Medical Cost Prediction

In [5]:
```
df = pd.read_csv('C:/Users/zizhe/Desktop/Leo Zhao/Medical Cost Prediction/Insurance.csv')
df.head(10)
```

Out[5]:

	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
5	31	female	25.740	0	no	southeast	3756.62160
6	46	female	33.440	1	no	southeast	8240.58960
7	37	female	27.740	3	no	northwest	7281.50560
8	37	male	29.830	2	no	northeast	6406.41070
9	60	female	25.840	0	no	northwest	28923.13692

Data Preprocessing

In [6]:
```
#number of rows and columns
df.shape
```

Out[6]:
```
(1338, 7)
```

In [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
```

In [8]:
```
#checking discriptive statistics
df.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 [12]:
```
#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
```

Replacing the categorical variables with numerical values.

- sex : 1 - male, 0 - female
- smoker : 1 - yes, 0 - no
- region : 0 - northeast, 1 - northwest, 2 - southeast, 3 - southwest

In [13]:
```
#changing categorical variables to numerical
df['Sex'] = df['Sex']({'male':1,'female':0})
df['Smoker'] = df['Smoker']({'yes':1,'no':0})
df['Region'] = df['Region']({'southwest':0,'southeast':1,'northwest':2,'northeast':3})
```

In [14]:
```
df.head(10)
```

Out[14]:

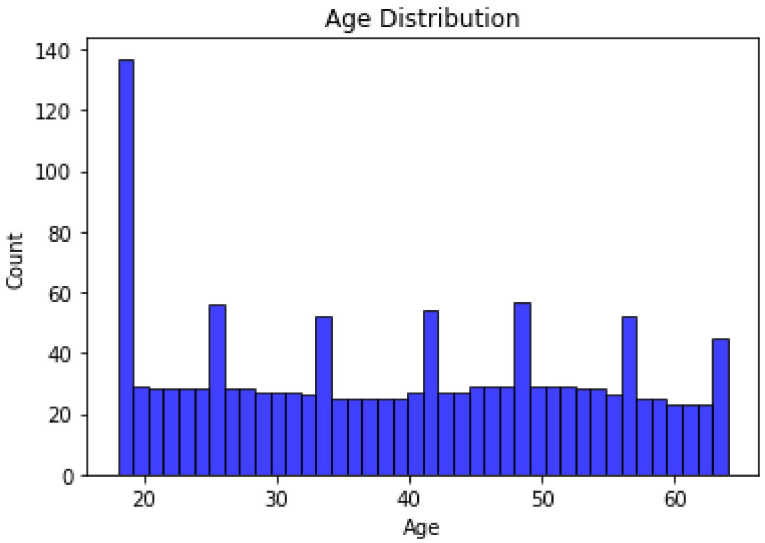
	Age	Sex	BMI	Children	Smoker	Region	Charges
0	19	0	27.900	0	1	0	16884.92400
1	18	1	33.770	1	0	1	1725.55230
2	28	1	33.000	3	0	1	4449.46200
3	33	1	22.705	0	0	2	21984.47061
4	32	1	28.880	0	0	2	3866.85520
5	31	0	25.740	0	0	1	3756.62160
6	46	0	33.440	1	0	1	8240.58960
7	37	0	27.740	3	0	2	7281.50560
8	37	1	29.830	2	0	3	6406.41070
9	60	0	25.840	0	0	2	28923.13692

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.

In [19]:
```
# Age distribution

sns.histplot(df.Age,bins=40,color='blue')
plt.title('Age Distribution')
plt.xlabel('Age')
plt.ylabel('Count')
plt.show()
```

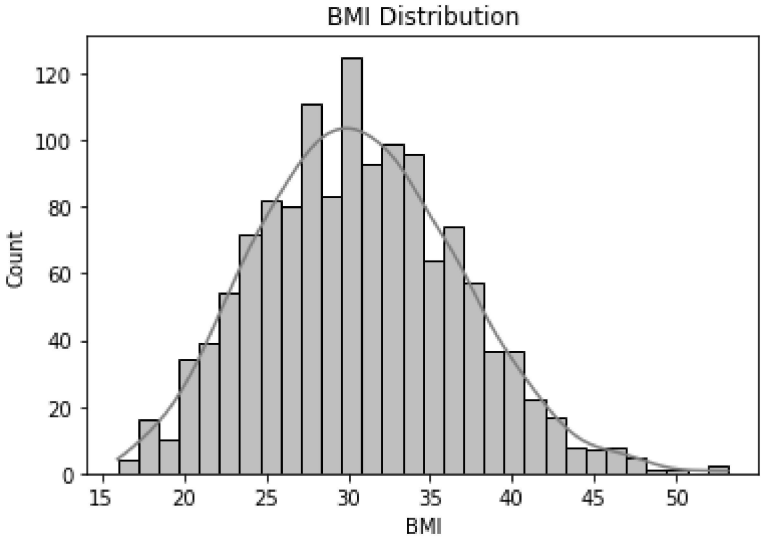


In [20]:
```
# Gender plot
sns.countplot(x ='Sex', data = df)
plt.title('Gender Distribution')
```

Out[20]:
```
Text(0.5, 1.0, 'Gender Distribution')
```

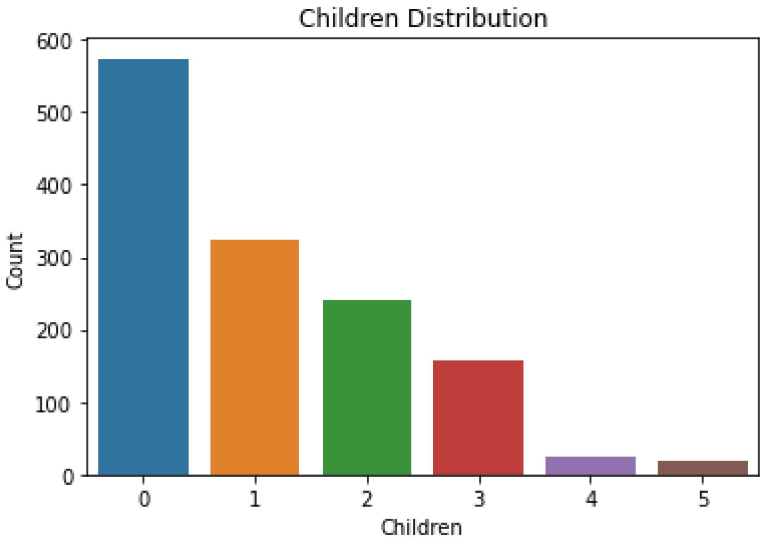
It is clear that number of Males and Females are almost equal in the dataset.

In [22]:
```
# BMI distribution
sns.histplot(df.BMI,bins=30,color='grey')
plt.title('BMI Distribution')
plt.xlabel('BMI')
plt.ylabel('Count')
plt.show()
```



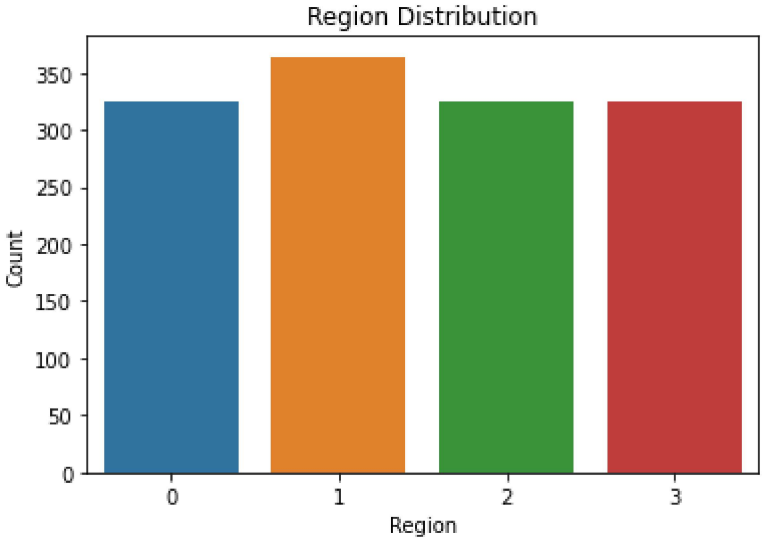
The majority of the patients have BMI between 24 and 38 which is considered as overweight and could be a major factor in increasing the medical cost.

In [23]:
```
#Child count distribution
sns.countplot(x = 'Children')
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.

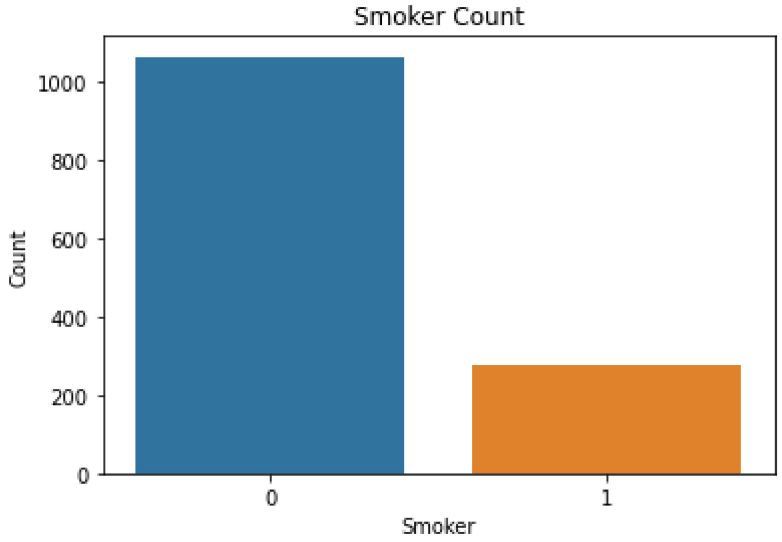
In [24]:
```
#Region plot
sns.countplot(x = 'Region')
plt.title('Region Distribution')
plt.xlabel('Region')
plt.ylabel('Count')
plt.show()
```



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

Region : 0 - Northeast, 1 - Northwest, 2 - Southeast, 3 - Southwest

In [25]:
```
#Count of Smokers
sns.countplot(x = 'Smoker')
plt.title('Smoker Count')
plt.xlabel('Smoker')
plt.ylabel('Count')
plt.show()
```

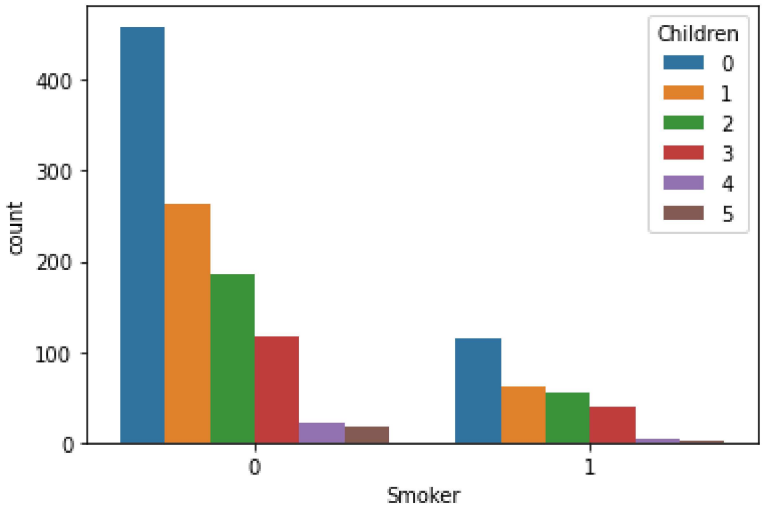


smokers are very few in the dataset. Nearly 80% of the patients are non-smokers.

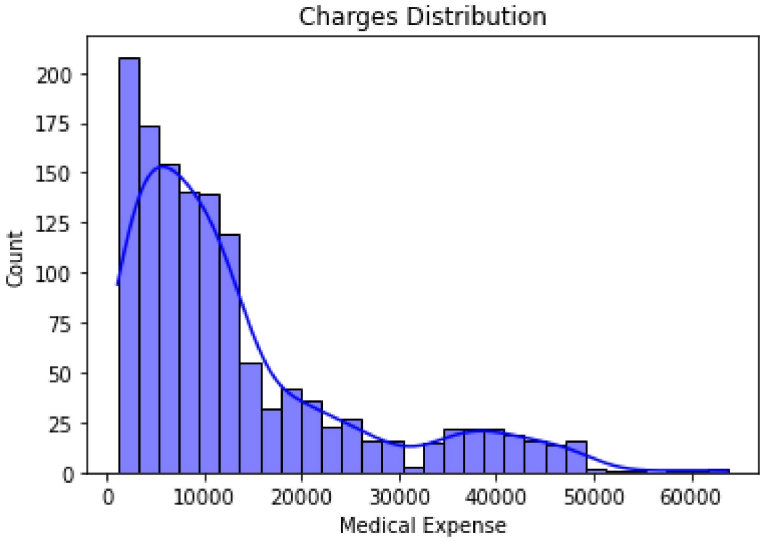
Smoker count with respect to the children count.

In [26]:
```
sns.plot(x = df.Smoker, hue = df.Children)
```

Out[26]:
```
<AxesSubplot:xlabel='Smoker', ylabel='count'>
```



In [27]:
```
#Charges distribution
sns.plot(df.Charges,bins=30,color='BLUE')
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 correlation between the variables.

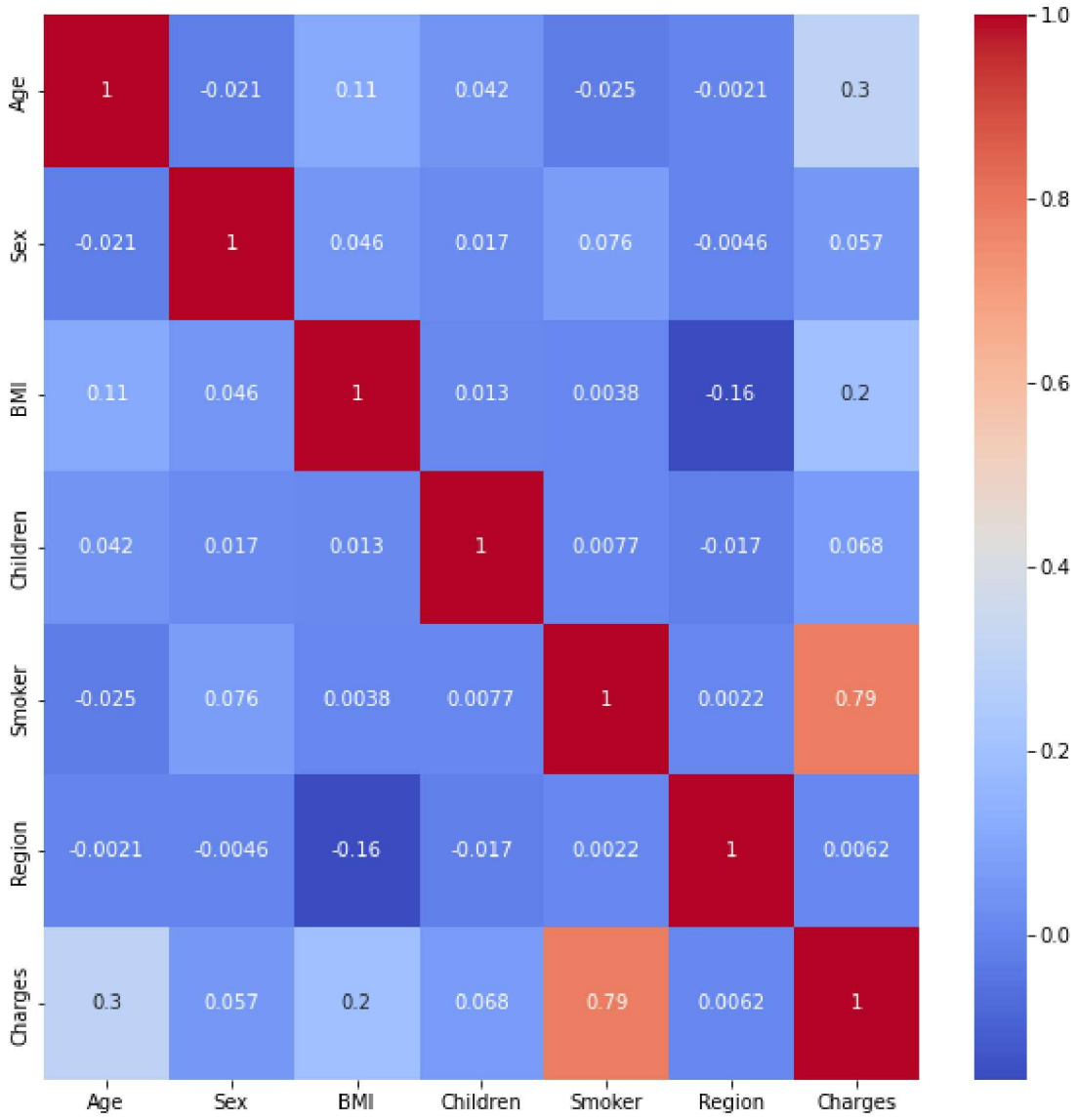
Correlation

In [28]:
```
# Correlation matrix
df.cor()
```

Out[28]:

	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

In [29]:
```
#plotting the correlation heatmap
plt.figure(figsize=(10,10))
sns.heatmap(df.cor(),annot=True,map='coolwarm')
plt.show()
```

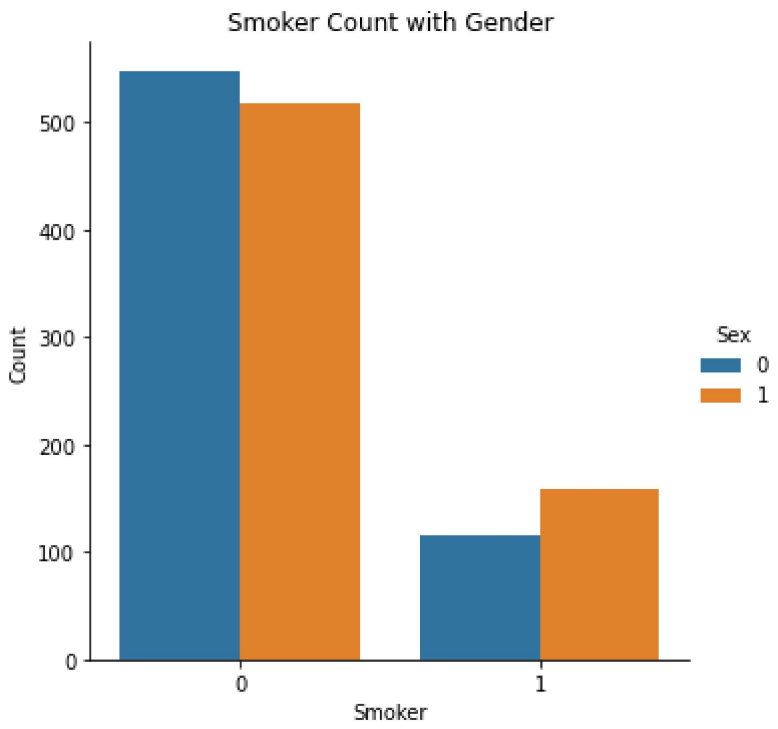


The variable smoker shows a significant correlation with the medical expenses. Now I will explore more into patients' smoking habits and their relations with other factors.

Plotting the smoker count with patient's gender

In [31]:
```
sns.catplot(x="Smoker", kind="count",hue = 'Sex')
plt.title('Smoker Count with Gender')
```

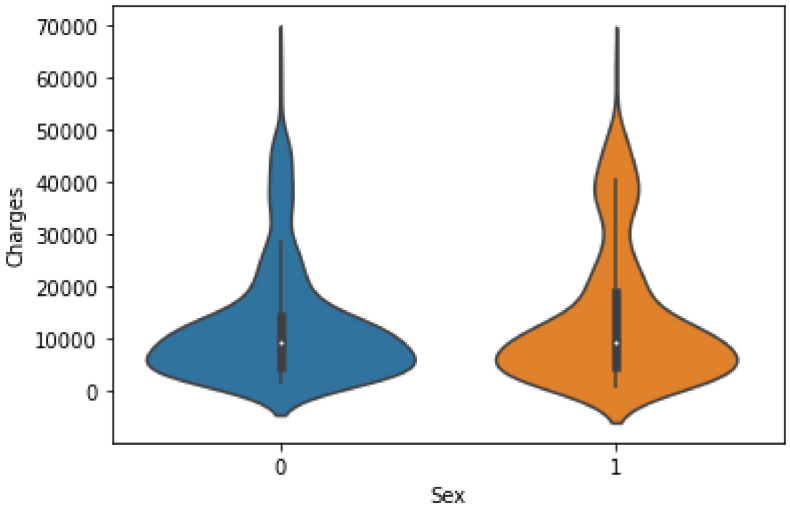
```
plt.xlabel('Smoker')
plt.ylabel('Count')
plt.show()
```



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.

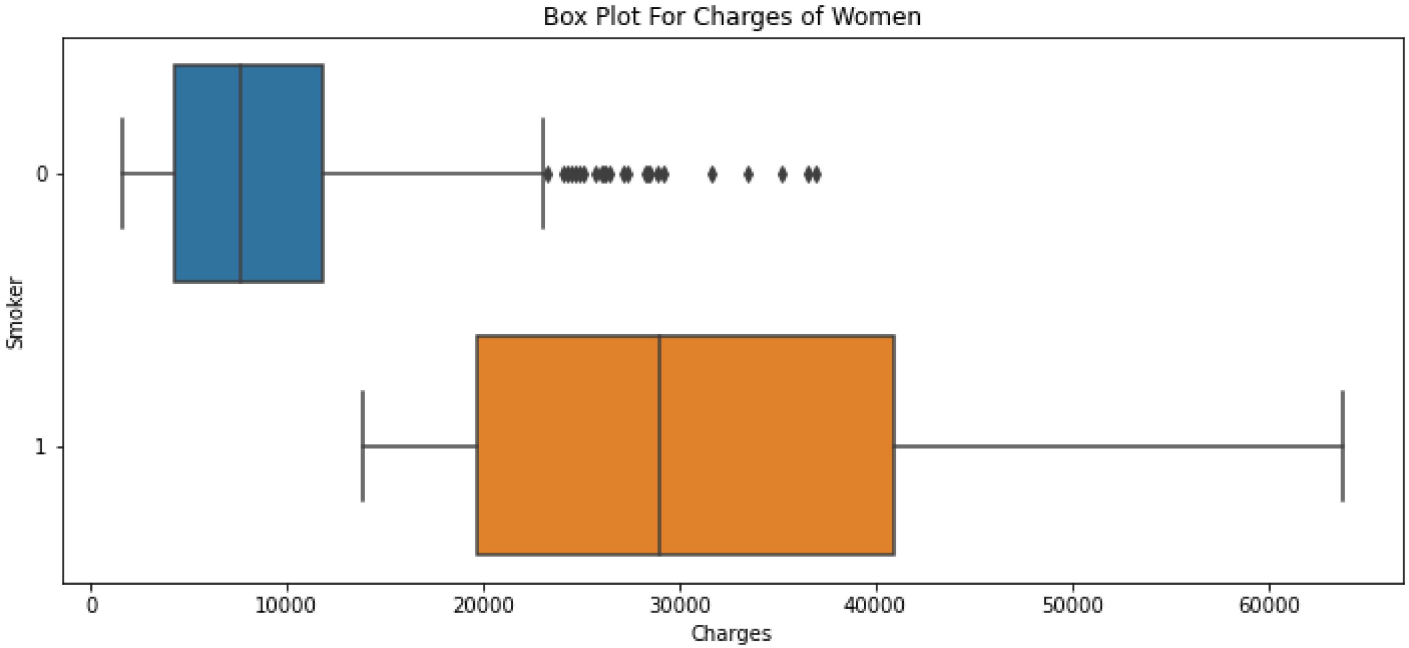
In [32]:
```
sns.vioplot(x = 'Sex', y = 'Charges', data = df)
```

Out[32]:
```
<AxesSubplot:xlabel='Sex', ylabel='Charges'>
```



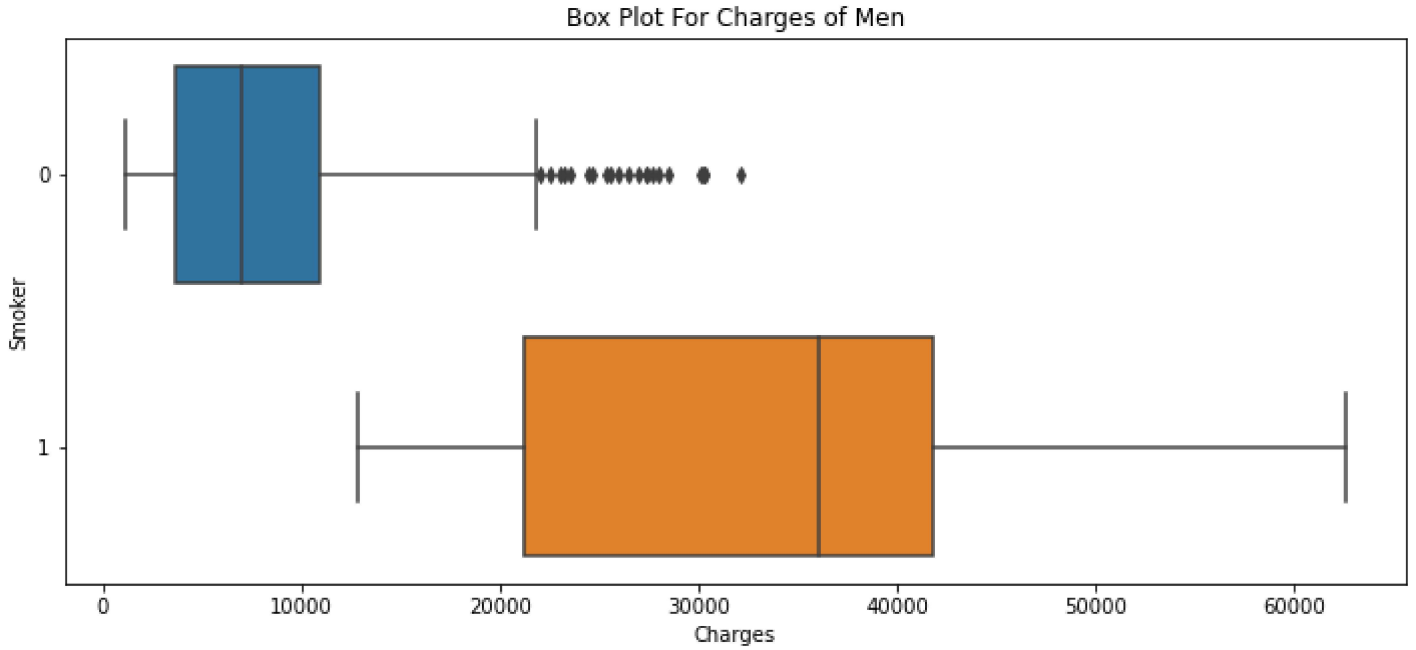
In [34]:
```
plt.figure(figsize=(12,5))
plt.title("Box Plot For Charges of Women")
sns.boxplot(y="Smoker", x="Charges", data =  df[(Sex == 0)] , rent="h")
```

Out[34]:
```
<AxesSubplot:title={'center':'Box Plot For Charges of Women'}, xlabel='Charges', ylabel='Smoker'>
```



In [35]:
```
plt.figure(figsize=(12,5))
plt.title("Box Plot For Charges of Men")
sns.plot(y="Smoker", x="Charges", data =  df[(df.Sex == 1)] , rent="h")
```

Out[35]:
```
<AxesSubplot:title={'center':'Box Plot For Charges of Men'}, xlabel='Charges', ylabel='Smoker'>
```



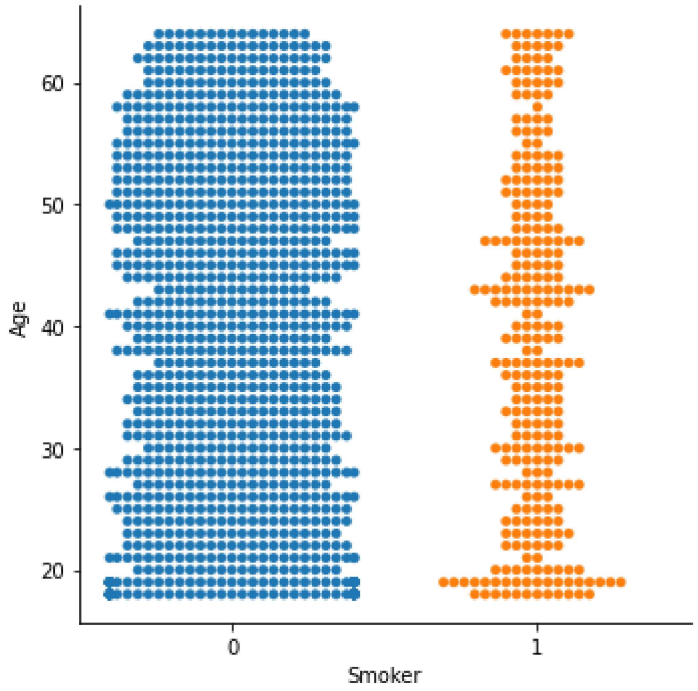
The assumption that the medical expenses of males are greater than those of females is true. Additionally, the medical expenses of smokers are greater than those of non-smokers.

Smokers and age distribution

In [36]:
```
#smokers and age distribution
sns.catplot(x="Smoker", y="Age", data=df)
```

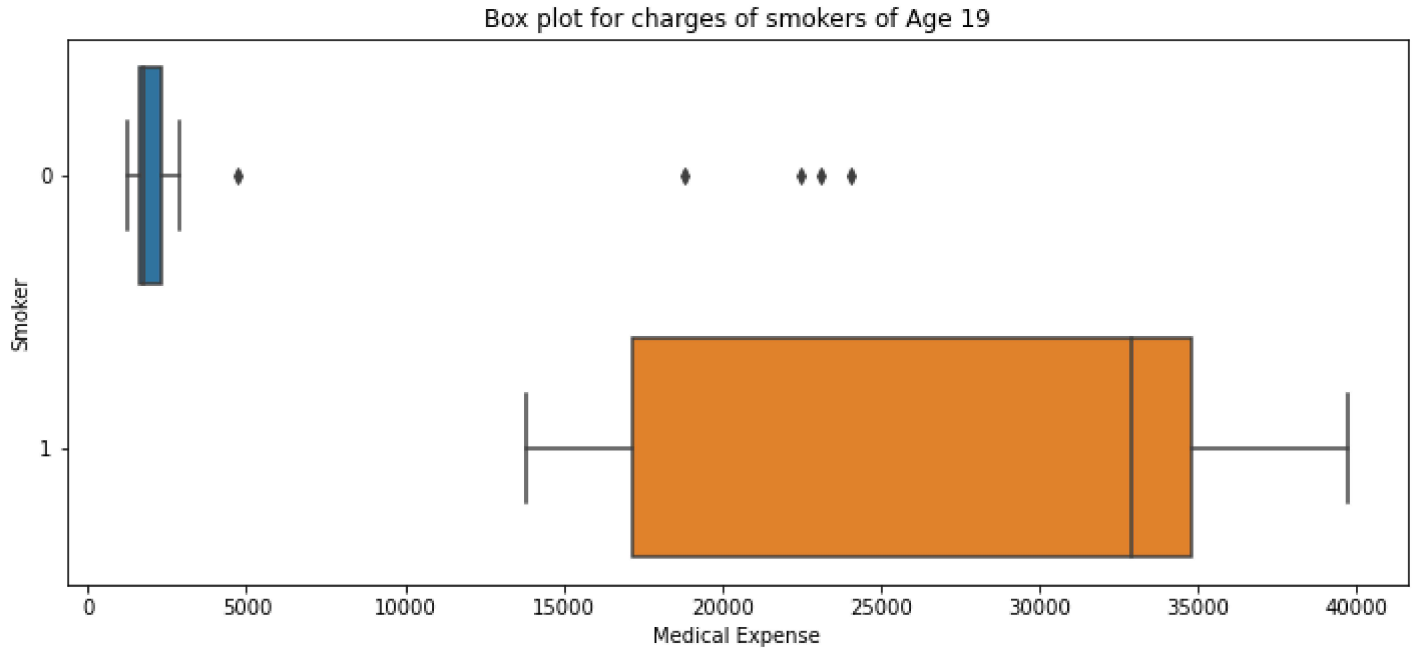
```
C:\Users\zizhe\New folder\lib\site-packages\seaborn\categorical.py:1296: UserWarning: 7.3% of the points cannot be plac
ed; you may want to decrease the size of the markers or use stripplot.
  warnings.warn(msg, UserWarning)
```

Out[36]:
```
<seaborn.axisgrid.FacetGrid at 0x1a3813c0d90>
```



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.

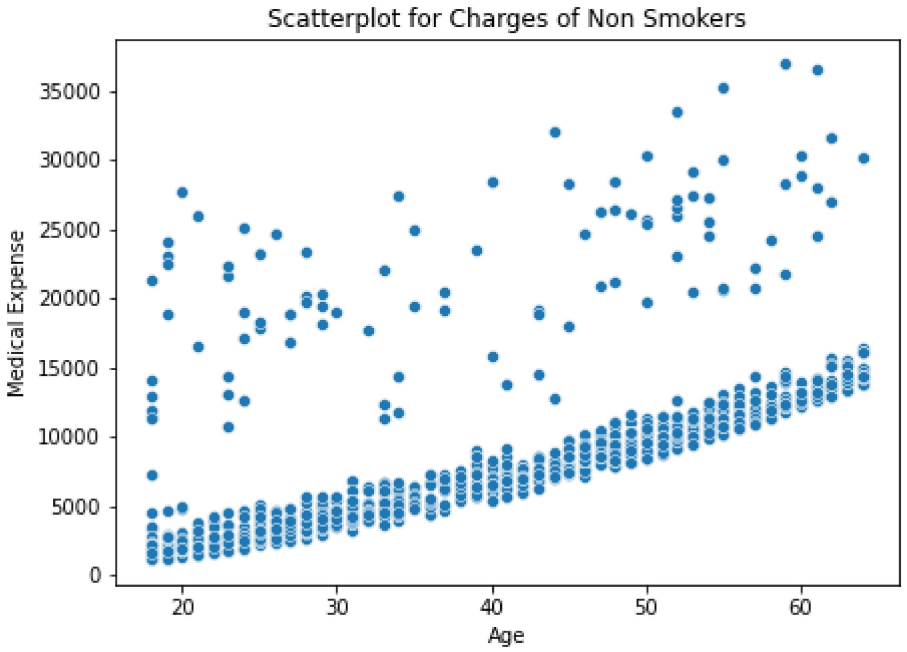
In [38]:
```
#smokers of age 19
plt.figure(figsize=(12,5))
plt.title("Box plot for charges of smokers of Age 19")
sns.boxplot(y="Smoker", x="Charges", data =  df[(df.Age == 19)])
plt.xlabel('Medical Expense')
plt.ylabel('Smoker')
plt.show()
```



Surprisingly, the medical expenses of smokers at the age of 19 are very high in comparison to non-smokers. Among non-smokers, we can observe some outliers, which may be due to illness or accidents.

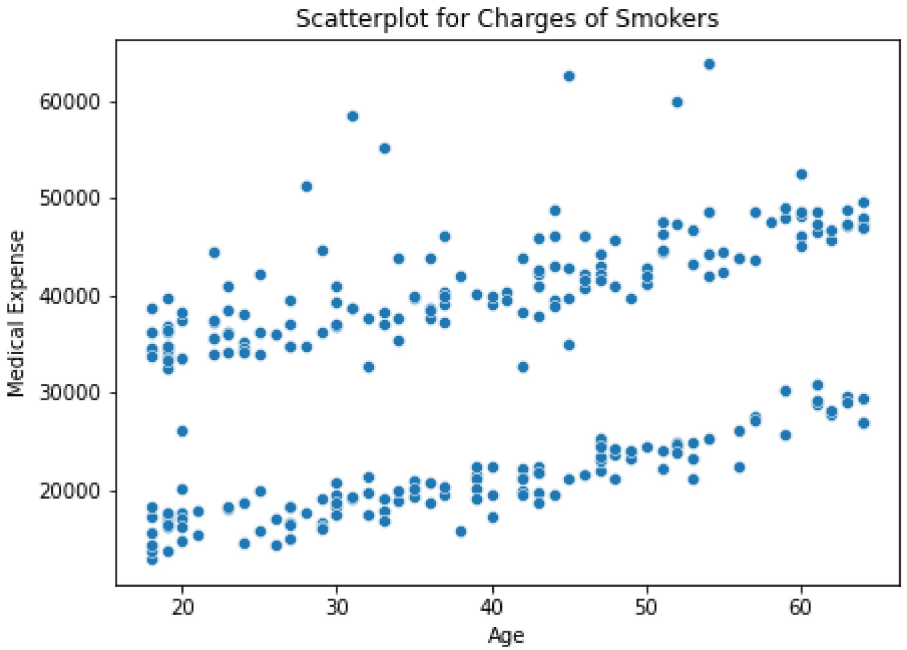
It is clear that the medical expenses of smokers are higher than those of non-smokers. Now, I will plot the distribution of charges with respect to the patients' age for both smokers and non-smokers.

In [39]:
```
#non smokers charge distribution
plt.figure(figsize=(7,5))
plt.title("Scatterplot for Charges of Non Smokers")
sns.scatterplot(x="Age", y="Charges")
plt.xlabel('Age')
plt.ylabel('Medical Expense')
plt.show()
```



The majority of the data points show that medical expenses increase with age, which may be due to the fact that older people are more prone to illness. However, there are some outliers that suggest other illnesses or accidents could be contributing to the increased medical expenses.

In [40]:
```
#smokers charge distribution
plt.figure(figsize=(7,5))
plt.title("Scatterplot for Charges of Smokers")
sns.scatterplot(x="Age", y="Charges")
plt.xlabel('Age')
plt.ylabel('Medical Expense')
plt.show()
```

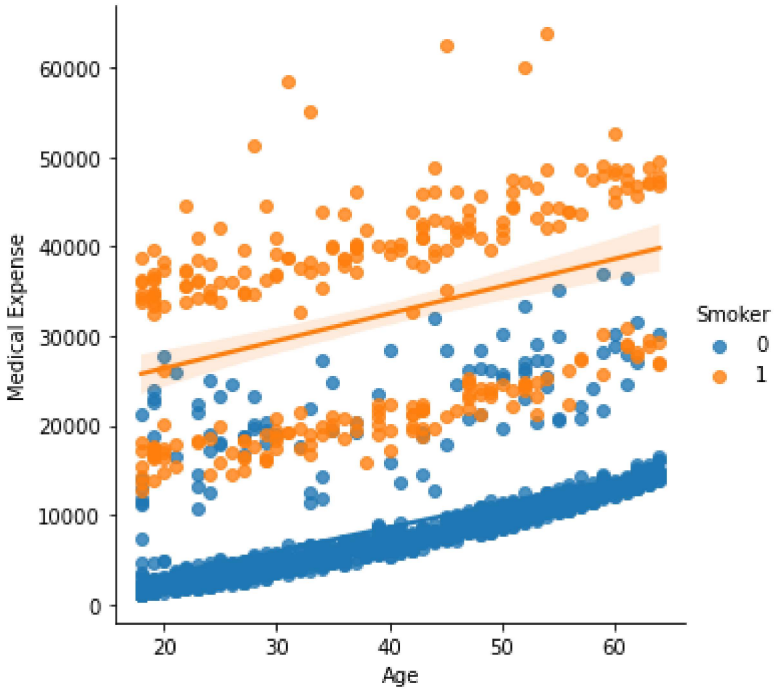


Here, we observe a peculiarity in the graph. The graph comprises two segments: one with high medical expenses, which may be attributed to smoking-related illnesses, and the other with low medical expenses, which may be attributed to age-related illnesses.

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

In [41]:
```
# Age charges distribution

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

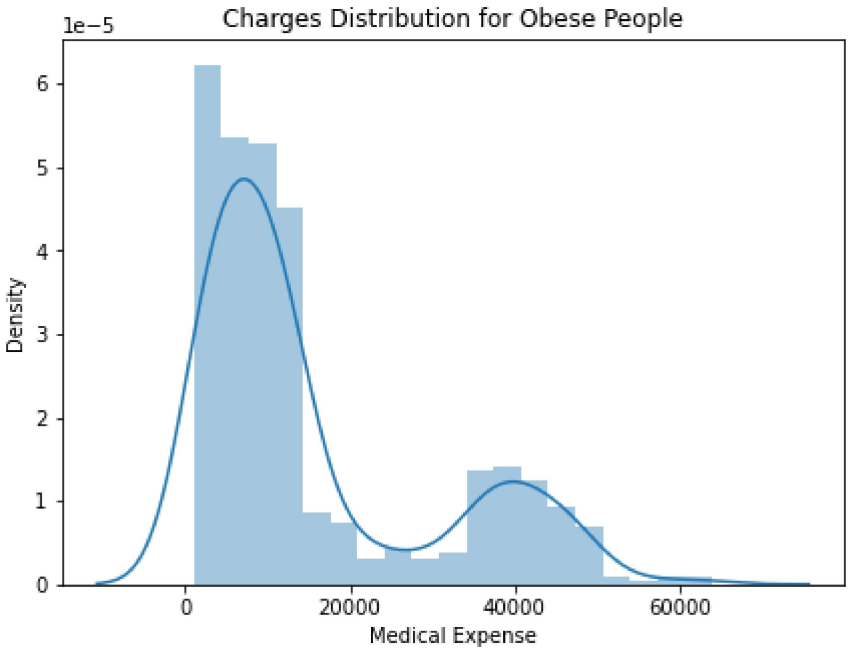


Now, we have a clear understanding of the variation in charges with respect to age and smoking habits. The medical expenses of smokers are higher than those of non-smokers. In non-smokers, the cost of treatment increases with age, which is expected. However, in smokers, the cost of treatment is high even for younger patients, indicating that smoking patients are incurring expenses for both smoking-related and age-related illnesses."

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

In [42]:
```
# 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()
```

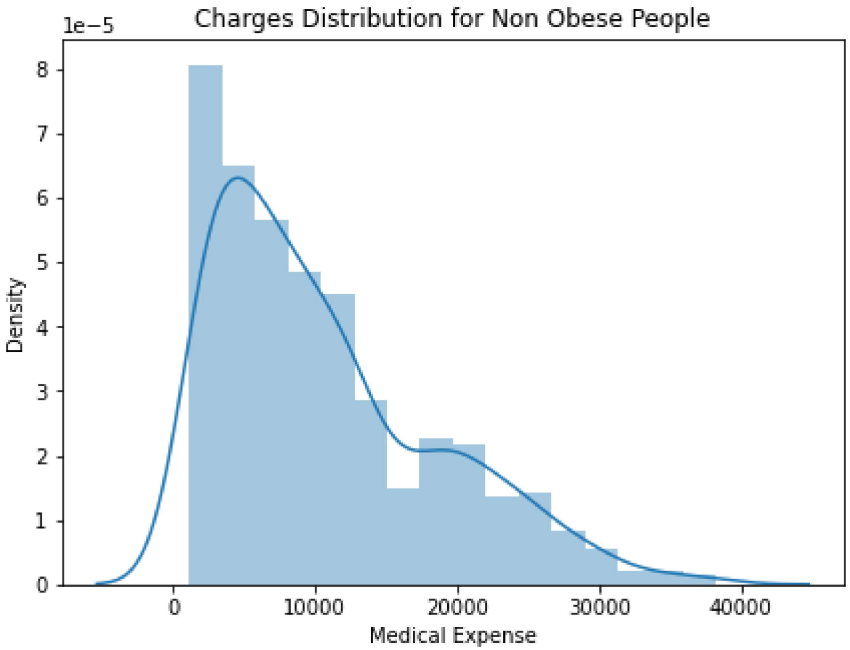
```
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. Please adapt your code to use either `displot` (a figure-level function
with similar flexibility) or `histplot` (an axes-level function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



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

In [43]:
```
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\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction 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)
```



Patients with a BMI less than 30 spend less on medical treatment than those with a BMI greater than 30.

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

Train Test Split

In [44]:
```
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_sta
```

Model Building

Linear Regression

In [45]:
```
#Linear Regression
from sklearn.model import LinearRegression
lr = LinearRegression()
lr
```

Out[45]:
```
LinearRegression()
```

In [46]:
```
#model training
lr.fit(x_train,y_train)
#model accuracy
lr.score(x_train,y_train)
```

Out[46]:
```
0.7368306228430945
```

In [47]:
```
#model prediction
y_pred = lr.predict(x_test)
```

Polynomial Regression

In [48]:
```
from sklearn.preprocessing import PolynomialFeatures
poly_reg = PolynomialFeatures(degree=3)
poly_reg
```

Out[48]:
```
PolynomialFeatures()
```

In [49]:
```
#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)
```

In [50]:
```
plr = LinearRegression()
#model training
plr.fit(x_train,y_train)
#model accuracy
plr.score(x_train,y_train)
```

Out[50]:
```
0.8372892318154602
```

In [51]:
```
#model prediction
y_pred = plr.predict(x_test)
```

Decision Tree Classifier

In [52]:
```
#decision tree regressor
from sklearn.tree import DecisionTreeRegressor
```

```
dtree = DecisionTreeRegressor()
dtree
```

Out[52]:
```
DecisionTreeRegressor()
```

In [53]:
```
#model training
dtree.fit(x_train,y_train)
#model accuracy
dtree.score(x_train,y_train)
```

Out[53]:
```
0.9993688476658964
```

In [54]:
```
#model prediction
dtree_pred = dtree.predict(x_test)
```

Random Forest Classifier

In [55]:
```
#random forest regressor
from sklearn.ensemble import RandomForestRegressor
rf = RandomForestRegressor(n_estimators=100)
rf
```

Out[55]:
```
RandomForestRegressor()
```

In [56]:
```
#model training
rf.fit(x_train,y_train)
#model accuracy
rf.score(x_train,y_train)
```

Out[56]:
```
0.9751126976784187
```

In [57]:
```
#model prediction
rf_pred = rf.predict(x_test)
```

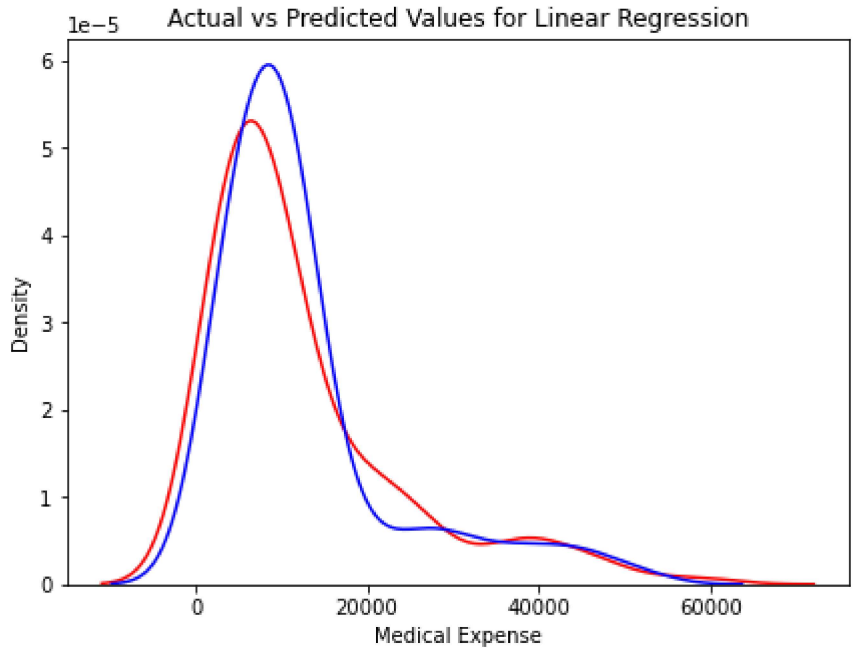
Model Evaluation

In [58]:
```
from sklearn.metrics import mean_squared_error,mean_absolute_error,r2_score
```

Linear Regression

In [59]:
```
#distribution of actual and predicted values
plt.figure(figsize=(7,5))
ax1 = sns.distplot(y_test,color='r',label='Actual Value')
sns.distplot(y_pred,hist=False,color='b',label='Predicted Value')
plt.title('Actual vs Predicted Values for Linear Regression')
plt.xlabel('Medical Expense')
plt.show()
```

```
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. 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).
  warnings.warn(msg, FutureWarning)
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. 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).
  warnings.warn(msg, FutureWarning)
```



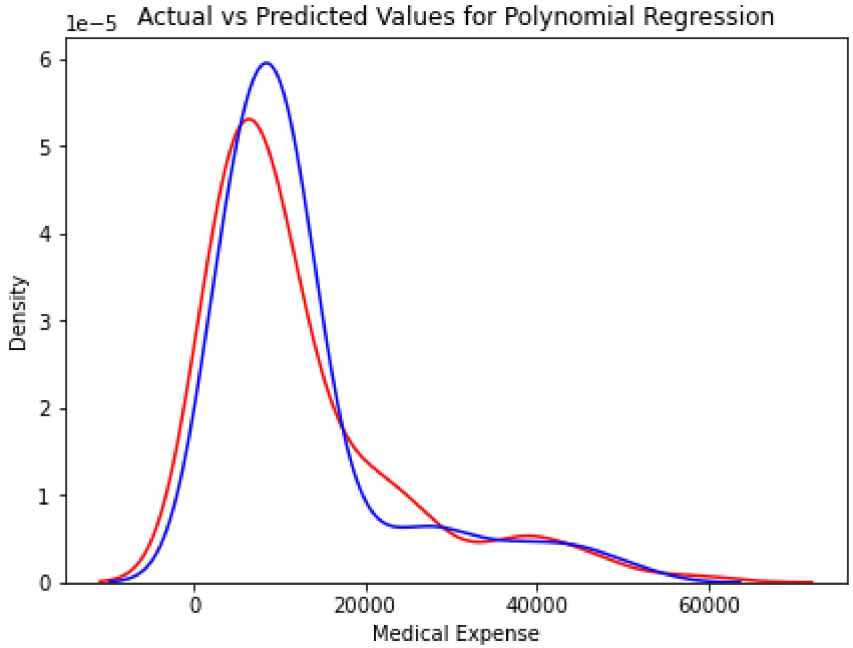
In [60]:
```
print('MAE:', mean_absolute_error(y_test, y_pred))
print('MSE:', mean_squared_error(y_test, y_pred))
print('RMSE:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('R2 Score:', r2_score(y_test, y_pred))
```

```
MAE: 2988.6210423247667
MSE: 24512839.635914266
RMSE: 4951.044297510805
R2 Score: 0.8221476642789292
```

Polynomial Regression

In [61]:
```
#acutal vs predicted values for polynomial regression
plt.figure(figsize=(7,5))
ax1 = sns.distplot(y_test,color='r',label='Actual Value')
sns.plot(y_pred,hist=False,color='b',label='Predicted Value')
plt.title('Actual vs Predicted Values for Polynomial Regression')
plt.xlabel('Medical Expense')
plt.show()
```

```
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. 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).
  warnings.warn(msg, FutureWarning)
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. 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).
  warnings.warn(msg, FutureWarning)
```



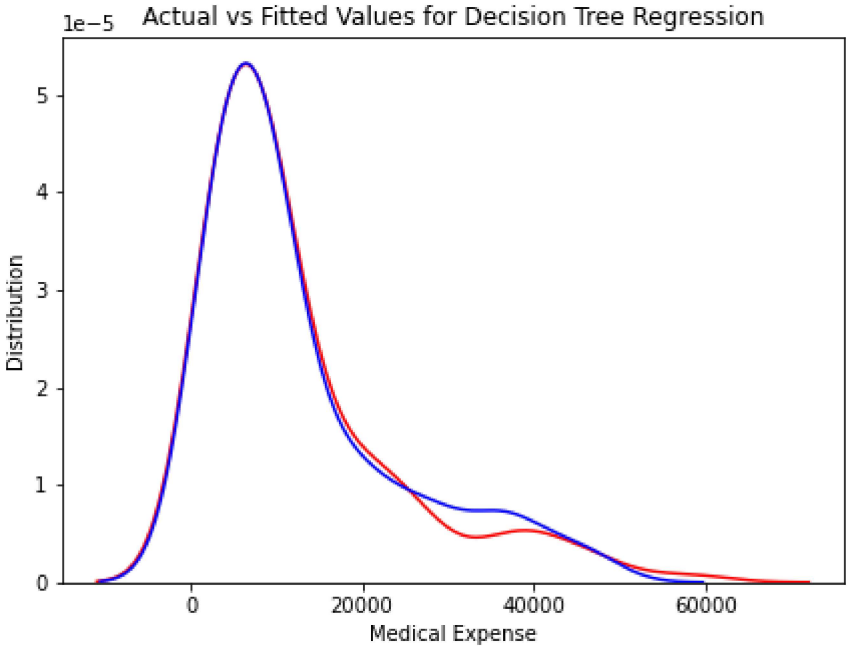
In [62]:
```
print('MAE:', mean_absolute_error(y_test, y_pred))
print('MSE:', mean_squared_error(y_test, y_pred))
print('RMSE:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('R2 Score:', r2_score(y_test, y_pred))
```

```
MAE: 2988.6210423247667
MSE: 24512839.635914266
RMSE: 4951.044297510805
R2 Score: 0.8221476642789292
```

Decision Tree Classifier

In [63]:
```
#distribution plot of actual and predicted values
plt.figure(figsize=(7,5))
ax = sns.plot(y_test, hist=False, color="r", label="Actual Value")
sns.distplot(dtree_pred, hist=False, color="b", label="Fitted Values")
plt.title('Actual vs Fitted Values for Decision Tree Regression')
plt.xlabel('Medical Expense')
plt.ylabel('Distribution')
plt.show()
```

```
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. 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).
  warnings.warn(msg, FutureWarning)
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. 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).
  warnings.warn(msg, FutureWarning)
```



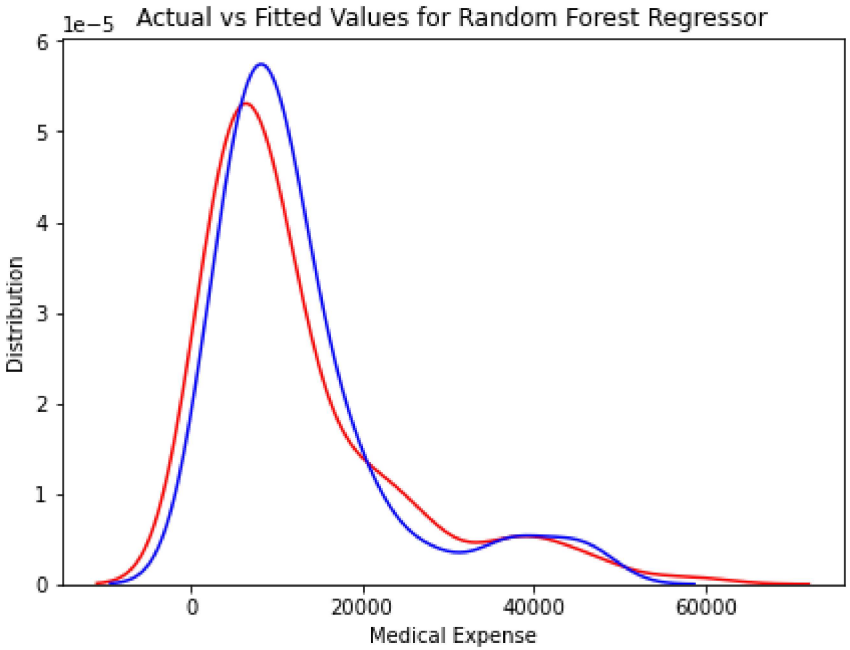
In [64]:
```
print('MAE:', mean_absolute_error(y_test, dtree_pred))
print('MSE:', mean_squared_error(y_test, dtree_pred))
print('RMSE:', np.sqrt(mean_squared_error(y_test, dtree_pred)))
print('Accuracy:', dtree.score(x_test,y_test))
```

```
MAE: 3303.3472760747663
MSE: 48673497.51782195
RMSE: 6976.639414347136
Accuracy: 0.6468505750522974
```

Random Forest Classifier

In [65]:
```
#distribution plot of actual and predicted values
plt.figure(figsize=(7,5))
ax = sns.distplot(y_test, color="r", label="Actual Value")
sns.distplot(rf_pred, hist=False, color="b", label="Fitted Values")
plt.title('Actual vs Fitted Values for Random Forest Regressor')
plt.xlabel('Medical Expense')
plt.ylabel('Distribution')
plt.show()
```

```
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. 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).
  warnings.warn(msg, FutureWarning)
C:\Users\zizhe\New folder\lib\site-packages\seaborn\distributions.py:2619: FutureWarning: `distplot` is a deprecated fu
nction and will be removed in a future version. 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).
  warnings.warn(msg, FutureWarning)
```



In [66]:
```
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: 2879.893381275039
MSE: 26494656.274785776
RMSE: 5147.296015850048
Accuracy: 0.8077686399215158
```

Conclusion

From the models mentioned above, it is evident that the Decision Tree Classifier and Random Forest Classifier yield the best results. However, the Random Forest Classifier outperforms with the lowest RMSE value. Therefore, I will employ the Random Forest Classifier to predict patients' medical expenses.

Furthermore, it is notable that smokers tend to have higher medical expenses compared to non-smokers. Patients with a BMI greater than 30 also exhibit higher medical expenses in comparison to those with a BMI less than 30. Additionally, older patients tend to have higher medical expenses than their younger counterparts.

In summary, our comprehensive analysis suggests that medical expenses are influenced by factors such as age, BMI, and smoking habits.

In [ ]: