INSURANCE DATASET

1. PROBLEM STATEMENT:To predict and analyze the Female and Male Smoker in the region

In [2]:
```
import pandas as pd
import numpy as np
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

DATA COLLECTION

In [3]:
```
df=pd.read_csv(r"C:\Users\91756\Documents\python\insurance.csv")
df
```

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
...	...	...	...	...	...	...	...
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 [4]:
```
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 [5]:
```
df['region'].value_counts()
```

Out[5]:
```
region
southeast    364
southwest    325
northwest    325
northeast    324
Name: count, dtype: int64
```

In [6]:
```
convert={'sex':{"female":1,"male":0}}
df=df.replace(convert)
df
```

Out[6]:

	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 [7]:
```
convert={'region':{"southeast":1,"southwest":2,"northwest":3,"northeast":4}}
df=df.replace(convert)
df
```

Out[7]:

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

1338 rows × 7 columns

In [8]:
```
x=['sex','bmi','children','region','charges']
y=["yes","no"]
```

In [9]:
```
all_inputs=df[x]
all_classes=df['smoker']
x_train,x_test,y_train,y_test=train_test_split(all_inputs,all_classes,train_si
```

In [10]:
```
dc=DecisionTreeClassifier()
dc.fit(x_train,y_train)
```

Out[10]:
```
DecisionTreeClassifier
DecisionTreeClassifier()
```

In [11]:
```
dc.score(x_test,y_test)
```

Out[11]:
```
0.9601990049751243
```

RANDOM FOREST

using insurance dataset

In [12]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [13]:
```
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
```

Out[13]:
```
RandomForestClassifier
RandomForestClassifier()
```

In [14]:
```
rf=RandomForestClassifier()
params={'max_depth':[2,3,4,5,6],'min_samples_leaf':[5,10,15,20,50,100],'n_esti
```

In [15]:
```
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[15]:
```
GridSearchCV
estimator: RandomForestClassifier
RandomForestClassifier
```

In [16]:
```
grid_search.best_score_
```

Out[16]:
```
0.9497863247863247
```

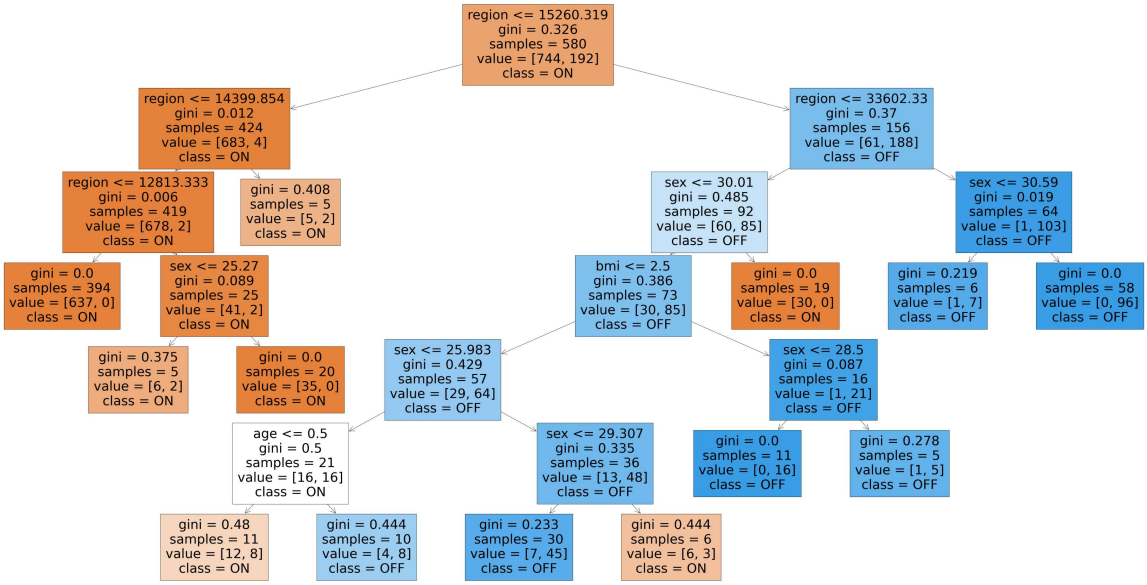
In [17]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=6, min_samples_leaf=5, n_estimators=33)
```

In [18]:
```
x=df.drop('smoker',axis=1)
y=df['smoker']
```

In [19]:
```
from sklearn.tree import plot_tree
from sklearn.tree import DecisionTreeClassifier
import matplotlib.pyplot as plt
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[5],feature_names=x.columns,class_names=['ON','OF
```

Out[19]:
```
[Text(0.46774193548387094, 0.9285714285714286, 'region <= 15260.319\ngini =
0.326\nsamples = 580\nvalue = [744, 192]\nclass = ON'),
 Text(0.1935483870967742, 0.7857142857142857, 'region <= 14399.854\ngini =
0.012\nsamples = 424\nvalue = [683, 4]\nclass = ON'),
 Text(0.12903225806451613, 0.6428571428571429, 'region <= 12813.333\ngini =
0.006\nsamples = 419\nvalue = [678, 2]\nclass = ON'),
 Text(0.06451612903225806, 0.5, 'gini = 0.0\nsamples = 394\nvalue = [637, 0]
\nclass = ON'),
 Text(0.1935483870967742, 0.5, 'sex <= 25.27\ngini = 0.089\nsamples = 25\nva
lue = [41, 2]\nclass = ON'),
 Text(0.12903225806451613, 0.35714285714285715, 'gini = 0.375\nsamples = 5\n
value = [6, 2]\nclass = ON'),
 Text(0.25806451612903225, 0.35714285714285715, 'gini = 0.0\nsamples = 20\nv
alue = [35, 0]\nclass = ON'),
 Text(0.25806451612903225, 0.6428571428571429, 'gini = 0.408\nsamples = 5\nv
alue = [5, 2]\nclass = ON'),
 Text(0.7419354838709677, 0.7857142857142857, 'region <= 33602.33\ngini = 0.
37\nsamples = 156\nvalue = [61, 188]\nclass = OFF'),
 Text(0.6129032258064516, 0.6428571428571429, 'sex <= 30.01\ngini = 0.485\ns
amples = 92\nvalue = [60, 85]\nclass = OFF'),
 Text(0.5483870967741935, 0.5, 'bmi <= 2.5\ngini = 0.386\nsamples = 73\nvalu
e = [30, 85]\nclass = OFF'),
 Text(0.3870967741935484, 0.35714285714285715, 'sex <= 25.983\ngini = 0.429
\nsamples = 57\nvalue = [29, 64]\nclass = OFF'),
 Text(0.25806451612903225, 0.21428571428571427, 'age <= 0.5\ngini = 0.5\nsam
ples = 21\nvalue = [16, 16]\nclass = ON'),
 Text(0.1935483870967742, 0.07142857142857142, 'gini = 0.48\nsamples = 11\nv
alue = [12, 8]\nclass = ON'),
 Text(0.3225806451612903, 0.07142857142857142, 'gini = 0.444\nsamples = 10\n
value = [4, 8]\nclass = OFF'),
 Text(0.5161290322580645, 0.21428571428571427, 'sex <= 29.307\ngini = 0.335
\nsamples = 36\nvalue = [13, 48]\nclass = OFF'),
 Text(0.45161290322580644, 0.07142857142857142, 'gini = 0.233\nsamples = 30
\nvalue = [7, 45]\nclass = OFF'),
 Text(0.5806451612903226, 0.07142857142857142, 'gini = 0.444\nsamples = 6\nv
alue = [6, 3]\nclass = ON'),
 Text(0.7096774193548387, 0.35714285714285715, 'sex <= 28.5\ngini = 0.087\ns
amples = 16\nvalue = [1, 21]\nclass = OFF'),
 Text(0.6451612903225806, 0.21428571428571427, 'gini = 0.0\nsamples = 11\nva
lue = [0, 16]\nclass = OFF'),
 Text(0.7741935483870968, 0.21428571428571427, 'gini = 0.278\nsamples = 5\nv
alue = [1, 5]\nclass = OFF'),
 Text(0.6774193548387096, 0.5, 'gini = 0.0\nsamples = 19\nvalue = [30, 0]\nc
lass = ON'),
 Text(0.8709677419354839, 0.6428571428571429, 'sex <= 30.59\ngini = 0.019\ns
amples = 64\nvalue = [1, 103]\nclass = OFF'),
 Text(0.8064516129032258, 0.5, 'gini = 0.219\nsamples = 6\nvalue = [1, 7]\nc
lass = OFF'),
 Text(0.9354838709677419, 0.5, 'gini = 0.0\nsamples = 58\nvalue = [0, 96]\nc
lass = OFF')]
```



In [20]:
```
rf_best.feature_importances_
```

Out[20]:
```
array([0.0091486 , 0.09391883, 0.01104503, 0.01288412, 0.87300342])
```

In [21]:
```
df1=pd.DataFrame({'Varname':x_train.columns,'Imp':rf_best.feature_importances_
```

In [22]:
```
df1.sort_values(by='Imp',ascending=False)
```

Out[22]:

	Varname	Imp
4	charges	0.873003
1	bmi	0.093919
3	region	0.012884
2	children	0.011045
0	sex	0.009149

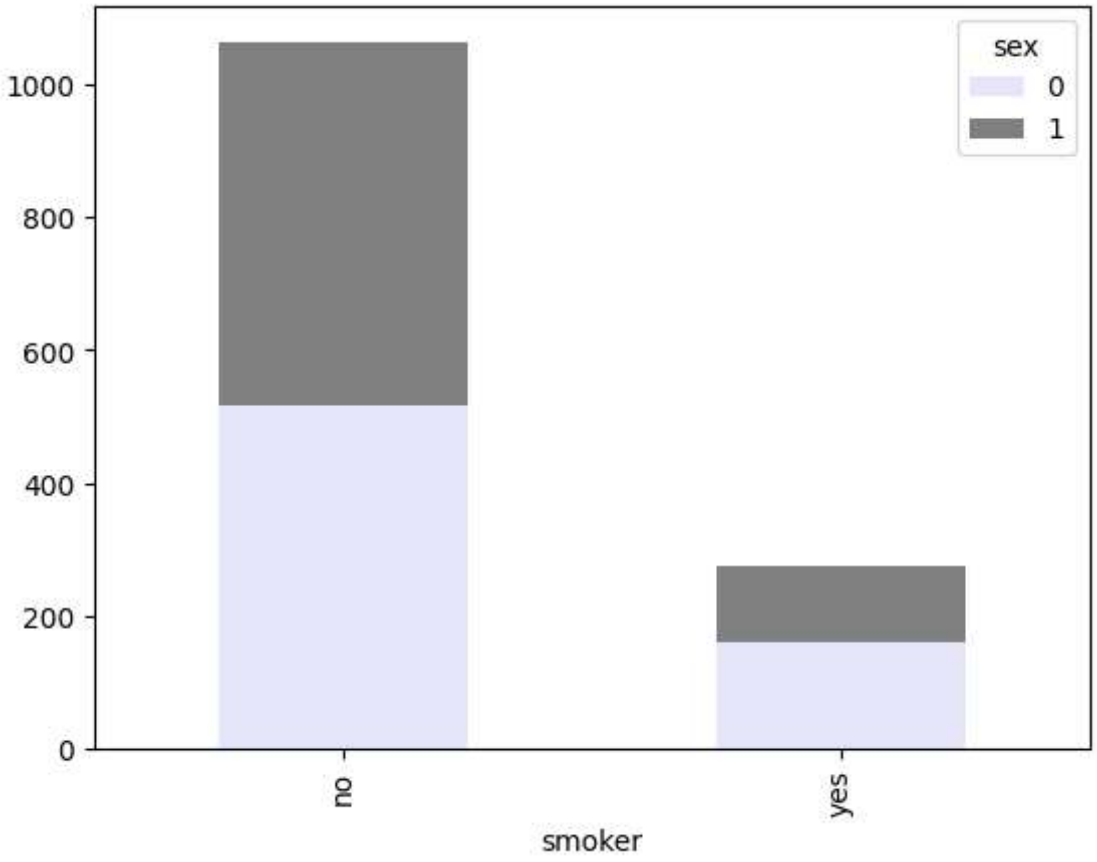
In [23]:
```
v=pd.crosstab(df['smoker'],df['sex'])
v
```

Out[23]:

sex	0	1
smoker		
no	517	547
yes	159	115

In [24]:
```
v.plot(kind='bar',stacked=True,color=["lavender","gray"],grid=False)
```

Out[24]: <Axes: xlabel='smoker'>



CONCLUSION

IN DECISION TREE THE SCORE OF X AND Y IS 94% AND IN THE RANDOM FOREST THE SCORE IS 95% COMPARING THE BOTH RANDOM FOREST IS HIGHEST IN THE ACCURACY AND AS PER THE PROBLEM STATEMENT MALE SMOKERS ARE HIGHER THEN FEMALE SMOKER

In [ ]:

In [ ]: