In [1]:
```
import numpy as np
import pandas as pd
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 [2]:
```
Medical_Insurance=pd.read_csv('E:/Machine learning dataset/Medical_Insurance.csv')
```

In [3]:
```
Medical_Insurance.head()
```

Out[3]:

	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 [4]:
```
Medical_Insurance.tail()
```

Out[4]:

	age	sex	bmi	children	smoker	region	charges
1333	50	male	30.97	3	no	northwest	10600.5483
1334	18	female	31.92	0	no	northeast	2205.9808
1335	18	female	36.85	0	no	southeast	1629.8335
1336	21	female	25.80	0	no	southwest	2007.9450
1337	61	female	29.07	0	yes	northwest	29141.3603

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

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

In [6]:
```
Medical_Insurance.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
```

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

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

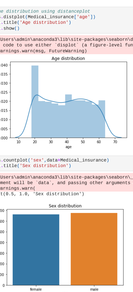
In [8]:
```
Medical_Insurance.describe()
```

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]:
```
#Age distribution using distanceplot
sns.distplot(Medical_Insurance['age'])
plt.title('Age distribution')
plt.show()
```

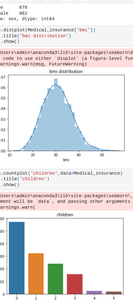
```
E:\Users\admin\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```



In [10]:
```
sns.countplot('sex',data=Medical_Insurance)
plt.title('Sex distribution')
```

```
E:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

Out[10]:
```
Text(0.5, 1.0, 'Sex distribution')
```



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

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

In [12]:
```
sns.distplot(Medical_Insurance['bmi'])
plt.title('bmi distribution')
plt.show()
```

```
E:\Users\admin\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

In [13]:
```
sns.countplot('children',data=Medical_Insurance)
plt.title('children')
plt.show()
```

```
E:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

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

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

In [15]:
```
sns.countplot('smoker',data=Medical_Insurance)
plt.title('No. of smokers')
plt.show()
```

```
E:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

In [16]:
```
Medical_Insurance['smoker'].value_counts()
```

Out[16]:
```
no     1064
yes     274
Name: smoker, dtype: int64
```

In [17]:
```
sns.countplot('region',data=Medical_Insurance)
plt.title('Region')
plt.show()
```

```
E:\Users\admin\anaconda3\lib\site-packages\seaborn\_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
  warnings.warn(
```

In [18]:
```
Medical_Insurance['region'].value_counts()
```

Out[18]:
```
southeast    364
southwest    325
northwest    325
northeast    324
Name: region, dtype: int64
```

In [19]:
```
sns.distplot(Medical_Insurance['charges'])
```

```
E:\Users\admin\anaconda3\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated function and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function with similar flexibility) or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

Out[19]:
```
<AxesSubplot:xlabel='charges', ylabel='Density'>
```

Data Preprocessing

Encoding the categorical features

In [20]:
```
Medical_Insurance['sex'] = Medical_Insurance['sex'].map({'male':1,'female':0})
```

In [21]:
```
Medical_Insurance.head()
```

Out[21]:

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	yes	southwest	16884.92400
1	18	1	33.770	1	no	southeast	1725.55230
2	28	1	33.000	3	no	southeast	4449.46200
3	33	1	22.705	0	no	northwest	21984.47061
4	32	1	28.880	0	no	northwest	3866.85520

In [22]:
```
Medical_Insurance.replace({'smoker':{'yes':0,'no':1}} ,inplace=True)

Medical_Insurance.replace({'region':{'southeast':0,'southwest':1,'northeast':2,'northwest':3}} ,inplace=True)
```

In [23]:
```
Medical_Insurance.head()
```

Out[23]:

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	0	1	16884.92400
1	18	1	33.770	1	1	0	1725.55230
2	28	1	33.000	3	1	0	4449.46200
3	33	1	22.705	0	1	3	21984.47061
4	32	1	28.880	0	1	3	3866.85520

In [24]:
```
X = Medical_Insurance.drop(columns='charges',axis=1)
Y = Medical_Insurance['charges']
```

In [25]:
```
print(X)
```

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

[1338 rows x 6 columns]
```

In [26]:
```
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
```

Split data into training and testing

In [27]:
```
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=2)
```

In [28]:
```
print(X_shape,X_train.shape,X_test.shape)
```

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

Model Training

In [29]:
```
regressor = LinearRegression()
```

In [30]:
```
regressor
```

Out[30]:
```
LinearRegression()
```

In [31]:
```
regressor.fit(X_train,Y_train)
```

Out[31]:
```
LinearRegression()
```

Model Evaluation

In [32]:
```
training_data_prediction = regressor.predict(X_train)
test_data_prediction = regressor.predict(X_test)

# R squared value
r2_train = metrics.r2_score(Y_train, training_data_prediction)
print('R squared value for train data: ',r2_train)

R squared value for train data: 0.7515956434111174

r2_test = metrics.r2_score(Y_test, test_data_prediction)
print('R squared value for test data: ',r2_test)

R squared value for test data: 0.7447273869684077
```

Building a predictive system

In [ ]:
```
31      female  25.74    0         no      southeast
```

In [36]:
```
input_data=(31,0,25.74,0,1,0)

# Change the input data to a numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = regressor.predict(input_data_reshaped)
print(prediction)

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

```
[3760.08057655]
The insurance cost is USD  3760.0805764960646
E:\Users\admin\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

In [38]:
```
input_data=(23,1,20.7,0,1,3)

# Change the input data to a numpy array
input_data_as_numpy_array = np.asarray(input_data)

# reshape the numpy array as we are predicting for only on instance
input_data_reshaped = input_data_as_numpy_array.reshape(1,-1)

prediction = regressor.predict(input_data_reshaped)
print(prediction)

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

```
[692.03260353]
The insurance cost is USD  692.0326035353537
E:\Users\admin\anaconda3\lib\site-packages\sklearn\base.py:450: UserWarning: X does not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

In [ ]:
```

```