# PROBLEM STATEMENT:HOW BESTFIT THE GIVEN DATASET IS

# 1.DATA COLLECTION

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
```

In [2]:
```
df=pd.read_csv(r"C:\Users\pavan\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

# 2.DATA CLEANING & PREPROCESSING

In [3]:
```
df.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]:
```
df.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]:
```
df.shape
```

Out[5]: (1338, 7)

In [6]:
```
df.describe()
```

Out[6]:

| | 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 [7]:
```
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]:
```
df.isna().sum()
```

Out[8]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [9]:
```
df['region'].value_counts()
```

Out[9]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [10]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

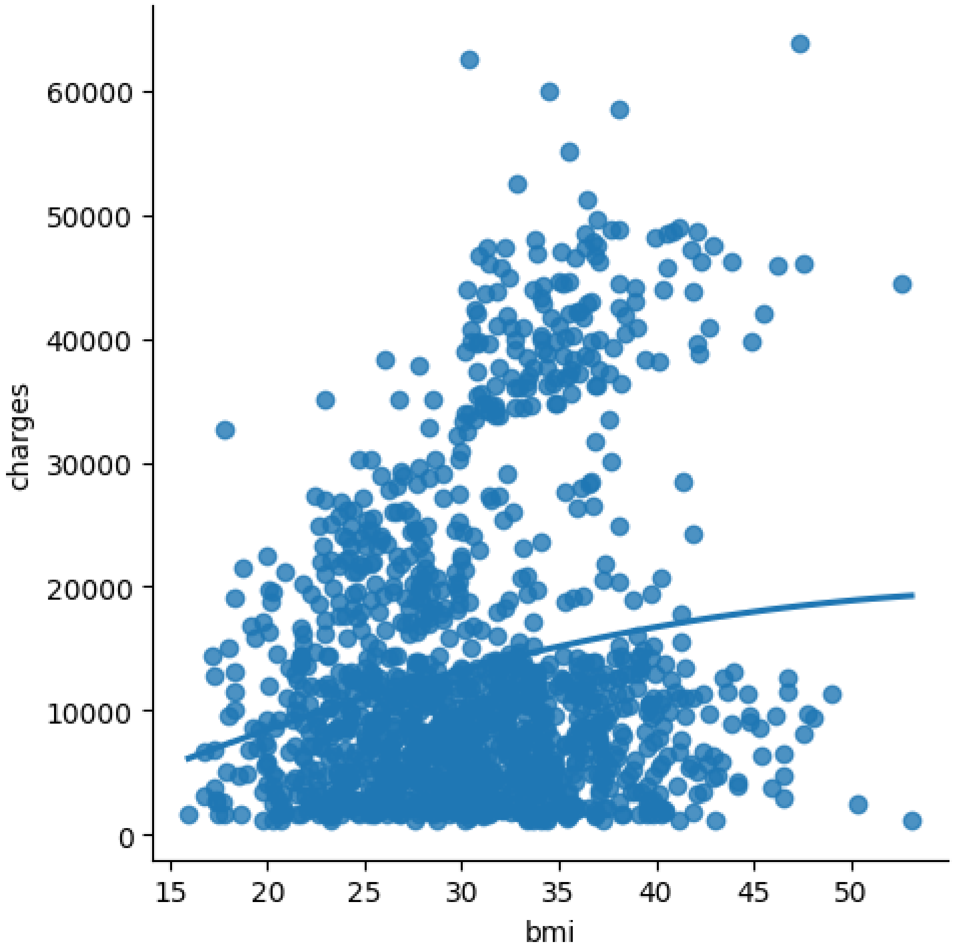
Out[10]:

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

1338 rows × 7 columns

# 3.DATA VISUALIZATION

In [11]:
```
sns.lmplot(x='bmi',y='charges',order=2,data=df,ci=None)
plt.show()
```

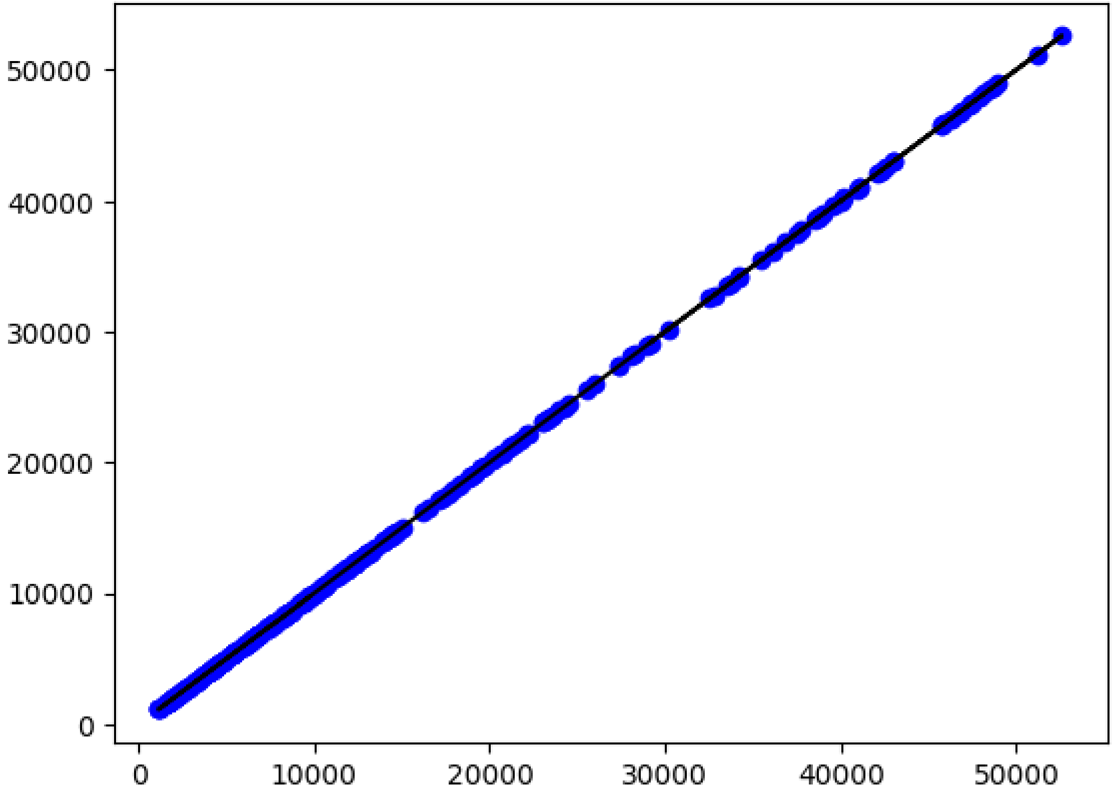



In [12]:
```
x=np.array(df['bmi']).reshape(-1,1)
y=x=np.array(df['charges']).reshape(-1,1)
```

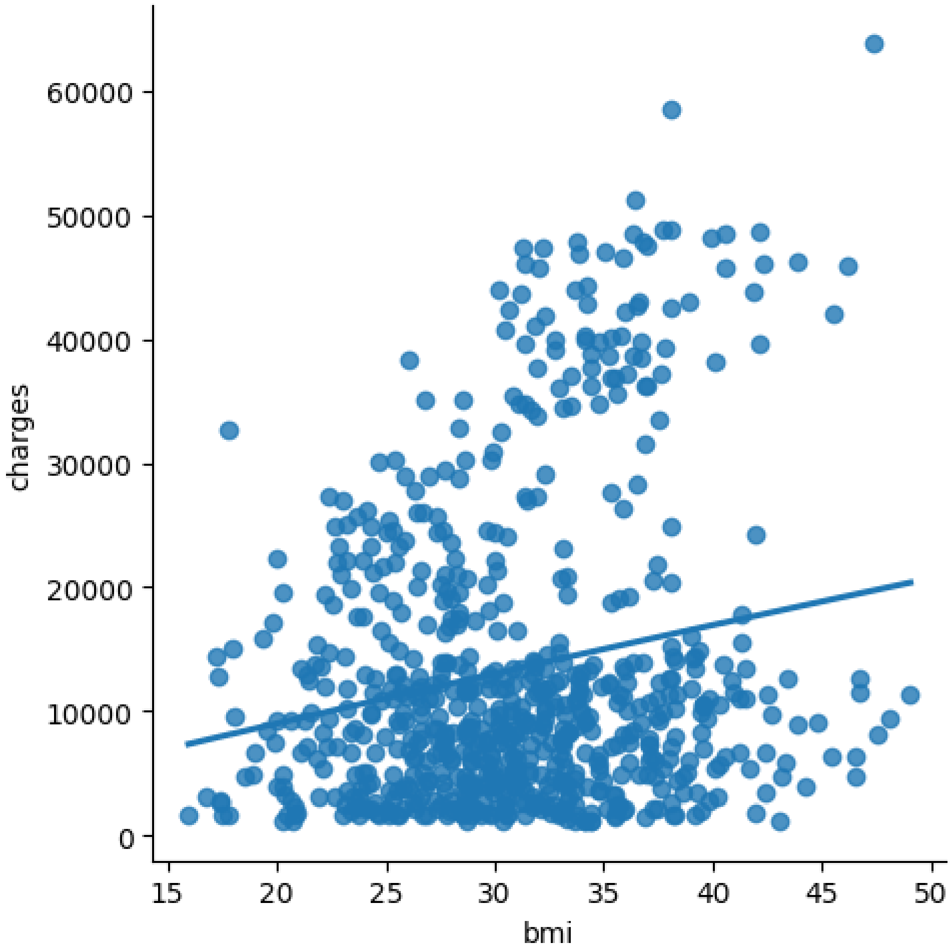
In [13]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=0)
lr=LinearRegression()
lr.fit(x_train,y_train)
print(lr.score(x_test,y_test))
```

```
1.0
```

In [14]:
```
y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



In [15]:
```
df700=df[:][:700]
sns.lmplot(x='bmi',y='charges',order=2,ci=None,data=df700)
plt.show()
```




In [16]:
```
df700.fillna(method='ffill',inplace=True)
```

In [17]:
```
x=np.array(df700["bmi"]).reshape(-1,1)
y=np.array(df700['charges']).reshape(-1,1)
```

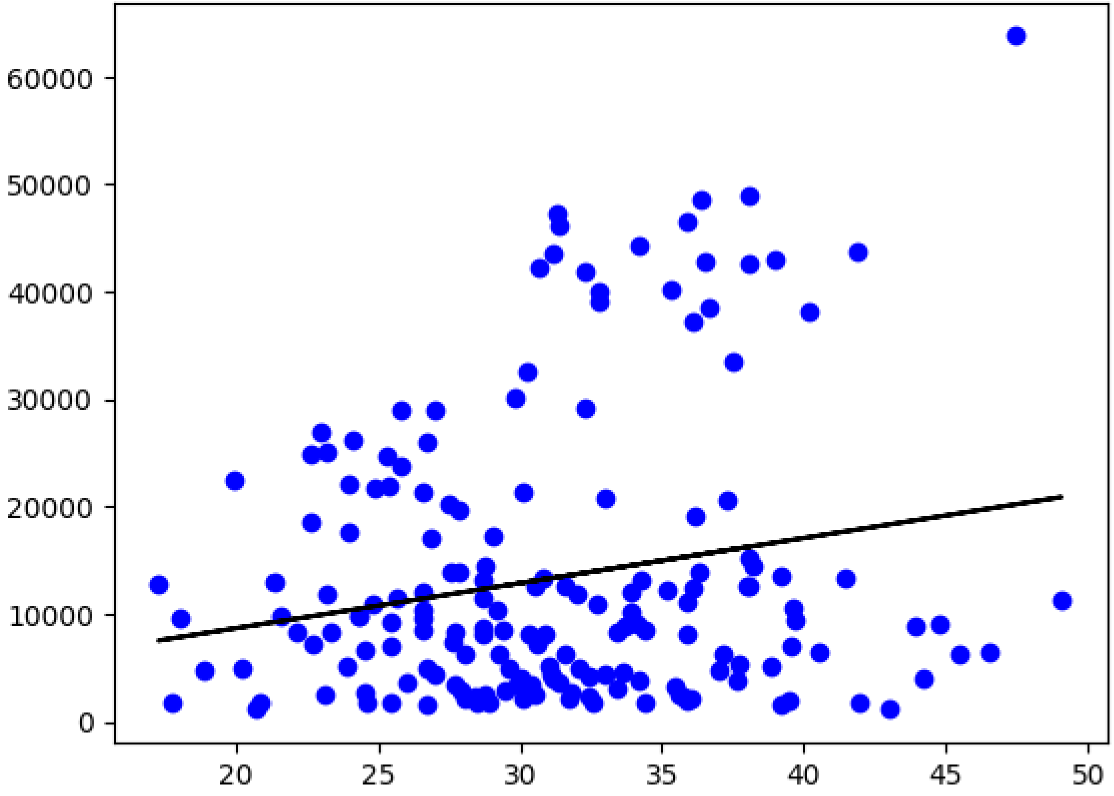
In [18]:
```
df700.dropna(inplace=True)
```

In [19]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
lr=LinearRegression()
lr.fit(x_train,y_train)
print(lr.score(x_test,y_test))
```

```
0.020508709612662157
```

In [20]:

```
y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```




# EVALUATION

In [21]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [22]:
```
lr=LinearRegression()
lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
r2=r2_score(y_test,y_pred)
print(r2)
```

```
0.020508709612662157
```

## accuracy is 0.03442

# RIDGE REGRESSION

Type *Markdown* and LaTeX: $\alpha^2$

In [23]:
```
convert={'smoker':{"yes":1,"no":0}}
df=df.replace(convert)
df
```

Out[23]:

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

1338 rows × 7 columns

In [24]:
```
convert={'region':{"southeast":1,"southwest":2,"northeast":3,"northwest":4}}
df=df.replace(convert)
df
```

Out[24]:

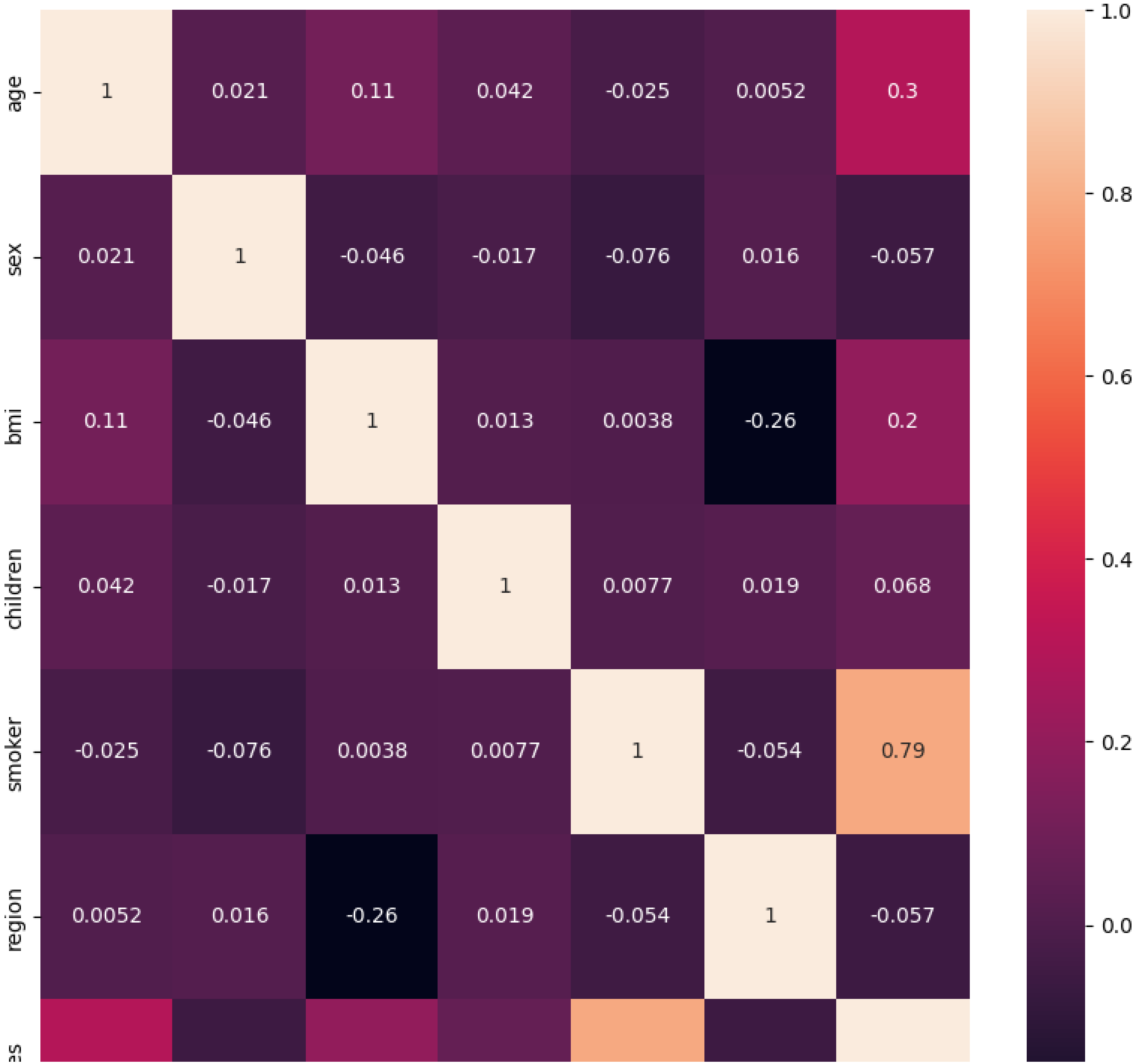
| | age | sex | bmi | children | smoker | region | charges |
|---|---|---|---|---|---|---|---|
| **0** | 19 | 1 | 27.900 | 0 | 1 | 2 | 16884.92400 |
| **1** | 18 | 0 | 33.770 | 1 | 0 | 1 | 1725.55230 |
| **2** | 28 | 0 | 33.000 | 3 | 0 | 1 | 4449.46200 |
| **3** | 33 | 0 | 22.705 | 0 | 0 | 4 | 21984.47061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

1338 rows × 7 columns

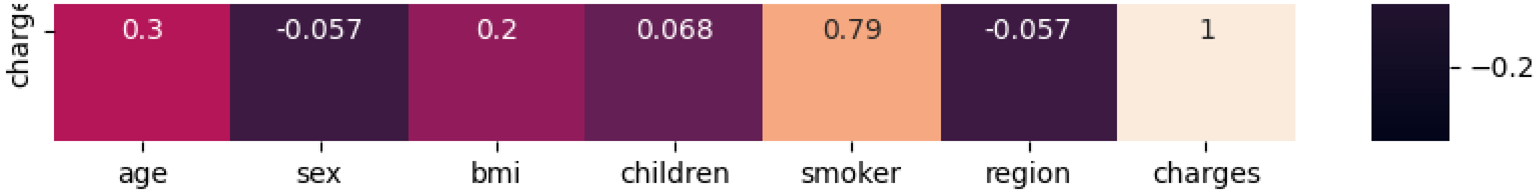
In [25]:
```
from sklearn.linear_model import Lasso,Ridge
from sklearn.preprocessing import StandardScaler
```

In [26]:
```
plt.figure(figsize = (10, 10))
sns.heatmap(df.corr(), annot = True)
```

Out[26]:
```
<Axes: >
```



age
sex
bmi
children
smoker
region
1 0.021 0.11 0.042 -0.025 0.0052 0.3
0.021 1 -0.046 -0.017 -0.076 0.016 -0.057
0.11 -0.046 1 0.013 0.0038 -0.26 0.2
0.042 -0.017 0.013 1 0.0077 0.019 0.068
-0.025 -0.076 0.0038 0.0077 1 -0.054 0.79
0.0052 0.016 -0.26 0.019 -0.054 1 -0.057
1.0
0.8
0.6
0.4
0.2
0.0




In [27]:
```
features=df.columns[0:1]
target=df.columns[-1]
```

In [28]:
```
x=df[features].values
y=df[target].values
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=1)
print("The dimension of X_train is {}".format(x_train.shape))
print("The dimension of X_test is {}".format(x_test.shape))
```

```
The dimension of X_train is (936, 1)
The dimension of X_test is (402, 1)
```

In [29]:
```
lr = LinearRegression()
#Fit model
lr.fit(x_train, y_train)
#predict
actual = y_test
train_score_lr = lr.score(x_train, y_train)
test_score_lr = lr.score(x_test, y_test)
print("\nLinear Regression Model:\n")
print("The train score for lr model is {}".format(train_score_lr))
print("The test score for lr model is {}".format(test_score_lr))
```

```

Linear Regression Model:

The train score for lr model is 0.09109639738057l4
The test score for lr model is 0.08490473916580776
```

In [30]:
```
ridgeReg = Ridge(alpha=10)
ridgeReg.fit(x_train,y_train)
#train and test scorefor ridge regression
train_score_ridge = ridgeReg.score(x_train, y_train)
test_score_ridge = ridgeReg.score(x_test, y_test)
print("\nRidge Model:\n")
print("The train score for ridge model is {}".format(train_score_ridge))
print("The test score for ridge model is {}".format(test_score_ridge))
```

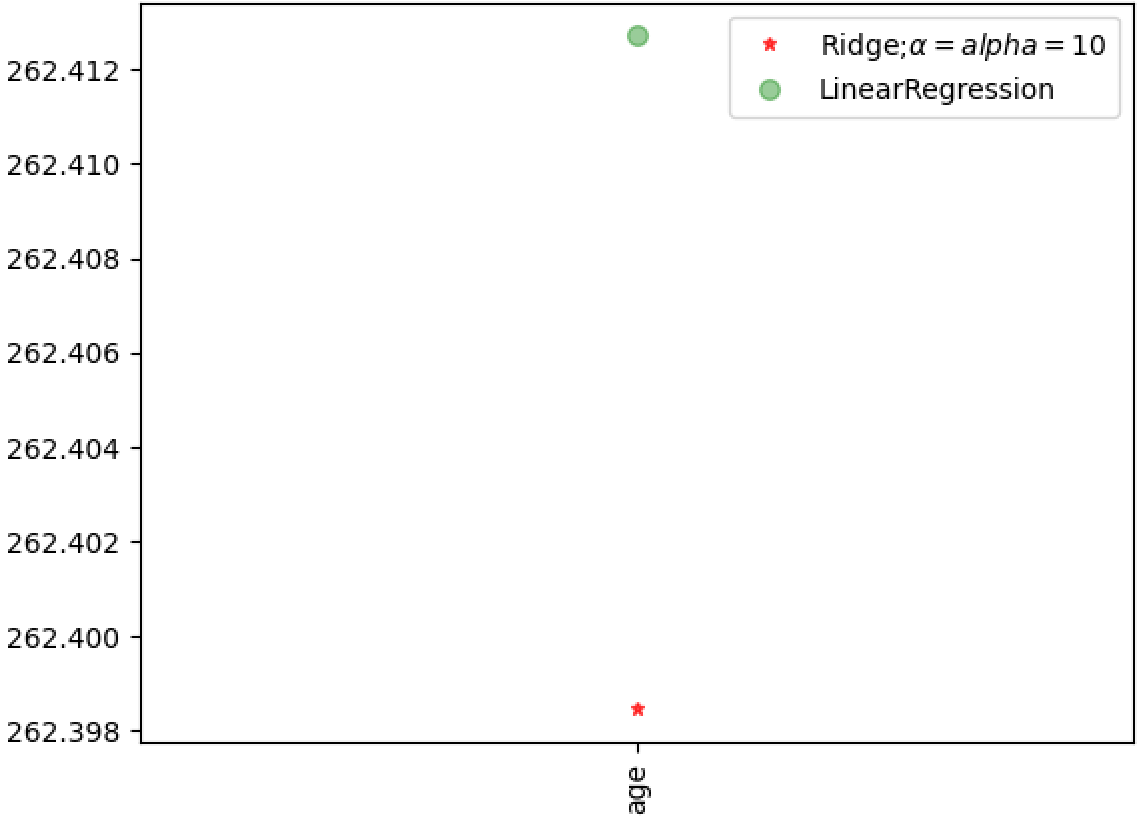
```
Ridge Model:

The train score for ridge model is 0.09109639711159634
The test score for ridge model is 0.08490538609860176
```

In [31]:
```
plt.figure(figsize=(10,10))
```

Out[31]:
```
<Figure size 1000x1000 with 0 Axes>
```

```
<Figure size 1000x1000 with 0 Axes>
```

In [32]:
```
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker="*",markersize=5,color='red',label=r'Ridge;$\alpha=
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker="o",markersize=7,color='green',label='LinearRegression')
plt.xticks(rotation=90)
plt.legend()
plt.show()
```




# LASSO REGRESSION

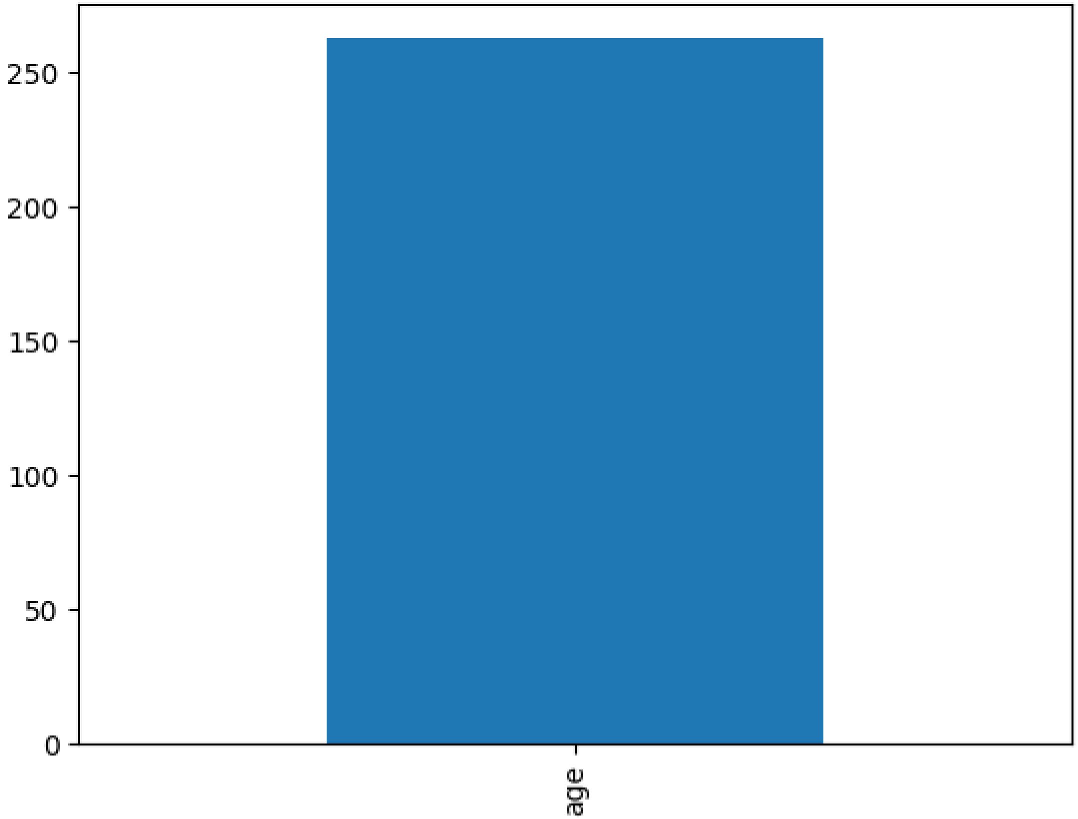
In [33]:
```
lasso= Lasso(alpha=10)
lasso.fit(x_train,y_train)
#train and test scorefor ridge regression
train_score_ls = lasso.score(x_train, y_train)
test_score_ls= lasso.score(x_test, y_test)
print("\nRidge Model:\n")
print("The train score for lasso model is {}".format(train_score_ls))
print("The test score for lasso model is {}".format(test_score_ls))
```

```
Ridge Model:

The train score for lasso model is 0.09109639395809055
The test score for lasso model is 0.08490704421828055
```

In [34]:
```
plt.figure(figsize=(10,10))
```

Out[34]:
```
<Figure size 1000x1000 with 0 Axes>

<Figure size 1000x1000 with 0 Axes>
```

In [35]:
```
pd.Series(lasso.coef_, features).sort_values(ascending = True).plot(kind = "bar")
plt.show()
```




In [36]:
```
from sklearn.linear_model import LassoCV
```

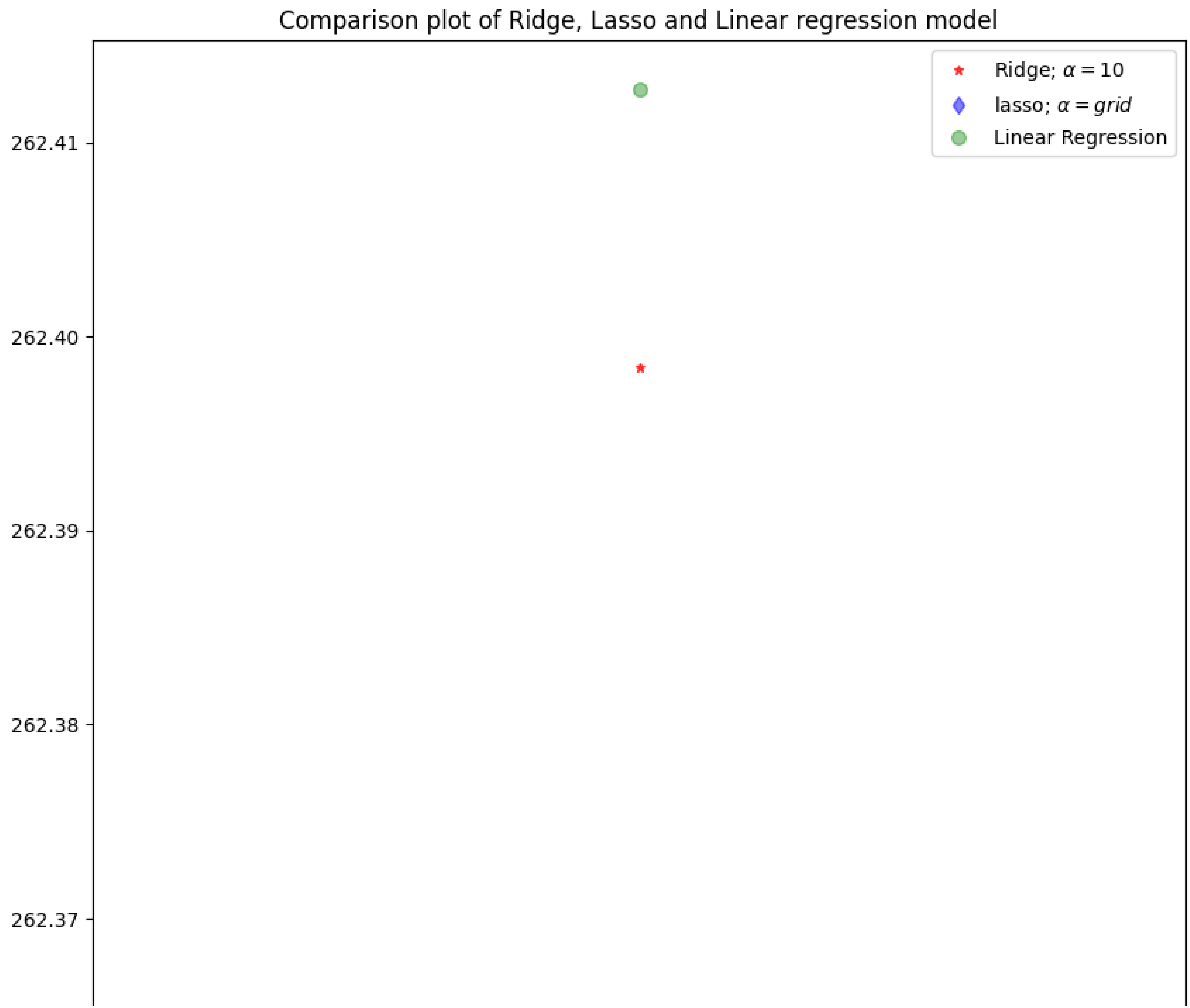
In [37]:
```
#using the linear cv model
from sklearn.linear_model import RidgeCV
#cross validation
ridge_cv=RidgeCV(alphas =[0.0001,0.001,0.01,0.1,1,10]).fit(x_train,y_train)
#score
print(ridge_cv.score(x_train,y_train))
print(ridge_cv.score(x_test,y_test))
```

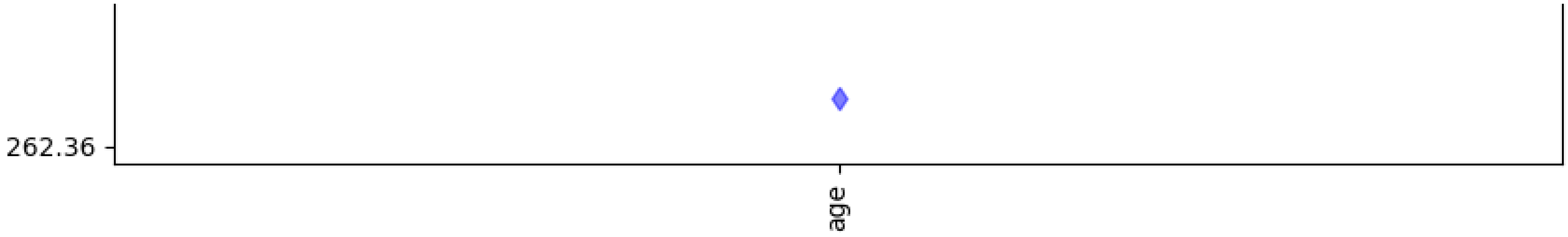
```
0.09109639711159612
0.08490538609884779
```

In [38]:
```
#using the linear cv model
from sklearn.linear_model import LassoCV
#cross validation
lasso_cv=LassoCV(alphas =[0.0001,0.001,0.01,0.1,1,10]).fit(x_train,y_train)
#score
print(lasso_cv.score(x_train,y_train))
print(lasso_cv.score(x_test,y_test))
```

```
0.09109639395809055
0.08490704421828055
```

In [39]:
```
ot size
figure(figsize = (10, 10))
 plot for ridge regression
plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,color='red',label=r'Ridge; $\alpha=10$
 plot for lasso regression
plot(lasso_cv.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,color='blue',label=r'lasso; $\alpha = grid$')
 plot for linear model
plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,color='green',label='Linear Regression')
ate axis
xticks(rotation = 90)
legend()
title("Comparison plot of Ridge, Lasso and Linear regression model")
show()
```


Comparison plot of Ridge, Lasso and Linear regression model
Ridge; $\alpha = 10$
lasso; $\alpha = grid$
Linear Regression
262.41
262.40
262.39
262.38
262.37




## ElasticNet

In [40]:
```
from sklearn.linear_model import ElasticNet
```

In [41]:
```
el=ElasticNet()
el.fit(x_train,y_train)
print(el.coef_)
print(el.intercept_)
el.score(x,y)
```

```
[261.74450967]
3115.083177426244
```

Out[41]:
```
0.08930616764094623
```

In [42]:
```
y_pred_elastic=el.predict(x_train)
```

In [43]:
```
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print(mean_squared_error)
```

```
135077142.70714515
```

## accuracy was 0.08930

## LOGISTIC REGRESSION

In [44]:
```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [45]:
```
df=pd.read_csv(r"C:\Users\pavan\Downloads\insurance.csv")
df
```

Out[45]:

| | 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 [46]:
```
df.shape
```
Out[46]:
```
(1338, 7)
```

In [47]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [48]:
```
print('This Dataset has %d rows and %d columns'%(df.shape))
```

```
This Dataset has 1338 rows and 7 columns
```

In [49]:
```
df.head()
```
Out[49]:

| | 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 [50]:
```
df.tail()
```
Out[50]:

| | 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 [51]:
```
df.describe()
```

Out[51]:

| | 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 [52]:
```
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 [53]:
```
df.isna().sum()
```

Out[53]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [54]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **184** | 44 | male | 30.690 | 2 | 0 | southeast | 7731.427100 |
| **185** | 36 | male | 41.895 | 3 | 1 | northeast | 43753.337050 |
| **186** | 26 | female | 29.920 | 2 | 0 | southeast | 3981.976800 |
| **187** | 30 | female | 30.900 | 3 | 0 | southwest | 5325.651000 |
| **188** | 41 | female | 32.200 | 1 | 0 | southwest | 6775.961000 |
| **189** | 29 | female | 32.110 | 2 | 0 | northwest | 4922.915900 |
| **190** | 61 | male | 31.570 | 0 | 0 | southeast | 12557.605300 |
| **191** | 36 | female | 26.200 | 0 | 0 | southwest | 4883.866000 |
| **192** | 25 | male | 25.740 | 0 | 0 | southeast | 2137.653600 |
| **193** | 56 | female | 26.600 | 1 | 0 | northwest | 12044.342000 |
| **194** | 18 | male | 34.430 | 0 | 0 | southeast | 1137.469700 |
| **195** | 19 | male | 30.590 | 0 | 0 | northwest | 1639.563100 |
| **196** | 39 | female | 32.800 | 0 | 0 | southwest | 5649.715000 |

In [55]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **45** | 55 | 0 | 37.300 | 0 | 0 | southwest | 20630.283510 |
| **46** | 18 | 1 | 38.665 | 2 | 0 | northeast | 3393.356350 |
| **47** | 28 | 1 | 34.770 | 0 | 0 | northwest | 3556.922300 |
| **48** | 60 | 1 | 24.530 | 0 | 0 | southeast | 12629.896700 |
| **49** | 36 | 0 | 35.200 | 1 | 1 | southeast | 38709.176000 |
| **50** | 18 | 1 | 35.625 | 0 | 0 | northeast | 2211.130750 |
| **51** | 21 | 1 | 33.630 | 2 | 0 | northwest | 3579.828700 |
| **52** | 48 | 0 | 28.000 | 1 | 1 | southwest | 23568.272000 |
| **53** | 36 | 0 | 34.430 | 0 | 1 | southeast | 37742.575700 |
| **54** | 40 | 1 | 28.690 | 3 | 0 | northwest | 8059.679100 |
| **55** | 58 | 0 | 36.955 | 2 | 1 | northwest | 47496.494450 |
| **56** | 58 | 1 | 31.825 | 2 | 0 | northeast | 13607.368750 |
| **57** | 18 | 0 | 31.680 | 2 | 1 | southeast | 34303.167200 |

In [56]:
```
convert={"region":{"southeast":1,"southwest":2,"northeast":3,"northwest":4}}
df=df.replace(convert)
df
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **64** | 20 | 1 | 22.420 | 0 | 1 | 4 | 14711.743800 |
| **65** | 19 | 1 | 28.900 | 0 | 0 | 2 | 1743.214000 |
| **66** | 61 | 1 | 39.100 | 2 | 0 | 2 | 14235.072000 |
| **67** | 40 | 0 | 26.315 | 1 | 0 | 4 | 6389.377850 |
| **68** | 40 | 1 | 36.190 | 0 | 0 | 1 | 5920.104100 |
| **69** | 28 | 0 | 23.980 | 3 | 1 | 1 | 17663.144200 |
| **70** | 27 | 1 | 24.750 | 0 | 1 | 1 | 16577.779500 |
| **71** | 31 | 0 | 28.500 | 5 | 0 | 3 | 6799.458000 |
| **72** | 53 | 1 | 28.100 | 3 | 0 | 2 | 11741.726000 |
| **73** | 58 | 0 | 32.010 | 1 | 0 | 1 | 11946.625900 |
| **74** | 44 | 0 | 27.400 | 2 | 0 | 2 | 7726.854000 |
| **75** | 57 | 0 | 34.010 | 0 | 0 | 4 | 11356.660900 |
| **76** | 29 | 1 | 29.590 | 1 | 0 | 1 | 3947.413100 |

In [57]:
```
features_matrix=df.iloc[:,0:4]
```

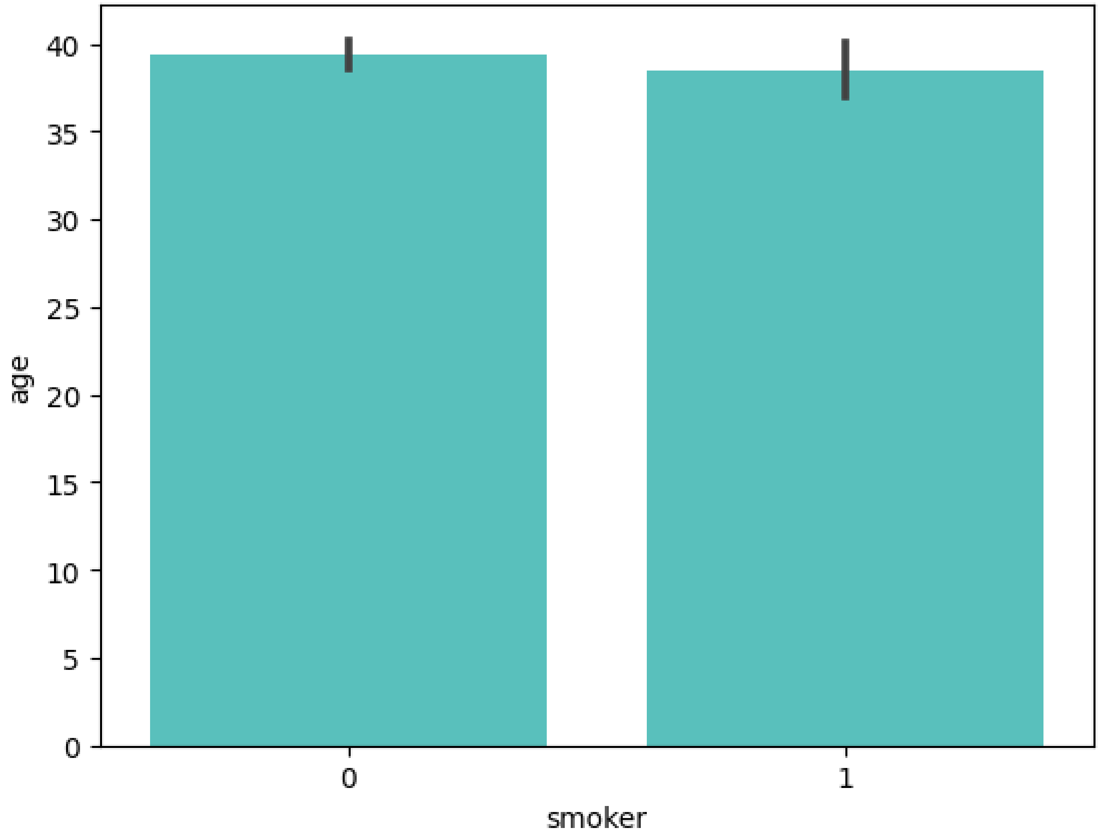
In [58]:
```
target_vector=df.iloc[:,-3]
```

In [59]:
```
print('The Feature Matrix has %d Rows and %d columns(s)'%(features_matrix.shape))
print('The Target Matrix has %d Rows and %d columns(s)'%(np.array(target_vector).reshape(-1,1).shape))
```

```
The Feature Matrix has 1338 Rows and 4 columns(s)
The Target Matrix has 1338 Rows and 1 columns(s)
```

In [60]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [61]:
```
sns.barplot(x='smoker', y='age', data=df, color="mediumturquoise")
plt.show()
```




In [62]:
```
features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

In [63]:
```
algorithm=LogisticRegression(max_iter=10000)
```

In [64]:
```
Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_vector)
```

In [65]:
```
observation=[[1,0,0.99539,-0.0588]]
```

In [66]:
```
predictions=Logistic_Regression_Model.predict(observation)
print('The model predicted the observation to belong to class %s'%(predictions))
```

```
The model predicted the observation to belong to class [0]
```

In [67]:
```
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 [68]:
```
print(" " "The Model says the probability of the observation we passed belonging to class[0] %s" " "%(algorithm.predic
print()
```

```
 The Model says the probability of the observation we passed belonging to class[0] 0.8057075871331396
```

In [69]:
```
print(" " "The Model says the probability of the observation we passed belonging to class['g'] Is %s" " "%(algorithm.p
```

```
 The Model says the probability of the observation we passed belonging to class['g'] Is 0.19429241286686041
```

In [70]:
```
x=np.array(df['age']).reshape(-1,1)
y=np.array(df['smoker']).reshape(-1,1)
```

In [71]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.05)
lo=LogisticRegression()
lo.fit(x_train,y_train)
print(lo.score(x_test,y_test))
```

```
0.835820895522388

C:\Users\pavan\AppData\Local\Programs\Python\Python311\Lib\site-packages\sklearn\utils\validation.py:1143: DataConve
rsionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,
), for example using ravel().
  y = column_or_1d(y, warn=True)
```

## for logistic regression the accuracy was 0.8059

## DECISIONTREE

In [72]:
```
import numpy as np
import pandas as pd
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [73]:
```
df=pd.read_csv(r"C:\Users\pavan\Downloads\insurance.csv")
df
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **8** | 37 | male | 29.830 | 2 | no | northeast | 6406.410700 |
| **9** | 60 | female | 25.840 | 0 | no | northwest | 28923.136920 |
| **10** | 25 | male | 26.220 | 0 | no | northeast | 2721.320800 |
| **11** | 62 | female | 26.290 | 0 | yes | southeast | 27808.725100 |
| **12** | 23 | male | 34.400 | 0 | no | southwest | 1826.843000 |
| **13** | 56 | female | 39.820 | 0 | no | southeast | 11090.717800 |
| **14** | 27 | male | 42.130 | 0 | yes | southeast | 39611.757700 |
| **15** | 19 | male | 24.600 | 1 | no | southwest | 1837.237000 |
| **16** | 52 | female | 30.780 | 1 | no | northeast | 10797.336200 |
| **17** | 23 | male | 23.845 | 0 | no | northeast | 2395.171550 |
| **18** | 56 | male | 40.300 | 0 | no | southwest | 10602.385000 |
| **19** | 30 | male | 35.300 | 0 | yes | southwest | 36837.467000 |
| **20** | 60 | female | 36.005 | 0 | no | northeast | 13228.846950 |

In [74]:
```
df.shape
```

Out[74]:
```
(1338, 7)
```

In [75]:
```
df.isnull().sum()
```

Out[75]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [76]:
```
df.isna().any()
```

Out[76]:
```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [77]:
```
df['region'].value_counts()
```

Out[77]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [78]:

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

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **55** | 58 | 0 | 36.955 | 2 | yes | northwest | 47496.494450 |
| **56** | 58 | 1 | 31.825 | 2 | no | northeast | 13607.368750 |
| **57** | 18 | 0 | 31.680 | 2 | yes | southeast | 34303.167200 |
| **58** | 53 | 1 | 22.880 | 1 | yes | southeast | 23244.790200 |
| **59** | 34 | 1 | 37.335 | 2 | no | northwest | 5989.523650 |
| **60** | 43 | 0 | 27.360 | 3 | no | northeast | 8606.217400 |
| **61** | 25 | 0 | 33.660 | 4 | no | southeast | 4504.662400 |
| **62** | 64 | 0 | 24.700 | 1 | no | northwest | 30166.618170 |
| **63** | 28 | 1 | 25.935 | 1 | no | northwest | 4133.641650 |
| **64** | 20 | 1 | 22.420 | 0 | yes | northwest | 14711.743800 |
| **65** | 19 | 1 | 28.900 | 0 | no | southwest | 1743.214000 |
| **66** | 61 | 1 | 39.100 | 2 | no | southwest | 14235.072000 |
| **67** | 40 | 0 | 26.315 | 1 | no | northwest | 6389.377850 |

In [79]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **34** | 28 | 0 | 36.400 | 1 | 1 | southwest | 51194.559140 |
| **35** | 19 | 0 | 20.425 | 0 | 0 | northwest | 1625.433750 |
| **36** | 62 | 1 | 32.965 | 3 | 0 | northwest | 15612.193350 |
| **37** | 26 | 0 | 20.800 | 0 | 0 | southwest | 2302.300000 |
| **38** | 35 | 0 | 36.670 | 1 | 1 | northeast | 39774.276300 |
| **39** | 60 | 0 | 39.900 | 0 | 1 | southwest | 48173.361000 |
| **40** | 24 | 1 | 26.600 | 0 | 0 | northeast | 3046.062000 |
| **41** | 31 | 1 | 36.630 | 2 | 0 | southeast | 4949.758700 |
| **42** | 41 | 0 | 21.780 | 1 | 0 | southeast | 6272.477200 |
| **43** | 37 | 1 | 30.800 | 2 | 0 | southeast | 6313.759000 |
| **44** | 38 | 0 | 37.050 | 1 | 0 | northeast | 6079.671500 |
| **45** | 55 | 0 | 37.300 | 0 | 0 | southwest | 20630.283510 |
| **46** | 18 | 1 | 38.665 | 2 | 0 | northeast | 3393.356350 |

In [80]:
```
x=["bmi","children"]
y=["Yes","No"]
all_inputs=df[x]
all_classes=df["sex"]
```

In [81]:
```
(x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,test_size=0.03)
```

In [82]:
```
clf=DecisionTreeClassifier(random_state=0)
```

In [83]:
```
clf.fit(x_train,y_train)
```

Out[83]:
```
▾ DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

In [84]:
```
score=clf.score(x_test,y_test)
print(score)
```

```
0.5853658536585366
```

# the accuracy for decision tree was 0.4878

# RANDOM FOREST

In [85]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt ,seaborn as sns
```

In [86]:
```python
df=pd.read_csv(r"C:\Users\pavan\Downloads\insurance.csv")
df
```

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **57** | 18 | male | 31.680 | 2 | yes | southeast | 34303.167200 |
| **58** | 53 | female | 22.880 | 1 | yes | southeast | 23244.790200 |
| **59** | 34 | female | 37.335 | 2 | no | northwest | 5989.523650 |
| **60** | 43 | male | 27.360 | 3 | no | northeast | 8606.217400 |
| **61** | 25 | male | 33.660 | 4 | no | southeast | 4504.662400 |
| **62** | 64 | male | 24.700 | 1 | no | northwest | 30166.618170 |
| **63** | 28 | female | 25.935 | 1 | no | northwest | 4133.641650 |
| **64** | 20 | female | 22.420 | 0 | yes | northwest | 14711.743800 |
| **65** | 19 | female | 28.900 | 0 | no | southwest | 1743.214000 |
| **66** | 61 | female | 39.100 | 2 | no | southwest | 14235.072000 |
| **67** | 40 | male | 26.315 | 1 | no | northwest | 6389.377850 |
| **68** | 40 | female | 36.190 | 0 | no | southeast | 5920.104100 |
| **69** | 28 | male | 23.980 | 3 | yes | southeast | 17663.144200 |

In [87]:
```python
df.shape
```

Out[87]:
```
(1338, 7)
```

In [88]:
```python
df['region'].value_counts()
```

Out[88]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [89]:
```
df['bmi'].value_counts()
```

```
28.595    6
37.100    6
31.730    6
33.000    6
27.740    6
29.830    6
25.175    6
26.410    6
27.835    5
33.660    5
28.900    5
25.080    5
28.500    5
26.695    5
23.210    5
32.395    5
31.825    5
33.155    5
29.640    5
36.850    5
```

In [90]:

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

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 35 | 19 | 0 | 20.425 | 0 | no | northwest | 1625.433750 |
| 36 | 62 | 1 | 32.965 | 3 | no | northwest | 15612.193350 |
| 37 | 26 | 0 | 20.800 | 0 | no | southwest | 2302.300000 |
| 38 | 35 | 0 | 36.670 | 1 | yes | northeast | 39774.276300 |
| 39 | 60 | 0 | 39.900 | 0 | yes | southwest | 48173.361000 |
| 40 | 24 | 1 | 26.600 | 0 | no | northeast | 3046.062000 |
| 41 | 31 | 1 | 36.630 | 2 | no | southeast | 4949.758700 |
| 42 | 41 | 0 | 21.780 | 1 | no | southeast | 6272.477200 |
| 43 | 37 | 1 | 30.800 | 2 | no | southeast | 6313.759000 |
| 44 | 38 | 0 | 37.050 | 1 | no | northeast | 6079.671500 |
| 45 | 55 | 0 | 37.300 | 0 | no | southwest | 20630.283510 |
| 46 | 18 | 1 | 38.665 | 2 | no | northeast | 3393.356350 |
| 47 | 28 | 1 | 34.770 | 0 | no | northwest | 3556.922300 |
| 48 | 60 | 1 | 24.530 | 0 | no | southeast | 12629.896700 |
| 49 | 36 | 0 | 35.200 | 1 | yes | southeast | 38709.176000 |
| 50 | 18 | 1 | 35.625 | 0 | no | northeast | 2211.130750 |
| 51 | 21 | 1 | 33.630 | 2 | no | northwest | 3579.828700 |
| 52 | 48 | 0 | 28.000 | 1 | yes | southwest | 23568.272000 |
| 53 | 36 | 0 | 34.430 | 0 | yes | southeast | 37742.575700 |
| 54 | 40 | 1 | 28.690 | 3 | no | northwest | 8059.679100 |

In [91]:

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

```
     age  sex     bmi  children  smoker     region      charges
0     19    1  27.900         0       1  southwest  16884.924000
1     18    0  33.770         1       0  southeast   1725.552300
2     28    0  33.000         3       0  southeast   4449.462000
3     33    0  22.705         0       0  northwest  21984.470610
4     32    0  28.880         0       0  northwest   3866.855200
5     31    1  25.740         0       0  southeast   3756.621600
6     46    1  33.440         1       0  southeast   8240.589600
7     37    1  27.740         3       0  northwest   7281.505600
8     37    0  29.830         2       0  northeast   6406.410700
9     60    1  25.840         0       0  northwest  28923.136920
10    25    0  26.220         0       0  northeast   2721.320800
11    62    1  26.290         0       1  southeast  27808.725100
12    23    0  34.400         0       0  southwest   1826.843000
13    56    1  39.820         0       0  southeast  11090.717800
14    27    0  42.130         0       1  southeast  39611.757700
15    19    0  24.600         1       0  southwest   1837.237000
16    52    1  30.780         1       0  northeast  10797.336200
17    23    0  23.845         0       0  northeast   2395.171550
```

In [92]:

```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[92]:

```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [93]:
```
rf=RandomForestClassifier()
params={'max_depth':[2,3,5,20],
 'min_samples_leaf':[5,10,20,50,100,200],
 'n_estimators':[10,25,30,50,100,200]}
```

In [94]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rf,param_grid=params,cv=2,scoring="accuracy")
grid_search.fit(x_train,y_train)
```

Out[94]:
```
▸ GridSearchCV
▸ estimator: RandomForestClassifier
    ▸ RandomForestClassifier
```

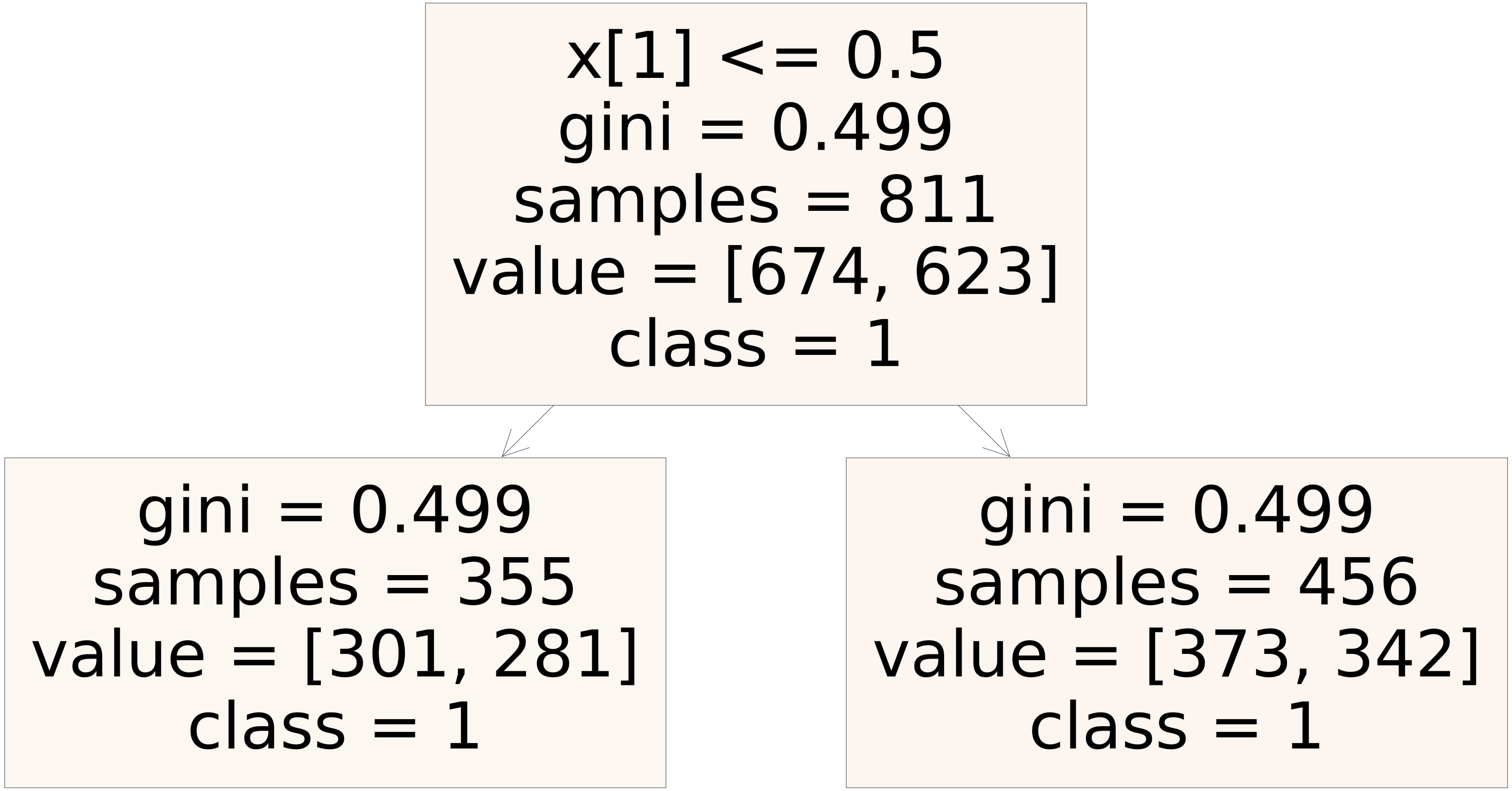
In [95]:
```
grid_search.best_score_
```

Out[95]:
```
0.5227462953451654
```

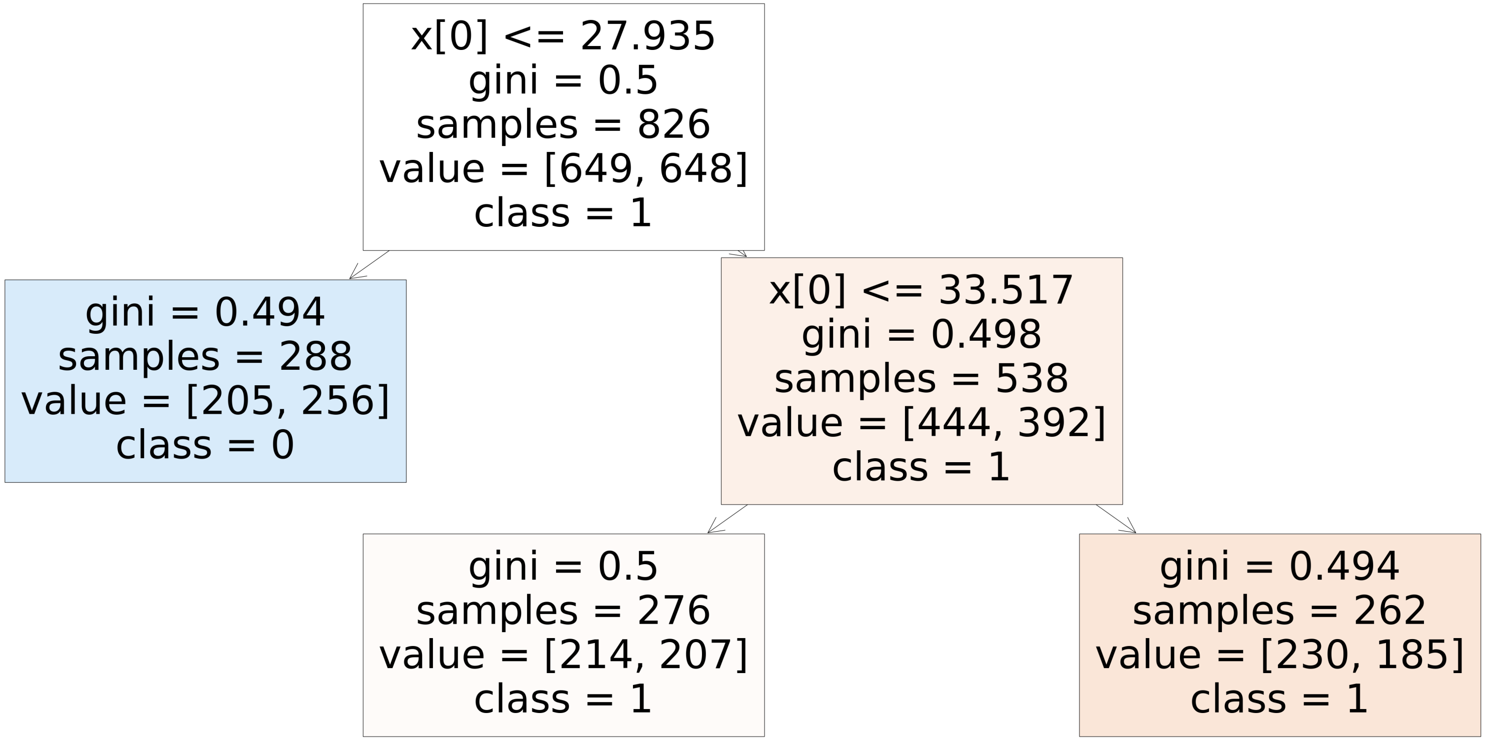
In [96]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=3, min_samples_leaf=200)
```

In [97]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[4],class_names=['1','0'],filled=True);
```



In [98]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(70,30))
plot_tree(rf_best.estimators_[6],class_names=["1","0"],filled=True);
```




In [99]:
```
rf_best.feature_importances_
```

Out[99]:
```
array([0.61264404, 0.38735596])
```

In [100]:
```
rf=RandomForestClassifier(random_state=0)
```

In [101]:
```
rf.fit(x_train,y_train)
```
Out[101]:
```
RandomForestClassifier
RandomForestClassifier(random_state=0)
```
In [102]:
```
score=rf.score(x_test,y_test)
print(score)
```
```
0.5853658536585366
```
# the accuracy is 0.43902

# CONCLUSION

```
 The given dataset is "Insurance",we need to find the bestfit Model. As per the data set ,we have used
different types of models,that different models got different types of accuracies.In this process Ridge
and Lasso got same accuracy i.e 0.091.so,we should not consider that.For the ElasticNet model I got the
accuracy of 0.089306.For the highest accuracy,I have done so many models among those ElasticNet got
highest accuracy.I have done so many visuvalization graphs as per the given Features.
```

So,the ElasticNet Model is the bestfit for the given Dataset "Insurance".

In [ ]:
```
```