Medical Cost Prediction

The aim of this analysis is to predict the medical expense based on the patients'information. The dataset used for this analysis is Insurance dataset from Kaggle. The dataset contains 1338 observations and 7 variables. The variables are as follows:

Variable	Description
age	age of primary beneficiary
bmi	body mass index
children	number of children covered by health insurance
smoker	smoking
region	the beneficiary's residential area in the US
charges	individual medical costs billed by health insurance

```
#importing the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df = pd.read_csv('insurance.csv')
df.head()
```

```
   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
```

Data Preprocessing

```
#number of rows and columns
df.shape
```

```
(1338, 7)
```

```
#checking for missing values
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

#checking discriptive statistics
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

#value counts for categorical variables
print(df.sex.value_counts(),'\n',df.smoker.value_counts(),'\
n',df.region.value_counts())

male      676
female    662
Name: sex, dtype: int64
 no     1064
yes     274
Name: smoker, dtype: int64
 southeast    364
southwest    325
northwest    325
northeast    324
Name: region, dtype: int64
```

Lable Encoding the categorical values

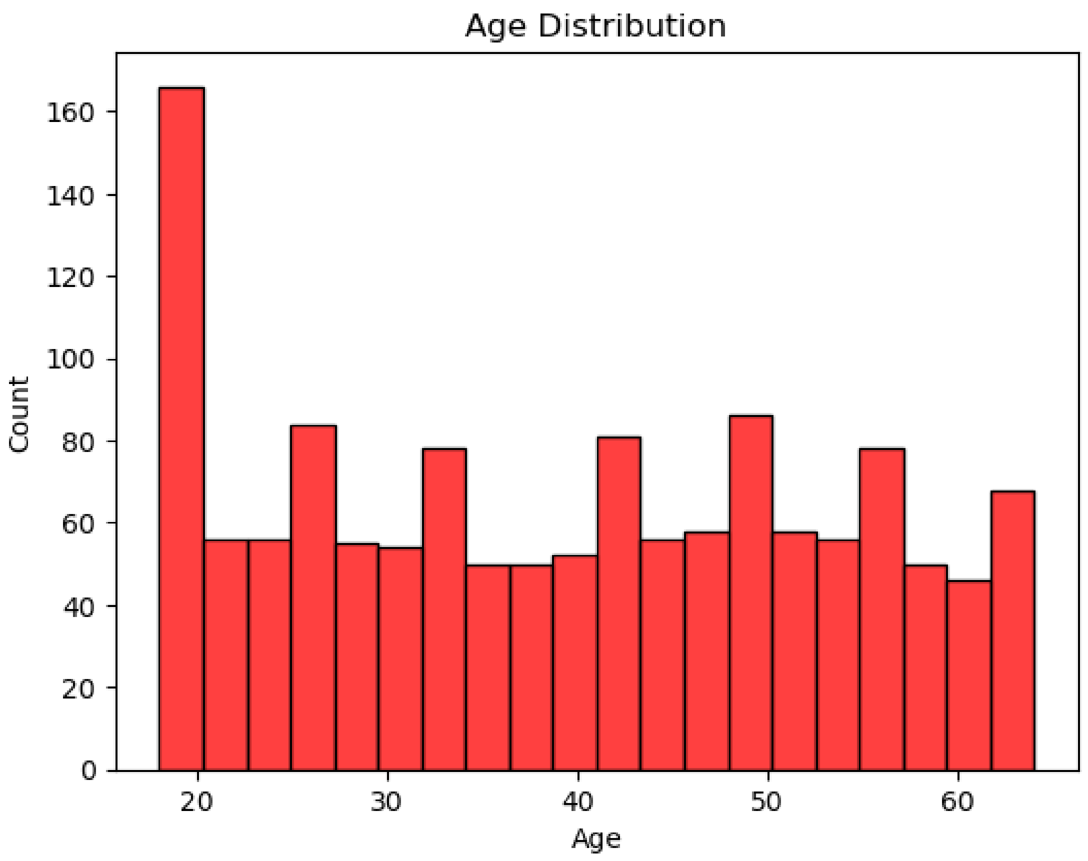
```
from sklearn.preprocessing import LabelEncoder

Le = LabelEncoder()

df['sex'] = Le.fit_transform(df['sex'])
df['smoker'] = Le.fit_transform(df['smoker'])
df['region'] = Le.fit_transform(df['region'])
```

```
df.head()

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

Exploratory Data Analysis

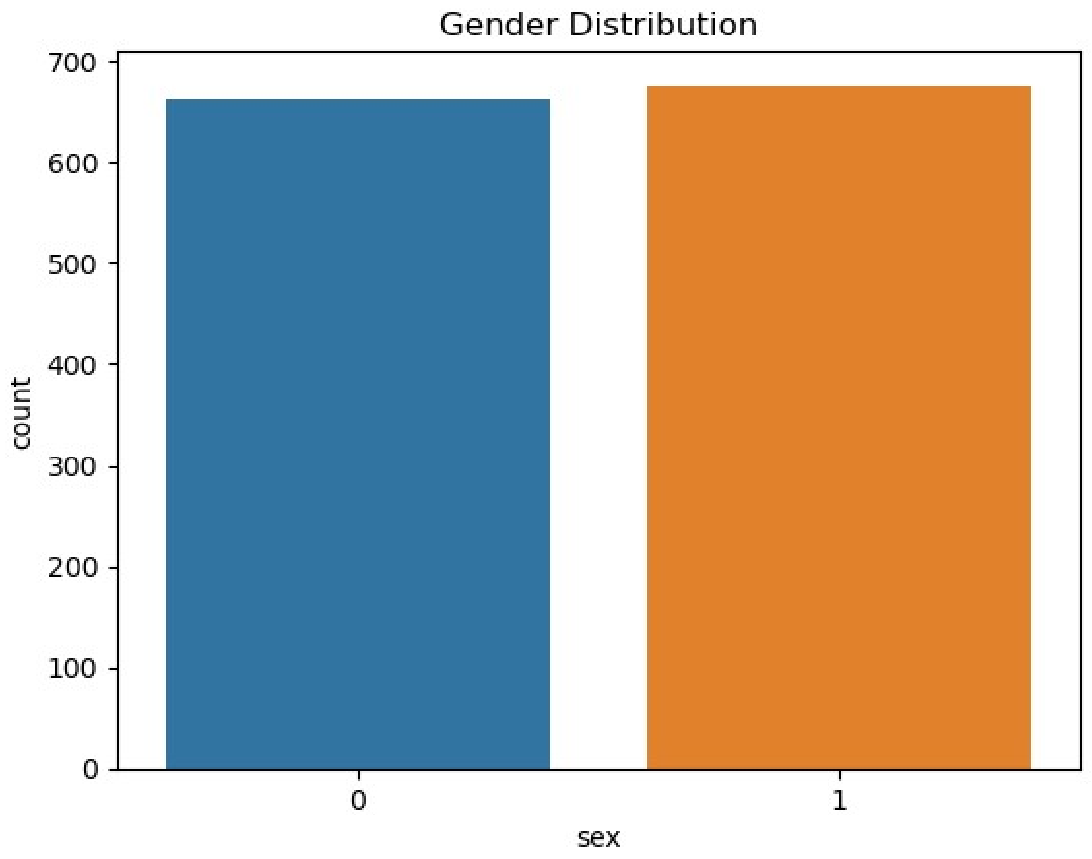
Visualization of the data is a good way to understand the data. In this section, I will plot the distribution of each variable to get an overview about their counts and distributions.

```
#age distribution
sns.histplot(df.age,bins=20, kde=False,color='red')
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```



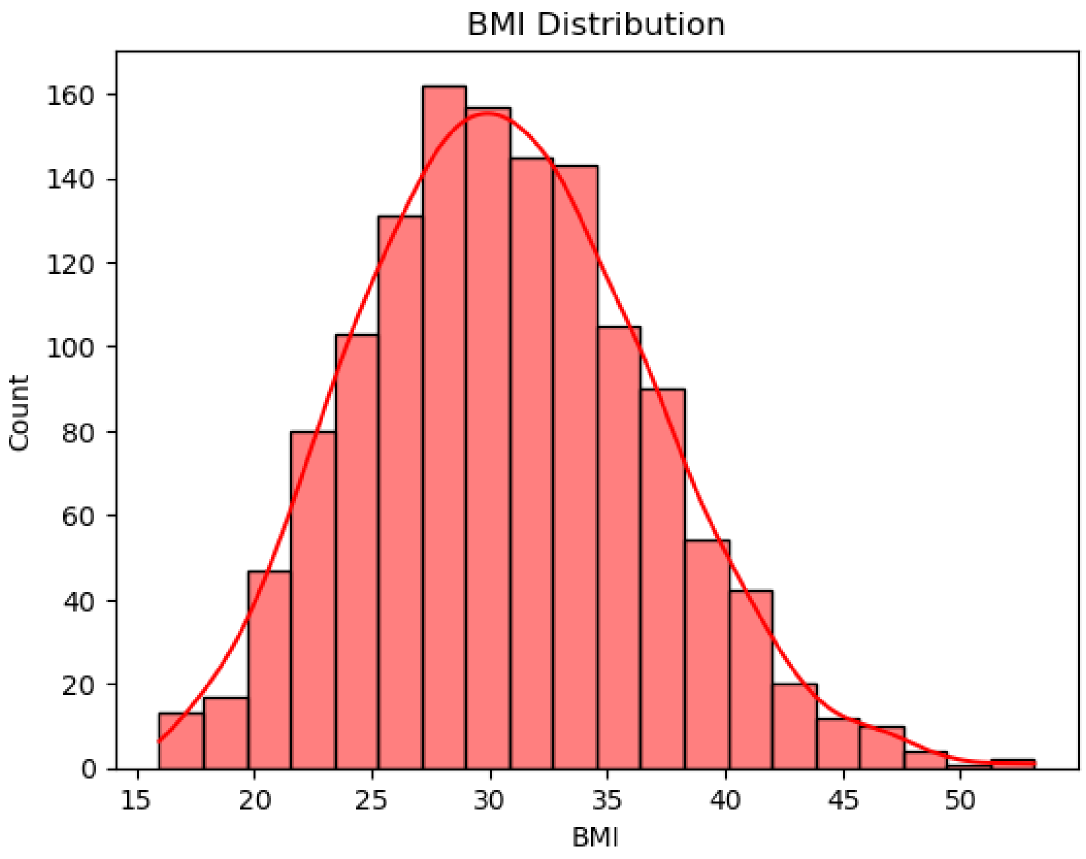
```
#gender plot
sns.countplot(x = 'sex', data = df)
plt.title('Gender Distribution')
```

```
Text(0.5, 1.0, 'Gender Distribution')
```



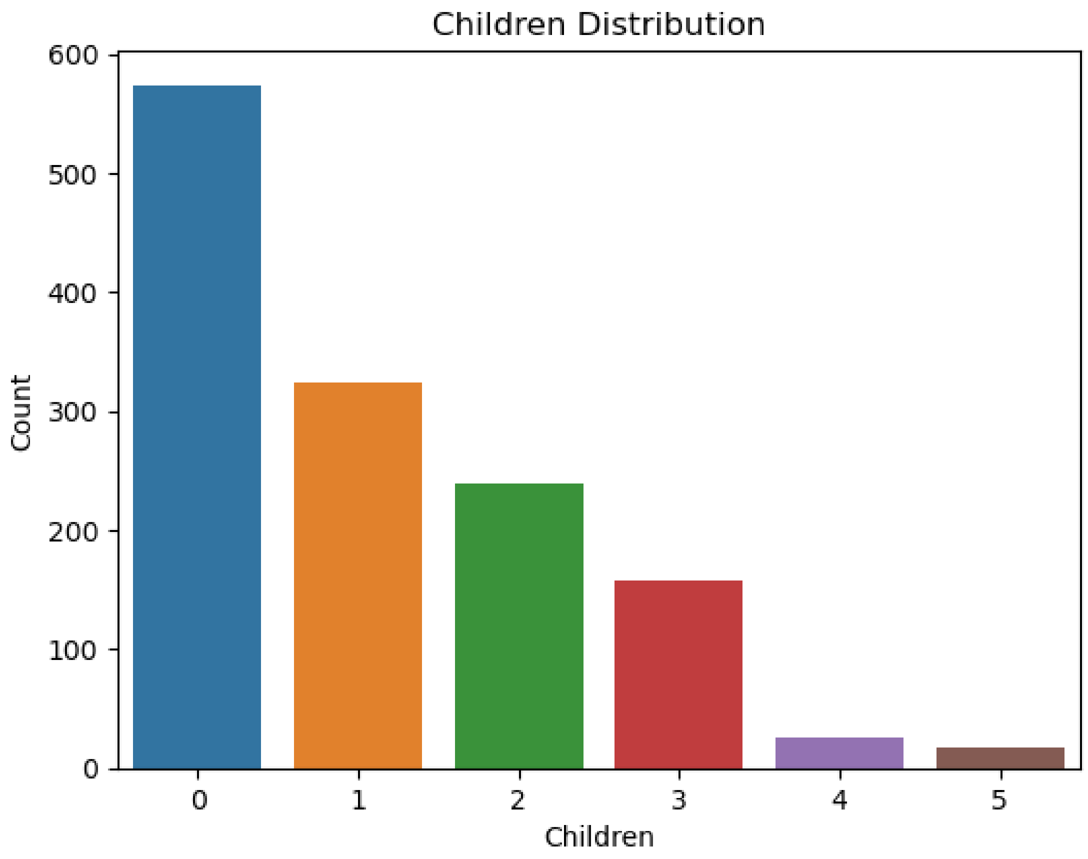
It is clear that number of males and females are almost equal in the dataset.

```
#bmi distribution
sns.histplot(df.bmi,bins=20, kde=True,color='red')
plt.title('BMI Distribution')
plt.xlabel('BMI')
plt.ylabel('Count')
plt.show()
```



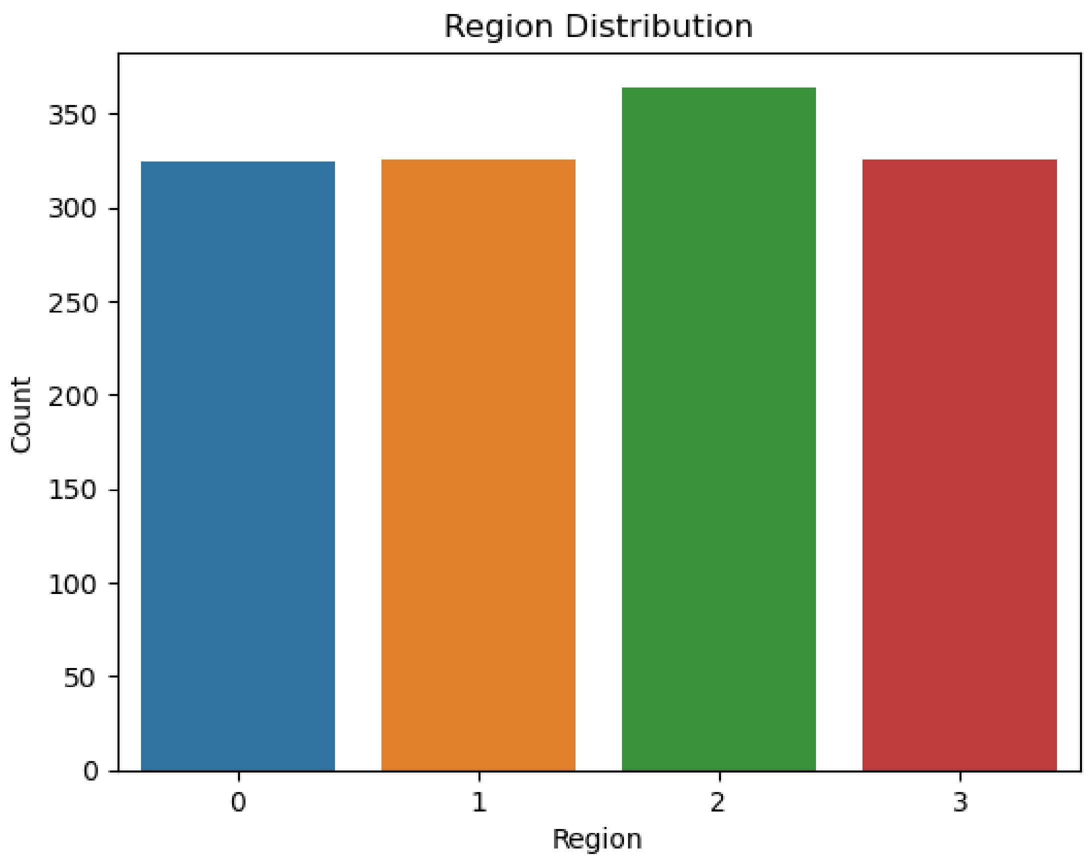
The majority of the patients have BMI between 25 and 40 which is considered as overweight and could be a major factor in increasing the medical cost.

```
#child count distribution
sns.countplot(x = 'children', data = df)
plt.title('Children Distribution')
plt.xlabel('Children')
plt.ylabel('Count')
plt.show()
```



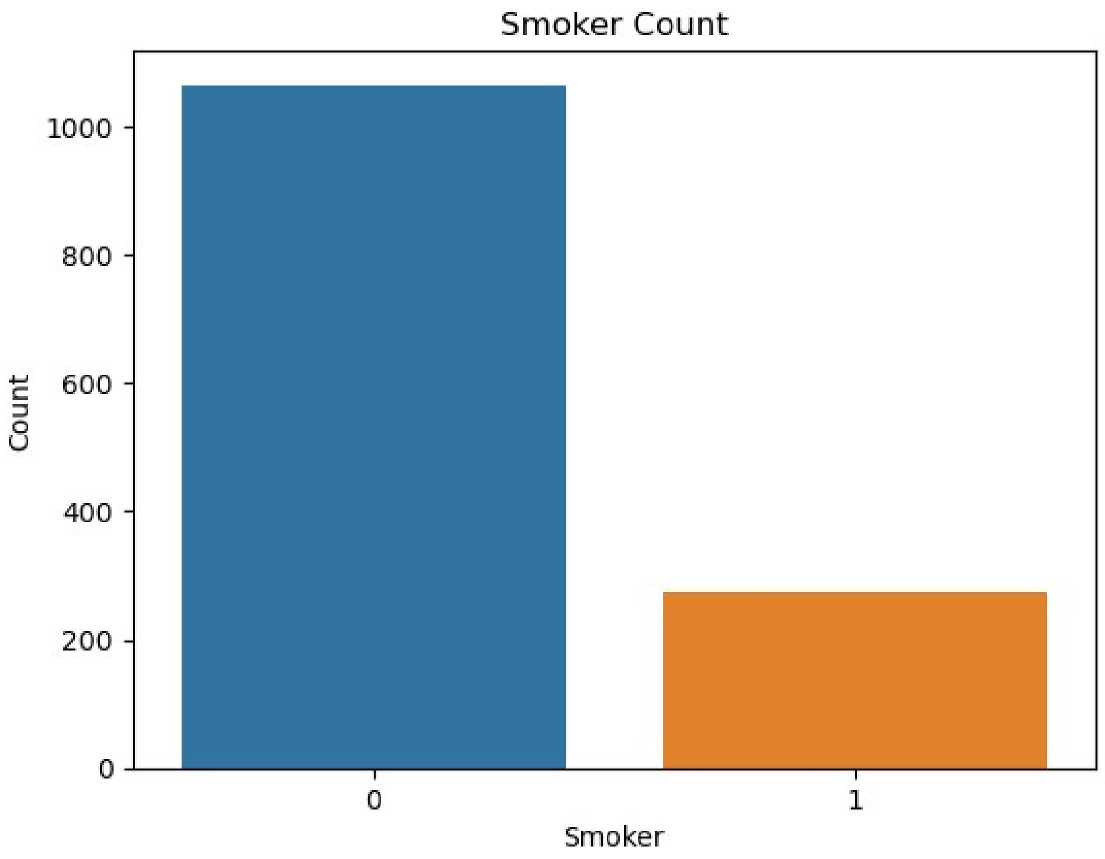
The graph clearly shows that most of the patients have no children and very few patients have more than 3 children.

```
#regionwise plot
sns.countplot(x = 'region', data = df)
plt.title('Region Distribution')
plt.xlabel('Region')
plt.ylabel('Count')
plt.show()
```



The count of patient from northwest is slighltly higher than the other regions, but the number of patients from other regions are almost equal.

```
#count of smokers
sns.countplot(x = 'smoker', data = df)
plt.title('Smoker Count')
plt.xlabel('Smoker')
plt.ylabel('Count')
plt.show()
```

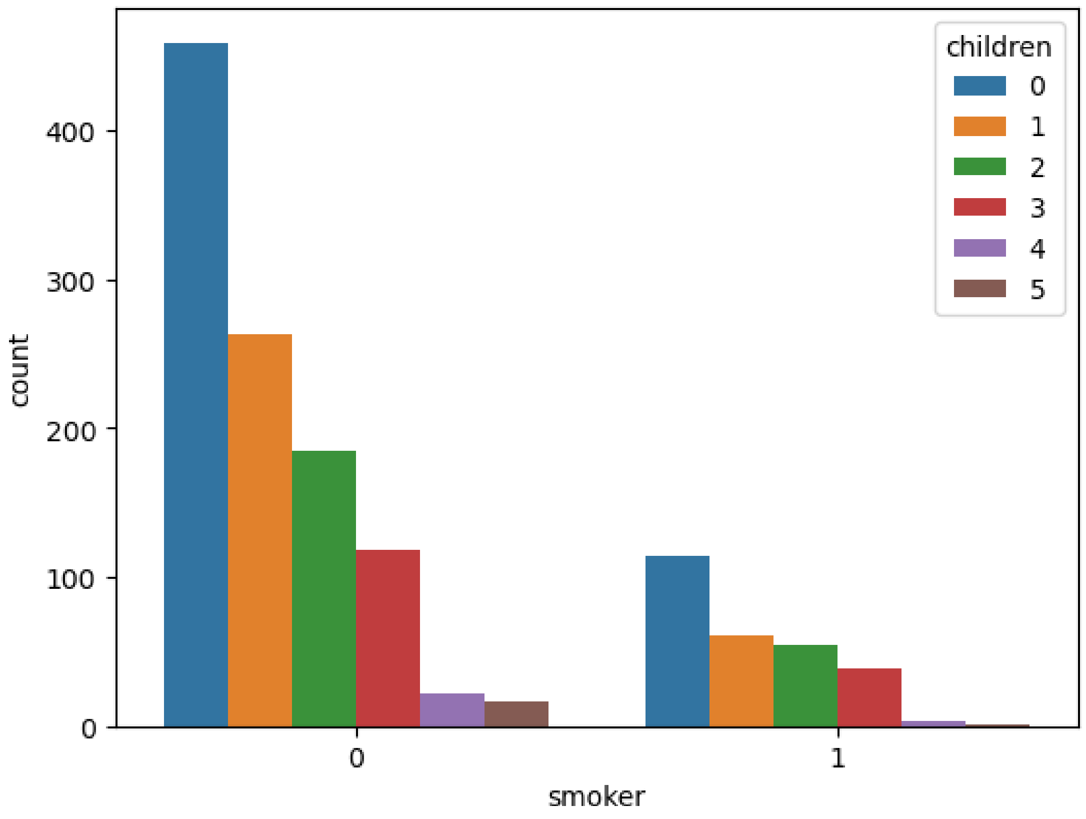


smokers are very few in the dataset. Nearly 80% of the patients are non-smokers.

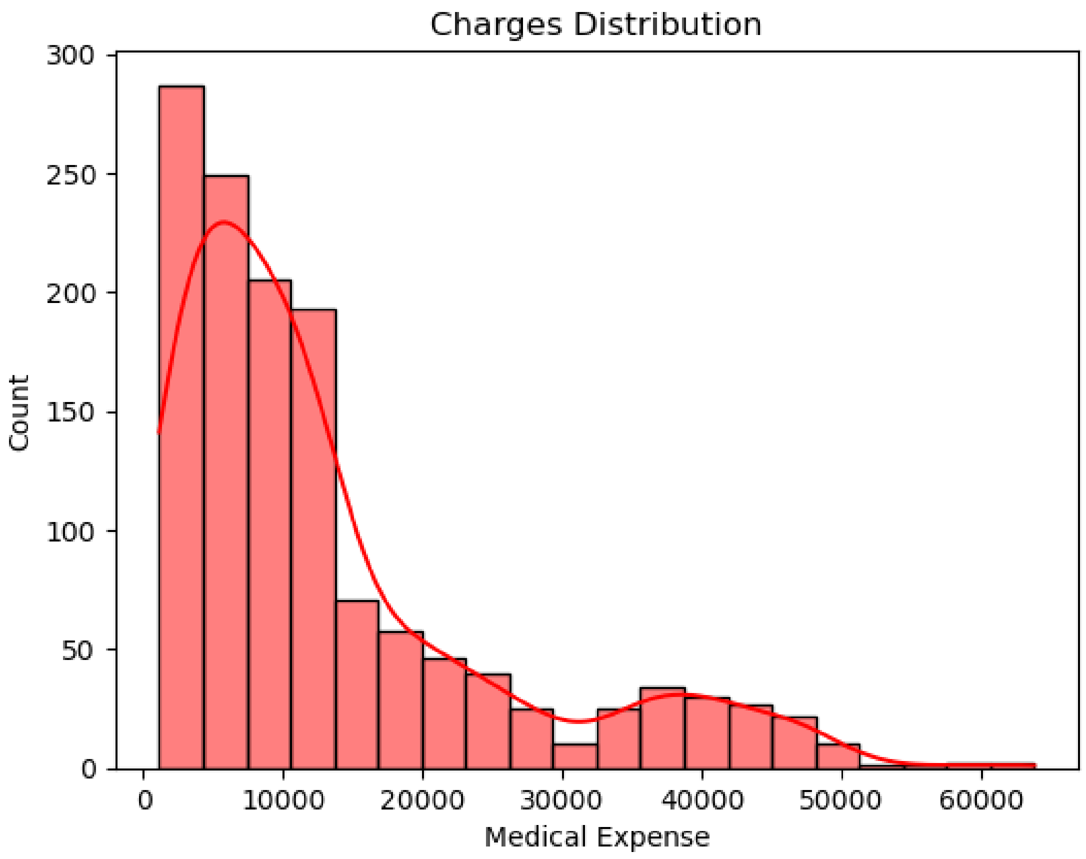
Smoker count with respect to the children count.

```
sns.countplot(x = df.smoker, hue = df.children)
```

```
<Axes: xlabel='smoker', ylabel='count'>
```



```
#charges distribution
sns.histplot(df.charges,bins=20, kde=True,color='red')
plt.title('Charges Distribution')
plt.xlabel('Medical Expense')
plt.ylabel('Count')
plt.show()
```



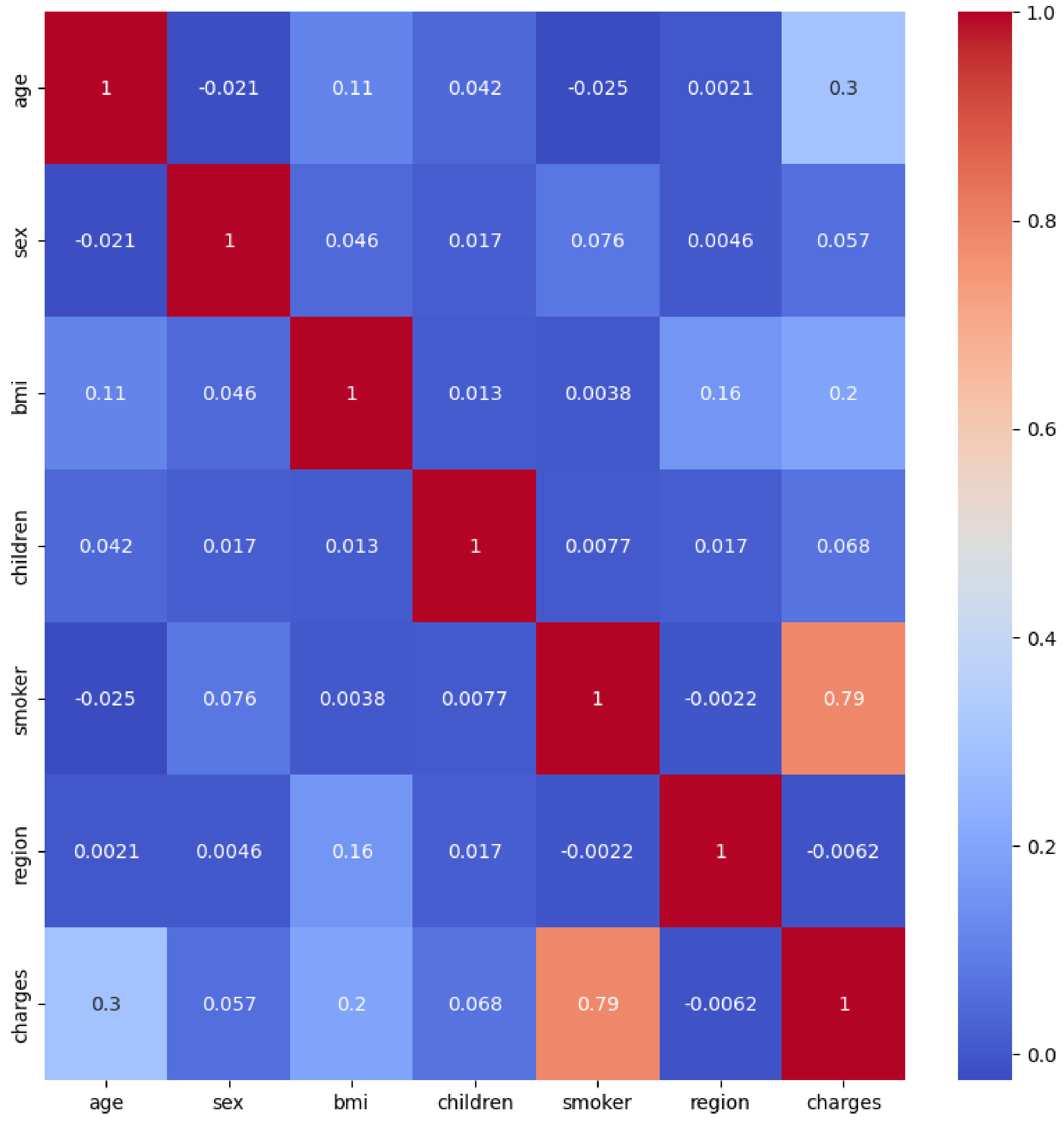
Most of the medical expenses are below 20000, with negligible number of patients having medical expenses above 50000.

From all the above plots, we have a clear understanding about the count of patients under each category of the variables. Now I will look into the coorelation between the variables.

Coorelation

```
#coorelation matrix
df.corr()

               age       sex       bmi  children    smoker    region
charges
age       1.000000 -0.020856  0.109272  0.042469 -0.025019  0.002127
0.299008
sex      -0.020856  1.000000  0.046371  0.017163  0.076185  0.004588
0.057292
bmi       0.109272  0.046371  1.000000  0.012759  0.003750  0.157566
0.198341
children  0.042469  0.017163  0.012759  1.000000  0.007673  0.016569
0.067998
smoker   -0.025019  0.076185  0.003750  0.007673  1.000000 -0.002181
0.787251
region    0.002127  0.004588  0.157566  0.016569 -0.002181  1.000000 -
```

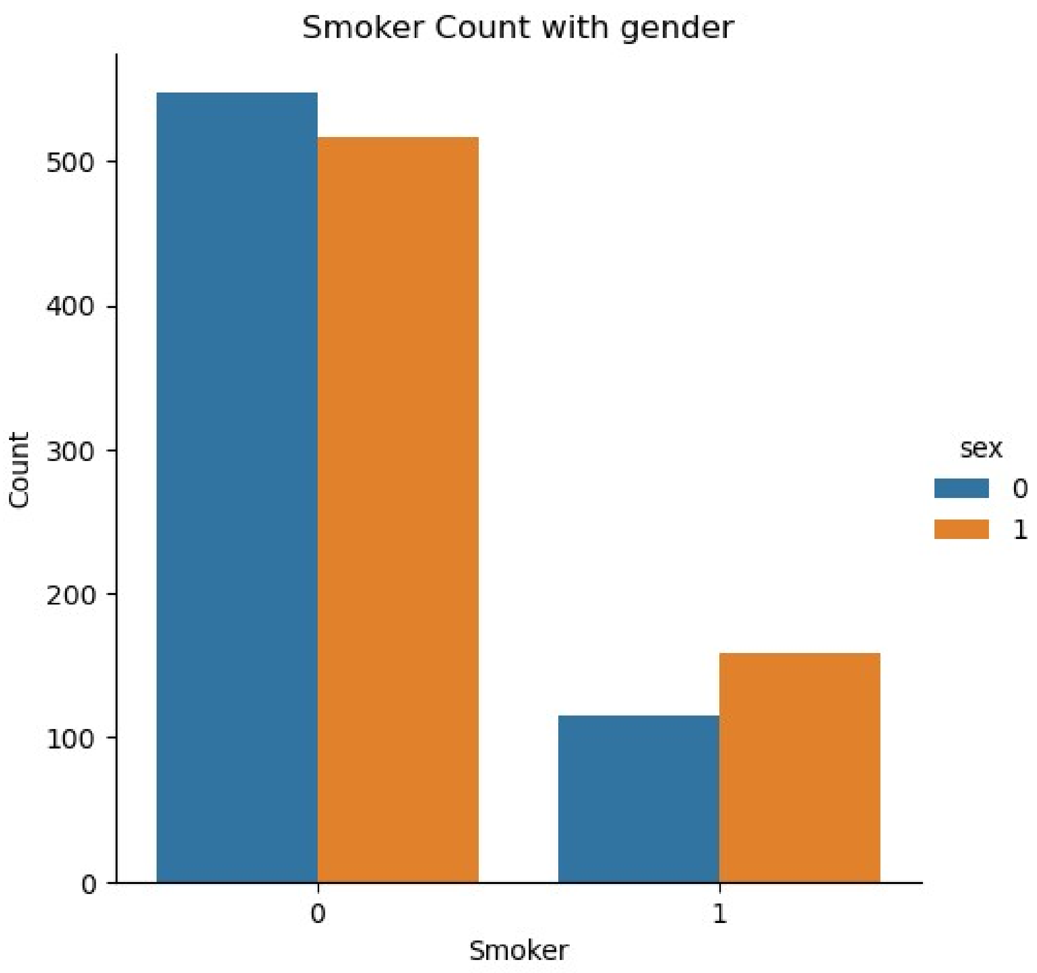
```
0.006208
charges    0.299008  0.057292  0.198341  0.067998  0.787251 -0.006208
1.000000

#plotting the coorelation heatmap
plt.figure(figsize=(10,10))
sns.heatmap(df.corr(),annot=True,cmap='coolwarm')
plt.show()
```



The variable smoker shows a significant coorelation with the medical expenses. Now I will explore more into patients' smoking habits and their relationa with other factors.

Plotting the smoker count with patient's gender

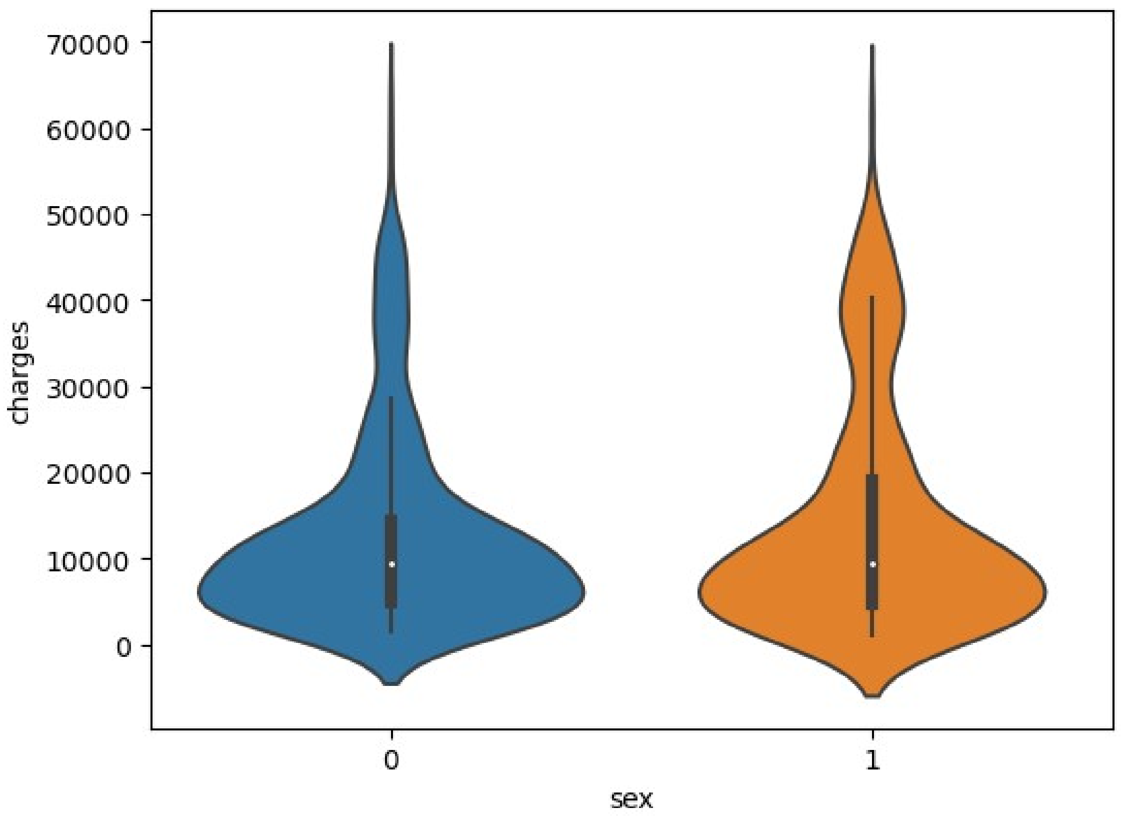
```
sns.catplot(x="smoker", kind="count",hue = 'sex', data=df)
plt.title('Smoker Count with gender')
plt.xlabel('Smoker')
plt.ylabel('Count')
plt.show()
```



We can notice more male smokers than female smokers. So, I will assume that medical treatment expense for males would be more than females, given the impact of smoking on the medical expenses.

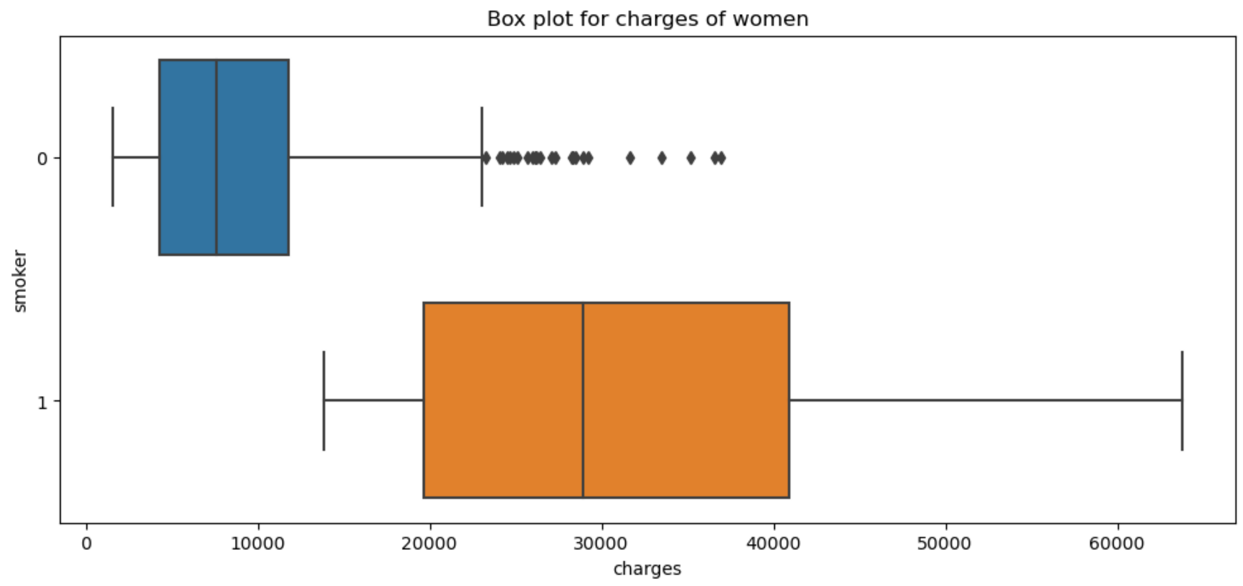
```
sns.violinplot(x = 'sex', y = 'charges', data = df)
```

```
<Axes: xlabel='sex', ylabel='charges'>
```



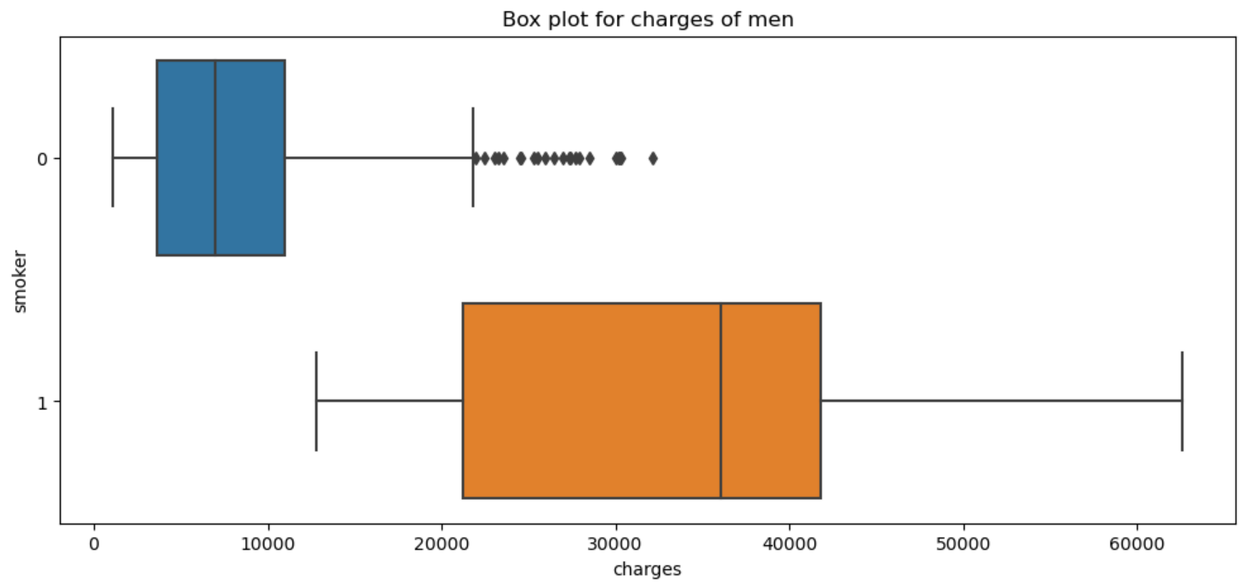
```
plt.figure(figsize=(12,5))
plt.title("Box plot for charges of women")
sns.boxplot(y="smoker", x="charges", data =  df[(df.sex == 0)] ,
orient="h")
```

```
<Axes: title={'center': 'Box plot for charges of women'},
xlabel='charges', ylabel='smoker'>
```



```
plt.figure(figsize=(12,5))
plt.title("Box plot for charges of men")
sns.boxplot(y="smoker", x="charges", data =  df[(df.sex == 1)] ,
orient="h")
```

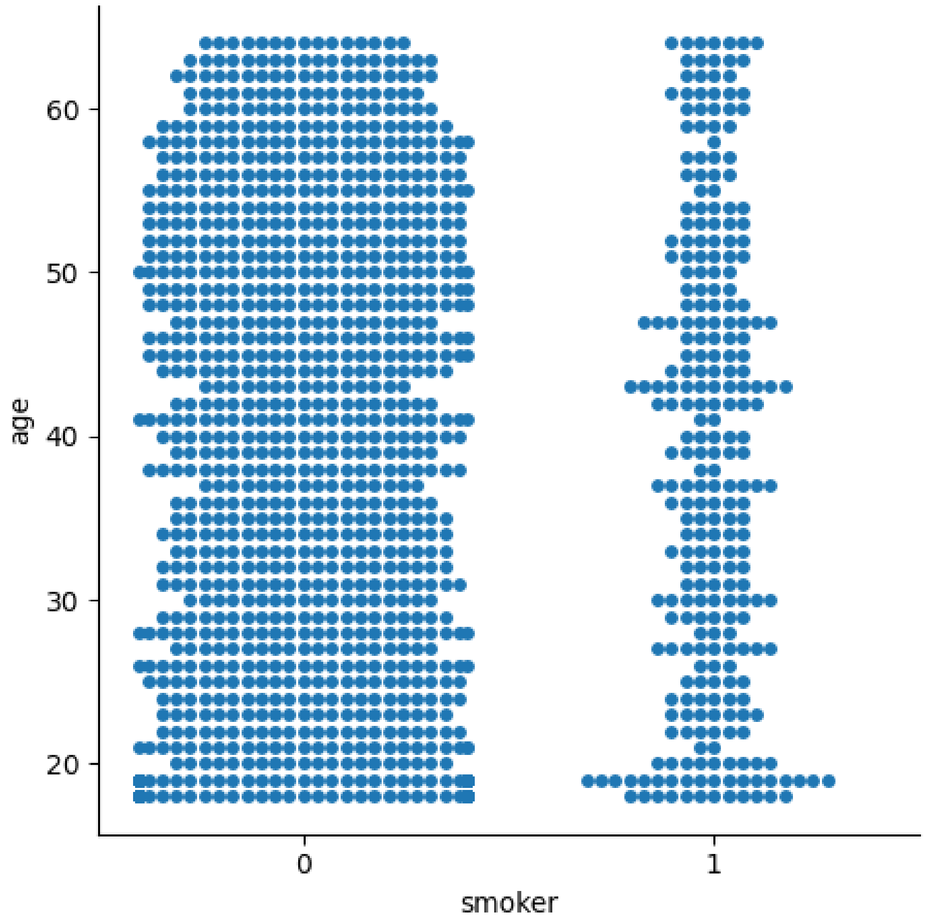
```
<Axes: title={'center': 'Box plot for charges of men'},
xlabel='charges', ylabel='smoker'>
```



The assumption is true, that the medical expense of males is greater than that of females. In addituion to that medical expense of smokers is greater than that of non-smokers.

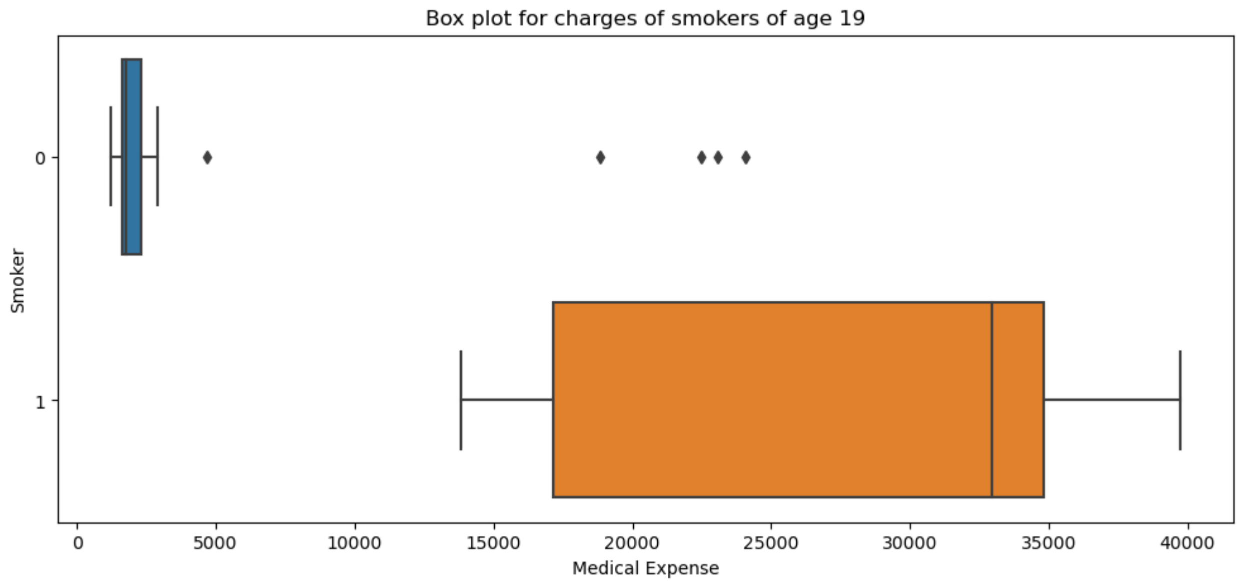
Smokers and age distribution

```
#smokers and age distribution
sns.catplot(x="smoker", y="age", kind="swarm", data=df)

<seaborn.axisgrid.FacetGrid at 0x21999778d10>

C:\Users\admin\anaconda3\Lib\site-packages\seaborn\
categorical.py:3544: UserWarning: 7.3% of the points cannot be placed;
you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```



From the graph, we can see that there significant number of smokers of age 19. Now I will study the medical expense of smokers of age 19.

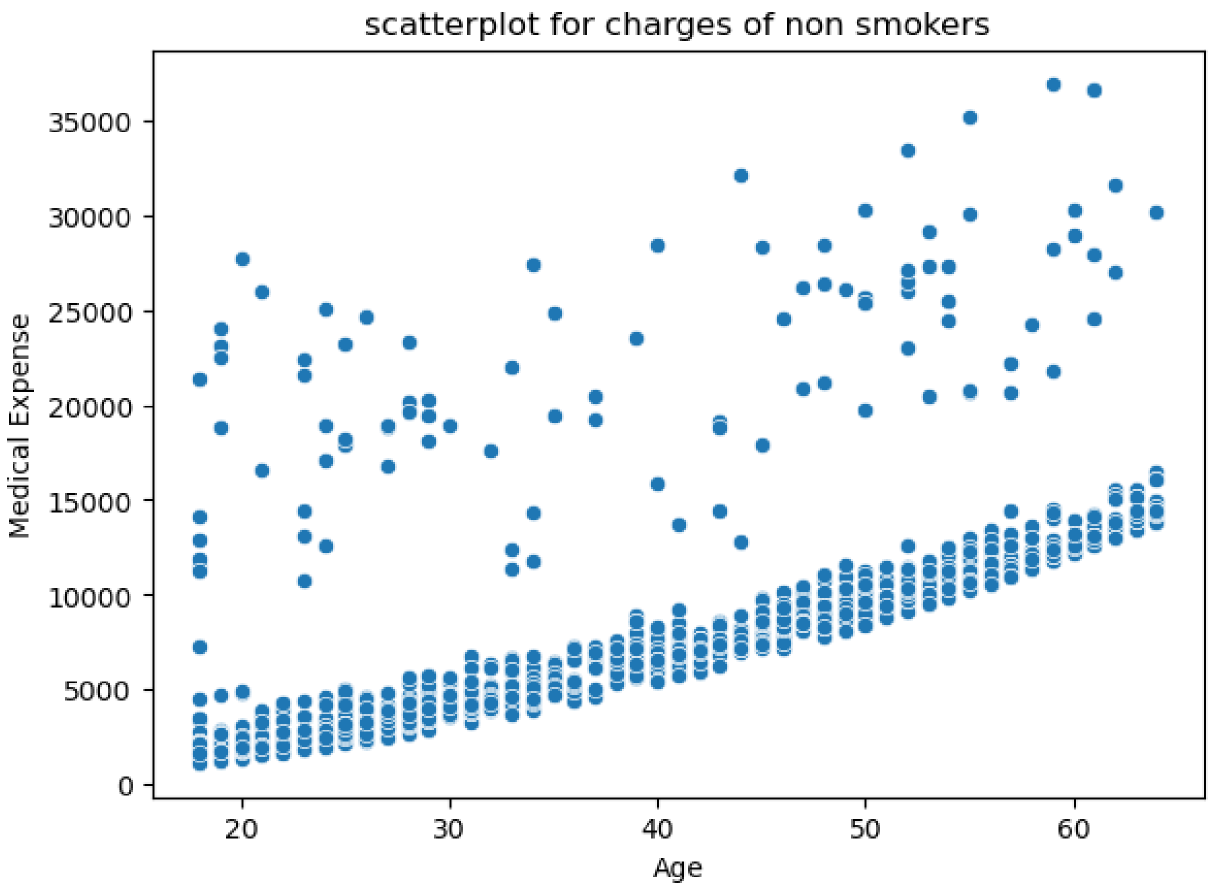
```
#smokers of age 19
plt.figure(figsize=(12,5))
plt.title("Box plot for charges of smokers of age 19")
sns.boxplot(y="smoker", x="charges", data =  df[(df.age == 19)] ,
orient="h")
plt.xlabel('Medical Expense')
plt.ylabel('Smoker')
plt.show()
```



Surprisingly the medical expense of smokers of age 19 is very high in comparison to non smokers. In non smokers we can see some outliners, which may be due to illness or accidents.

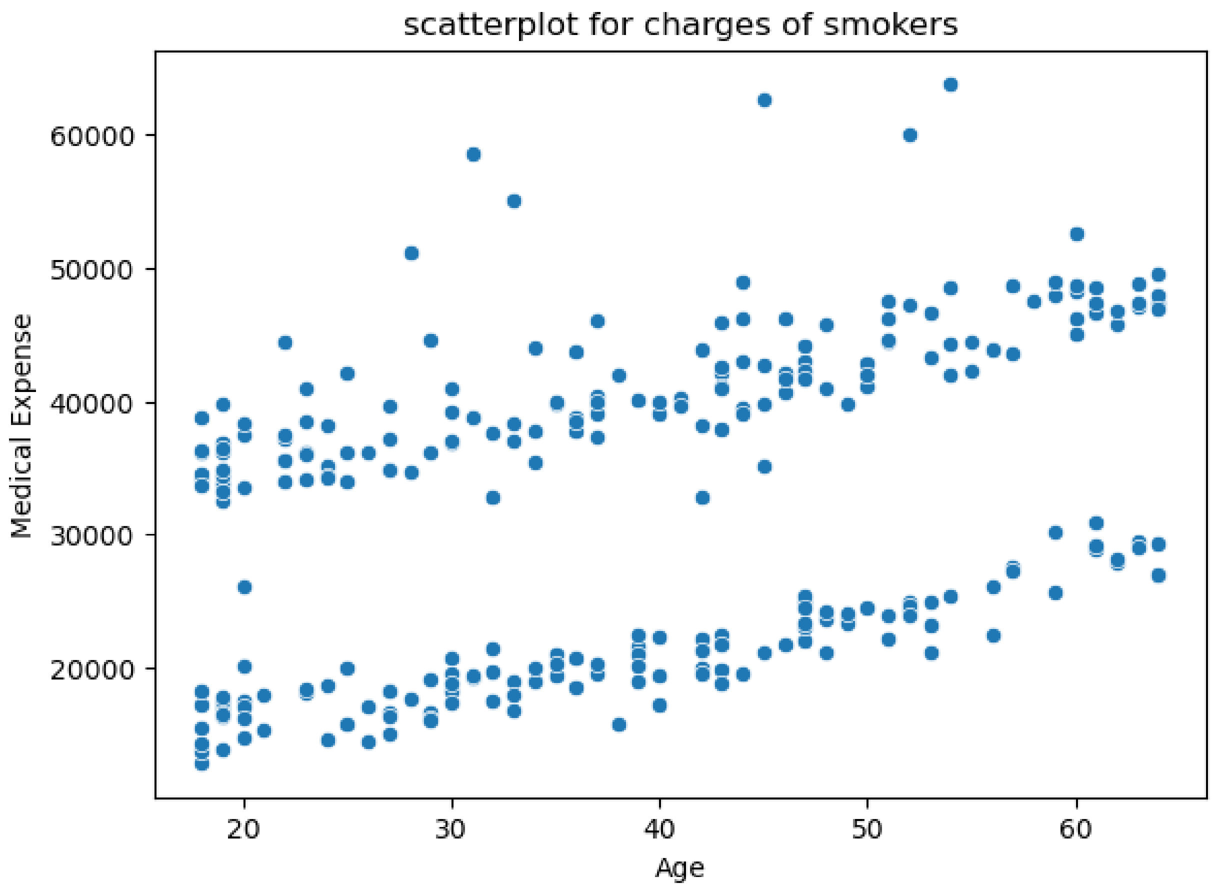
It is clear that the medical expense of smokers is higher than that of non-smokers. Now I will plot the charges distribution with repect to patients age of smokers and non-smokers.

```
#non smokers charge distribution
plt.figure(figsize=(7,5))
plt.title("scatterplot for charges of non smokers")
sns.scatterplot(x="age", y="charges", data =  df[(df.smoker == 0)])
plt.xlabel('Age')
plt.ylabel('Medical Expense')
plt.show()
```



Majority of the points shows that medical expense increases with age which may be due to the fact that older people are more prone to illness. But there are some outliners which shows that there are other illness or accidents which may increase the medical expense.

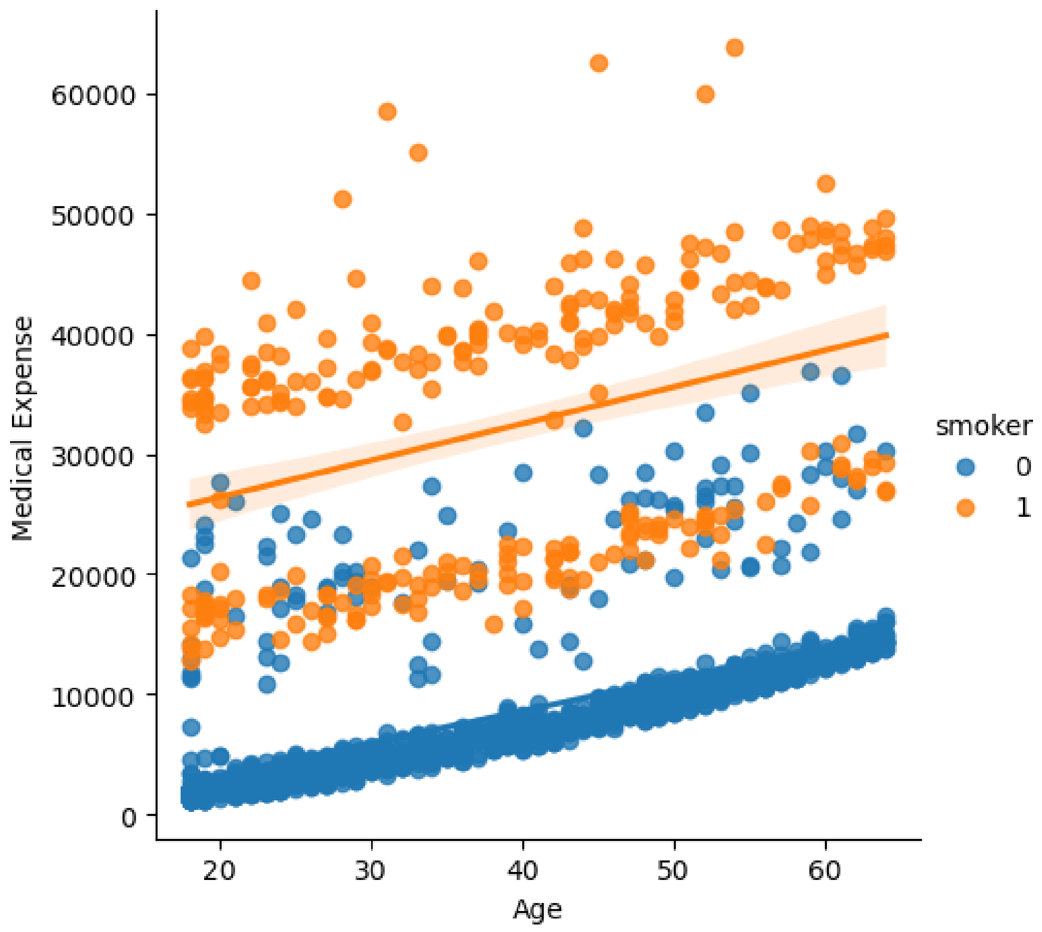
```
#smokers charge distribution
plt.figure(figsize=(7,5))
plt.title("scatterplot for charges of smokers")
sns.scatterplot(x="age", y="charges", data =  df[(df.smoker == 1)])
plt.xlabel('Age')
plt.ylabel('Medical Expense')
plt.show()
```



Here we see pecularity in the graph. In the graph there are two segments, one with high medical expense which may be due to smoking related illness and the other with low medical expense which may be due age related illness.

Now, in order to get a more clear picture, I will combine these two graphs.

```
#age charges distribution

sns.lmplot(x="age", y="charges", data =  df, hue = 'smoker')
plt.xlabel('Age')
plt.ylabel('Medical Expense')
plt.show()
```



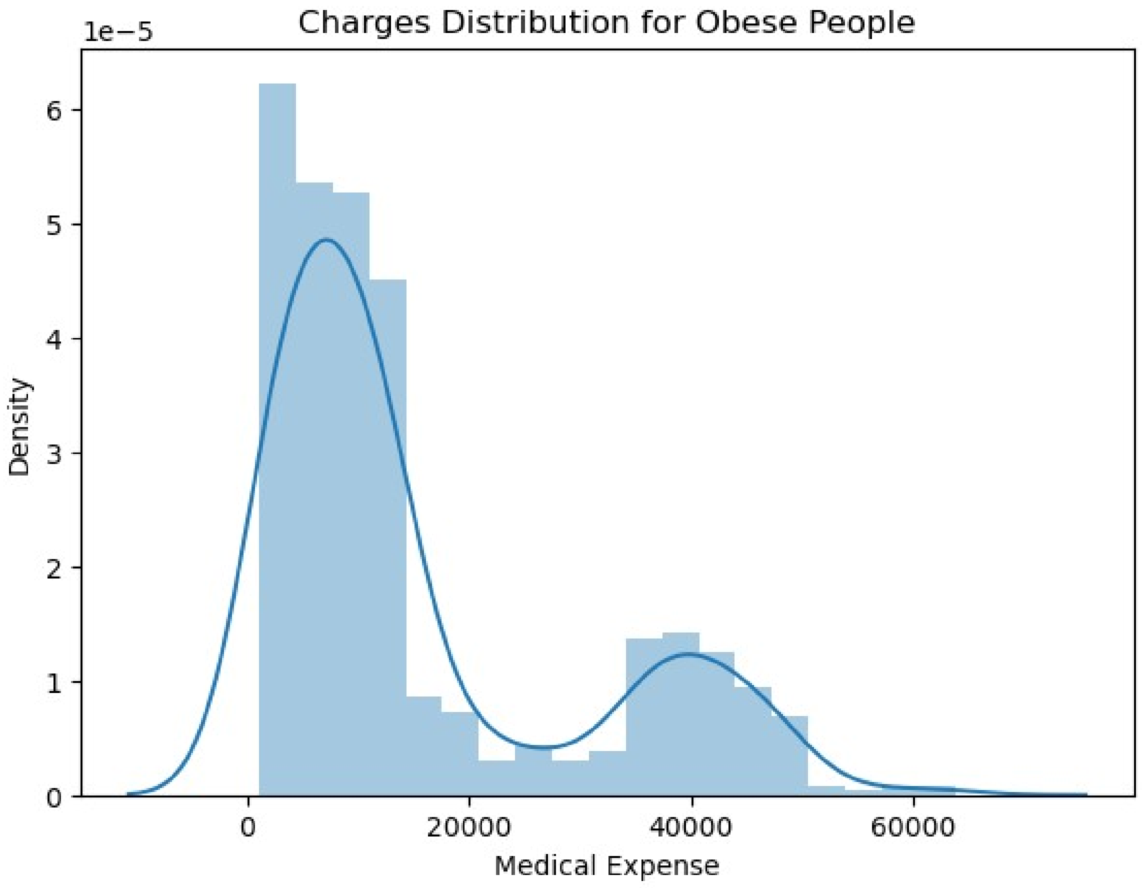
Now, we clearly understand the variation in charges with respect to age and smoking habits. The medical expense of smokers is higher than that of non-smokers. In non-smokers, the cost of treatment increases with age which is obvious. But in smokers, the cost of treatment is high even for younger patients, which means the smoking patients are spending upon their smoking related illness as well as age related illness.

Charges distribution for patients with BMI greater than 30 i.e. obese patients

```
#bmi charges distribution for obese people
plt.figure(figsize=(7,5))
sns.distplot(df[(df.bmi >= 30)]['charges'])
plt.title('Charges Distribution for Obese People')
plt.xlabel('Medical Expense')
plt.show()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\836969744.py:3:
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(df[(df.bmi >= 30)]['charges'])
```

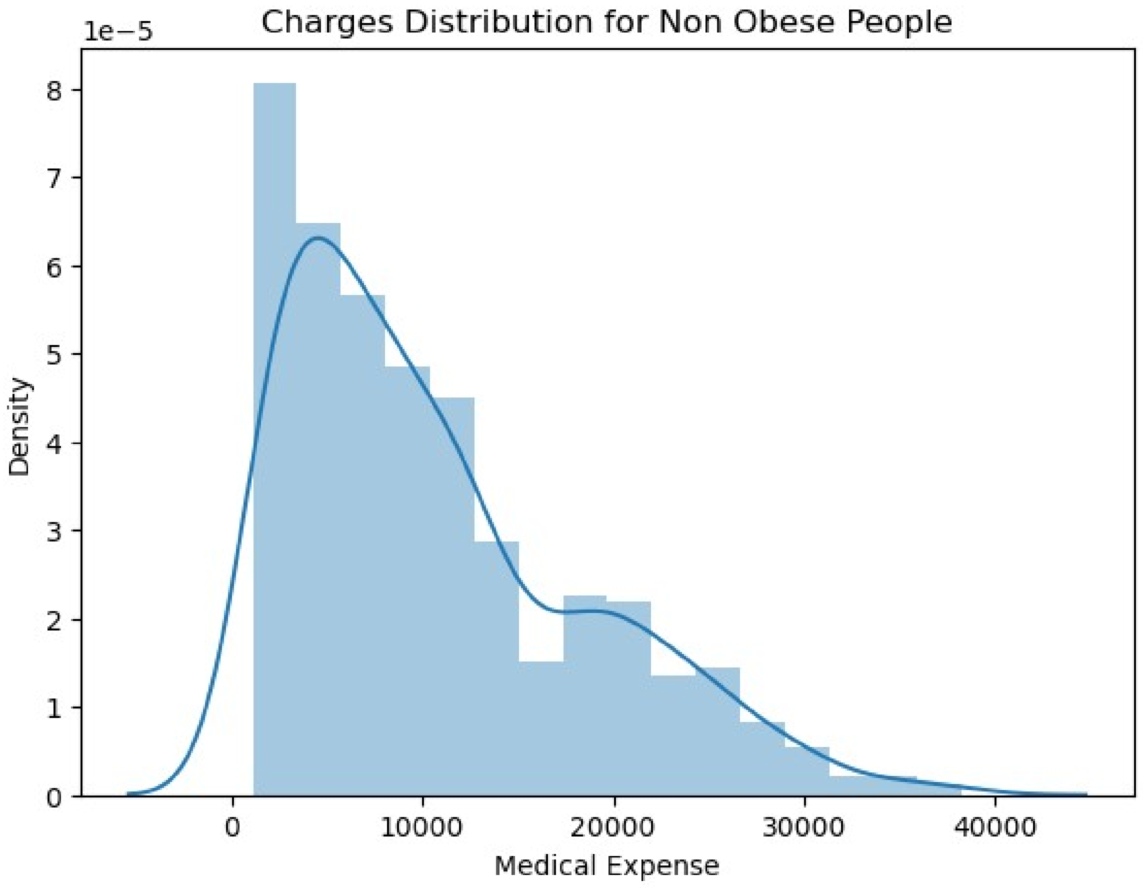


Charges distribution for patients with BMI less than 30 i.e. healthy patients

```
plt.figure(figsize=(7,5))
sns.distplot(df[(df.bmi < 30)]['charges'])
plt.title('Charges Distribution for Non Obese People')
plt.xlabel('Medical Expense')
plt.show()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\887406048.py:2:
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(df[(df.bmi < 30)]['charges'])
```



Therefore, patients with BMI less than 30 are spending less on medical treatment than those with BMI greater than 30.

Through the EDA, we have a clear understanding about the data and the coorelation between the variables. Now, I will build a model to predict the medical expense of patients.

Train Test Split

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test =
train_test_split(df.drop('charges',axis=1), df['charges'],
test_size=0.2, random_state=0)
```

Model Building

Linear Regression

```
#Linear Regression
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr
```

```
LinearRegression()
```

```
#model training
lr.fit(x_train,y_train)
#model accuracy
lr.score(x_train,y_train)
```

```
0.7368306228430945
```

```
#model prediction
y_pred = lr.predict(x_test)
```

Polynomial Regression

```
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree=2)
poly_reg
```

```
PolynomialFeatures()
```

```
#transforming the features to higher degree
x_train_poly = poly_reg.fit_transform(x_train)
#splitting the data
x_train, x_test, y_train, y_test = train_test_split(x_train_poly,
y_train, test_size=0.2, random_state=0)
```

```
plr = LinearRegression()
#model training
plr.fit(x_train,y_train)
#model accuracy
plr.score(x_train,y_train)
```

```
0.83737417416858
```

```
#model prediction
y_pred = plr.predict(x_test)
```

Decision Tree Regressor

```
#decision tree regressor
from sklearn.tree import DecisionTreeRegressor
dtree = DecisionTreeRegressor()
dtree
```

```
DecisionTreeRegressor()
```

```
#model training
dtree.fit(x_train,y_train)
#model accuracy
dtree.score(x_train,y_train)
```

```
0.9993688476658964
```

```
#model prediction
dtree_pred = dtree.predict(x_test)
```

Random Forest Regressor

```
#random forest regressor
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(n_estimators=100)
rf
```

```
RandomForestRegressor()
```

```
#model training
rf.fit(x_train,y_train)
#model accuracy
rf.score(x_train,y_train)
```

```
0.9747594816961677
```

```
#model prediction
rf_pred = rf.predict(x_test)
```

Model Evaluation

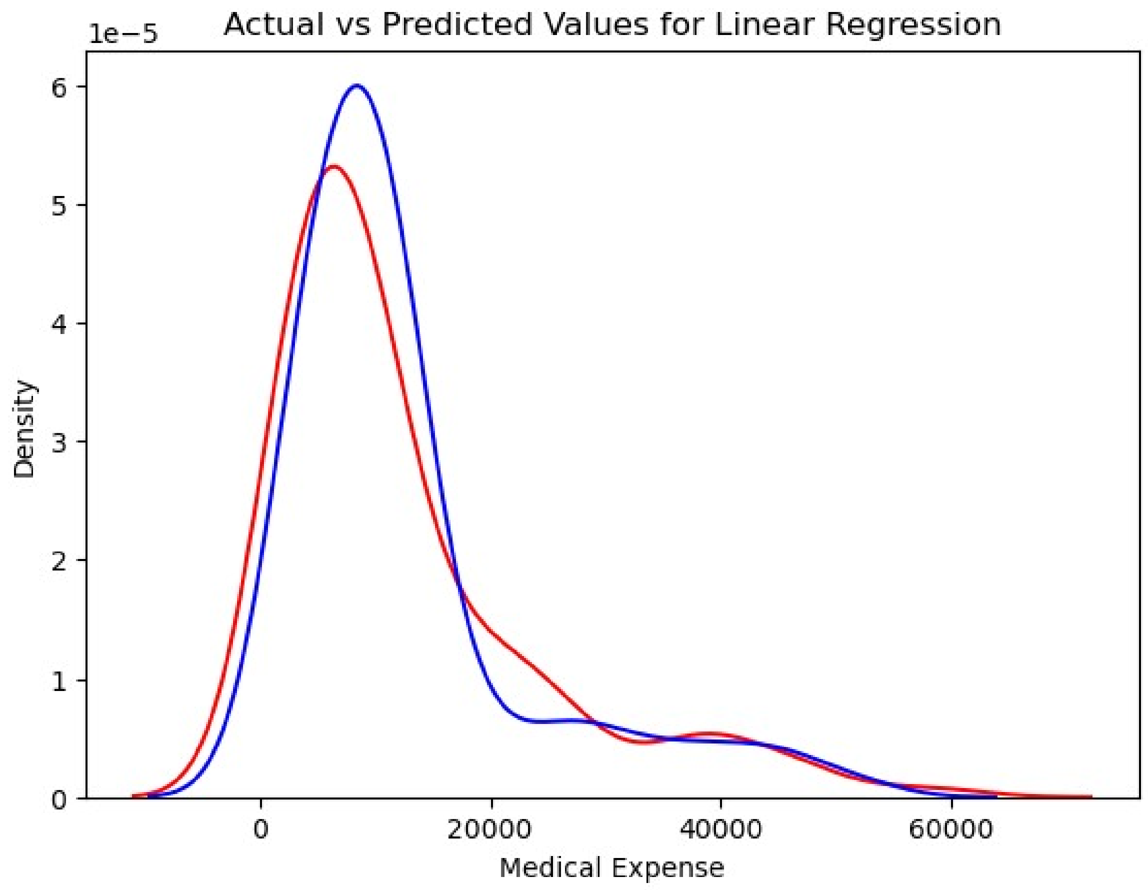
```
from sklearn.metrics import
mean_squared_error,mean_absolute_error,r2_score
```

Linear Regression

```
#distribution of actual and predicted values
plt.figure(figsize=(7,5))
ax1 = sns.distplot(y_test,hist=False,color='r',label='Actual Value')
```

```
sns.distplot(y_pred,hist=False,color='b',label='Predicted
Value',ax=ax1)
plt.title('Actual vs Predicted Values for Linear Regression')
plt.xlabel('Medical Expense')
plt.show()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\767213784.py:3:
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 `kdeplot` (an axes-level function for kernel
density plots).

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

  ax1 = sns.distplot(y_test,hist=False,color='r',label='Actual Value')
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\767213784.py:4:
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 `kdeplot` (an axes-level function for kernel
density plots).

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

  sns.distplot(y_pred,hist=False,color='b',label='Predicted
Value',ax=ax1)
```



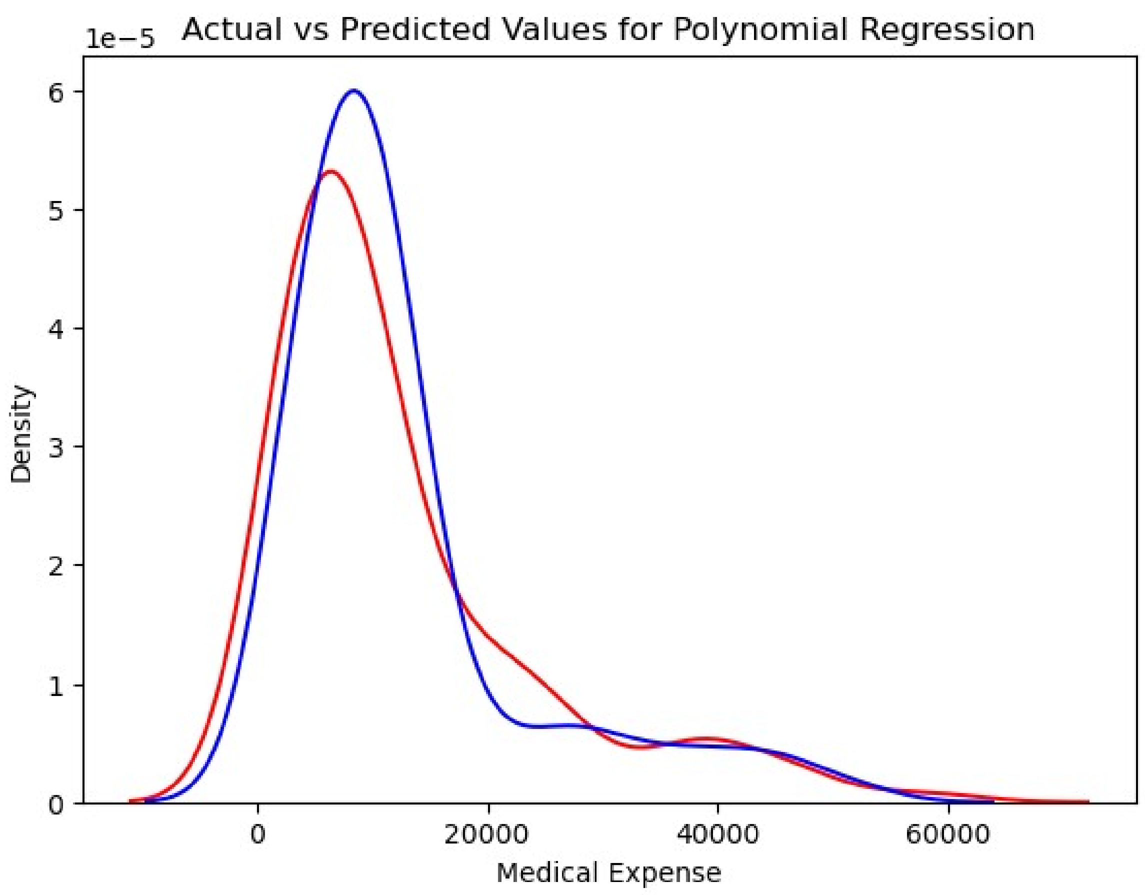
```
print('MAE:', mean_absolute_error(y_test, y_pred))
print('MSE:', mean_squared_error(y_test, y_pred))
print('RMSE:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('R2 Score:', r2_score(y_test, y_pred))
```

```
MAE: 2980.4106330957943
MSE: 24389106.013968404
RMSE: 4938.532779476957
R2 Score: 0.8230454106843704
```

Polynomial Regression

```
#acutal vs predicted values for polynomial regression
plt.figure(figsize=(7,5))
ax1 = sns.distplot(y_test,hist=False,color='r',label='Actual Value')
sns.distplot(y_pred,hist=False,color='b',label='Predicted
Value',ax=ax1)
plt.title('Actual vs Predicted Values for Polynomial Regression')
plt.xlabel('Medical Expense')
plt.show()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\1377664896.py:3:
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 `kdeplot` (an axes-level function for kernel
density plots).

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

  ax1 = sns.distplot(y_test,hist=False,color='r',label='Actual Value')
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\1377664896.py:4:
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 `kdeplot` (an axes-level function for kernel
density plots).

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

  sns.distplot(y_pred,hist=False,color='b',label='Predicted
Value',ax=ax1)
```

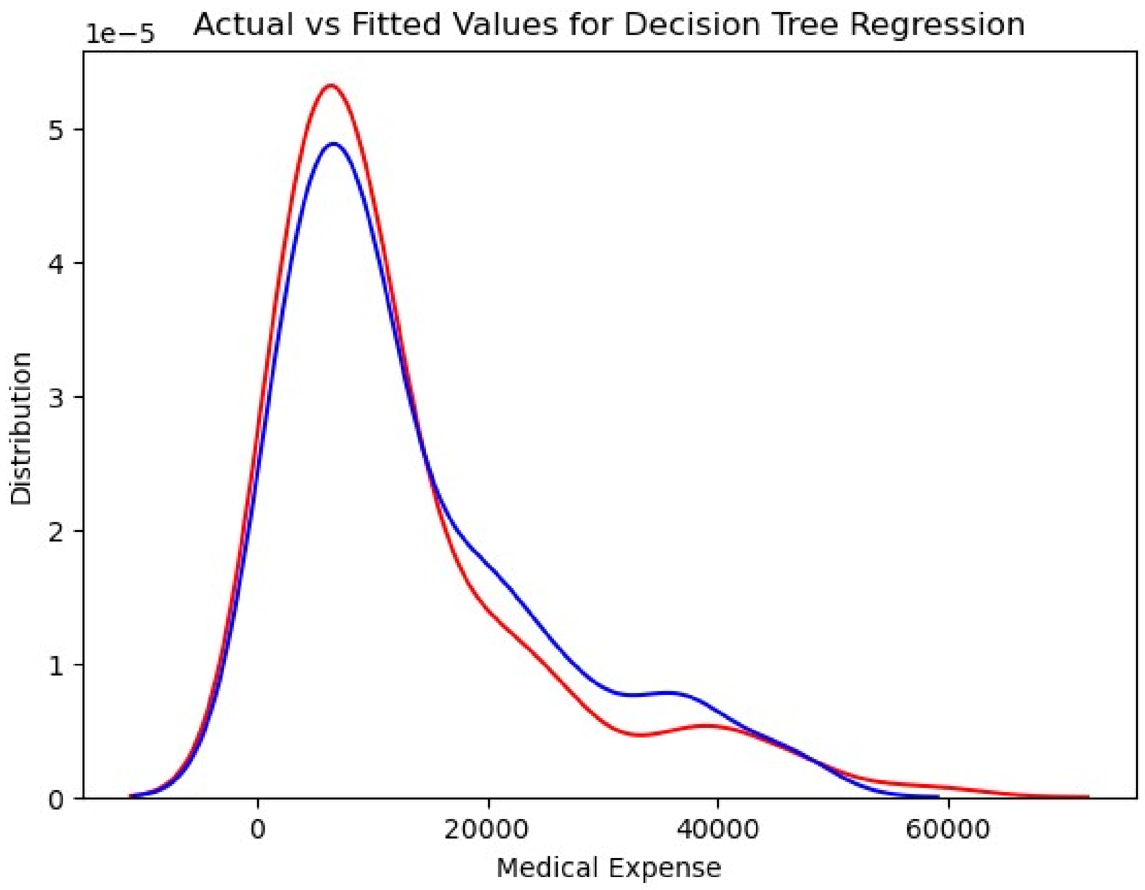


```
print('MAE:', mean_absolute_error(y_test, y_pred))
print('MSE:', mean_squared_error(y_test, y_pred))
print('RMSE:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('R2 Score:', r2_score(y_test, y_pred))

MAE: 2980.4106330957943
MSE: 24389106.013968404
RMSE: 4938.532779476957
R2 Score: 0.8230454106843704
```

Decision Tree Regressor

```
#distribution plot of actual and predicted values
plt.figure(figsize=(7,5))
ax = sns.distplot(y_test, hist=False, color="r", label="Actual Value")
sns.distplot(dtree_pred, hist=False, color="b", label="Fitted
Values" , ax=ax)
plt.title('Actual vs Fitted Values for Decision Tree Regression')
plt.xlabel('Medical Expense')
plt.ylabel('Distribution')
plt.show()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\1470506590.py:3:
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 `kdeplot` (an axes-level function for kernel
density plots).

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

  ax = sns.distplot(y_test, hist=False, color="r", label="Actual
Value")
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\1470506590.py:4:
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 `kdeplot` (an axes-level function for kernel
density plots).

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

  sns.distplot(dtree_pred, hist=False, color="b", label="Fitted
Values" , ax=ax)
```



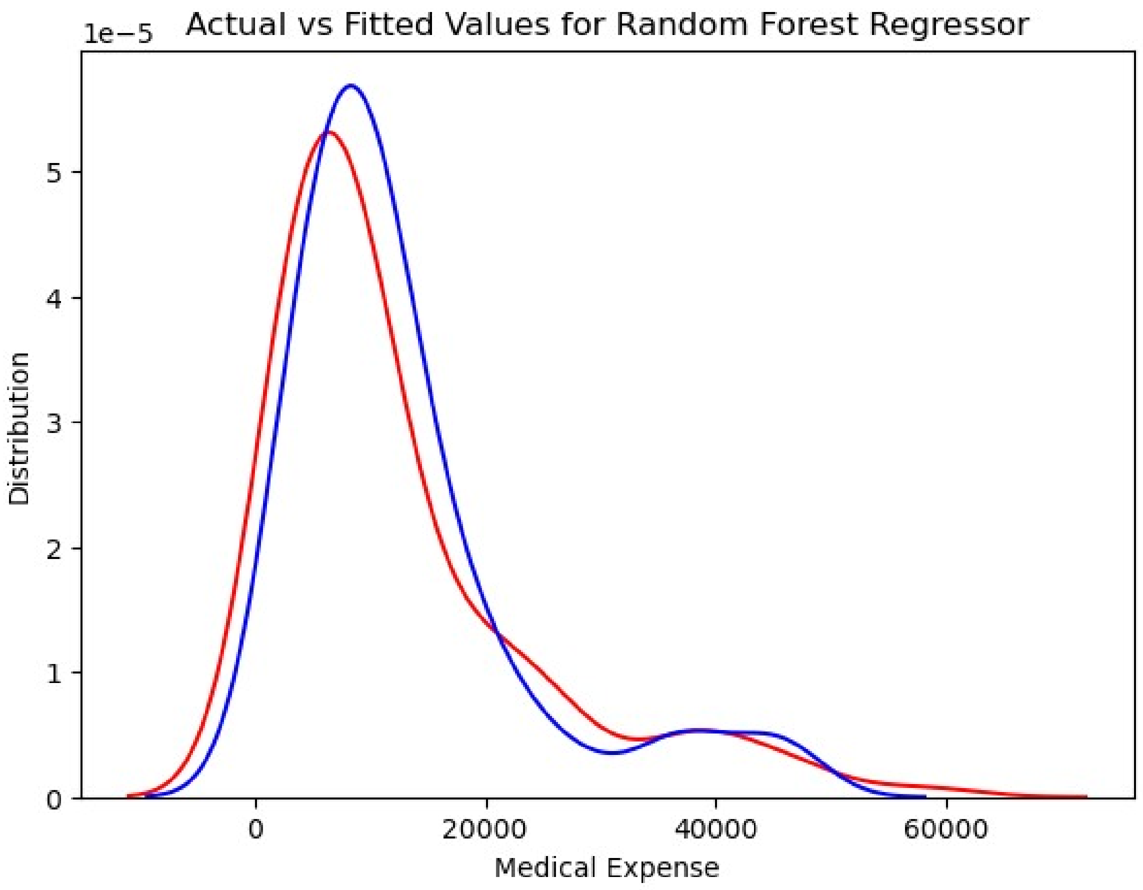
```
print('MAE:', mean_absolute_error(y_test, dtree_pred))
print('MSE:', mean_squared_error(y_test, dtree_pred))
print('RMSE:', np.sqrt(mean_squared_error(y_test, dtree_pred)))
print('Accuracy:', dtree.score(x_test,y_test))
```

```
MAE: 4124.423047570094
MSE: 60986222.048794106
RMSE: 7809.367583152563
Accuracy: 0.5575158896609267
```

Random Forest Regressor

```
#distribution plot of actual and predicted values
plt.figure(figsize=(7,5))
ax = sns.distplot(y_test, hist=False, color="r", label="Actual Value")
sns.distplot(rf_pred, hist=False, color="b", label="Fitted Values" ,
ax=ax)
plt.title('Actual vs Fitted Values for Random Forest Regressor')
plt.xlabel('Medical Expense')
plt.ylabel('Distribution')
plt.show()
```

```
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\4145278179.py:3:
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 `kdeplot` (an axes-level function for kernel
density plots).

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

  ax = sns.distplot(y_test, hist=False, color="r", label="Actual
Value")
C:\Users\admin\AppData\Local\Temp\ipykernel_9176\4145278179.py:4:
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 `kdeplot` (an axes-level function for kernel
density plots).

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

  sns.distplot(rf_pred, hist=False, color="b", label="Fitted Values" ,
ax=ax)
```



```
print('MAE:', mean_absolute_error(y_test, rf_pred))
print('MSE:', mean_squared_error(y_test, rf_pred))
print('RMSE:', np.sqrt(mean_squared_error(y_test, rf_pred)))
print('Accuracy:', rf.score(x_test,y_test))
```

```
MAE: 2973.1179516809593
MSE: 26534975.30772119
RMSE: 5151.211052531355
Accuracy: 0.8074761061193091
```

Conclusion

From the above models, we can see that Decision Tree Regressor and Random Forest Regressor are giving the best results. But, Random Forest Regressor is giving the best results with the least RMSE value. Therefore, I will use Random Forest Regressor to predict the medical expense of patients.

Moreover, the medical expense of smokers is higher than that of non-smokers. The medical expense of patients with BMI greater than 30 is higher than that of patients with BMI less than 30. The medical expense of older patients is higher than that of younger patients.

Thus, from the overall analysis, we can conclude that the medical expense of patients depends on their age, BMI, smoking habits.