MINI PROJECT

PROBLEM STATEMENT:WHICH METHOD IS SUITABLE FOR INSURANCE DATASET

In [63]:

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

# DATA COLLECTION

# READ THE DATA

In [64]:

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

Out[64]:

	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

# DATA CLEANING AND PREPROCESSING

In [65]:
```
df.head()
```

Out[65]:

	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 [66]:
```
df.tail()
```

Out[66]:

	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 [67]:
```
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 [68]:
```
df.shape
```

Out[68]:
```
(1338, 7)
```

In [69]:
```
df.describe()
```

Out[69]:

	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 [70]:
```
df['age'].value_counts()
```

Out[70]:
```
18    69
19    68
20    29
51    29
45    29
46    29
47    29
48    29
50    29
52    29
28    28
54    28
21    28
27    28
26    28
49    28
25    28
24    28
23    28
22    28
53    28
42    27
44    27
43    27
41    27
40    27
31    27
30    27
29    27
56    26
34    26
33    26
32    26
57    26
55    26
35    25
59    25
58    25
36    25
39    25
38    25
37    25
60    23
61    23
62    23
63    23
64    22
Name: age, dtype: int64
```

```
In [71]: df['bmi'].value_counts()

Out[71]: 32.300    13
         28.310     9
         30.800     8
         34.100     8
         28.880     8
                   ..
         44.745     1
         26.070     1
         27.300     1
         37.715     1
         29.200     1
         Name: bmi, Length: 548, dtype: int64
```

```
In [72]: 
         df['children'].value_counts()

Out[72]: 0    574
         1    324
         2    240
         3    157
         4     25
         5     18
         Name: children, dtype: int64
```

```
In [73]: df['charges'].value_counts()

Out[73]: 1639.56310     2
         11987.16820    1
         7624.63000     1
         12523.60480    1
         10355.64100    1
                       ..
         62592.87309    1
         18903.49141    1
         8538.28845     1
         11165.41765    1
         60021.39897    1
         Name: charges, Length: 1337, dtype: int64
```

```
In [74]: df['smoker'].value_counts()

Out[74]: no     1064
         yes     274
         Name: smoker, dtype: int64
```

```
In [75]: df['sex'].value_counts()

Out[75]: male      676
         female    662
         Name: sex, dtype: int64
```

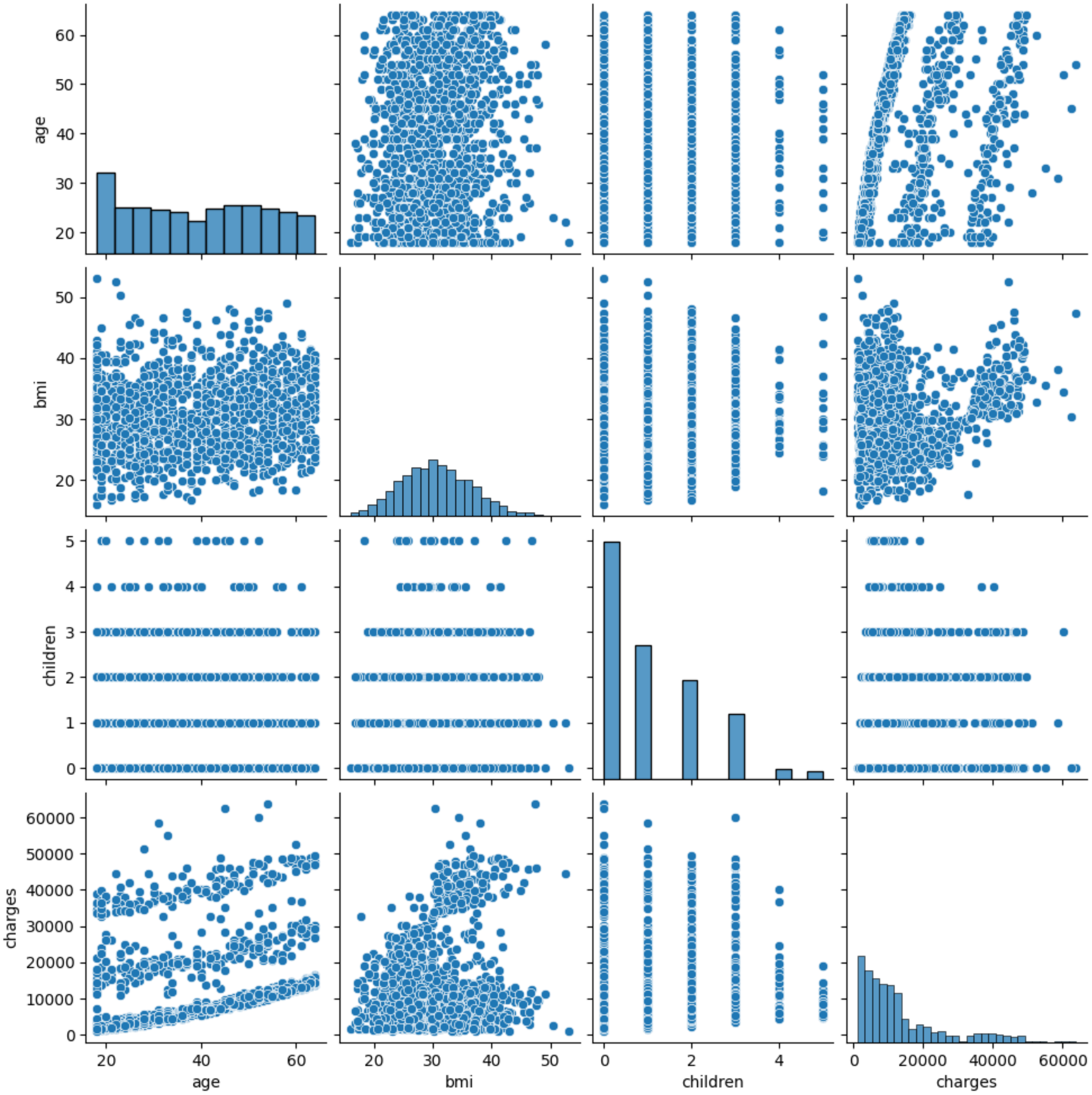
In [76]:
```
df.isnull().sum()
```

Out[76]:
```
age         0
sex         0
bmi         0
children    0
smoker      0
region      0
charges     0
dtype: int64
```

# DATA VISUALIZATION

In [77]:
```
sns.pairplot(df)
```

Out[77]:
```
<seaborn.axisgrid.PairGrid at 0xf7a3699488>
```



In [78]:

```
df.columns
```

Out[78]:

```
Index(['age', 'sex', 'bmi', 'children', 'smoker', 'region', 'charges'], dtype='object')
```

In [79]:

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

Out[79]:

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

1338 rows × 7 columns

In [80]:

```
region={"region":{"southwest":1,"southeast":0,"northwwest":2}}
df=df.replace(region)
df
```

Out[80]:

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

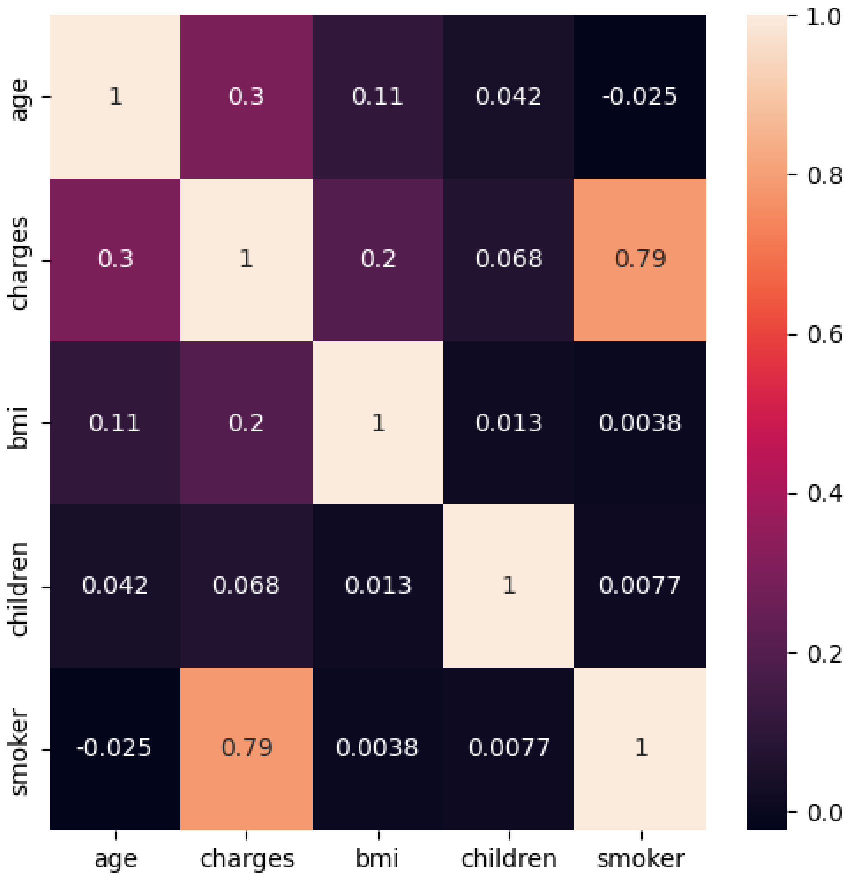
1338 rows × 7 columns

In [81]:

```
idf=df[['age', 'charges', 'bmi', 'children', 'smoker', 'sex']]
plt.figure(figsize=(6,6))
sns.heatmap(idf.corr(),annot=True)
```

Out[81]:

```
<AxesSubplot:>
```



# Feature Scaling : To Split the data into training data and test data

In [82]:

```

x=df[['age', 'sex', 'bmi', 'children', 'smoker']]
y=df[['charges']]
```

In [83]:

```
x=np.array(df['age']).reshape(-1,1)
y=np.array(df['bmi']).reshape(-1,1)
```

In [84]:

```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25)
regr=LinearRegression()
regr.fit(x_train,y_train)
print(regr.score(x_test,y_test))
```

```
-0.01865680701646455
```

In [85]:
```
print(regr.intercept_)
```

```
[28.19476708]
```

In [86]:
```
coeff_df=pd.DataFrame(regr.coef_)
coeff_df
```

Out[86]:

	0
0	0.062626

In [ ]:

# Logistic Regression

In [87]:
```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing,svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
```

In [88]:
```
df=pd.read_csv(r"C:\Users\Mastan Reddy\Downloads\insurance.csv")
df
```

Out[88]:

	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

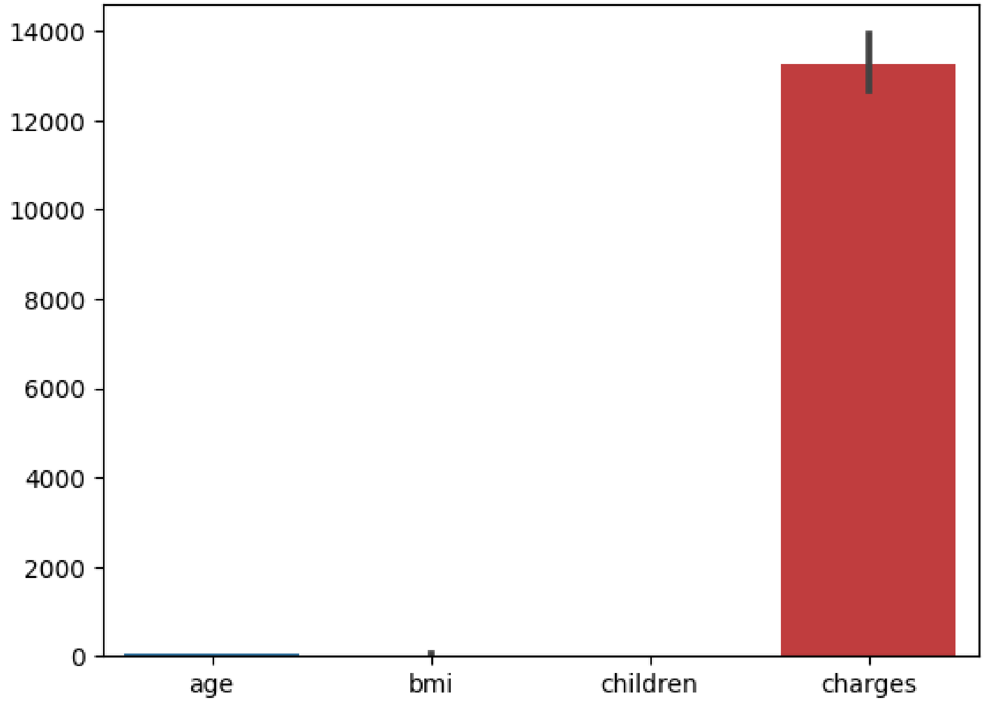
1338 rows × 7 columns

In [89]:

```
sns.barplot(df)
```

Out[89]:

```
<AxesSubplot:>
```

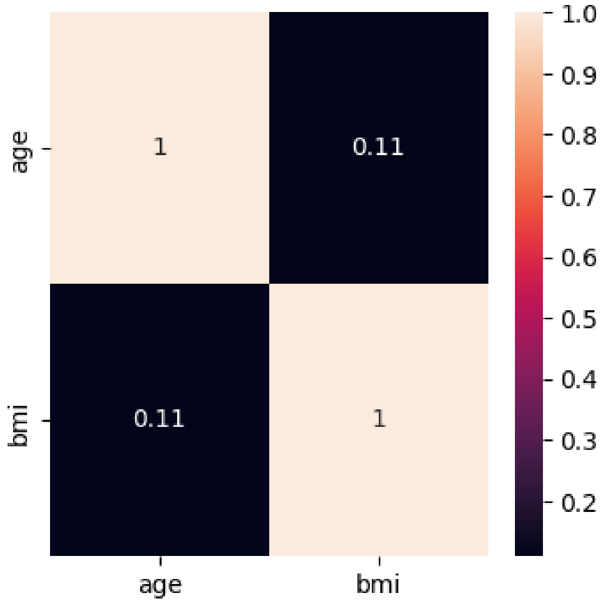


In [90]:

```
Insuranced=df[['age','bmi']]
plt.figure(figsize=(4,4))
sns.heatmap(Insuranced.corr(),annot=True)
```

Out[90]:

```
<AxesSubplot:>
```



In [91]:

```
x = df.iloc[:,:-1].values
y = df.iloc[:,1].values
```

In [92]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2)
```

In [93]:
```
ml = LogisticRegression()
```

In [94]:
```
x=np.array(df['smoker']).reshape(-1,1)
x=np.array(df['age']).reshape(-1,1)
df.dropna(inplace=True)
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.25,random_state=1)
from sklearn.linear_model import LogisticRegression
lr=LogisticRegression(max_iter=10000)
```

In [95]:
```
lr.fit(x_train,y_train)
```

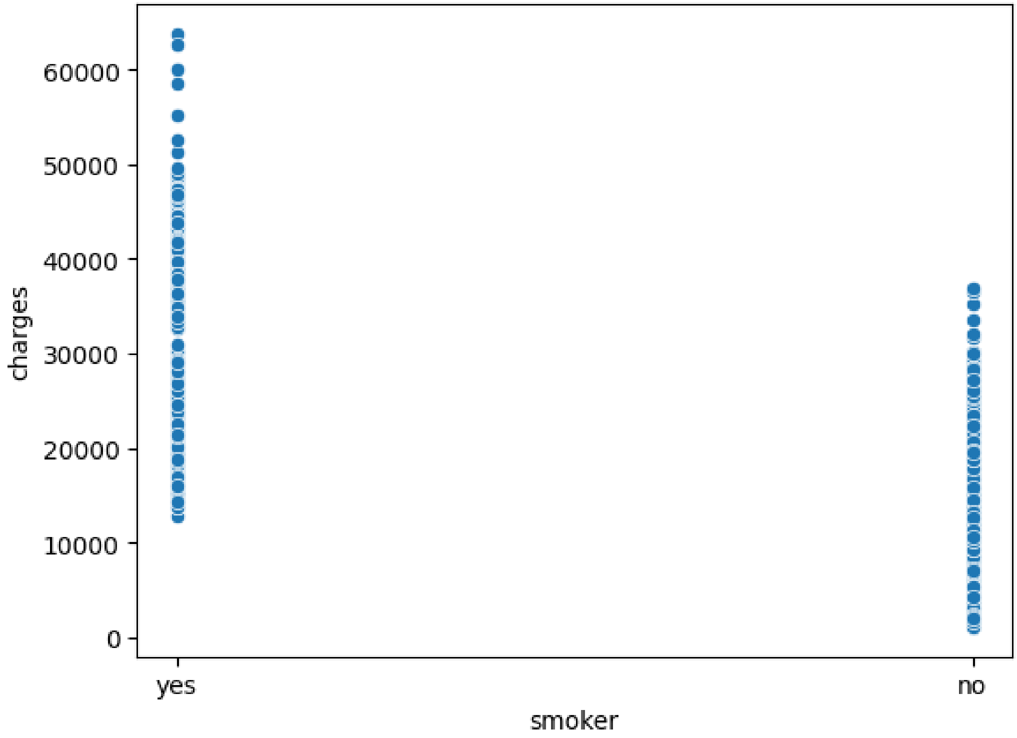
Out[95]:
```
LogisticRegression(max_iter=10000)
```

In [96]:
```
score=lr.score(x_test,y_test)
print(score)
```

```
0.48059701492537316
```

In [97]:
```
sns.scatterplot(data=df,x='smoker',y='charges')
```

Out[97]:
```
<AxesSubplot:xlabel='smoker', ylabel='charges'>
```



# Decesion Tree

In [98]:
```
# Decision Tree
from sklearn.tree import DecisionTreeClassifier
clf=DecisionTreeClassifier()
clf.fit(x_train,y_train)
```

Out[98]:
```
DecisionTreeClassifier()
```

In [ ]:

In [99]:
```
convert={'sex':{'female':0,'male':1}}
df=df.replace(convert)
df
```

Out[99]:

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

1338 rows × 7 columns

In [100]:
```
X=['age','sex']
y=['yes','no']
all_inputs=df[X]
all_classes=df['smoker']
```

In [101]:
```
X_train,x_test,y_train,y_test=train_test_split(all_inputs,all_classes,test_size=0.7)
```

In [102]:
```
clf=DecisionTreeClassifier(random_state=0)
```

In [103]:
```
clf.fit(X_train,y_train)
```

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

In [104]:
```
score=clf.score(X_train,y_train)
print(score)
```

```
0.8054862842892768
```

# Random Forest

In [105]:

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

In [106]:

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

In [107]:

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

Out[107]:

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

1338 rows × 7 columns

In [108]:

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

Out[108]:

```
RandomForestClassifier()
```

In [109]:

```
score=rfc.score(x_test,y_test)
print(score)
```

```
0.7417289220917823
```

In [110]:

```
params={'max_depth':[2,3,5,10,20],
 'min_samples_leaf':[5,10,20,50,100,200],
 'n_estimators':[10,25,30,50,100,200]}
```

In [111]:

```
from sklearn.model_selection import GridSearchCV
grid_search=GridSearchCV(estimator=rfc,param_grid=params,cv=2,scoring="accuracy")
grid_search.fit(X_train,y_train)
```

Out[111]:

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

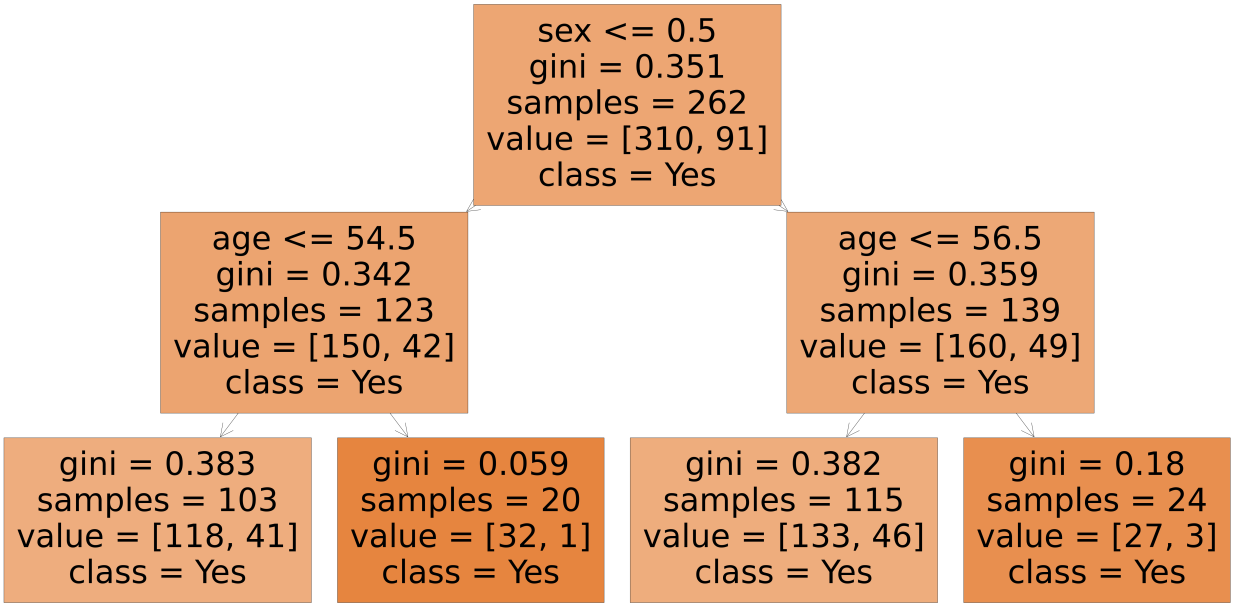
In [112]:
```
grid_search.best_score_
```

Out[112]:
```
0.7755597014925373
```

In [113]:
```
rf_best=grid_search.best_estimator_
```

In [114]:
```
from sklearn.tree import plot_tree
from sklearn.tree import DecisionTreeClassifier
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[5],feature_names=x.columns,class_names=['Yes','No'],filled=True
```

Out[114]:
```
[Text(0.5, 0.8333333333333334, 'sex <= 0.5\ngini = 0.351\nsamples = 262\nvalue = [310, 91]\n
class = Yes'),
 Text(0.25, 0.5, 'age <= 54.5\ngini = 0.342\nsamples = 123\nvalue = [150, 42]\nclass = Ye
s'),
 Text(0.125, 0.16666666666666666, 'gini = 0.383\nsamples = 103\nvalue = [118, 41]\nclass = Y
es'),
 Text(0.375, 0.16666666666666666, 'gini = 0.059\nsamples = 20\nvalue = [32, 1]\nclass = Ye
s'),
 Text(0.75, 0.5, 'age <= 56.5\ngini = 0.359\nsamples = 139\nvalue = [160, 49]\nclass = Ye
s'),
 Text(0.625, 0.16666666666666666, 'gini = 0.382\nsamples = 115\nvalue = [133, 46]\nclass = Y
es'),
 Text(0.875, 0.16666666666666666, 'gini = 0.18\nsamples = 24\nvalue = [27, 3]\nclass = Ye
s')]
```



In [115]:
```
imp_df=pd.DataFrame({"varname":X_train.columns,"Imp":rf_best.feature_importances_})
imp_df.sort_values(by="Imp",ascending=False)
```

Out[115]:

	varname	Imp
0	age	0.963412
1	sex	0.036588

# Coclusion: From the above implemented models the accuracy score is high in "Decision Tree"so it is the best model

```
In [ ]:
```