Problem Statement: To Predict How Best The Data Fits

1.DATA COLLECTION

In [127]:
```
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 [128]:
```
df=pd.read_csv(r"C:\Users\sneha\Downloads\insurance.csv")
df
```

Out[128]:

	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

2.DATA CLEANING AND PREPROCESSING

EXPLORATORY DATA ANALYSIS

In [129]:
```
df.head()
```

Out[129]:

	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 [130]:
```
df.tail()
```

Out[130]:

	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 [131]:
```
df.shape
```

Out[131]:
```
(1338, 7)
```

In [132]:
```
df.describe
```

Out[132]:
```
<bound method NDFrame.describe of       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
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
```

In [133]:
```
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 [134]:
```
df.isnull().any()
```

Out[134]:
```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [135]:
```
df.isna().sum()
```

Out[135]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [136]:
```
df['region'].value_counts()
```

Out[136]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [137]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[137]:

	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

In [138]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

Out[138]:

	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

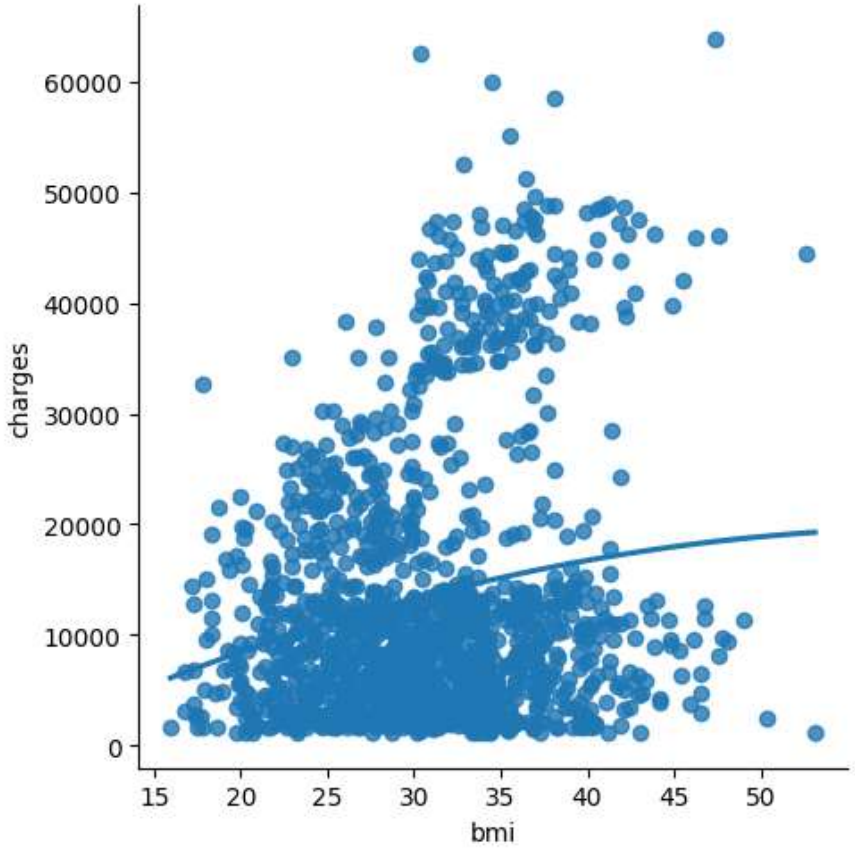
In [139]:
```
convert={"region":{"southwest":1,"southeast":2,"northwest":3,"northeast":4}}
df=df.replace(convert)
df
```

Out[139]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	1	16884.924000
1	18	0	33.770	1	0	2	1725.552300
2	28	0	33.000	3	0	2	4449.462000
3	33	0	22.705	0	0	3	21984.470610
4	32	0	28.880	0	0	3	3866.855200
5	31	1	25.740	0	0	2	3756.621600
6	46	1	33.440	1	0	2	8240.589600
7	37	1	27.740	3	0	3	7281.505600
8	37	0	29.830	2	0	4	6406.410700
9	60	1	25.840	0	0	3	28923.136920
10	25	0	26.220	0	0	4	2721.320800

3.DATA VISUALIZATION

In [140]:
```
sns.lmplot(x='bmi',y='charges',order=2,data=df,ci=None)
plt.show()
```

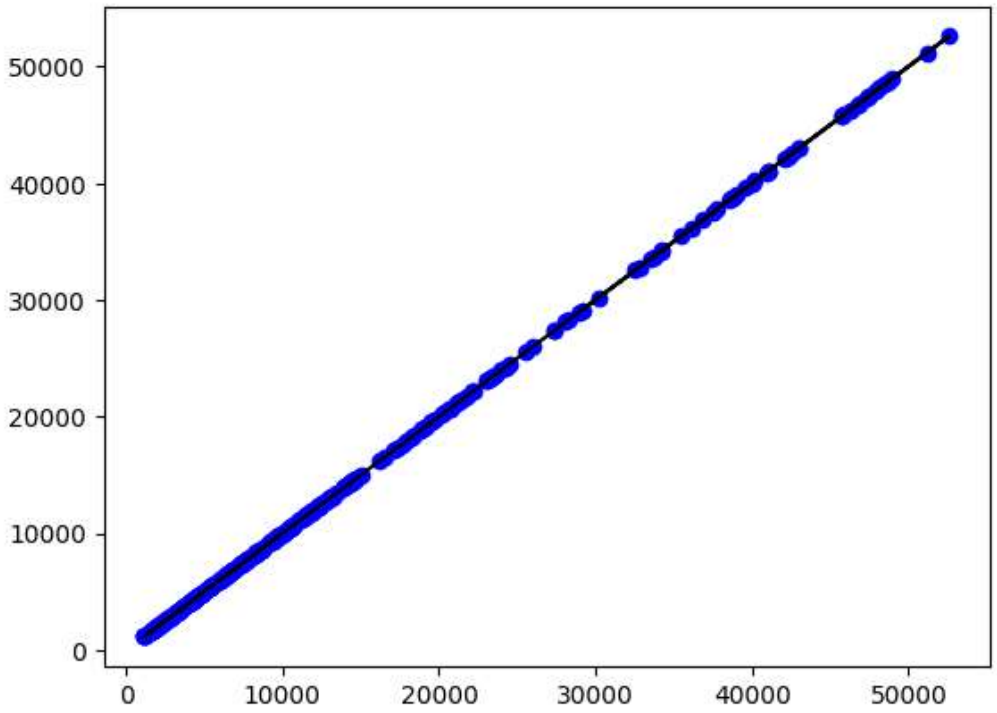


In [141]:
```
x=np.array(df['bmi']).reshape(-1,1)
y=x=np.array(df['charges']).reshape(-1,1)
```

In [142]:
```
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))
```

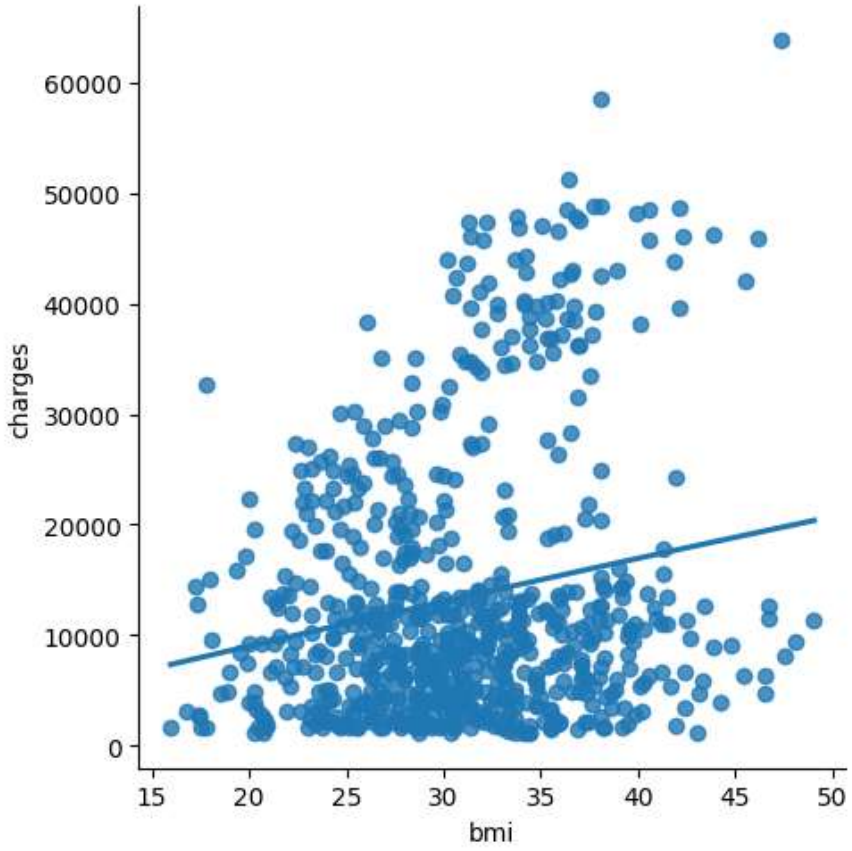
```
1.0
```

In [143]:
```
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 [144]:
```
df700=df[:][:700]
sns.lmplot(x='bmi',y='charges',order=2,ci=None,data=df700)
plt.show()
```



In [145]:
```
df700.fillna(method='ffill',inplace=True)
```

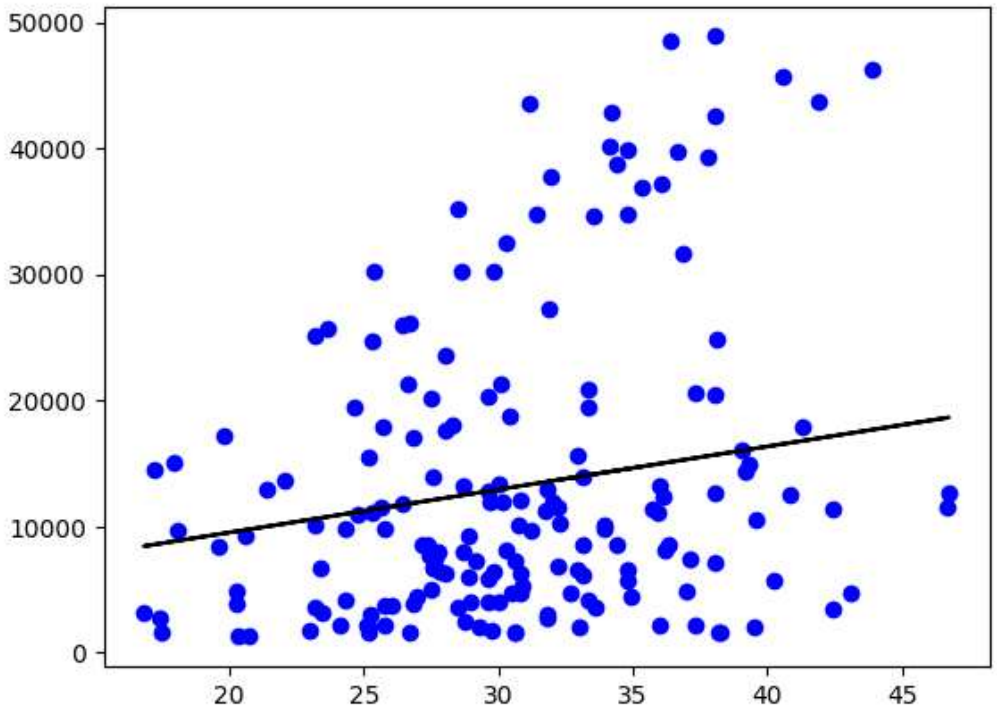
In [146]:
```
x=np.array(df700["bmi"]).reshape(-1,1)
y=np.array(df700['charges']).reshape(-1,1)
```

In [147]:
```
df700.dropna(inplace=True)
```

In [148]:
```
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))
```

```
0.06525423248224327
```

In [149]:
```
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 [150]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

In [151]:
```
lr=LinearRegression()
lr.fit(x_train,y_train)
y_pred=lr.predict(x_test)
r2=r2_score(y_test,y_pred)
print(r2)
```

```
0.06525423248224327
```

Ridge Regression

In [152]:
```
from sklearn.linear_model import Lasso,Ridge
from sklearn.preprocessing import StandardScaler
```

In [153]:
```
plt.figure(figsize=(10,10))
sns.heatmap(df700.corr(),annot=True)
plt.show()
```

In [154]:
```
features=df.columns[0:1]
target=df.columns[-1]
```

In [155]:
```
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 [156]:
```
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 [157]:
```
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))
```

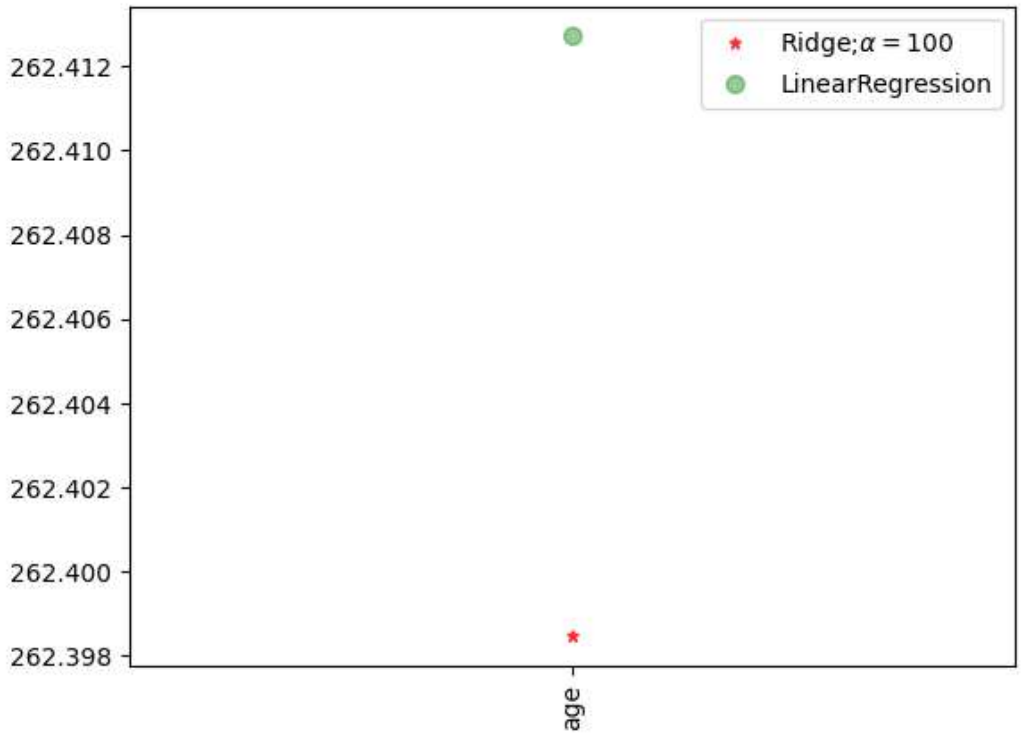
```
Ridge Model:

The train score for ridge model is 0.09109639711159634
The test score for ridge model is 0.08490538609860199
```

In [158]:
```
plt.figure(figsize=(10,10))
```

Out[158]:
```
<Figure size 1000x1000 with 0 Axes>
```

<Figure size 1000x1000 with 0 Axes>

In [159]:
```
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker="*",markersize=5
         ,color='red',label=r'Ridge;$\alpha=100$')
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 [160]:
```
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))
```

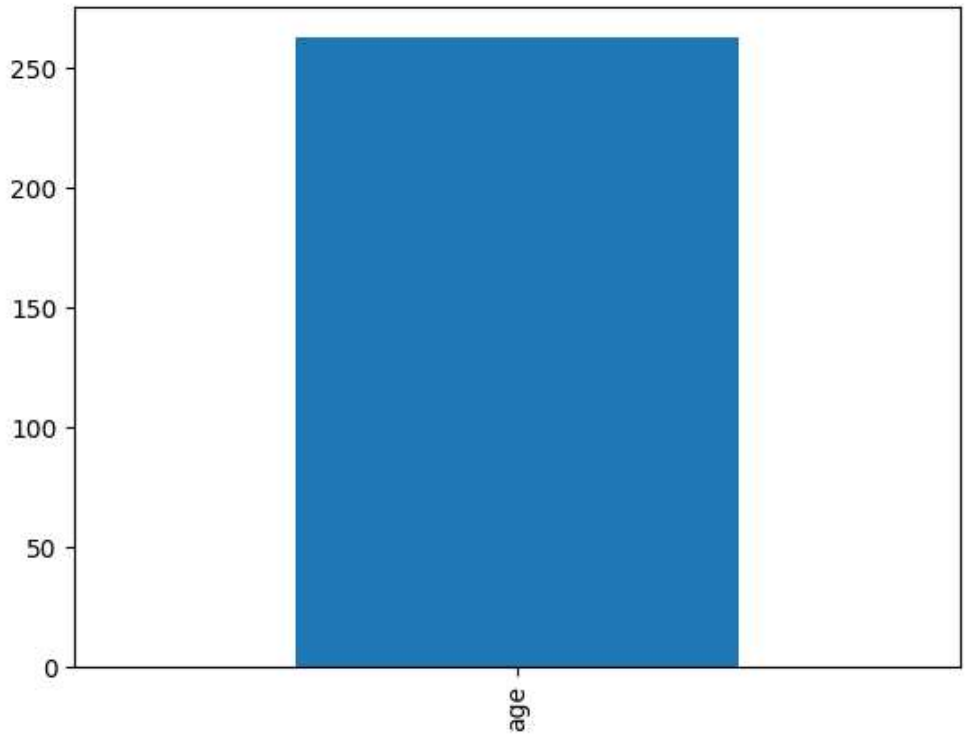
```
Ridge Model:

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

In [161]:
```
plt.figure(figsize=(10,10))
```

Out[161]:
```
<Figure size 1000x1000 with 0 Axes>
```

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

In [162]:
```
pd.Series(lasso.coef_, features).sort_values(ascending = True).plot(kind = "bar")
plt.show()
```



In [163]:
```
from sklearn.linear_model import LassoCV
```

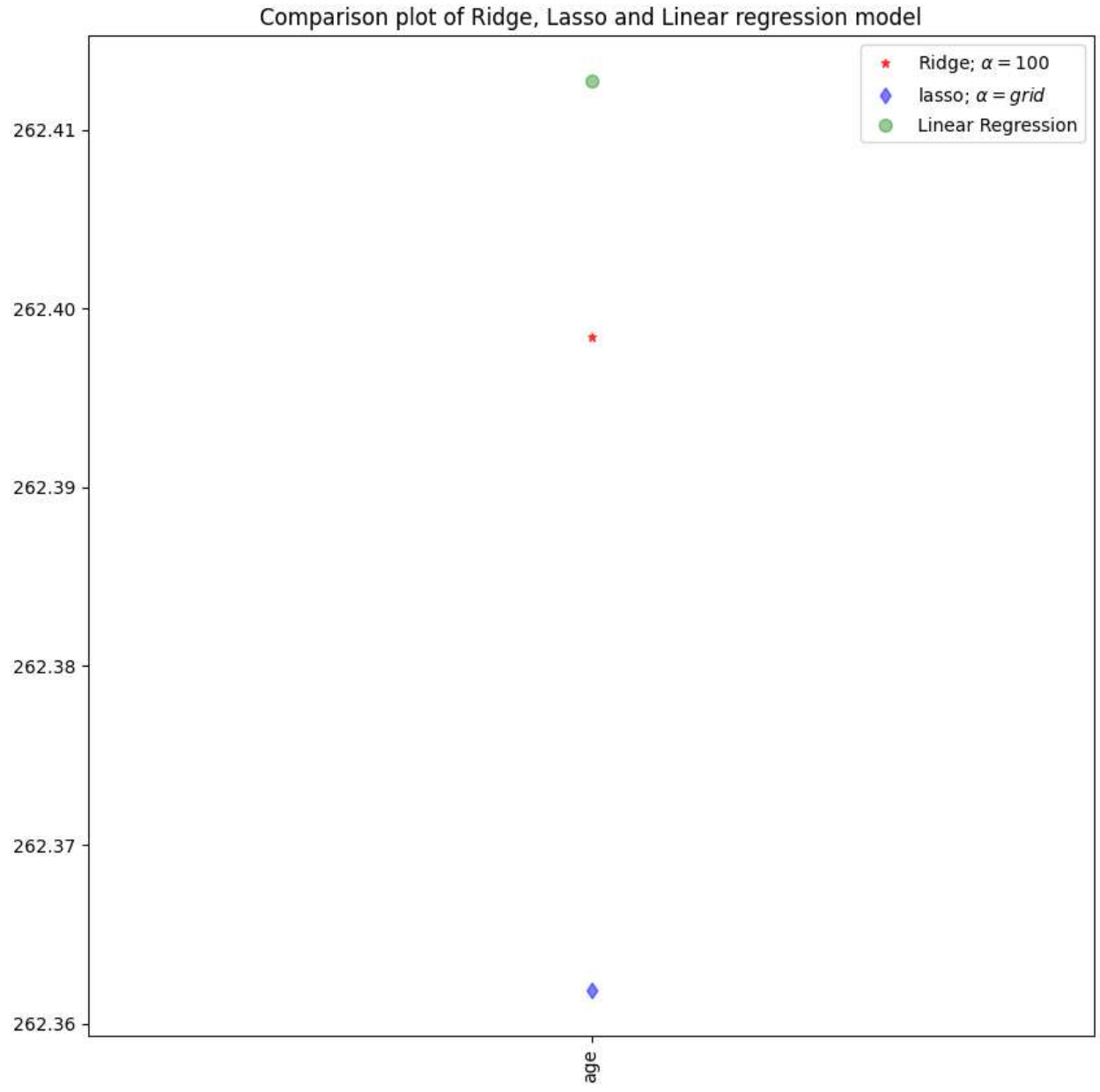
In [164]:
```
#using the linear cv model
from sklearn.linear_model import RidgeCV
#cross validation
ridge_cv=RidgeCV(alphas =[0.0001,0.001,0.01,0.1,1,10]).fit(x_train,y_train)
#score
print(ridge_cv.score(x_train,y_train))
print(ridge_cv.score(x_test,y_test))
```

```
0.09109639711159612
0.08490538609885023
```

In [165]:
```
#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))
```

```
0.09109639395809044
0.08490704421828055
```

In [166]:
```
plt.figure(figsize = (10, 10))
#add plot for ridge regression
plt.plot(features,ridgeReg.coef_,alpha=0.7,linestyle='none',marker='*',markersize=5,
         color='red',label=r'Ridge; $\alpha=100$')
#add plot for lasso regression
plt.plot(lasso_cv.coef_,alpha=0.5,linestyle='none',marker='d',markersize=6,
         color='blue',label=r'lasso; $\alpha = grid$')
#add plot for linear model
plt.plot(features,lr.coef_,alpha=0.4,linestyle='none',marker='o',markersize=7,
         color='green',label='Linear Regression')
#rotate axis
plt.xticks(rotation = 90)
plt.legend()
plt.title("Comparison plot of Ridge, Lasso and Linear regression model")
plt.show()
```



ElasticNet Regression

In [167]:
```
from sklearn.linear_model import ElasticNet
```

In [168]:
```
el=ElasticNet()
el.fit(x_train,y_train)
print(el.coef_)
print(el.intercept_)
```

```
[261.74450967]
3115.0831774262424
```

In [169]:
```
y_pred_elastic=el.predict(x_train)
```

In [170]:
```
mean_squared_error=np.mean((y_pred_elastic-y_train)**2)
print(mean_squared_error)
```

```
135077142.70714515
```

In [171]:
```
el=ElasticNet()
el.fit(x_train,y_train)
print(el.score(x_train,y_train))
```

```
0.09109580670592365
```

Logistic Regression

In [172]:
```
import numpy as np
import pandas as pd
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
```

In [173]:
```
df=pd.read_csv(r"C:\Users\sneha\Downloads\insurance.csv")
df
```

Out[173]:

	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

In [174]:
```
df.shape
```

Out[174]:
```
(1338, 7)
```

In [175]:
```
pd.set_option('display.max_rows',10000000000)
pd.set_option('display.max_columns',10000000000)
pd.set_option('display.width',95)
```

In [176]:
```
print('This Dataset has %d rows and %d columns'%(df.shape))
```

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

In [177]:
```
df.head()
```

Out[177]:

	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 [178]:
```
df.describe
```

```
284    52  female  31.200         0     no  southwest   9625.920000
285    46    male  26.620         1     no  southeast   7742.109800
286    46  female  48.070         2     no  northeast   9432.925300
287    63  female  26.220         0     no  northwest  14256.192800
288    59  female  36.765         1    yes  northeast  47896.791350
289    52    male  26.400         3     no  southeast  25992.821040
290    28  female  33.400         0     no  southwest   3172.018000
291    29    male  29.640         1     no  northeast  20277.807510
292    25    male  45.540         2    yes  southeast  42112.235600
293    22  female  28.820         0     no  southeast   2156.751800
294    25    male  26.800         3     no  southwest   3906.127000
295    18    male  22.990         0     no  northeast   1704.568100
296    19    male  27.700         0    yes  southwest  16297.846000
297    47    male  25.410         1    yes  southeast  21978.676900
298    31    male  34.390         3    yes  northwest  38746.355100
299    48  female  28.880         1     no  northwest   9249.495200
300    36    male  27.550         3     no  northeast   6746.742500
301    53  female  22.610         3    yes  northeast  24873.384900
302    56  female  37.510         2     no  southeast  12265.506900
```

In [179]:
```
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 [180]:
```
df.isnull().sum()
```

Out[180]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

In [181]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

50	18	female	35.625	0	0	northeast	2211.130750
51	21	female	33.630	2	0	northwest	3579.828700
52	48	male	28.000	1	1	southwest	23568.272000
53	36	male	34.430	0	1	southeast	37742.575700
54	40	female	28.690	3	0	northwest	8059.679100
55	58	male	36.955	2	1	northwest	47496.494450
56	58	female	31.825	2	0	northeast	13607.368750
57	18	male	31.680	2	1	southeast	34303.167200
58	53	female	22.880	1	1	southeast	23244.790200
59	34	female	37.335	2	0	northwest	5989.523650
60	43	male	27.360	3	0	northeast	8606.217400
61	25	male	33.660	4	0	southeast	4504.662400
62	64	male	24.700	1	0	northwest	30166.618170

In [182]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

72	53	1	28.100	3	0	southwest	11741.726000
73	58	0	32.010	1	0	southeast	11946.625900
74	44	0	27.400	2	0	southwest	7726.854000
75	57	0	34.010	0	0	northwest	11356.660900
76	29	1	29.590	1	0	southeast	3947.413100
77	21	0	35.530	0	0	southeast	1532.469700
78	22	1	39.805	0	0	northeast	2755.020950
79	41	1	32.965	0	0	northwest	6571.024350
80	31	0	26.885	1	0	northeast	4441.213150
81	45	1	38.285	0	0	northeast	7935.291150
82	22	0	37.620	1	1	southeast	37165.163800
83	48	1	41.230	4	0	northwest	11033.661700
84	37	1	34.800	2	1	southwest	39836.519000

In [183]:
```
convert={"region":{"southeast":1,"southwest":2,"northeast":3,"northwest":4}}
df=df.replace(convert)
df
```

Out[183]:

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	2	16884.924000
1	18	0	33.770	1	0	1	1725.552300
2	28	0	33.000	3	0	1	4449.462000
3	33	0	22.705	0	0	4	21984.470610
4	32	0	28.880	0	0	4	3866.855200
5	31	1	25.740	0	0	1	3756.621600
6	46	1	33.440	1	0	1	8240.589600
7	37	1	27.740	3	0	4	7281.505600
8	37	0	29.830	2	0	3	6406.410700
9	60	1	25.840	0	0	4	28923.136920
10	25	0	26.220	0	0	3	2721.320800

In [184]:
```
features_matrix=df.iloc[:,0:4]
```

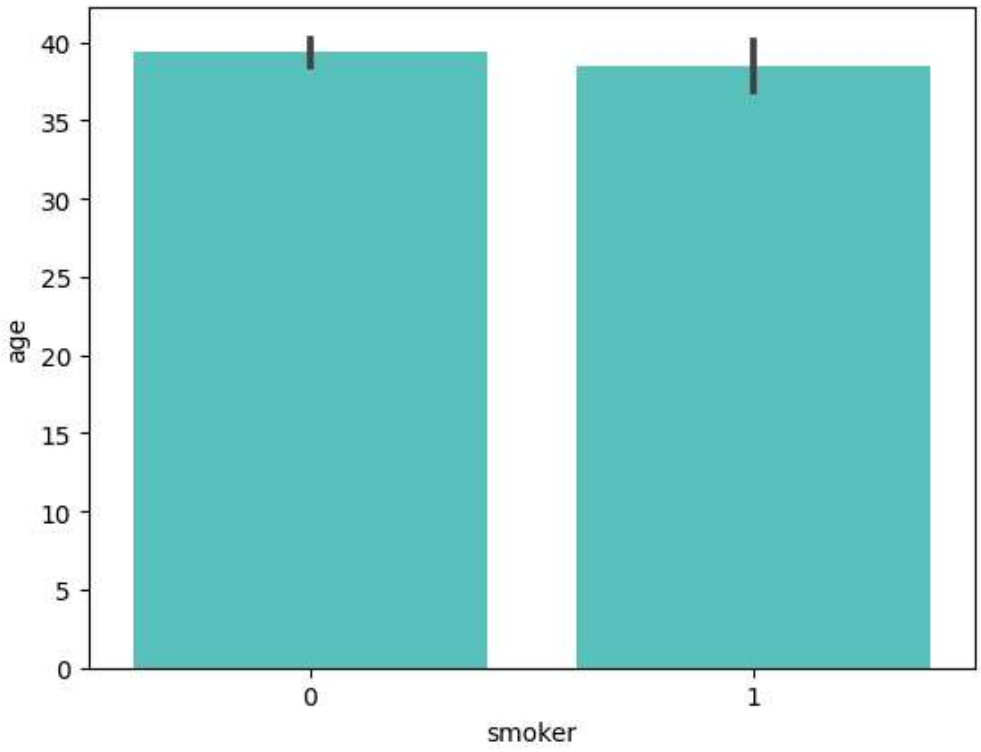
In [185]:
```
target_vector=df.iloc[:,-3]
```

In [186]:
```
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 [187]:
```
import matplotlib.pyplot as plt
import seaborn as sns
```

In [188]:
```
sns.barplot(x='smoker', y='age', data=df, color="mediumturquoise")
plt.show()
```



In [189]:
```
features_matrix_standardized=StandardScaler().fit_transform(features_matrix)
```

In [190]:
```
algorithm=LogisticRegression(max_iter=10000)
```

In [191]:
```
Logistic_Regression_Model=algorithm.fit(features_matrix_standardized,target_vector)
```

In [192]:
```
observation=[[1,0,0.99539,-0.0588]]
```

In [193]:
```
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 [194]:
```
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 [195]:
```
print(" " "The Model says the probability of the observation we passed belonging to class[0]
      %(algorithm.predict_proba(observation)[0][0]))
print()
```

```
 The Model says the probability of the observation we passed belonging to class[0] 0.805707
5871331396
```

In [196]:
```
print(" " "The Model says the probability of the observation we passed belonging to class['g
      %(algorithm.predict_proba(observation)[0][1]))
```

```
 The Model says the probability of the observation we passed belonging to class['g'] Is 0.1
9429241286686041
```

In [197]:
```
x=np.array(df['age']).reshape(-1,1)
y=np.array(df['smoker']).reshape(-1,1)
```

In [198]:
```
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))
```

```
0.7910447761194029

C:\Users\sneha\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\utils\vali
dation.py:1143: DataConversionWarning: A column-vector y was passed when a 1d array was exp
ected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Desicion Tree

In [199]:
```
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 [200]:
```
df=pd.read_csv(r"C:\Users\sneha\Downloads\insurance.csv")
df
```

74	44	male	27.400	2	no	southwest	7726.854000
75	57	male	34.010	0	no	northwest	11356.660900
76	29	female	29.590	1	no	southeast	3947.413100
77	21	male	35.530	0	no	southeast	1532.469700
78	22	female	39.805	0	no	northeast	2755.020950
79	41	female	32.965	0	no	northwest	6571.024350
80	31	male	26.885	1	no	northeast	4441.213150
81	45	female	38.285	0	no	northeast	7935.291150
82	22	male	37.620	1	yes	southeast	37165.163800
83	48	female	41.230	4	no	northwest	11033.661700
84	37	female	34.800	2	yes	southwest	39836.519000
85	45	male	22.895	2	yes	northwest	21098.554050
86	57	female	31.160	0	yes	northwest	43578.939400

In [201]:
```
df.shape
```

Out[201]:
```
(1338, 7)
```

In [202]:
```
df.isnull().any()
```

Out[202]:
```
age         False
sex         False
bmi         False
children    False
smoker      False
region      False
charges     False
dtype: bool
```

In [203]:
```
df['region'].value_counts()
```

Out[203]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [204]:
```
convert={"sex":{"female":1,"male":0}}
df=df.replace(convert)
df
```

159	50	1	27.830	3	no	southeast	19749.383380
160	42	1	26.600	0	yes	northwest	21348.706000
161	18	1	36.850	0	yes	southeast	36149.483500
162	54	0	39.600	1	no	southwest	10450.552000
163	32	1	29.800	2	no	southwest	5152.134000
164	37	0	29.640	0	no	northwest	5028.146600
165	47	0	28.215	4	no	northeast	10407.085850
166	20	1	37.000	5	no	southwest	4830.630000
167	32	1	33.155	3	no	northwest	6128.797450
168	19	1	31.825	1	no	northwest	2719.279750
169	27	0	18.905	3	no	northeast	4827.904950
170	63	0	41.470	0	no	southeast	13405.390300
171	49	0	30.300	0	no	southwest	8116.680000

In [205]:
```
convert={"smoker":{"yes":1,"no":0}}
df=df.replace(convert)
df
```

Out[205]:

	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

In [206]:
```
x=["bmi","children"]
y=["Yes","No"]
all_inputs=df[x]
all_classes=df["sex"]
```

In [207]:
```
(x_train,x_test,y_train,y_test)=train_test_split(all_inputs,all_classes,test_size=0.03)
```

In [208]:
```
clf=DecisionTreeClassifier(random_state=0)
```

In [209]:
```
clf.fit(x_train,y_train)
```

Out[209]:
```
DecisionTreeClassifier
DecisionTreeClassifier(random_state=0)
```

In [210]:
```
score=clf.score(x_test,y_test)
print(score)
```

```
0.4634146341463415
```

Random Forest

In [211]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt ,seaborn as sns
```

In [212]:
```
df=pd.read_csv(r"C:\Users\sneha\Downloads\insurance.csv")
df
```

Out[212]:

	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

In [213]:
```
df.shape
```

Out[213]:
```
(1338, 7)
```

In [214]:
```
df['region'].value_counts()
```

Out[214]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [215]:
```
df['bmi'].value_counts()
```

Out[215]:
```
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 [216]:
```
m={"sex":{"female":1,"male":0}}
df=df.replace(m)
print(df)
```

```
      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
12     23    0  34.400         0     no  southwest   1826.843000
13     56    1  39.820         0     no  southeast  11090.717800
14     27    0  42.130         0    yes  southeast  39611.757700
15     19    0  24.600         1     no  southwest   1837.237000
16     52    1  30.780         1     no  northeast  10797.336200
17     23    0  23.845         0     no  northeast   2395.171550
```

In [217]:
```
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 [218]:
```
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier()
rfc.fit(x_train,y_train)
```

Out[218]:
```
▾ RandomForestClassifier

RandomForestClassifier()
```

In [219]:
```
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 [220]:
```
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[220]:

▸ GridSearchCV

▸ estimator: RandomForestClassifier

▸ RandomForestClassifier

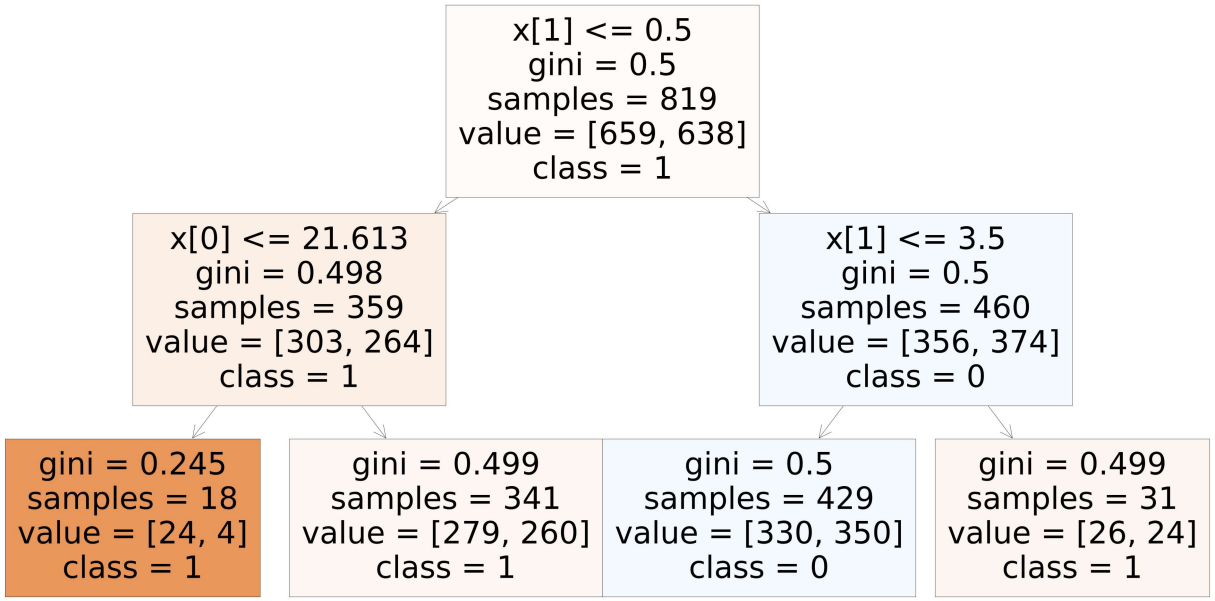
In [221]:
```
grid_search.best_score_
```

Out[221]:
```
0.5258267705301604
```

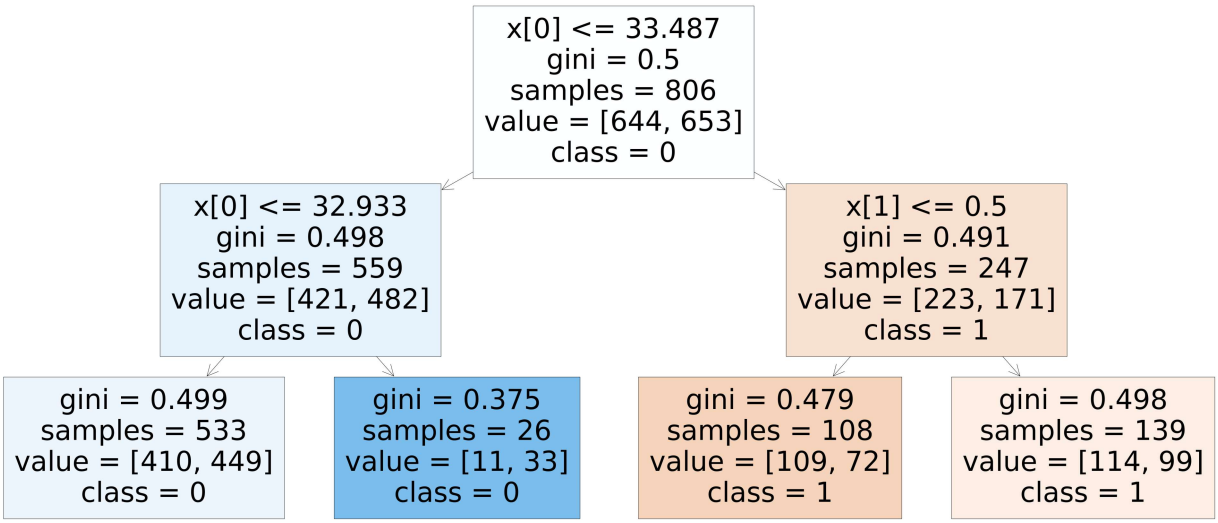
In [222]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=2, min_samples_leaf=5, n_estimators=10)
```

In [223]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[4],class_names=['1','0'],filled=True);
```



In [224]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(70,30))
plot_tree(rf_best.estimators_[6],class_names=["1","0"],filled=True);
```



In [225]:
```
rf_best.feature_importances_
```

Out[225]:
```
array([0.59997769, 0.40002231])
```

In [226]:
```
rf=RandomForestClassifier(random_state=0)
```

In [227]:
```
rf.fit(x_train,y_train)
```

Out[227]:
```
RandomForestClassifier
RandomForestClassifier(random_state=0)
```

In [228]:
```
score=rf.score(x_test,y_test)
print(score)
```

```
0.4878048780487805
```

Conclusion:

For the given insurance dataset we have performed Linear,Logistic,Random Forest and Desicion Tree models of regression and classification and have concluded that the most accuracy is occured in LogisticRegression i.e 83 percent when compare to other Regression models.And concluded that " LOGISTIC REGRESSION" model best fits for the data.

In [ ]:
```
```