```python
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
import warnings
warnings.filterwarnings('ignore')
```

```python
df=pd.read_csv("health insurance.csv")
df
```

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

1338 rows × 7 columns

```python
df.shape
```

```
(1338, 7)
```

We can observe that the dataset contains 1338 rows and 7 columns

```python
df.columns.tolist()
```

```
['age', 'sex', 'bmi', 'children', 'smoker', 'region', 'charges']
```

```python
df.dtypes
```

```
age           int64
sex          object
bmi         float64
children      int64
smoker       object
region       object
charges     float64
dtype: object
```

```python
df.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
```

As we can observe there is no missing data also there are three datatype (float, int and object)

```python
df.nunique().to_frame("No.of unique frame")
```

	No.of unique frame
age	47
sex	2
bmi	548
children	6
smoker	2
region	4
charges	1337

We can check for unique value for each column by nunique function

```python
df.describe()
```

	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

This gives the statistical information of the numerical columns. The summary of the dataset looks perfectly fine as there is no negative value present in the dataset.

From the above description we can observe the following:

1. It is showing the result for only numerical columns not for categorical columns

2. The count of all the columns are same which means there are no missing value present in the dataset

3. The mean value is greater than the median value in all the columns

4. By summarizing the data we can observe there is a huge difference 75% and maximum in charges and bmi and little difference in age and children hence outliers are present in the data

5. We also got the standard deviation, min, 25th quantile and 75th quantile values from the describe method.

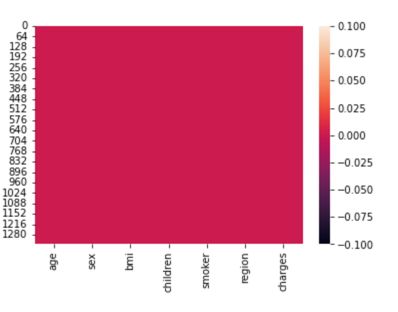
```python
df.isnull().sum()
```

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

As we can see there is no missing value in the dataset

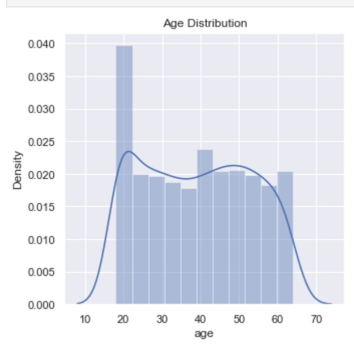
```python
sns.heatmap(df.isnull())
```

```
<AxesSubplot:>
```



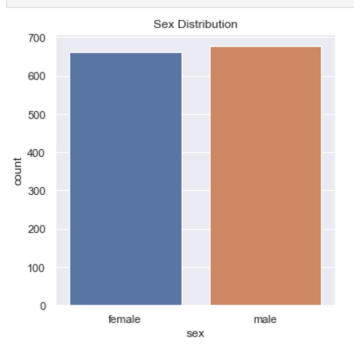
This is a visual representation of no missing data present in the dataset

```python
sns.set()
plt.figure(figsize=(5,5))
sns.distplot(df['age'])
plt.title('Age Distribution')
plt.show()
```



Age range is between 10 to 70 in which the most number of value is around 20 to 23 which means the people in age group 20 to 23 is maximum in our data and between 25 to 70 the distribution is almost normal

```python
plt.figure(figsize=(5,5))
sns.countplot(x='sex', data=df)
plt.title('Sex Distribution')
plt.show()
```

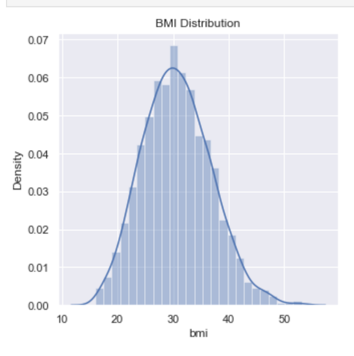


The sex distribution is almost equal for both the gender

```python
df['sex'].value_counts()
```

```
male      676
female    662
Name: sex, dtype: int64
```

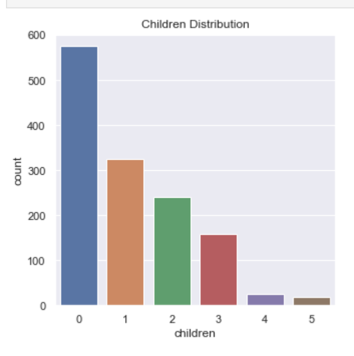
```python
plt.figure(figsize=(5,5))
sns.distplot(df['bmi'])
plt.title('BMI Distribution')
plt.show()
```



The distribution of BMI(Body Mass Index) which means whether the person is under weight or over weight, is a normal distribution

Normal BMI range => 18.5 to 24.9, so according to this graph most of the people are over weighted

```python
plt.figure(figsize=(5,5))
sns.countplot(x='children', data=df)
plt.title('Children Distribution')
plt.show()
```

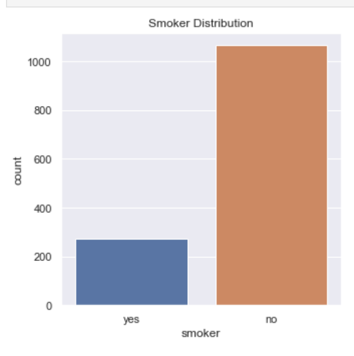


As we can observe that most of the people do not have child, and some people have 1 child and so on. The graph is decreasing for the increasing number of children

```python
df['children'].value_counts()
```

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

```python
plt.figure(figsize=(5,5))
sns.countplot(x='smoker', data=df)
plt.title('Smoker Distribution')
plt.show()
```

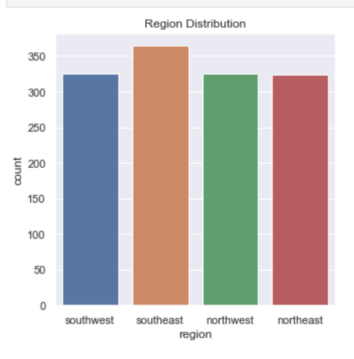


As we can see that there are huge no. of people who do not smoke and comparitively less number of people who smoke.

```python
df['smoker'].value_counts()
```

```
no     1064
yes     274
Name: smoker, dtype: int64
```

```python
plt.figure(figsize=(5,5))
sns.countplot(x='region', data=df)
plt.title('Region Distribution')
plt.show()
```

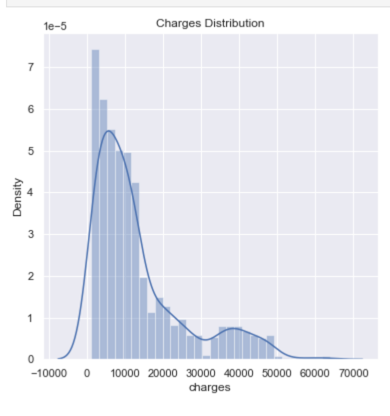


As we can observe that we have 4 region for which the data is almost similar, but it is little more for the southeast

```python
df['region'].value_counts()
```

```
southeast    364
southwest    325
northwest    325
northeast    324
Name: region, dtype: int64
```

```python
plt.figure(figsize=(5,5))
sns.distplot(df['charges'])
plt.title('Charges Distribution')
plt.show()
```



The high distribution of charges is around 0-10000 for rest we have very little value distributed

DATA PREPROCESSING

```python
df.replace({'sex':{'male':0,'female':1}}, inplace=True)
df.replace({'smoker':{'yes':0,'no':1}}, inplace=True)
df.replace({'region':{'southeast':0,'southwest':1,'northeast':2,'northwest':3}}, inplace=True)
```

```python
df
```

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

1338 rows × 7 columns

As we can see, we have some text data(sex, smoker and region) in our dataset which we have converted into numerical data

```python
X=df.drop(columns='charges', axis=1)
Y=df['charges']
```

```python
print(X)
```

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

[1338 rows x 6 columns]
```

```python
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
```

we have splited the data with X and Y variable in which X has all the inputs and Y has the output(charges)

```python
X_train,X_test,Y_train,Y_test=train_test_split(X,Y, test_size=0.2, random_state=2)
```

```python
print(X.shape,X_train.shape,X_test.shape)
```

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

MODEL TRAINING

```python
regressor=LinearRegression()
```

```python
regressor.fit(X_train,Y_train)
```

```
LinearRegression()
```

MODEL EVALUATION

```python
training_data_prediction=regressor.predict(X_train)
```

PERFORMANCE MATRIX

```python
r2_train=metrics.r2_score(Y_train,training_data_prediction)
print("R squared value : ",r2_train)
```

```
R squared value :  0.751505643411174
```

```python
testing_data_prediction=regressor.predict(X_test)
```

```python
r2_test=metrics.r2_score(Y_test,testing_data_prediction)
print("R squared value : ",r2_test)
```

```
R squared value :  0.7447273869684076
```

we are predicting for both the training data and testing data as to find out the overfitting(or over training) , in that case the model will over learn on training data, we get the the similar results which means our data is not overfitted

BUILDING A PREDICTIVE SYSTEM

```python
input_data=(31,1,25.74,0,1,0)
input_data_as_numpy_array=np.asarray(input_data)   #changing input-data in a numpy array
input_data_reshaped=input_data_as_numpy_array.reshape(1,-1)
prediction=regressor.predict(input_data_reshaped)

print("The Insurance Cost is USD ",prediction[0])
```

```
The Insurance Cost is USD  3760.0807968657
```

The predicted value is very close to the actual value, which means our model is working good