In [1]:

```
import numpy as np,pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import seaborn as sb
```

In [2]:

```
df=pd.read_csv(r"C:\Users\91950\Downloads\insurance.csv")
df
```

Out[2]:

	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 [3]:

```
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 [4]:

```
df.shape
```

Out[4]:

```
(1338, 7)
```

In [5]:

```
df.describe()
```

Out[5]:

	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 [6]:

```
df.isna().any()
```

Out[6]:

```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [7]:

```
df.head()
```

Out[7]:

	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 [8]:

```
df.tail()
```

Out[8]:

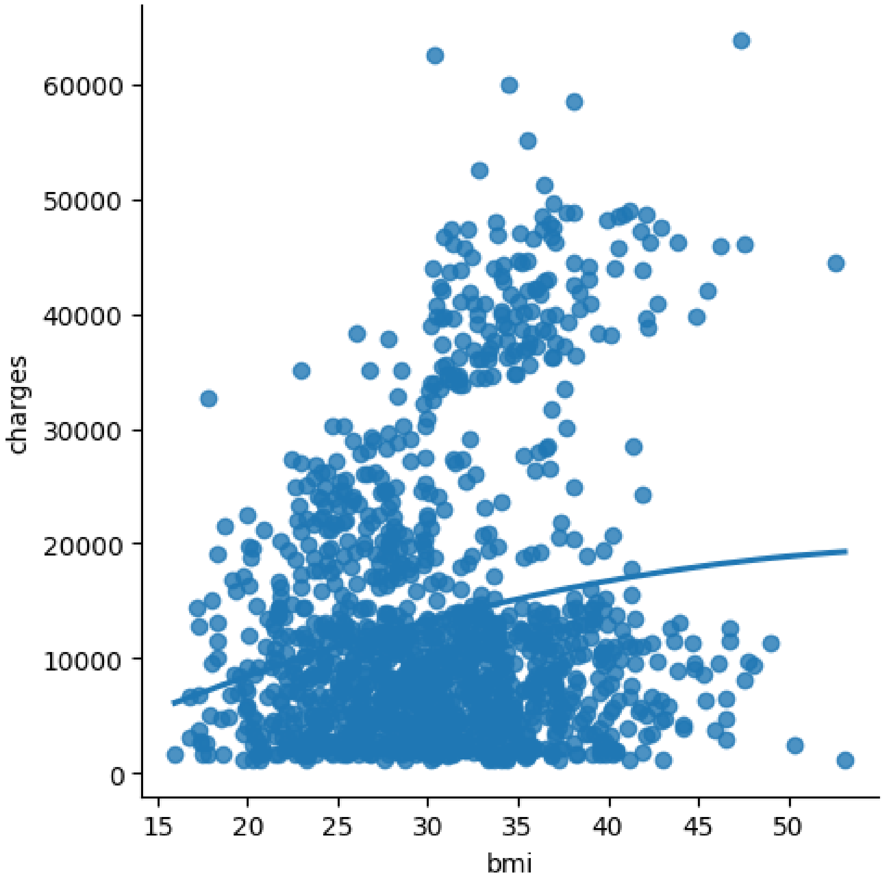
	age	sex	bmi	children	smoker	region	charges
1333	50	male	30.97	3	no	northwest	10600.5483
1334	18	female	31.92	0	no	northeast	2205.9808
1335	18	female	36.85	0	no	southeast	1629.8335
1336	21	female	25.80	0	no	southwest	2007.9450
1337	61	female	29.07	0	yes	northwest	29141.3603

In [9]:

```
sb.lmplot(x="bmi",y="charges",data=df,order=2,ci=None)
```

Out[9]:

```
<seaborn.axisgrid.FacetGrid at 0x1b0007e2b50>
```



In [10]:

```
df.drop("charges",axis=1)
```

Out[10]:

	age	sex	bmi	children	smoker	region
0	19	female	27.900	0	yes	southwest
1	18	male	33.770	1	no	southeast
2	28	male	33.000	3	no	southeast
3	33	male	22.705	0	no	northwest
4	32	male	28.880	0	no	northwest
...	...	...	...	...	...	...
1333	50	male	30.970	3	no	northwest
1334	18	female	31.920	0	no	northeast
1335	18	female	36.850	0	no	southeast
1336	21	female	25.800	0	no	southwest
1337	61	female	29.070	0	yes	northwest

1338 rows × 6 columns

In [11]:

```
sex={"sex":{"female":0,"male":1}}
df=df.replace(sex)
df
```

Out[11]:

	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
...	...	...	...	...	...	...	...
1333	50	1	30.970	3	no	northwest	10600.54830
1334	18	0	31.920	0	no	northeast	2205.98080
1335	18	0	36.850	0	no	southeast	1629.83350
1336	21	0	25.800	0	no	southwest	2007.94500
1337	61	0	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [12]:

```
smoker={"smoker":{"yes":1,"no":0}}
df=df.replace(smoker)
df
```

Out[12]:

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

1338 rows × 7 columns

In [13]:

```
features=df.columns[0:5]
target=df.columns[-1]
```

In [14]:

```
x=df[features].values
y=df[target].values
```

In [15]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.5)
```

In [16]:

```
a=LinearRegression()
a.fit(x_train,y_train)
```

Out[16]:

```
▾ LinearRegression
LinearRegression()
```

In [17]:

```
print(a.score(x_test,y_test))
```

```
0.7482473945814735
```

In [18]:

```
a=LinearRegression()
a.fit(x_train,y_train)
train_score_a=a.score(x_train,y_train)
test_score_a=a.score(x_test,y_test)
print("\nLinearModel\n")
print("The train score for lr model is {}".format(train_score_a))
print("The train score for lr model is {}".format(test_score_a))
```

```

LinearModel

The train score for lr model is 0.75018912067726
The train score for lr model is 0.7482473945814735
```

ridge and lasso Regression

In [19]:

```
from sklearn.linear_model import Ridge, RidgeCV, Lasso
```

In [20]:

```
ridge=Ridge(alpha=2)
ridge.fit(x_train,y_train)
train_score_ridge=ridge.score(x_train,y_train)
test_score_ridge=ridge.score(x_test,y_test)
print("\nLinearRegression\n")
print(train_score_ridge)
print(test_score_ridge)
```

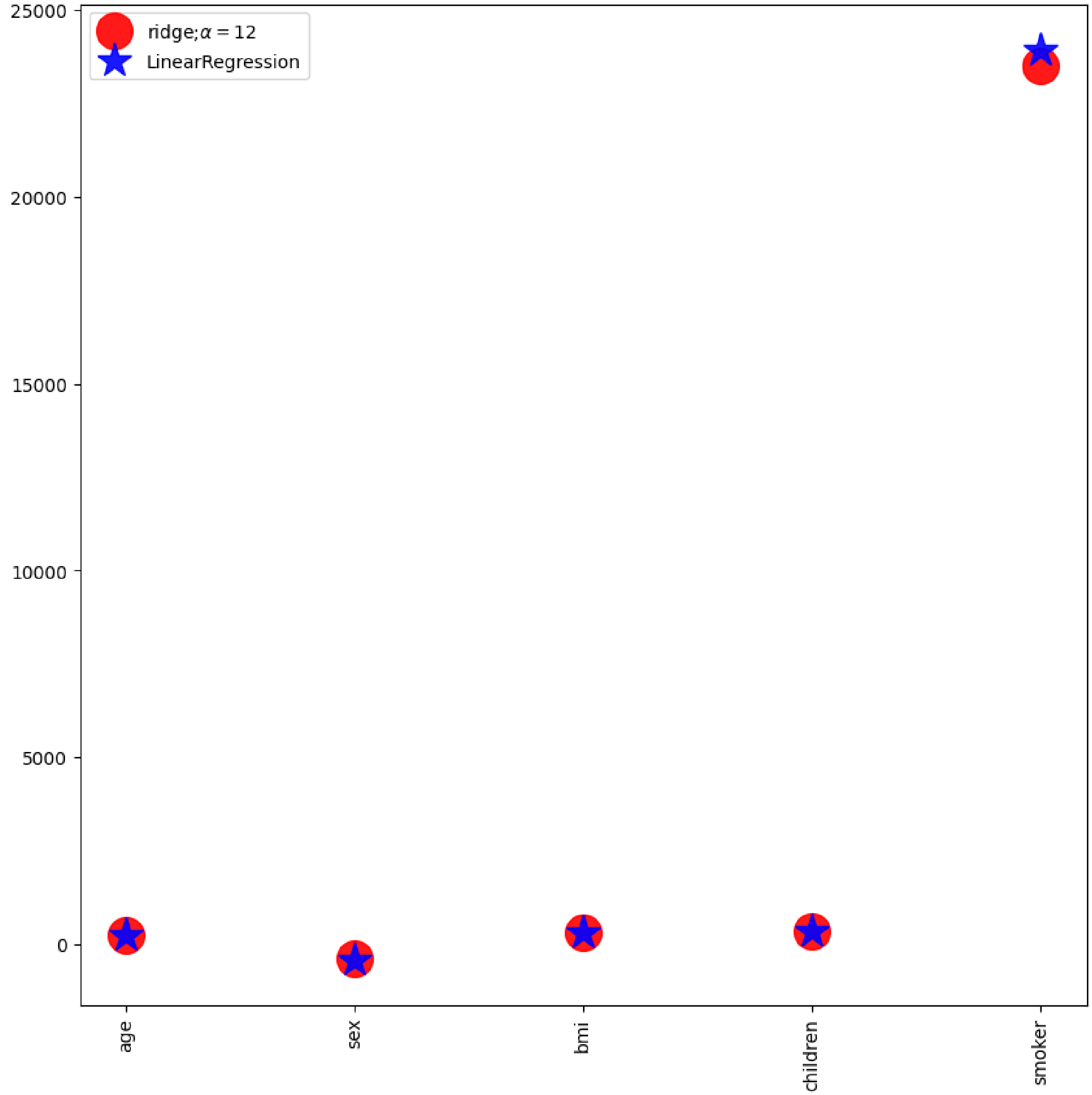
```

LinearRegression

0.7499908458339131
0.7482219743820813
```

In [21]:

```
plt.figure(figsize=(10,10))
plt.plot(features,ridge.coef_,alpha=0.9,linestyle="None",markersize=20,color="Red",label=r"ridge;$
plt.plot(features,a.coef_,alpha=0.9,linestyle="None",markersize=20,color="blue",label="LinearRegre
plt.xticks(rotation=90)
plt.legend()
plt.show()
5
6
```



Out[21]:

```
6
```

Lasso

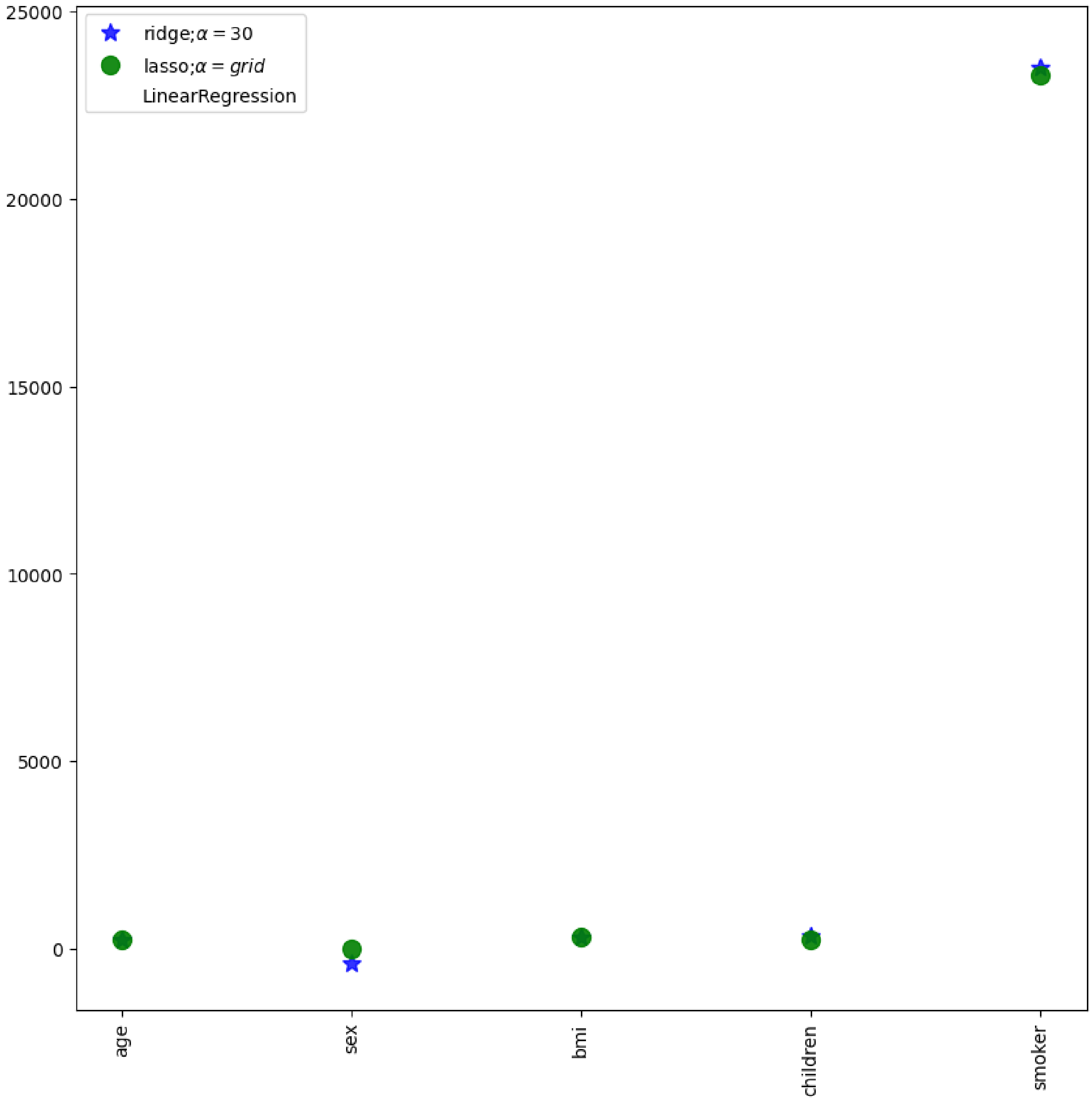
In [22]:

```
lasso=Lasso(alpha=100)
lasso=lasso.fit(x_train,y_train)
train_score_lasso=lasso.score(x_train,y_train)
test_score_lasso=lasso.score(x_test,y_test)
print(train_score_lasso)
print(test_score_lasso)
```

```
0.749484667839464
0.748096359553873
```

In [23]:

```
plt.figure(figsize=(10,10))
plt.plot(features,ridge.coef_,alpha=0.8,marker="*",markersize=10,linestyle="none",color="blue",lab
plt.plot(lasso.coef_,alpha=0.9,marker='o',markersize=10,linestyle="none",color="green",label=r"las
plt.plot(features,a.coef_,alpha=0.7,linestyle="None",markersize=5,color="blue",label=r"LinearRegre
plt.xticks(rotation=90)
plt.legend()
plt.show()
```



In [24]:

```
from sklearn.linear_model import ElasticNet
```

In [25]:

```
a=ElasticNet()
a.fit(x,y)
print(a.coef_)
print(a.intercept_)
```

```
[ 244.74498193  323.34788404  324.21935152  389.31828171 5839.32681943]
-8052.400589902743
```

calculating the error rate

In [26]:

```
y_pred_elastic=a.predict(x_train)
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print(mean_squared_error)
```

```
91743245.76096947
```

logistic regression

In [27]:

```
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from sklearn import metrics
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
```

In [28]:

```
df=pd.read_csv(r"C:\Users\91950\Downloads\insurance.csv")
df
```

Out[28]:

	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 [29]:

```
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 [30]:

```
df.describe()
```

Out[30]:

	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 [31]:

```
df.shape
```

Out[31]:

```
(1338, 7)
```

In [32]:

```
df=df[["sex","smoker"]]
df.columns=["sex","smoker"]
```

In [33]:

```
df.head()
```

Out[33]:

	sex	smoker
0	female	yes
1	male	no
2	male	no
3	male	no
4	male	no

In [34]:

```
sex={"sex":{"female":0,"male":1}}
df=df.replace(sex)
df
```

Out[34]:

```
      sex  smoker
   0    0     yes
   1    1      no
   2    1      no
   3    1      no
   4    1      no
 ...  ...     ...
1333    1      no
1334    0      no
1335    0      no
1336    0      no
1337    0     yes

1338 rows × 2 columns
```

In [35]:

```
smoker={"smoker":{"no":0,"yes":1}}
df=df.replace(smoker)
df
```

Out[35]:

```
      sex  smoker
   0    0       1
   1    1       0
   2    1       0
   3    1       0
   4    1       0
 ...  ...     ...
1333    1       0
1334    0       0
1335    0       0
1336    0       0
```

In [36]:

```
features_matrix=df.iloc[:,0:2]
target_vector=df.iloc[:,-1]
```

In [37]:

```
print('The target matrix has %d rows and %d column(S)'%(np.array(target_vector).reshape(-1,1).shap
```

```
The target matrix has 1338 rows and 1 column(S)
```

In [38]:

```
features_matrix_Standardized=StandardScaler().fit_transform(features_matrix)
```

In [47]:

```
algorithm = LogisticRegression(penalty=None,dual=False,tol=1e-1,C= 1.0,fit_intercept=True,intercep
```

In [48]:

```
Logistic_Regression_Model=algorithm.fit(features_matrix_Standardized,target_vector)
```

In [49]:

```
observation=[[0,1]]
```

In [50]:

```
print('The algoritham was trained to predict one of the two classes:%s'%(algorithm.classes_))
```

```
The algoritham was trained to predict one of the two classes:[0 1]
```

In [54]:

```
print(" " "The Model says the probability of the observation we passed belonging to class['0'] Is %
print()
```

```
 The Model says the probability of the observation we passed belonging to class['0']
Is 0.10473683208905549
```

In [55]:

```
print(" " "The Model says the probability of the observation we passed belonging to class['1'] Is %
print()
```

```
 The Model says the probability of the observation we passed belonging to class['1']
Is 0.10473683208905549
```

Decision tree

In [59]:

```
import numpy as np
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
```

In [61]:

```
df=pd.read_csv(r"C:\Users\91950\Downloads\insurance.csv")
df
```

Out[61]:

	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 [62]:

```
df["region"].value_counts()
```

Out[62]:

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

In [63]:

```
convert={"sex":{"female":0,"male":1}}
df=df.replace(convert)
df
```

Out[63]:

	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
...	...	...	...	...	...	...	...
1333	50	1	30.970	3	no	northwest	10600.54830
1334	18	0	31.920	0	no	northeast	2205.98080
1335	18	0	36.850	0	no	southeast	1629.83350
1336	21	0	25.800	0	no	southwest	2007.94500
1337	61	0	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [64]:

```
convert={"smoker":{"yes":0,"no":1}}
df=df.replace(smoker)
df
```

Out[64]:

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

1338 rows × 7 columns

In [65]:

```
x=["age","sex","children","bmi","charges"]
y=["0","1"]
all_inputs=df[x]
all_classes=df["smoker"]
```

In [66]:

```
x_train,X_test,y_train,y_test=train_test_split(all_inputs,all_classes,test_size=0.5)
x_train.shape,x_test.shape
```

Out[66]:

```
((669, 5), (669, 5))
```

In [67]:

```
s=DecisionTreeClassifier(random_state=20)
s.fit(x_train,y_train)
score=s.score(x_test,y_test)
print(score)
```

```
0.6756352765321375

C:\Users\91950\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\bas
e.py:439: UserWarning: X does not have valid feature names, but DecisionTreeClassifi
er was fitted with feature names
  warnings.warn(
```

Random forest

In [68]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt,seaborn as sb
```

In [69]:

```
df=pd.read_csv(r"C:\Users\91950\Downloads\insurance.csv")
df
```

Out[69]:

	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 [70]:

```
df.describe()
```

Out[70]:

	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 [71]:

```
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 [72]:

```
df.isna().any()
```

Out[72]:

```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [73]:

```
df.shape
```

Out[73]:

```
(1338, 7)
```

In [74]:

```
convert={"sex":{"female":0,"male":1}}
df=df.replace(convert)
df
```

Out[74]:

	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
...	...	...	...	...	...	...	...
1333	50	1	30.970	3	no	northwest	10600.54830
1334	18	0	31.920	0	no	northeast	2205.98080
1335	18	0	36.850	0	no	southeast	1629.83350
1336	21	0	25.800	0	no	southwest	2007.94500
1337	61	0	29.070	0	yes	northwest	29141.36030

1338 rows × 7 columns

In [75]:

```
df["region"].value_counts()
```

Out[75]:

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

In [76]:

```
r={"region":{"southeast":0,"southwest":1,"northeast":2,"northwest":3}}
df=df.replace(r)
df
```

Out[76]:

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

1338 rows × 7 columns

In [77]:

```
x=df.drop("smoker",axis=1)
y=df["smoker"]
```

In [78]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.5)
```

In [79]:

```
from sklearn.ensemble import RandomForestClassifier
```

In [80]:

```
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
```

Out[80]:

```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [81]:

```
params={"max_depth":[1,23,4,56,85],"min_samples_leaf":[4,6,8,10,12],"n_estimators":[8,9,10,65,42]}
```

In [82]:

```
from sklearn.model_selection import GridSearchCV
```

In [83]:

```
grid_search=GridSearchCV(estimator=rf,param_grid=params,cv=2)
grid_search.fit(x_train,y_train)
print(grid_search.score(x_test,y_test))
```

```
0.9461883408071748
```

In [84]:

```
p=grid_search.best_estimator_
print(p)
```

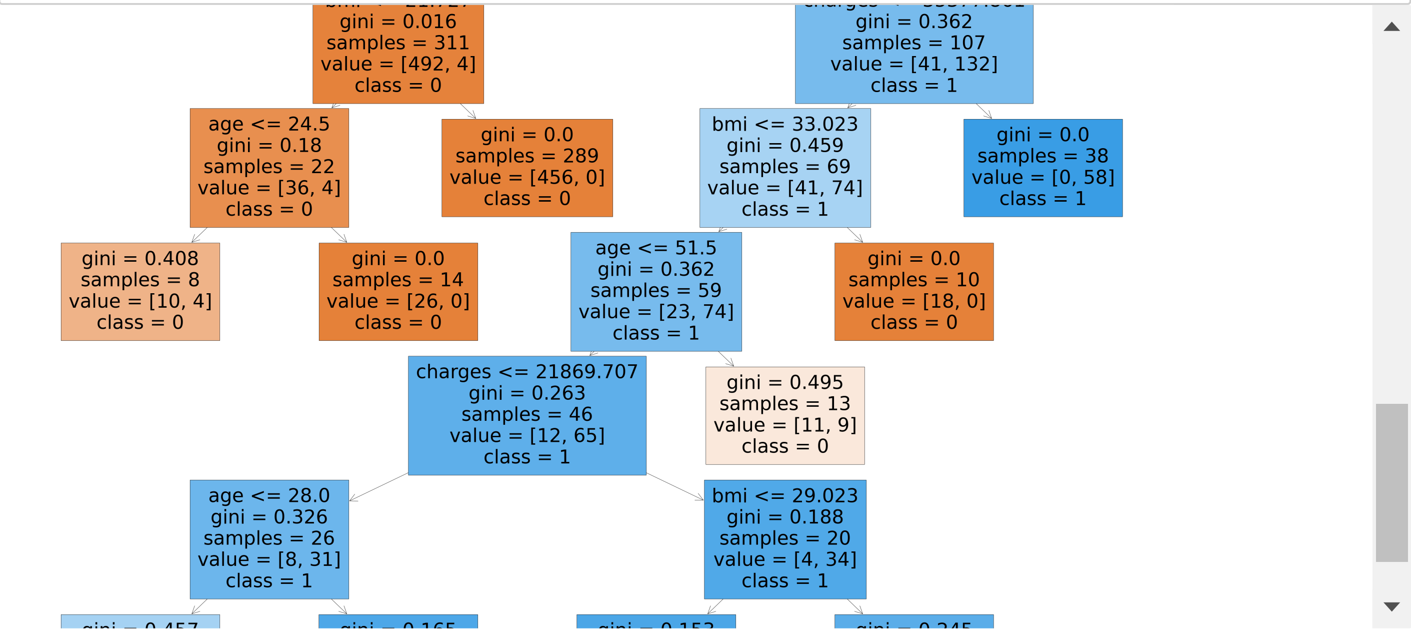
```
RandomForestClassifier(max_depth=56, min_samples_leaf=8, n_estimators=10)
```

In [85]:

```
from sklearn.tree import plot_tree
```

In [100]:

```
plt.figure(figsize=(80,60))
plot_tree(p.estimators_[5],feature_names=x.columns,class_names=["0","1"],filled=True)
```



In [101]:

```
p.feature_importances_
```

Out[101]:

```
array([0.04394906, 0.01297194, 0.05731956, 0.00837061, 0.00910233,
       0.8682865 ])
```

In [102]:

```
imp=pd.DataFrame({"varname":x_train.columns,"Imp":p.feature_importances_})
```

In [103]:

```
imp.sort_values(by="Imp",ascending=True)
```

Out[103]:

	varname	Imp
3	children	0.008371
4	region	0.009102
1	sex	0.012972
0	age	0.043949
2	bmi	0.057320
5	charges	0.868287

In [ ]:

```
from the above models i have concluded that random forest is the best model and the accuracy is 94
```