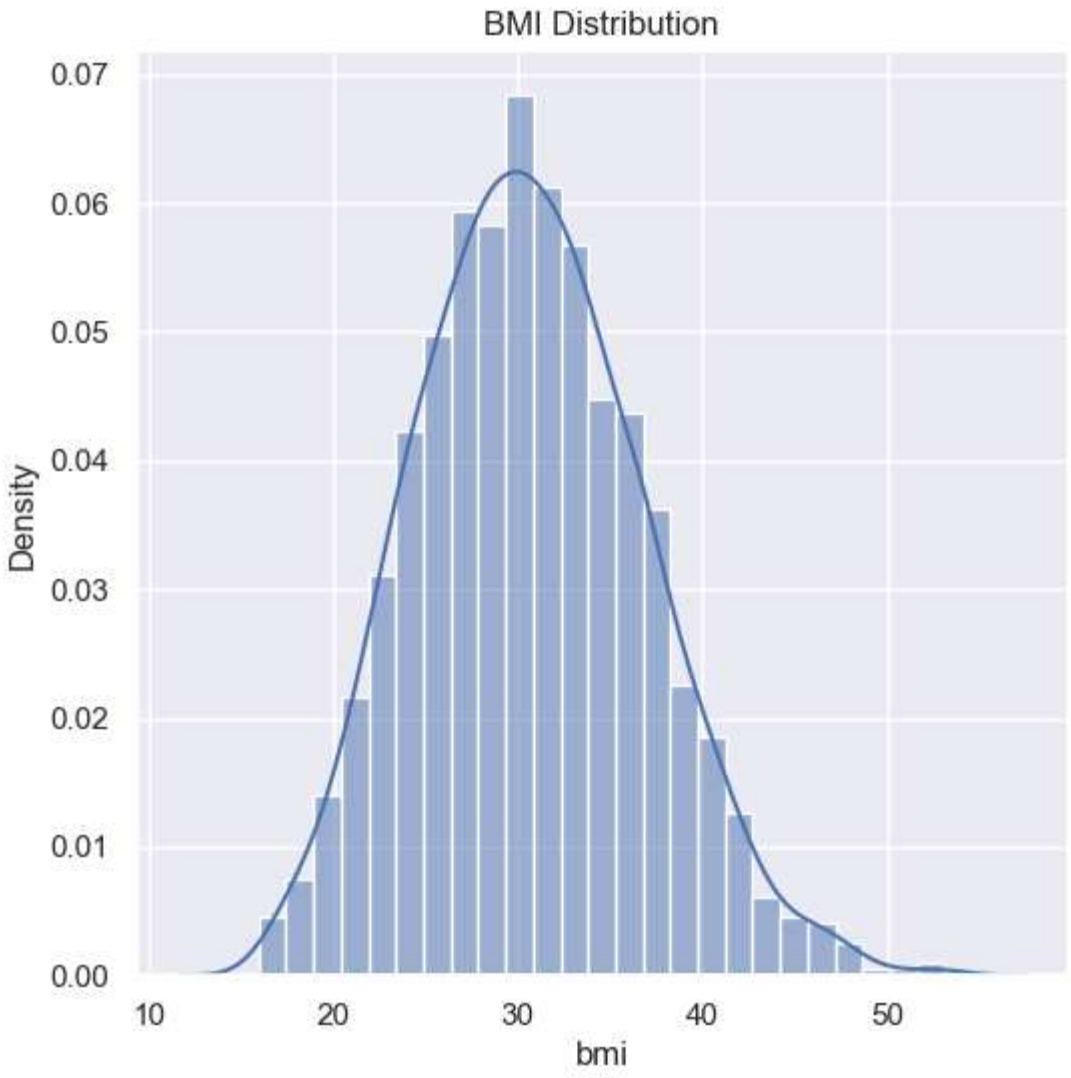


In [11]:
```
insurance_dataset['sex'].value_counts()
```

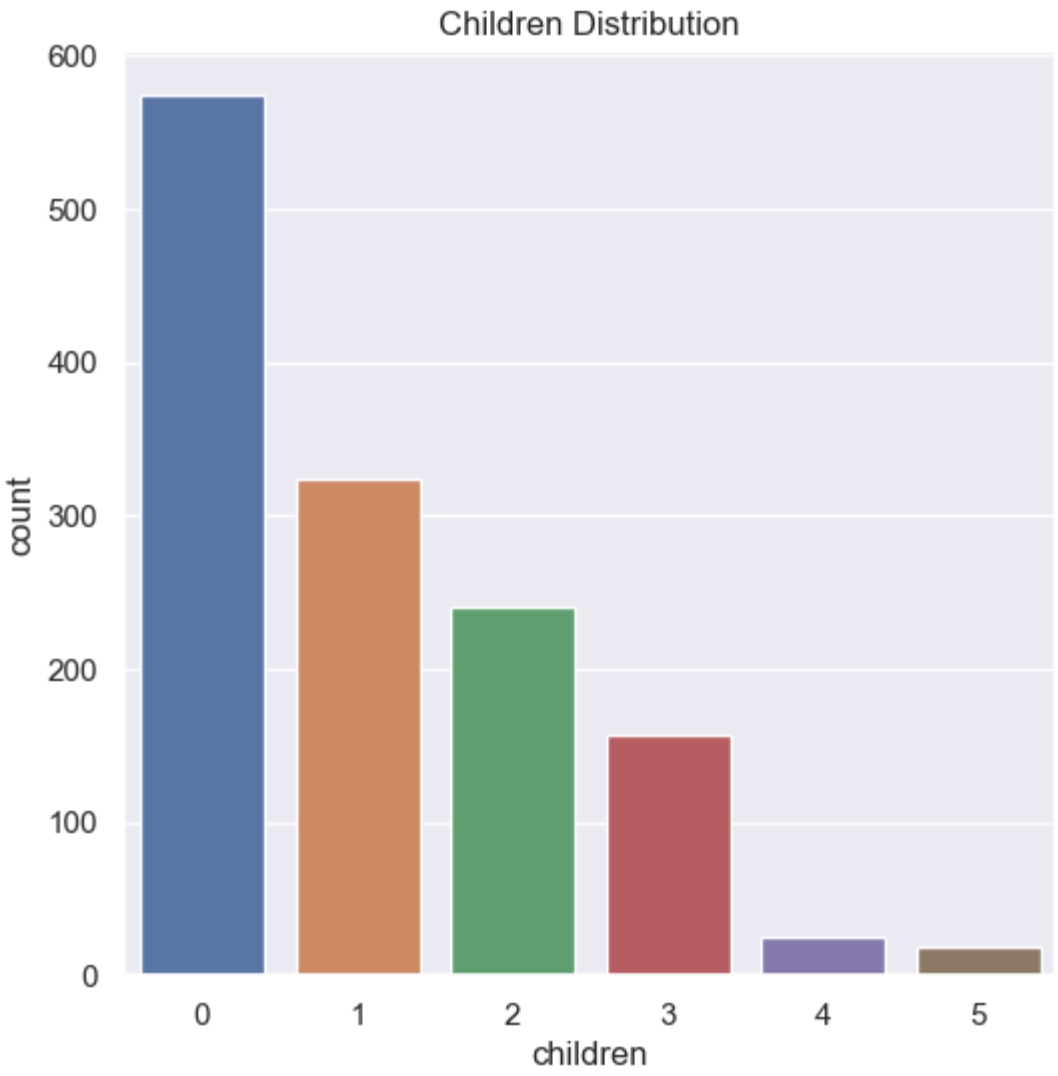
Out[11]:
```
male      676
female    662
Name: sex, dtype: int64
```

In [12]:
```
#BMI Distribution

sns.set()
plt.figure(figsize=(6,6))
sns.histplot(insurance_dataset['bmi'],kde=True,stat="density", kde_kws=dict(cut=3))
plt.title('BMI Distribution')
plt.show()
```



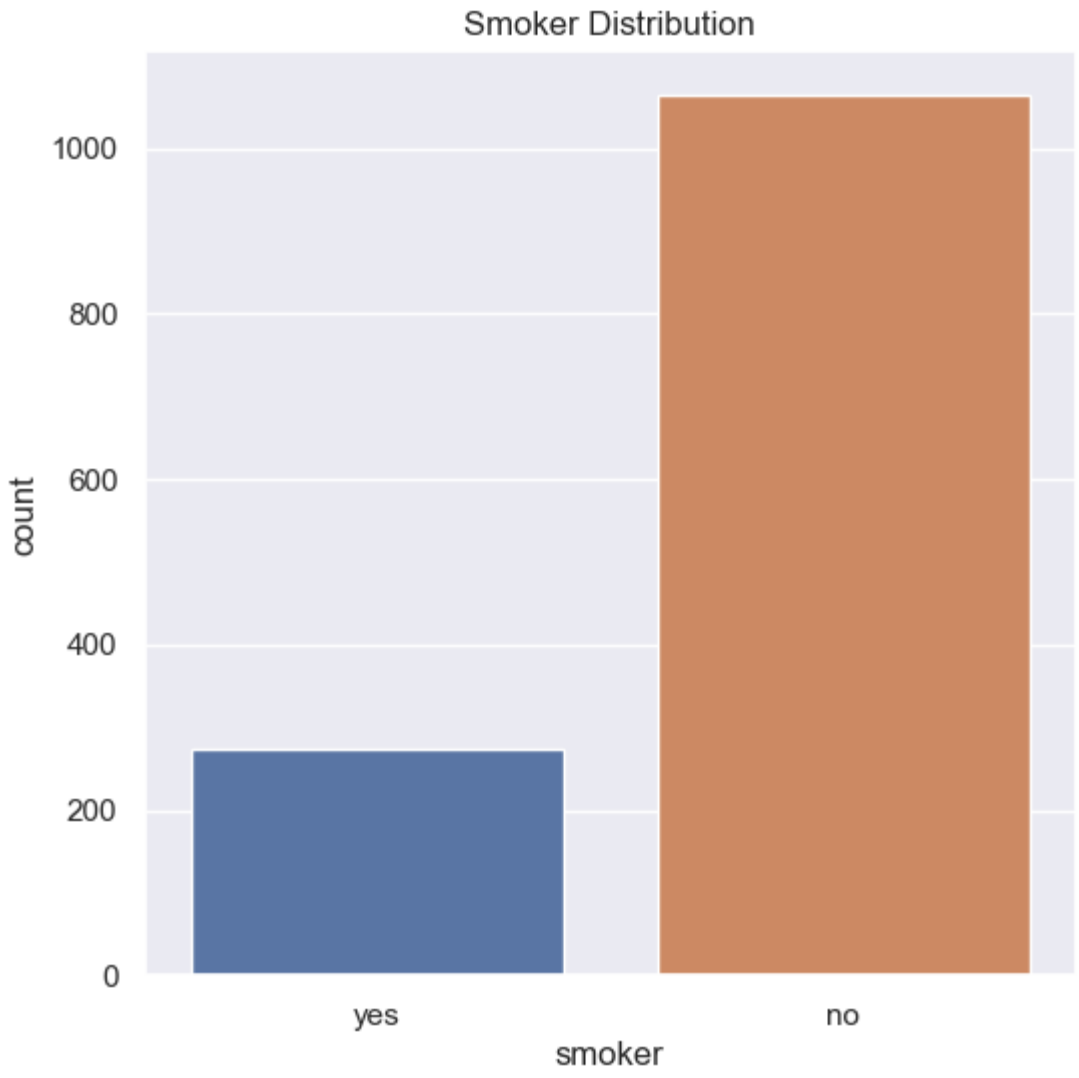
In [13]:
```
#distribution of children column
plt.figure(figsize=(6,6))
sns.countplot(x='children',data=insurance_dataset)
plt.title('Children Distribution')
plt.show()
```



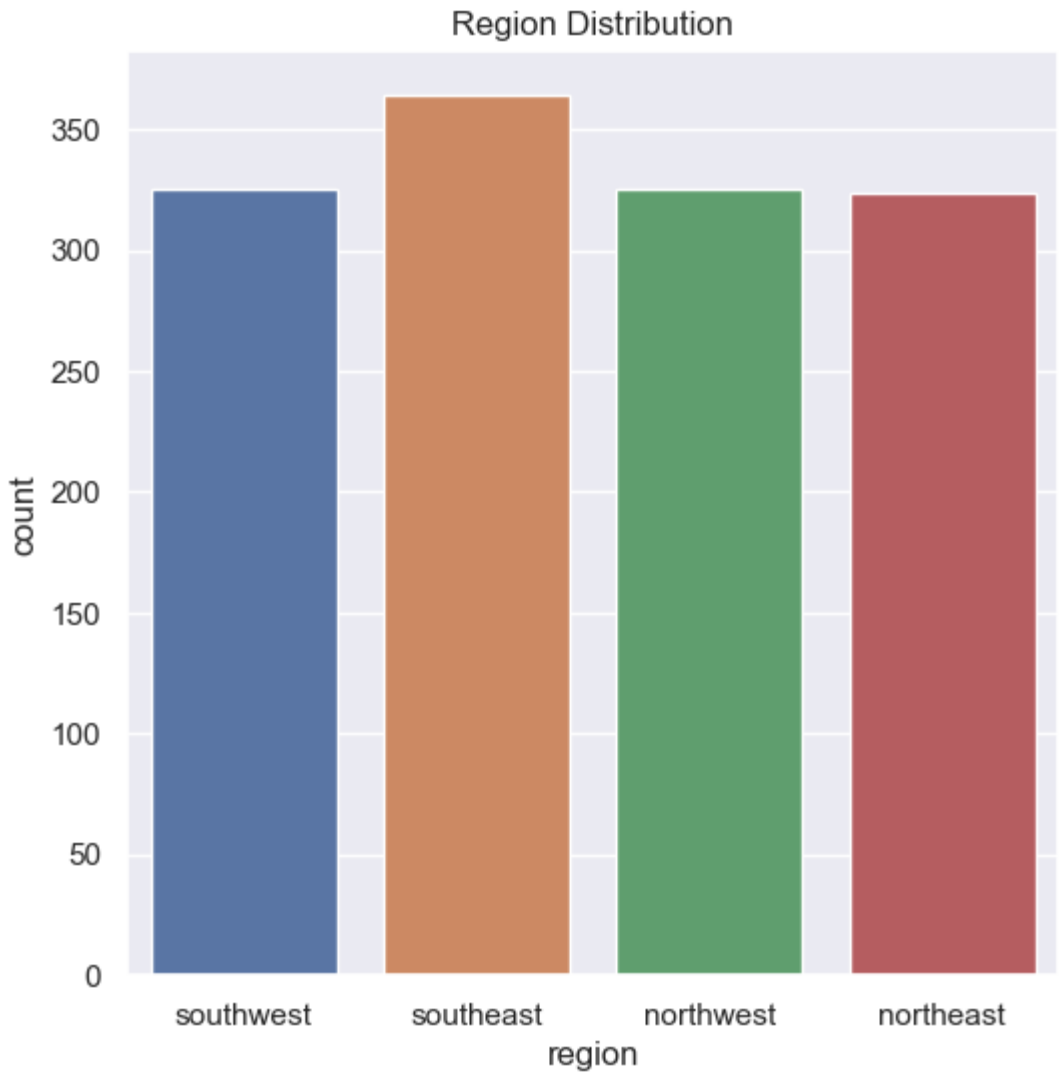
In [14]:
```
insurance_dataset['children'].value_counts()
```

Out[14]:
```
0    574
1    324
2    240
3    157
4     25
5     18
Name: children, dtype: int64
```

In [15]:
```
#distribution of smoker column
plt.figure(figsize=(6,6))
sns.countplot(x='smoker',data=insurance_dataset)
plt.title('Smoker Distribution')
plt.show()
```



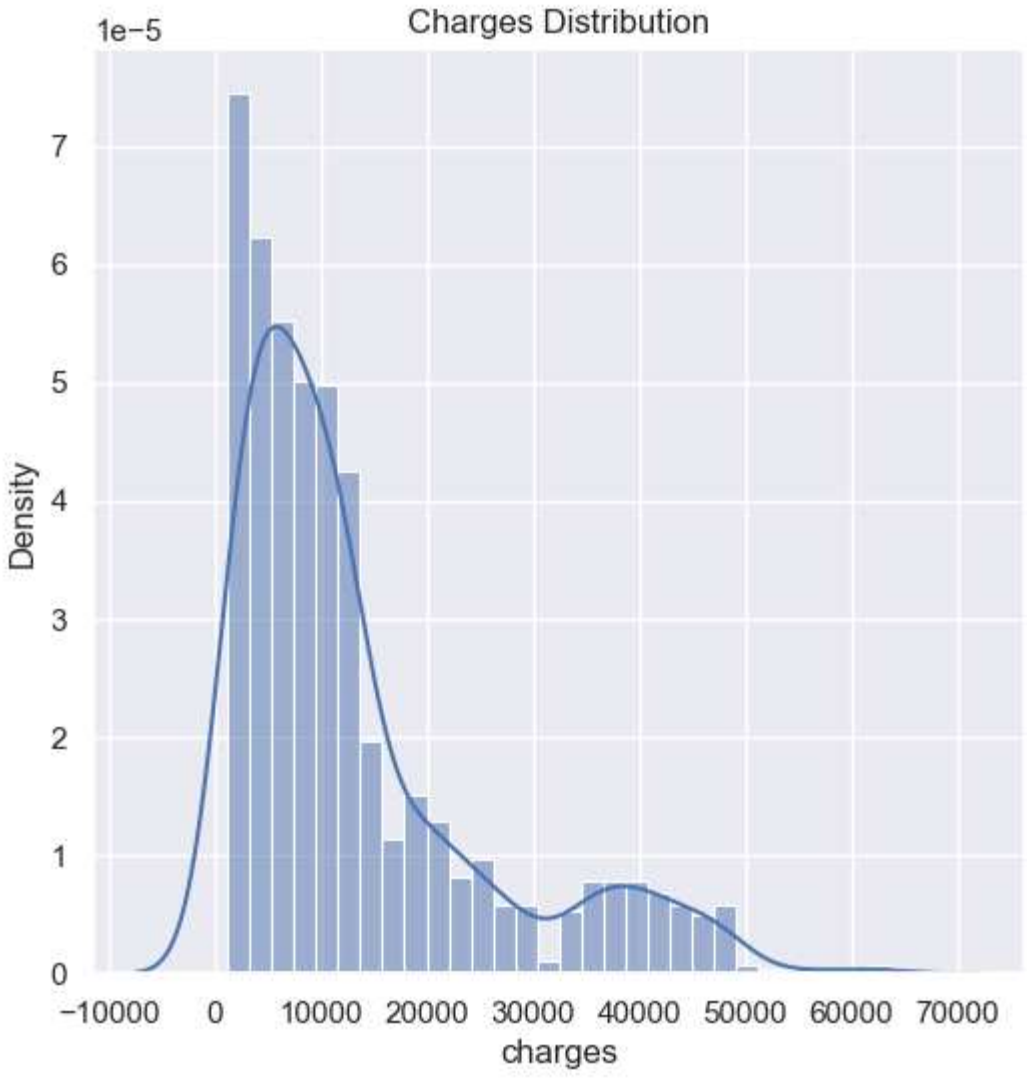
In [16]:
```
#distribution of Region column
plt.figure(figsize=(6,6))
sns.countplot(x='region',data=insurance_dataset)
plt.title('Region Distribution')
plt.show()
```



In [17]:
```
insurance_dataset['region'].value_counts()
```

Out[17]:
```
southeast    364
southwest    325
northwest    325
northeast    324
Name: region, dtype: int64
```

In [18]:
```
#Distribution of Charges column
sns.set()
plt.figure(figsize=(6,6))
sns.histplot(insurance_dataset['charges'],kde=True,stat="density", kde_kws=dict(cut
plt.title('Charges Distribution')
plt.show()
```



Data pre-processiong

In [19]:
```
insurance_dataset.head()
```

Out[19]:

	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

In [20]:
```
#Encoding of the categorical variables
#Encoding sex column
insurance_dataset.replace({'sex':{'male':0,'female':1}}, inplace=True)
#encoding smoker column
insurance_dataset.replace({'smoker':{'yes':1,'no':0}}, inplace=True)
#Encoding region column
insurance_dataset.replace({'region':{'southeast':0,'southwest':1, 'northeast':2, 'n
```

In [21]:
```
insurance_dataset
```

Out[21]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	1	16884.92400
1	18	0	33.770	1	0	0	1725.55230
2	28	0	33.000	3	0	0	4449.46200
3	33	0	22.705	0	0	3	21984.47061
4	32	0	28.880	0	0	3	3866.85520
...	...	...	...	...	...	...	...
1333	50	0	30.970	3	0	3	10600.54830
1334	18	1	31.920	0	0	2	2205.98080
1335	18	1	36.850	0	0	0	1629.83350
1336	21	1	25.800	0	0	1	2007.94500
1337	61	1	29.070	0	1	3	29141.36030

1338 rows × 7 columns

Splitting the features and target variable

In [22]:
```
x=insurance_dataset.drop(columns='charges',axis=1)
y=insurance_dataset['charges']
```

In [23]:
```
print(x)
```

```
      age  sex     bmi  children  smoker  region
0      19    1  27.900         0       1       1
1      18    0  33.770         1       0       0
2      28    0  33.000         3       0       0
3      33    0  22.705         0       0       3
4      32    0  28.880         0       0       3
...   ...  ...     ...       ...     ...     ...
1333   50    0  30.970         3       0       3
1334   18    1  31.920         0       0       2
1335   18    1  36.850         0       0       0
1336   21    1  25.800         0       0       1
1337   61    1  29.070         0       1       3

[1338 rows x 6 columns]
```

In [24]:
```
print(y)
```

```
0       16884.92400
1        1725.55230
2        4449.46200
3       21984.47061
4        3866.85520
            ...
1333    10600.54830
1334     2205.98080
1335     1629.83350
1336     2007.94500
1337    29141.36030
Name: charges, Length: 1338, dtype: float64
```

Splitting the data into train and test

In [25]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=2)
```

In [26]:
```
print(x.shape,x_train.shape,x_test.shape)
```

```
(1338, 6) (1070, 6) (268, 6)
```

Model training

In [27]:
```
#Loading the linear regression model
regressor = LinearRegression()
```

In [28]:
```
regressor.fit(x_train,y_train)
```

Out[28]:
```
▾ LinearRegression

LinearRegression()
```

In [29]:
```
#model evaluation
#prediction on trainig data
training_data_prediction=regressor.predict(x_train)
```

In [30]:
```
#finding r squared value
r2_train = metrics.r2_score(y_train,training_data_prediction)
print('R squared value:',r2_train)
```

```
R squared value: 0.751505643411174
```

In [31]:
```
test_data_prediction=regressor.predict(x_test)
```

In [32]:
```
r2_test = metrics.r2_score(y_test,test_data_prediction)
print('R squared value:',r2_test)
```

```
R squared value: 0.7447273869684076
```

Building a prediction system

In [33]:
```
input_data = (31,1,25.74,0,0,0)
#Changing tuple(input_data) to numpy array
input_array=np.asarray(input_data)
#reshape the array
input_reshape = input_array.reshape(1,-1)
```

In [34]:
```
prediction = regressor.predict(input_reshape)
```

```
C:\Users\shant\anaconda3\lib\site-packages\sklearn\base.py:420: UserWarning: X does
not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

In [35]:
```
print('The insurance cost is USD:',prediction[0])
```

```
The insurance cost is: $3760.080576496046
```