

Problem statement: The model which is best for the insurance

```
In [78]: import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
from sklearn.model_selection import train_test_split
```

READ THE DATASET

```
In [79]: df=pd.read_csv(r"C:\Users\pucha\Downloads\insurance.csv")
df
```

Out[79]:

	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 [98]: df.head()
```

```
Out[98]:
```

	age	sex	bmi	children	smoker	region	charges
0	19	0	27.900	0	yes	1	16884.92400
1	18	1	33.770	1	no	3	1725.55230
2	28	1	33.000	3	no	3	4449.46200
3	33	1	22.705	0	no	2	21984.47061
4	32	1	28.880	0	no	2	3866.85520

```
In [80]: ins={"sex":{"male":1,"female":0}}
df=df.replace(ins)
print(df)
```

	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 x 7 columns]
```

```
In [99]: df.tail()
```

```
Out[99]:
```

	age	sex	bmi	children	smoker	region	charges
1333	50	1	30.97	3	no	2	10600.5483
1334	18	0	31.92	0	no	4	2205.9808
1335	18	0	36.85	0	no	3	1629.8335
1336	21	0	25.80	0	no	1	2007.9450
1337	61	0	29.07	0	yes	2	29141.3603

```
In [81]: r={"region":{"southwest":1,"northwest":2,"southeast":3,"northeast":4}}
df=df.replace(r)
print(df)
```

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

```
[1338 rows x 7 columns]
```

In [122]: `df.describe()`

Out[122]:

	age	sex	bmi	children	region	charges
count	1338.000000	1338.000000	1338.000000	1338.000000	1338.000000	1338.000000
mean	39.207025	0.505232	30.663397	1.094918	2.513453	13270.422265
std	14.049960	0.500160	6.098187	1.205493	1.104915	12110.011237
min	18.000000	0.000000	15.960000	0.000000	1.000000	1121.873900
25%	27.000000	0.000000	26.296250	0.000000	2.000000	4740.287150
50%	39.000000	1.000000	30.400000	1.000000	3.000000	9382.033000
75%	51.000000	1.000000	34.693750	2.000000	3.000000	16639.912515
max	64.000000	1.000000	53.130000	5.000000	4.000000	63770.428010

FEATURE SCALLING:-Split the dataset into independent and dependent variables

Split your dataset in two catagories 1.Train data 2.Test data

In [101]: `x=["sex","bmi","children","region","charges"]`
`y=["yes","no"]`

In [109]: `all_inputs=df[x]`
`all_classes=df["smoker"]`
`x_train,x_test,y_train,y_test=train_test_split(all_inputs,all_classes,train_size=0.7)`

DECISION TREE CALSSIFIER

```
In [127]: from sklearn.tree import DecisionTreeClassifier
```

```
In [128]: dc=DecisionTreeClassifier()  
dc.fit(x_train,y_train)
```

```
Out[128]: ▾ DecisionTreeClassifier  
DecisionTreeClassifier()
```

```
In [124]: dc.score(x_test,y_test)
```

```
Out[124]: 0.9328358208955224
```

RANDOM FOREST CLASSIFIER

```
In [125]: from sklearn.ensemble import RandomForestClassifier  
rfc=RandomForestClassifier()  
rfc.fit(x_train,y_train)
```

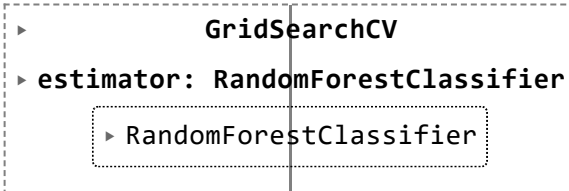
```
Out[125]: ▾ RandomForestClassifier  
RandomForestClassifier()
```

```
In [126]: rf=RandomForestClassifier()
```

```
In [107]: params={'max_depth':[2,3,5,1,20],  
                 'min_samples_leaf':[5,1,2,50,10,20],  
                 'n_estimators':[10,25,30,50,100,200]}
```

```
In [108]: 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[108]:
```



```
  ▸ GridSearchCV
  ▸ estimator: RandomForestClassifier
    ▸ RandomForestClassifier
```

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

```
Out[112]: 0.9615384615384615
```

```
In [113]: rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=20, min_samples_leaf=5)
```

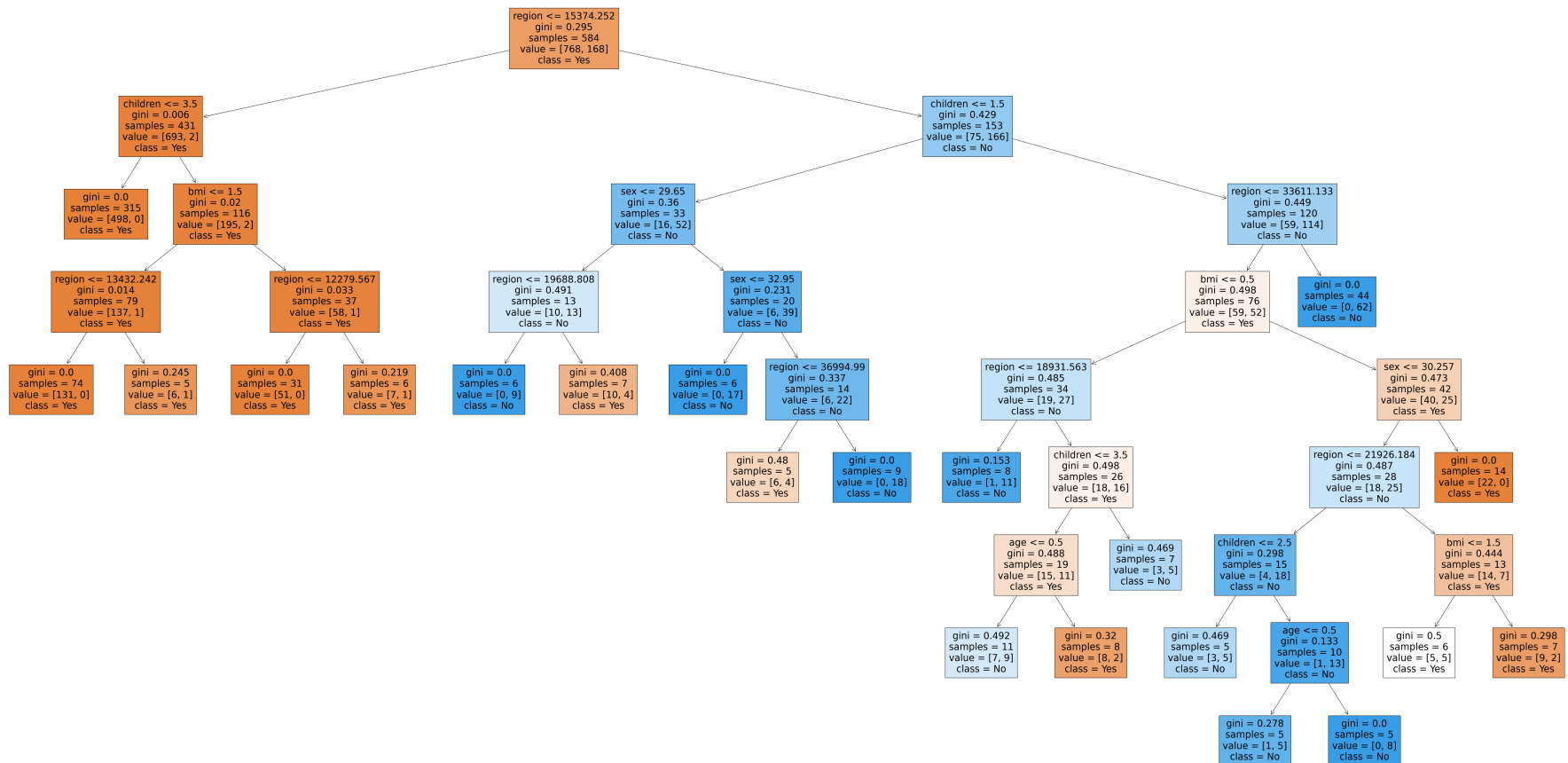
```
In [114]: x=df.drop("smoker",axis=1)
y=df["smoker"]
```

```

graph TD
    Root["sex <= 47.08  
gini = 0.354  
samples = 593  
value = [723, 215]  
class = ON"]
    Root --> Node1["age <= 0.5  
gini = 0.349  
samples = 588  
value = [710, 209]  
class = ON"]
    Root --> Node2["sex <= 0.375  
samples = 5  
value = [2, 6]  
class = OFF"]
    Node1 --> Node3["sex <= 28.39  
gini = 0.314  
samples = 299  
value = [371, 90]  
class = ON"]
    Node1 --> Node4["sex <= 22.8  
gini = 0.38  
samples = 289  
value = [148, 139]  
class = ON"]
    Node3 --> Node5["bmi <= 2.5  
gini = 0.386  
samples = 115  
value = [136, 46]  
class = ON"]
    Node3 --> Node6["region <= 28714.629  
gini = 0.257  
samples = 184  
value = [235, 42]  
class = ON"]
    Node3 --> Node7["sex <= 19.875  
gini = 0.149  
samples = 23  
value = [14, 3]  
class = ON"]
    Node4 --> Node8["region <= 15846.975  
gini = 0.394  
samples = 266  
value = [114, 116]  
class = ON"]
    Node4 --> Node9["sex <= 29.375  
gini = 0.334  
samples = 86  
value = [31, 115]  
class = OFF"]
    Node5 --> Node10["sex <= 27.623  
gini = 0.414  
samples = 98  
value = [111, 46]  
class = ON"]
    Node5 --> Node11["children <= 2.5  
gini = 0.491  
samples = 15  
value = [103, 33]  
class = OFF"]
    Node6 --> Node12["sex <= 23.323  
gini = 0.137  
samples = 17  
value = [25, 2]  
class = ON"]
    Node6 --> Node13["region <= 36994.99  
gini = 0.045  
samples = 11  
value = [1, 42]  
class = OFF"]
    Node6 --> Node14["gini = 0.469  
samples = 6  
value = [5, 3]  
class = ON"]
    Node6 --> Node15["gini = 0.0  
samples = 17  
value = [29, 0]  
class = ON"]
    Node6 --> Node16["children <= 3.5  
gini = 0.007  
samples = 140  
value = [283, 1]  
class = ON"]
    Node8 --> Node17["sex <= 1.5  
gini = 0.14  
samples = 4  
value = [4, 49]  
class = OFF"]
    Node8 --> Node18["bmi <= 1.5  
gini = 0.027  
samples = 48  
value = [73, 1]  
class = ON"]
    Node8 --> Node19["region <= 13342.304  
gini = 0.027  
samples = 23  
value = [10, 34]  
class = OFF"]
    Node8 --> Node20["gini = 0.0  
samples = 131  
value = [210, 0]  
class = ON"]
    Node8 --> Node21["gini = 0.278  
samples = 5  
value = [5, 1]  
class = ON"]
    Node9 --> Node22["region <= 22328.501  
gini = 0.412  
samples = 31  
value = [27, 66]  
class = OFF"]
    Node9 --> Node23["sex <= 30.685  
gini = 0.128  
samples = 18  
value = [127, 3]  
class = ON"]
    Node9 --> Node24["gini = 0.0  
samples = 37  
value = [0, 64]  
class = OFF"]
    Node10 --> Node25["sex <= 26.937  
gini = 0.397  
samples = 83  
value = [101, 33]  
class = ON"]
    Node10 --> Node26["children <= 3.5  
gini = 0.0  
samples = 75  
value = [90, 331]  
class = ON"]
    Node10 --> Node27["gini = 0.397  
samples = 9  
value = [3, 8]  
class = OFF"]
    Node10 --> Node28["gini = 0.486  
samples = 7  
value = [7, 5]  
class = ON"]
    Node25 --> Node29["sex <= 21.585  
gini = 0.369  
samples = 54  
value = [68, 22]  
class = ON"]
    Node25 --> Node30["gini = 0.0  
samples = 10  
value = [118, 0]  
class = ON"]
    Node25 --> Node31["bmi <= 0.5  
gini = 0.424  
samples = 44  
value = [50, 22]  
class = ON"]
    Node29 --> Node32["sex <= 26.245  
gini = 0.494  
samples = 22  
value = [23, 17]  
class = ON"]
    Node29 --> Node33["sex <= 21.405  
gini = 0.251  
samples = 5  
value = [29, 5]  
class = ON"]
    Node29 --> Node34["gini = 0.5  
samples = 5  
value = [3, 3]  
class = ON"]
    Node29 --> Node35["bmi <= 0.5  
gini = 0.5  
samples = 10  
value = [6, 6]  
class = ON"]
    Node32 --> Node36["sex <= 24.73  
gini = 0.475  
samples = 17  
value = [19, 11]  
class = ON"]
    Node32 --> Node37["gini = 0.375  
samples = 5  
value = [2, 6]  
class = OFF"]
    Node33 --> Node38["gini = 0.49  
samples = 5  
value = [4, 3]  
class = ON"]
    Node33 --> Node39["region <= 14564.805  
gini = 0.137  
samples = 17  
value = [25, 2]  
class = ON"]
    Node34 --> Node40["gini = 0.0  
samples = 6  
value = [9, 0]  
class = ON"]
    Node34 --> Node41["gini = 0.486  
samples = 5  
value = [7, 5]  
class = ON"]
    Node36 --> Node42["region <= 14278.125  
gini = 0.475  
samples = 11  
value = [17, 11]  
class = OFF"]
    Node36 --> Node43["gini = 0.0  
samples = 6  
value = [20, 0]  
class = ON"]
    Node39 --> Node44["gini = 0.0  
samples = 12  
value = [5, 2]  
class = ON"]
    Node39 --> Node45["gini = 0.408  
samples = 5  
value = [5, 2]  
class = ON"]
    Node42 --> Node46["gini = 0.245  
samples = 5  
value = [6, 13]  
class = ON"]
    Node42 --> Node47["gini = 0.165  
samples = 6  
value = [11, 10]  
class = OFF"]
  
```



```
In [116]: from sklearn.tree import plot_tree
plt.figure(figsize=(80,40))
plot_tree(rf_best.estimators_[7],feature_names=x.columns,class_names=['Yes','No'],filled=True);
```



```
In [117]: rf_best.feature_importances_
```

```
Out[117]: array([0.00799748, 0.09304862, 0.01537871, 0.0113572 , 0.87221798])
```

```
In [120]: imp_df=pd.DataFrame({"varname":x_train.columns,"Imp":rf_best.feature_importances_})
```

```
In [121]: imp_df.sort_values(by="Imp",ascending=False)
```

Out[121]:

	varname	Imp
4	charges	0.872218
1	bmi	0.093049
2	children	0.015379
3	region	0.011357
0	sex	0.007997

CONCLUSION:-

FROM THE ABOVE DATASET I HAVE TO DEFINE THAT WHICH MODEL HAS THE BEST ACCURACY FROM THE AMONG MODELS

STEP:-1. IMPORT THE LIBRARY

STEP:-2. I HAVE TO READ THE DATASET

STEP:-3. TAKE DATA CLEANING AND PREPROCESSING

STEP:-4. SPLITTING THE DATASET INTO TRAIN DATA AND TEST DATA

STEP:-5. IMPORTING THE DECISIONTREECLASSIFIER

STEP:-6. IMPORTING THE RANDOMFORESTCLASSIFIER

FROM THE DATASET DECISIONTREECLASSIFIER I GOT THE ACCURACY: 0.9328358208955224 AND
RANDOMFORESTCLASSIFIER I GOT THE ACCURACY: 0.9615384615384615.

FINALLY I CONCLUDED THAT RANDOMFORESTCLASSIFIER HAS GOT MORE ACCURACY THAN DECISIONTREECLASSIFIER.
SO THE RANDOMFORESTCLASSIFIER THE BEST FIT MODEL FOR THE INSURANCE DATASET

In []:

