Problem statement:To predict How Best the data fits

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\Niranjan\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 and Preprocessing

Exploratory data anlysis

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]:
```
<bound method NDFrame.describe of       age     sex     bmi  children smoke
r     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 x 7 columns]>
```

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.isnull().any()
```

Out[8]:
```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [9]:
```
df.isna().sum()
```

Out[9]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [10]:
```
df['region'].value_counts()
```

Out[10]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [11]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[11]:

	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

In [12]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

Out[12]:

	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 [13]:
```
convert={"region":{"southwest":1,"southeast":2,"northwest":3,"northeast"
df=df.replace(convert)
df
```

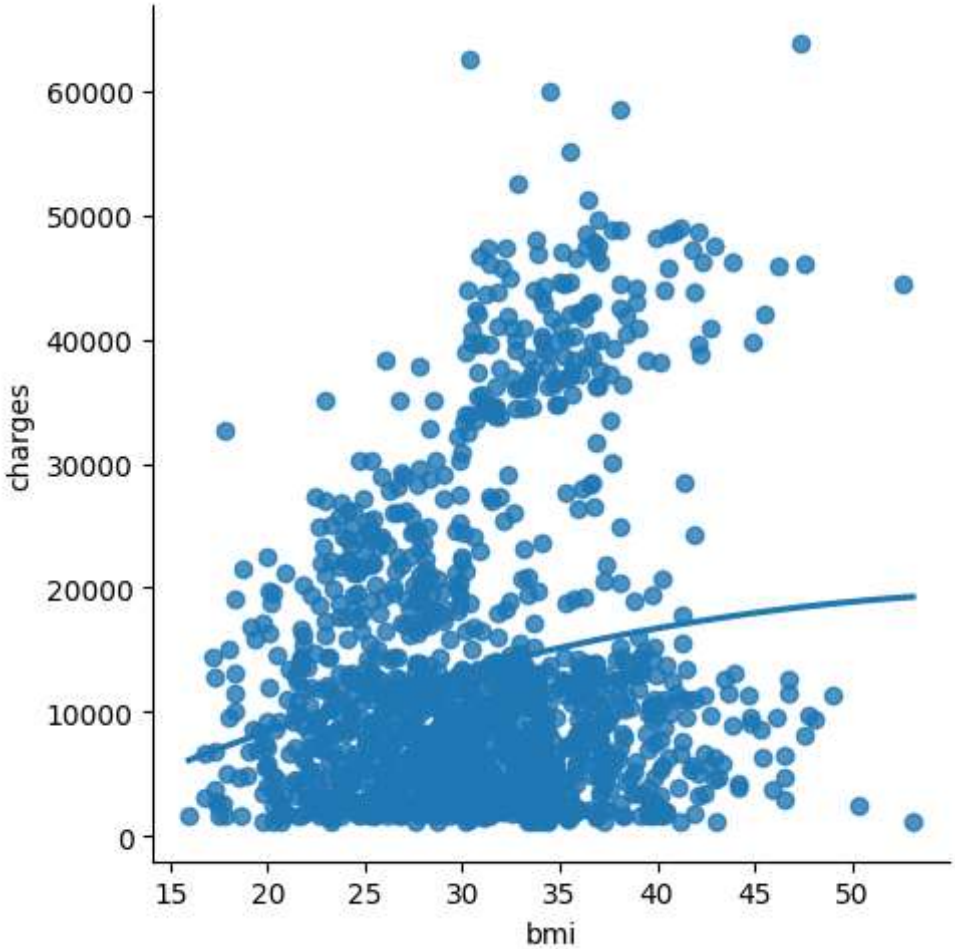
Out[13]:

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

1338 rows × 7 columns

3)Data Visualization

In [14]:
```
sns.lmplot(x='bmi',y='charges',order=2,data=df,ci=None)
plt.show()
```

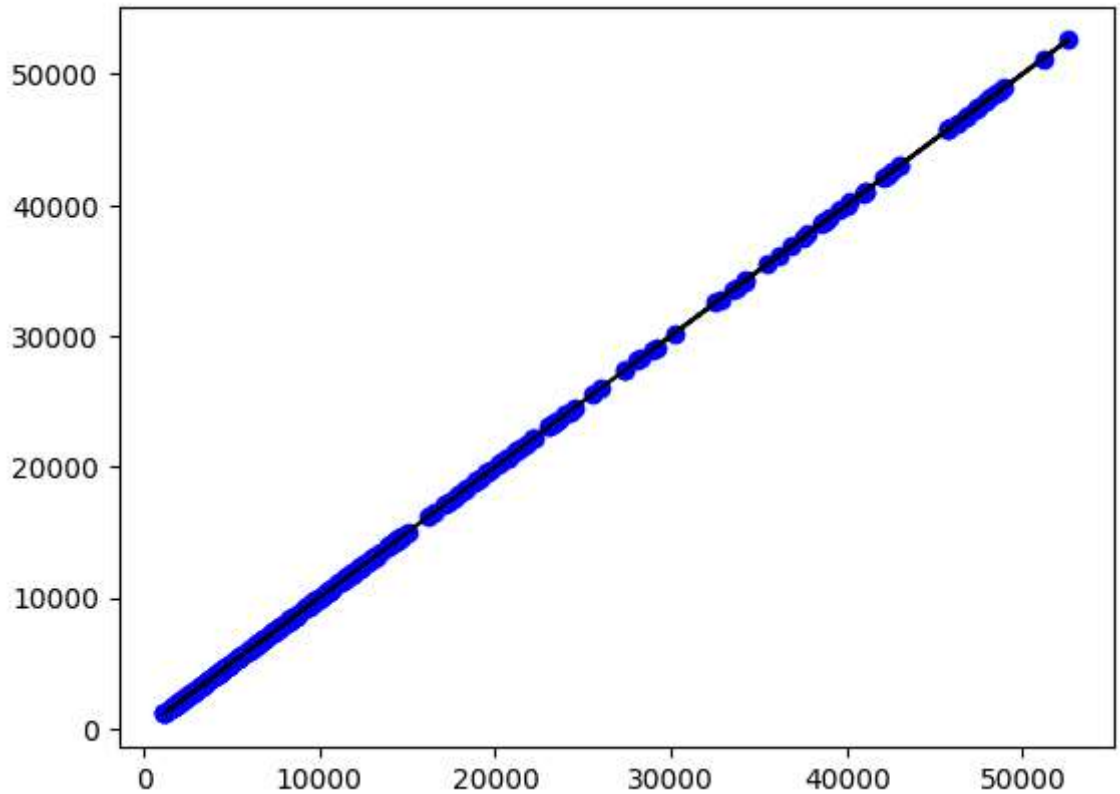


In [15]:
```
x=np.array(df['bmi']).reshape(-1,1)
y=x=np.array(df['charges']).reshape(-1,1)
```

In [16]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random
lr=LinearRegression()
lr.fit(x_train,y_train)
print(lr.score(x_test,y_test))
```

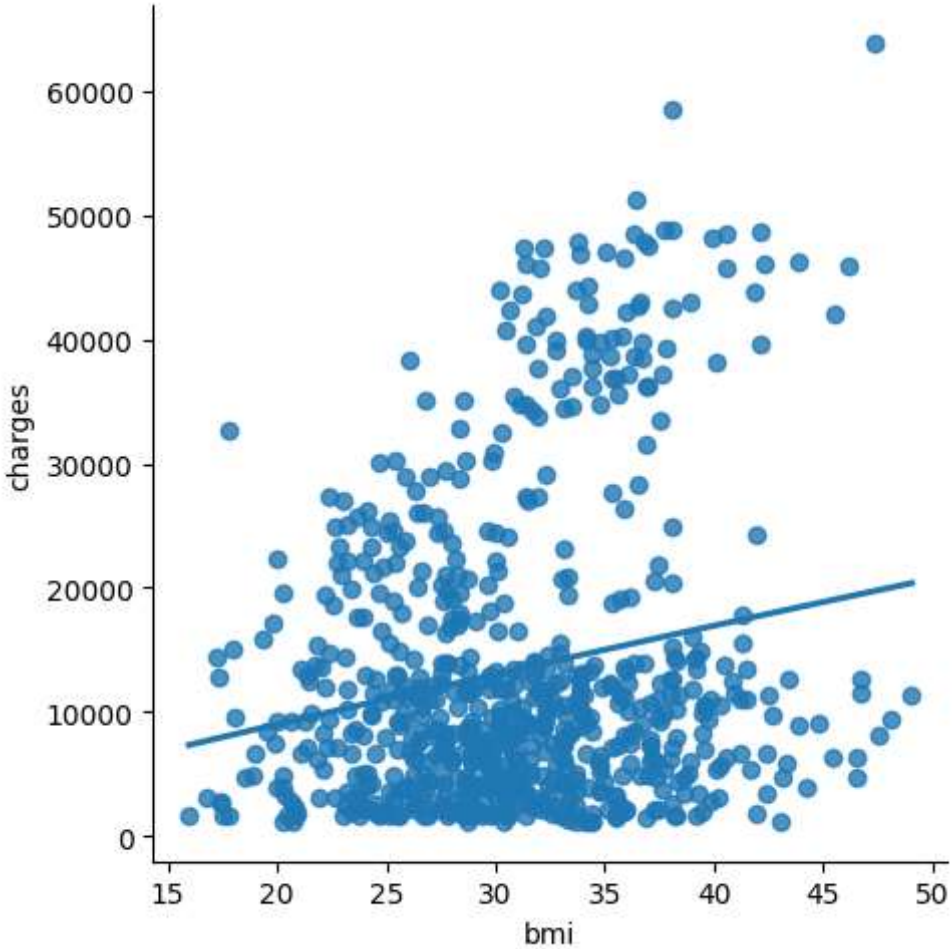
```
1.0
```

In [17]:
```
y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



working with subset of data

In [18]:
```
df700=df[:][:700]
sns.lmplot(x='bmi',y='charges',order=2,ci=None,data=df700)
plt.show()
```



In [19]:
```
df700.fillna(method='ffill',inplace=True)
```

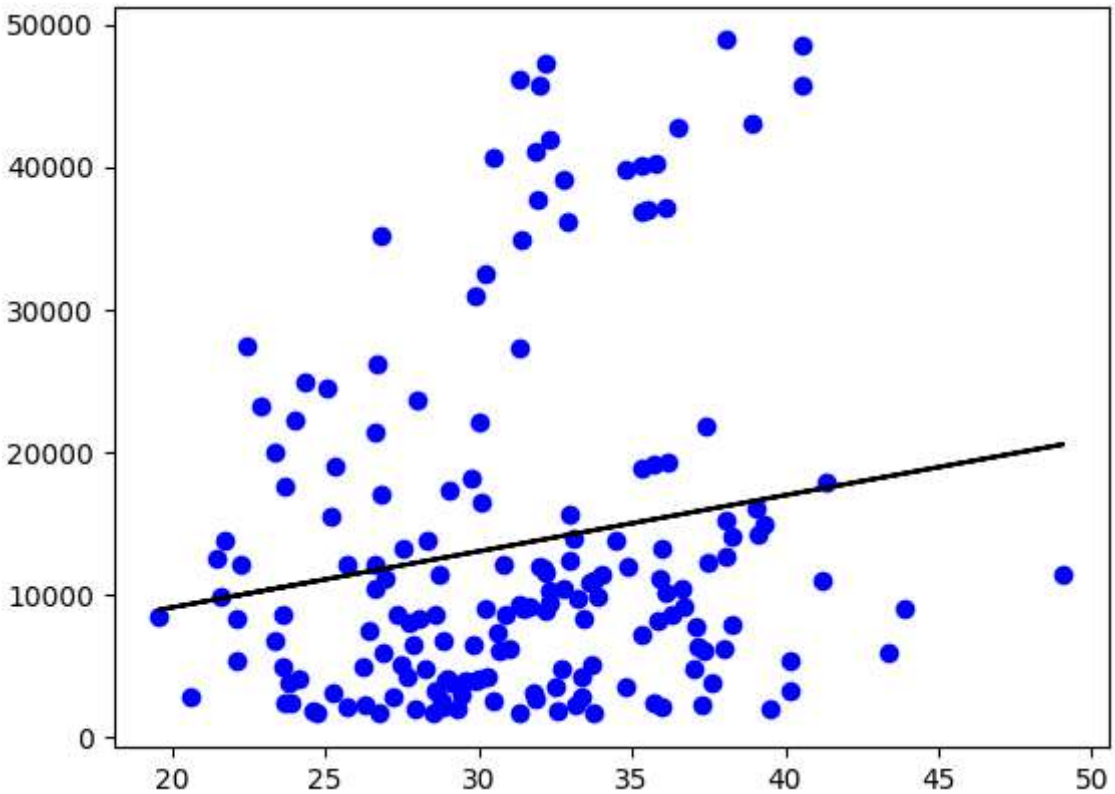
In [20]:
```
x=np.array(df700["bmi"]).reshape(-1,1)
y=np.array(df700['charges']).reshape(-1,1)
```

In [21]:
```
df700.dropna(inplace=True)
```

In [22]:
```
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))
```

```
0.029666204468372648
```

In [23]:
```
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 of model

In [24]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [25]:
```
lr=LinearRegression()
lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
r2=r2_score(y_test,y_pred)
print(r2)
```

```
0.029666204468372648
```

Ridge Regression

In [26]:
```
from sklearn.linear_model import Lasso,Ridge
from sklearn.preprocessing import StandardScaler
```

In [27]:
```
plt.figure(figsize=(10,10))
sns.heatmap(df700.corr(),annot=True)
plt.show()
```

In [28]:
```
features=df.columns[0:1]
target=df.columns[-1]
```

In [29]:
```
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
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 [30]:
```
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 0.0910963973805714
The test score for lr model is 0.08490473916580776
```

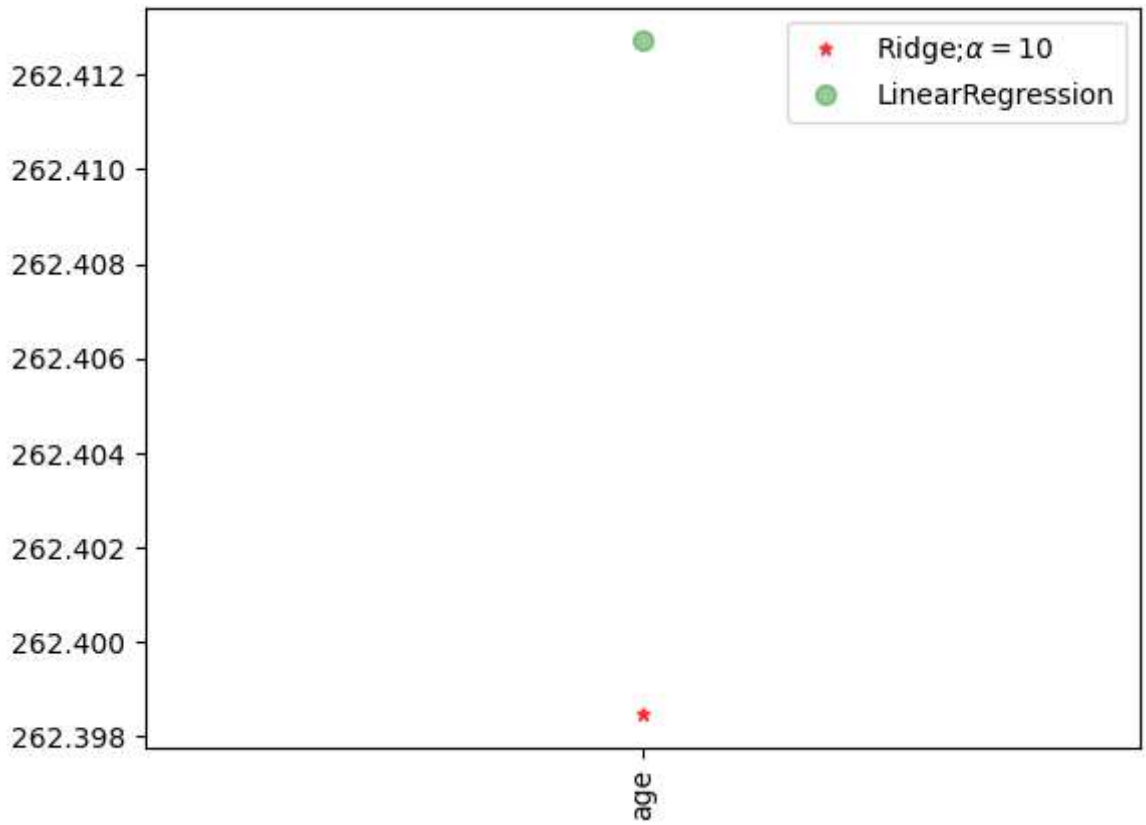
In [31]:
```
ridgeReg = Ridge(alpha=10)
ridgeReg.fit(x_train,y_train)
#train and test scorefor ridge regression
train_score_ridge = ridgeReg.score(x_train, y_train)
test_score_ridge = ridgeReg.score(x_test, y_test)
print("\nRidge Model:\n")
print("The train score for ridge model is {}".format(train_score_ridge))
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 0.08490538609860199
```

In [32]:
```
plt.figure(figsize=(10,10))
```

Out[32]:
```
<Figure size 1000x1000 with 0 Axes>

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

In [34]:
```
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker="*",m
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker="o",markers
plt.xticks(rotation=90)
plt.legend()
plt.show()
```



Lasso Regression

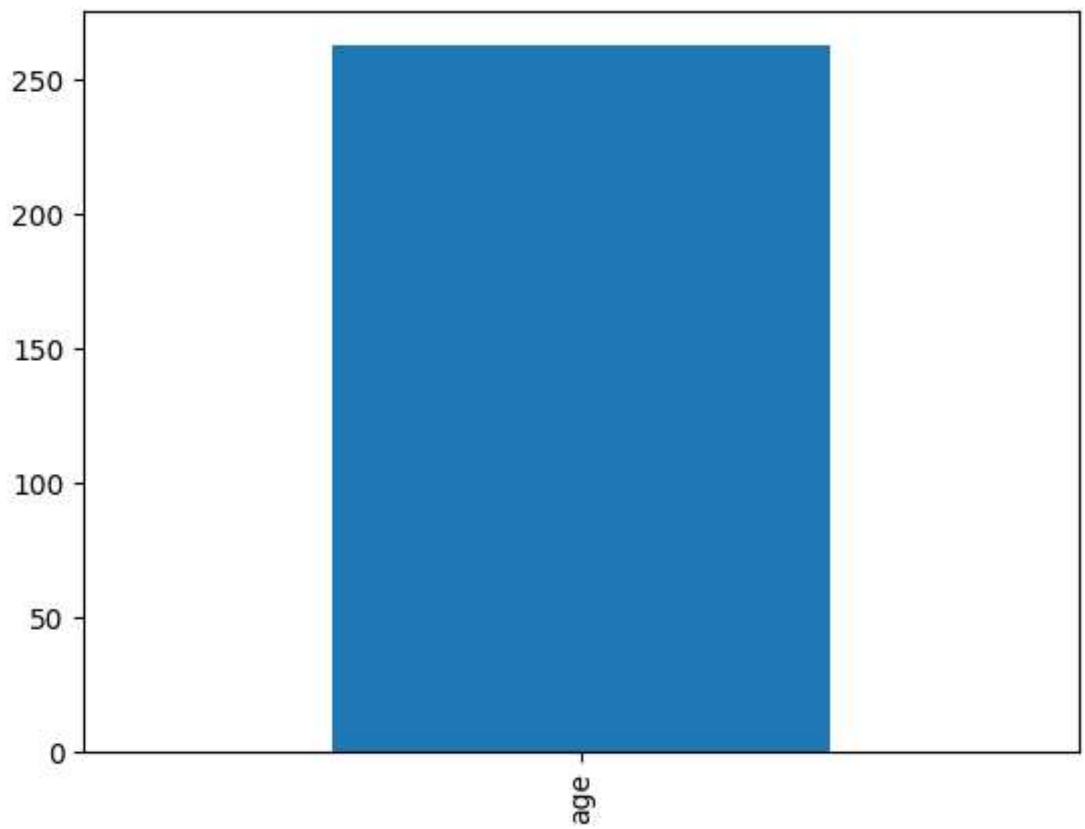
In [35]:
```
lasso= Lasso(alpha=10)
lasso.fit(x_train,y_train)
#train and test scorefor ridge regression
train_score_ls = lasso.score(x_train, y_train)
test_score_ls= lasso.score(x_test, y_test)
print("\nRidge Model:\n")
print("The test score for lasso model is {}".format(test_score_ls))
```

```
Ridge Model:

The train score for lasso model is 0.09109639395809044
The test score for lasso model is 0.08490704421828055
```

In [36]:
```
plt.figure(figsize=(10,10))
```

Out[36]:
```
<Figure size 1000x1000 with 0 Axes>

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

In [37]:
```
pd.Series(lasso.coef_, features).sort_values(ascending = True).plot(kind
plt.show()
```



In [38]:
```
from sklearn.linear_model import LassoCV
```

In [39]:
```
#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_tra
#score
print(ridge_cv.score(x_train,y_train))
print(ridge_cv.score(x_test,y_test))
```

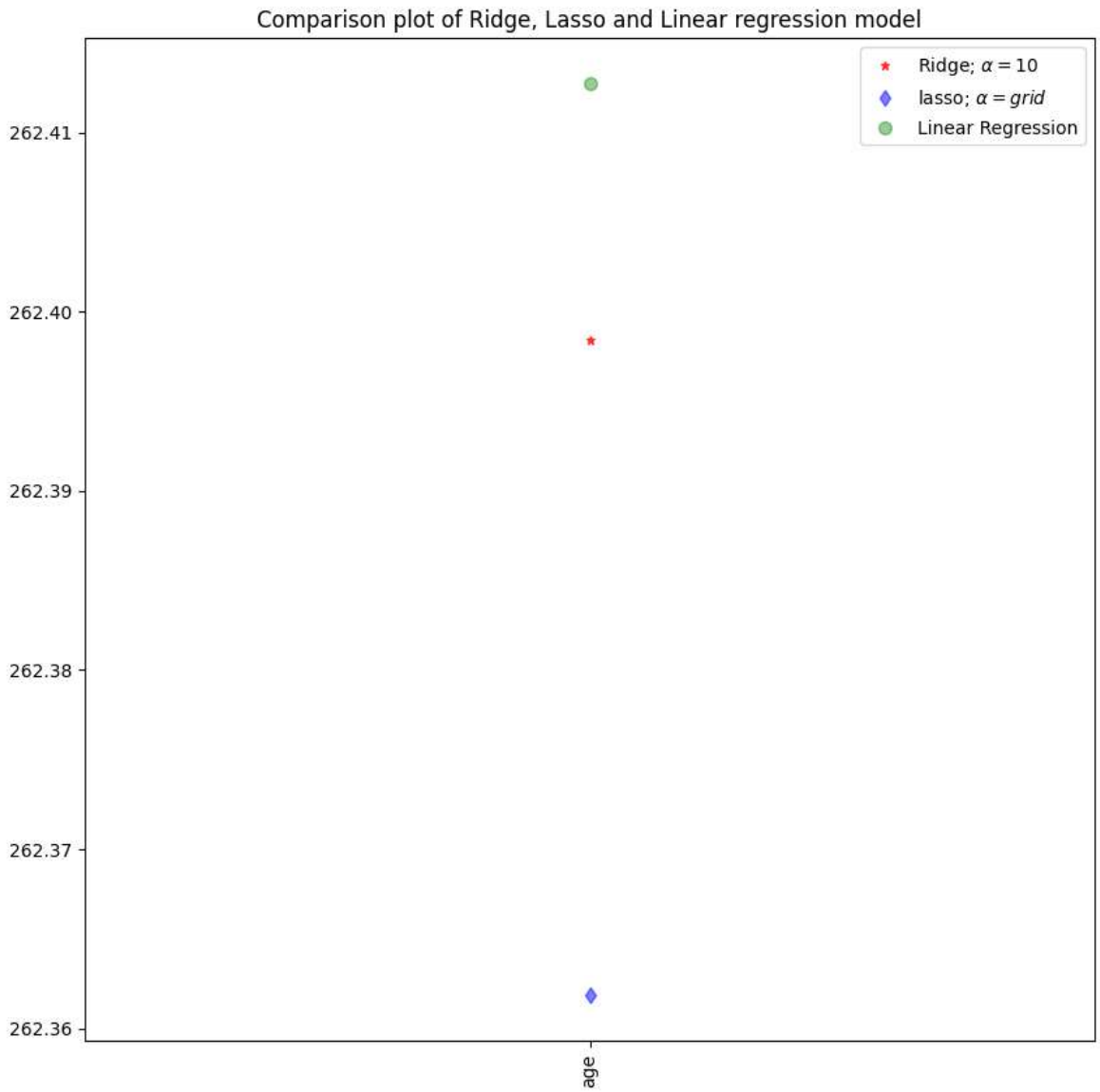
```
0.09109639711159612
0.08490538609885023
```

In [40]:

```
#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_tra
#score
print(lasso_cv.score(x_train,y_train))
print(lasso_cv.score(x_test,y_test))
```

```
0.09109639395809044
0.08490704421828055
```

In [42]:
```
plt.figure(figsize = (10, 10))
#add plot for ridge regression
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',m
#add plot for lasso regression
plt.plot(lasso_cv.coef_,alpha=0.5,linestyle='none',marker='d',markersize
#add plot for linear model
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markers
#rotate axis
plt.xticks(rotation = 90)
plt.legend()
plt.title("Comparison plot of Ridge, Lasso and Linear regression model")
plt.show()
```



ElasticNet Regression

In [43]:
```
from sklearn.linear_model import ElasticNet
```

In [44]:
```
el=ElasticNet()
el.fit(x_train,y_train)
print(el.coef_)
print(el.intercept_)
```

```
[261.74450967]
3115.0831774262424
```

In [45]:
```
y_pred_elastic=el.predict(x_train)
```

In [46]:
```
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print(mean_squared_error)
```

```
135077142.70714515
```

In [47]:
```
el=ElasticNet()
el.fit(x_train,y_train)
print(el.score(x_train,y_train))
```

```
0.09109580670592365
```

Logistic Regression

In [48]:
```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [49]:
```
df=pd.read_csv(r"C:\Users\Niranjan\Downloads\insurance.csv")
df
```

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
...	...	...	...	...	...	...	...
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 [50]:
```
df.shape
```

Out[50]:
```
(1338, 7)
```

In [51]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [52]:
```
print('This Dataset has %d rows and %d columns'%(df.shape))
```

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

In [53]:
```
df.head()
```

Out[53]:

	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 [54]:
```
df.describe
```

```
335    64    male  34.500         0     no  southwest  13822.803000
336    60    male  25.740         0     no  southeast  12142.578600
337    62    male  27.550         1     no  northwest  13937.666500
338    50    male  32.300         1    yes  northeast  41919.097000
339    46  female  27.720         1     no  southeast   8232.638800
340    24  female  27.600         0     no  southwest  18955.220170
341    62    male  30.020         0     no  northwest  13352.099800
342    60  female  27.550         0     no  northeast  13217.094500
343    63    male  36.765         0     no  northeast  13981.850350
344    49  female  41.470         4     no  southeast  10977.206300
345    34  female  29.260         3     no  southeast   6184.299400
346    33    male  35.750         2     no  southeast   4889.999500
347    46    male  33.345         1     no  northeast   8334.457550
348    36  female  29.920         1     no  southeast   5478.036800
349    19    male  27.835         0     no  northwest   1635.733650
350    57  female  23.180         0     no  northwest  11830.607200
351    50  female  25.600         0     no  southwest   8932.084000
352    30  female  27.700         0     no  southwest   3554.203000
353    33    male  35.245         0     no  northeast  12404.879100
354    18  female  38.280         0     no  southeast  14133.037750
```

In [55]:
```
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 [56]:
```
df.isnull().sum()
```

Out[56]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [57]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

102	18	female	30.115	0	0	northeast	21344.846700
103	61	female	29.920	3	1	southeast	30942.191800
104	34	female	27.500	1	0	southwest	5003.853000
105	20	male	28.025	1	1	northwest	17560.379750
106	19	female	28.400	1	0	southwest	2331.519000
107	26	male	30.875	2	0	northwest	3877.304250
108	29	male	27.940	0	0	southeast	2867.119600
109	63	male	35.090	0	1	southeast	47055.532100
110	54	male	33.630	1	0	northwest	10825.253700
111	55	female	29.700	2	0	southwest	11881.358000
112	37	male	30.800	0	0	southwest	4646.759000
113	21	female	35.720	0	0	northwest	2404.733800
114	52	male	32.205	3	0	northeast	11488.316950

In [58]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[58]:

	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

In [59]:
```
convert={"region":{"southeast":1,"southwest":2,"northeast":3,"northwest"
df=df.replace(convert)
df
```

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
77	21	0	35.530	0	0	1	1532.469700
78	22	1	39.805	0	0	3	2755.020950
79	41	1	32.965	0	0	4	6571.024350
80	31	0	26.885	1	0	3	4441.213150
81	45	1	38.285	0	0	3	7935.291150
82	22	0	37.620	1	1	1	37165.163800
83	48	1	41.230	4	0	4	11033.661700
84	37	1	34.800	2	1	2	39836.519000
85	45	0	22.895	2	1	4	21098.554050

In [60]:
```
features_matrix=df.iloc[:,0:4]
```

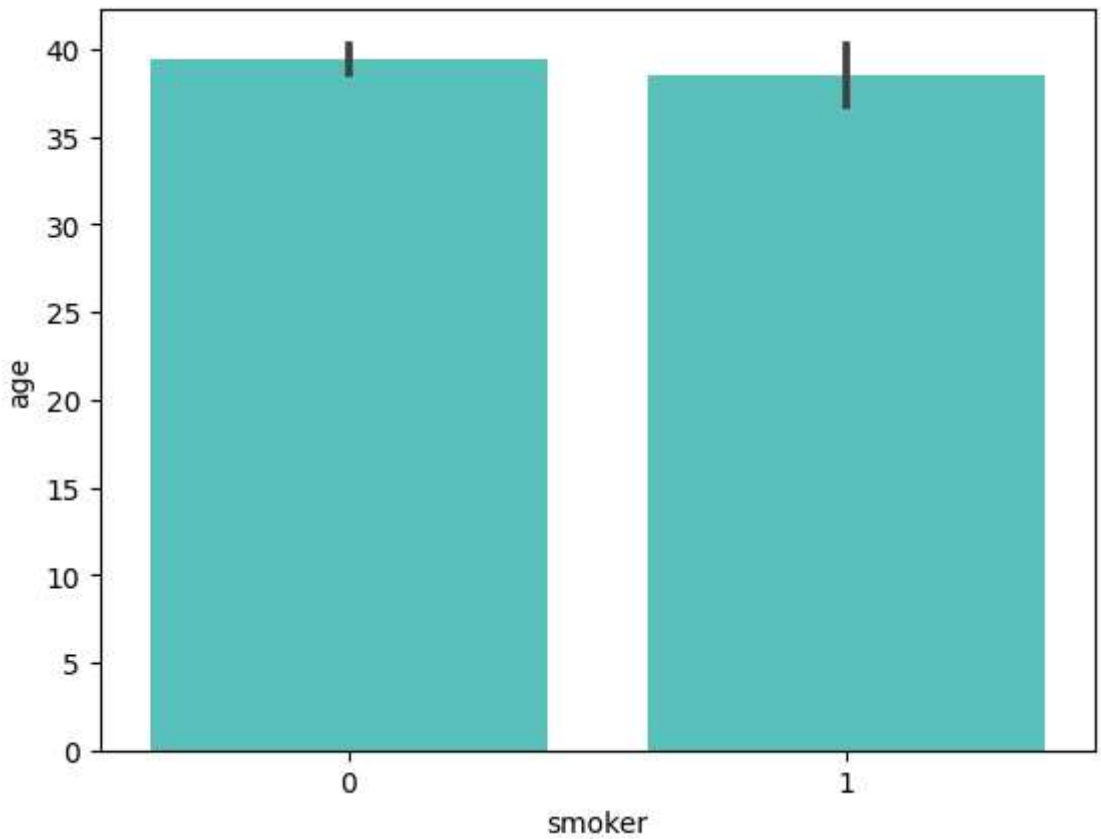
In [61]:
```
target_vector=df.iloc[:,-3]
```

In [62]:
```
print('The Feature Matrix has %d Rows and %d columns(s)'%(features_matri
print('The Target Matrix has %d Rows and %d columns(s)'%(np.array(target_
```

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

In [63]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [64]:
```
sns.barplot(x='smoker', y='age', data=df, color="mediumturquoise")
plt.show()
```



In [65]:
```
features_matrix_standardized=StandardScaler().fit_transform(features_mat
```

In [66]:
```
algorithm=LogisticRegression(max_iter=10000)
```

In [67]:
```
Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,tar
```

In [68]:
```
observation=[[1,0,0.99539,-0.0588]]
```

In [69]:
```
predictions=Logistic_Regression_Model.predict(observation)
print('The model predicted the observation to belong to class %s'%(predi
```

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

In [70]:
```
print('The algoritham was trained to predict one of the two classes:%s'%
```

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

In [71]:
```
print(" " "The Model says the probability of the observation we passed b
print()
print(" " "The Model says the probability of the observation we passed b
```

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

 The Model says the probability of the observation we passed belonging to c
lass[1] 0.19429241286686041
```

In [72]:
```
x=np.array(df['age']).reshape(-1,1)
y=np.array(df['smoker']).reshape(-1,1)
```

In [73]:
```
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\Niranjan\AppData\Local\Programs\Python\Python311\Lib\site-packages
\sklearn\utils\validation.py:1143: DataConversionWarning: 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)
```

Decision Tree

In [74]:
```
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 [75]:
```
df=pd.read_csv(r"C:\Users\Niranjan\Downloads\insurance.csv")
df
```

Out[75]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.924000
1	18	male	33.770	1	no	southeast	1725.552300
2	28	male	33.000	3	no	southeast	4449.462000
3	33	male	22.705	0	no	northwest	21984.470610
4	32	male	28.880	0	no	northwest	3866.855200
5	31	female	25.740	0	no	southeast	3756.621600
6	46	female	33.440	1	no	southeast	8240.589600
7	37	female	27.740	3	no	northwest	7281.505600
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

In [76]:
```
df.shape
```

Out[76]:
```
(1338, 7)
```

In [77]:
```
df.isnull().any()
```

Out[77]:
```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [78]:
```
df['region'].value_counts()
```

Out[78]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [79]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[79]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	yes	southwest	16884.924000
1	18	0	33.770	1	no	southeast	1725.552300
2	28	0	33.000	3	no	southeast	4449.462000
3	33	0	22.705	0	no	northwest	21984.470610
4	32	0	28.880	0	no	northwest	3866.855200
5	31	1	25.740	0	no	southeast	3756.621600
6	46	1	33.440	1	no	southeast	8240.589600
7	37	1	27.740	3	no	northwest	7281.505600
8	37	0	29.830	2	no	northeast	6406.410700
9	60	1	25.840	0	no	northwest	28923.136920
10	25	0	26.220	0	no	northeast	2721.320800
11	62	1	26.290	0	yes	southeast	27808.725100

In [80]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

Out[80]:

	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

In [81]:
```
x=["bmi","children"]
y=["Yes","No"]
all_inputs=df[x]
all_classes=df["sex"]
```

In [82]:
```
(x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,
```

In [83]:
```
clf=DecisionTreeClassifier(random_state=0)
```

In [84]:
```
clf.fit(x_train,y_train)
```

Out[84]:
```
DecisionTreeClassifier(random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [85]:
```
score=clf.score(x_test,y_test)
print(score)
```

```
0.36585365853658536
```

Random Forest

In [86]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt ,seaborn as sns
```

In [87]:
```
df=pd.read_csv(r"C:\Users\Niranjan\Downloads\insurance.csv")
df
```

Out[87]:

	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	yes	southwest	16884.924000
1	18	male	33.770	1	no	southeast	1725.552300
2	28	male	33.000	3	no	southeast	4449.462000
3	33	male	22.705	0	no	northwest	21984.470610
4	32	male	28.880	0	no	northwest	3866.855200
5	31	female	25.740	0	no	southeast	3756.621600
6	46	female	33.440	1	no	southeast	8240.589600
7	37	female	27.740	3	no	northwest	7281.505600
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

In [88]:
```
df.shape
```

Out[88]:
```
(1338, 7)
```

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

Out[89]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [90]:
```
df['bmi'].value_counts()
```

Out[90]:
```
bmi
32.300    13
28.310     9
30.495     8
30.875     8
31.350     8
30.800     8
34.100     8
28.880     8
33.330     7
35.200     7
25.800     7
32.775     7
27.645     7
32.110     7
38.060     7
25.460     7
30.590     7
27.360     7
```

In [92]:
```
m={"sex":{"female":1,"male":0}}
df=df.replace(m)
print(df)
```

```
170    63    0  41.470         0     no  southeast  13405.390300
171    49    0  30.300         0     no  southwest   8116.680000
172    18    0  15.960         0     no  northeast   1694.796400
173    35    1  34.800         1     no  southwest   5246.047000
174    24    1  33.345         0     no  northwest   2855.437550
175    63    1  37.700         0    yes  southwest  48824.450000
176    38    0  27.835         2     no  northwest   6455.862650
177    54    0  29.200         1     no  southwest  10436.096000
178    46    1  28.900         2     no  southwest   8823.279000
179    41    1  33.155         3     no  northeast   8538.288450
180    58    0  28.595         0     no  northwest  11735.879050
181    18    1  38.280         0     no  southeast   1631.821200
182    22    0  19.950         3     no  northeast   4005.422500
183    44    1  26.410         0     no  northwest   7419.477900
184    44    0  30.690         2     no  southeast   7731.427100
185    36    0  41.895         3    yes  northeast  43753.337050
186    26    1  29.920         2     no  southeast   3981.976800
187    30    1  30.900         3     no  southwest   5325.651000
188    41    1  32.200         1     no  southwest   6775.961000
189    29    1  32.110         2     no  northwest   4922.915900
```

In [93]:
```
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 [94]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[94]:
```
RandomForestClassifier()
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [95]:
```
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 [96]:
```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rf,param_grid=params,cv=2,scoring="ac
grid_search.fit(x_train,y_train)
```

Out[96]:
```
GridSearchCV(cv=2, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [2, 3, 5, 20],
                         'min_samples_leaf': [5, 10, 20, 50, 100, 200],
                         'n_estimators': [10, 25, 30, 50, 100, 200]},
             scoring='accuracy')
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

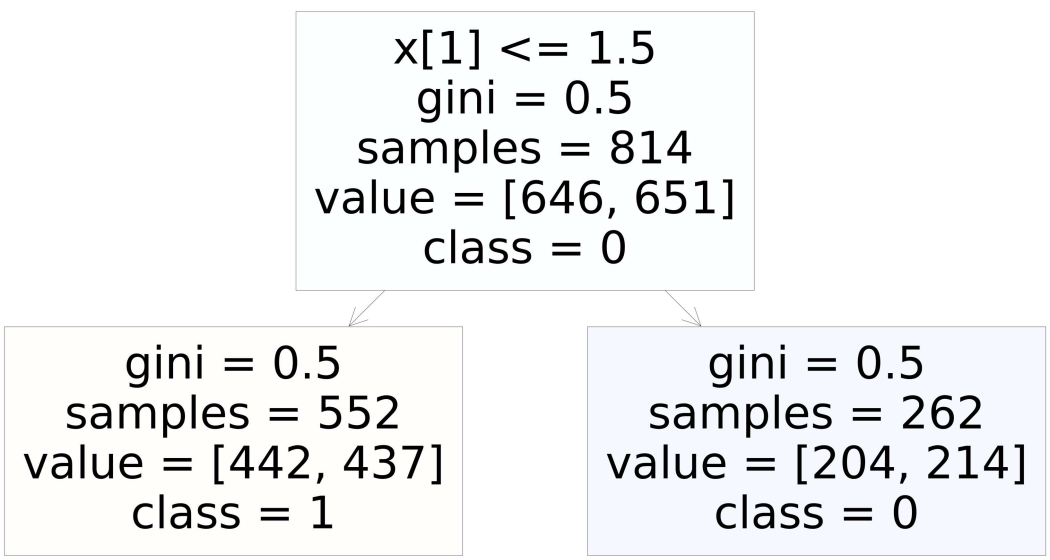
In [97]:
```
grid_search.best_score_
```

Out[97]:
```
0.5235131446289638
```

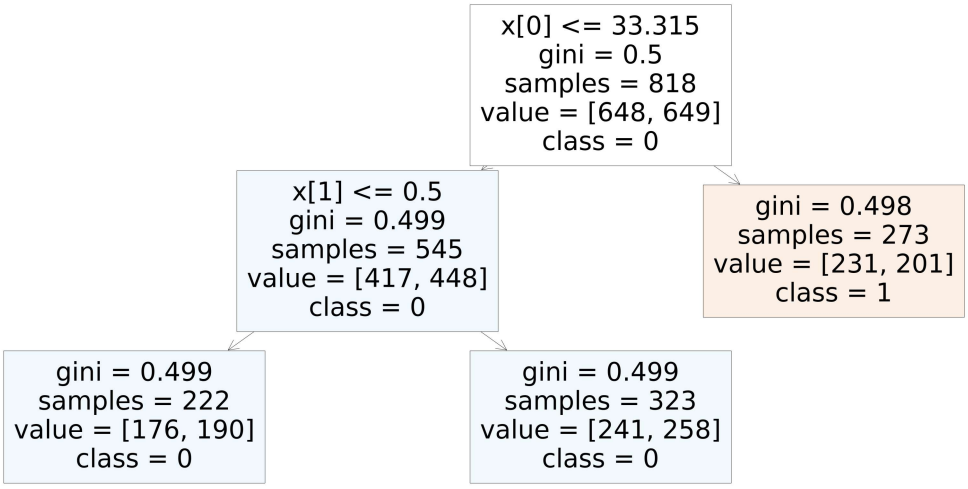
In [98]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=2, min_samples_leaf=200, n_estimators=200)
```

In [99]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[4],class_names=['1','0'],filled=True);
```



In [100]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(70,30))
plot_tree(rf_best.estimators_[6],class_names=["1","0"],filled=True);
```



In [101]:
```
rf_best.feature_importances_
```

Out[101]:
```
array([0.58866949, 0.41133051])
```

In [102]:
```
rf=RandomForestClassifier(random_state=0)
```

In [103]:
```
rf.fit(x_train,y_train)
```

Out[103]:
```
RandomForestClassifier(random_state=0)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [104]:
```
score=rf.score(x_test,y_test)
print(score)
```

```
0.2682926829268293
```

conclusion:For the given insurance data set have performed linear,logistic,random forest and decision tree models of regression and classifications.

In [ ]:

```
1
```