Problem Statement: Which model is best fit for given dataset

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.linear_model import LogisticRegression
```

Data Collection

In [2]:
```
df=pd.read_csv(r"C:\Users\pappu\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

Data Cleaning & Preprocessing

In [3]:
```
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 [4]:
```
df.head()
```

Out[4]:

	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 [5]:
```
df.tail()
```

Out[5]:

	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 [6]:
```
df.describe()
```

Out[6]:

	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 [7]:
```
r={'smoker':{'yes':1,'no':0}}
df=df.replace(r)
df
```

Out[7]:

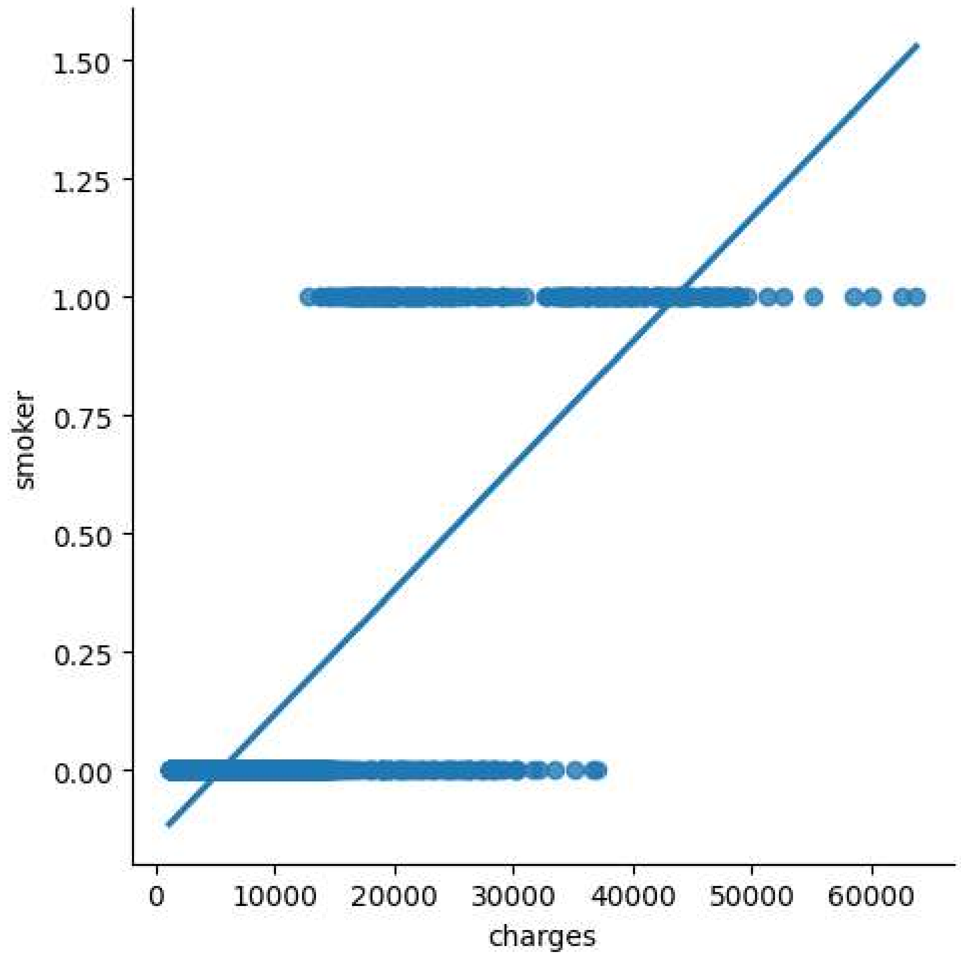
	age	sex	bmi	children	smoker	region	charges
0	19	female	27.900	0	1	southwest	16884.92400
1	18	male	33.770	1	0	southeast	1725.55230
2	28	male	33.000	3	0	southeast	4449.46200
3	33	male	22.705	0	0	northwest	21984.47061
4	32	male	28.880	0	0	northwest	3866.85520
...	...	...	...	...	...	...	...
1333	50	male	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

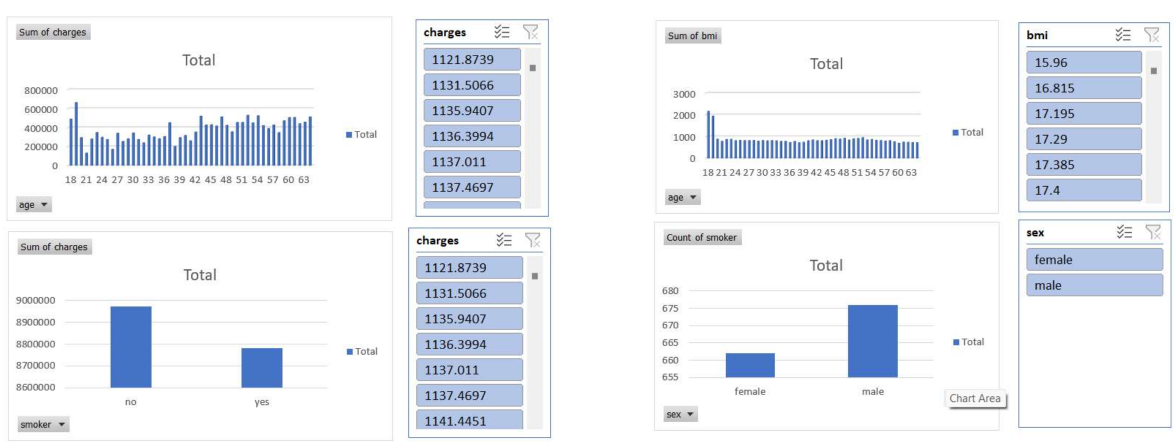
Data Visualization

In [8]:
```
sns.lmplot(x='charges',y='smoker',data=df,order=1,ci=None)
```

Out[8]: <seaborn.axisgrid.FacetGrid at 0x25f0ffcf370>



Dashboard



In [9]:
```
x=df[['charges']]
y=df['smoker']
```

Data Modelling

Linear Regression:

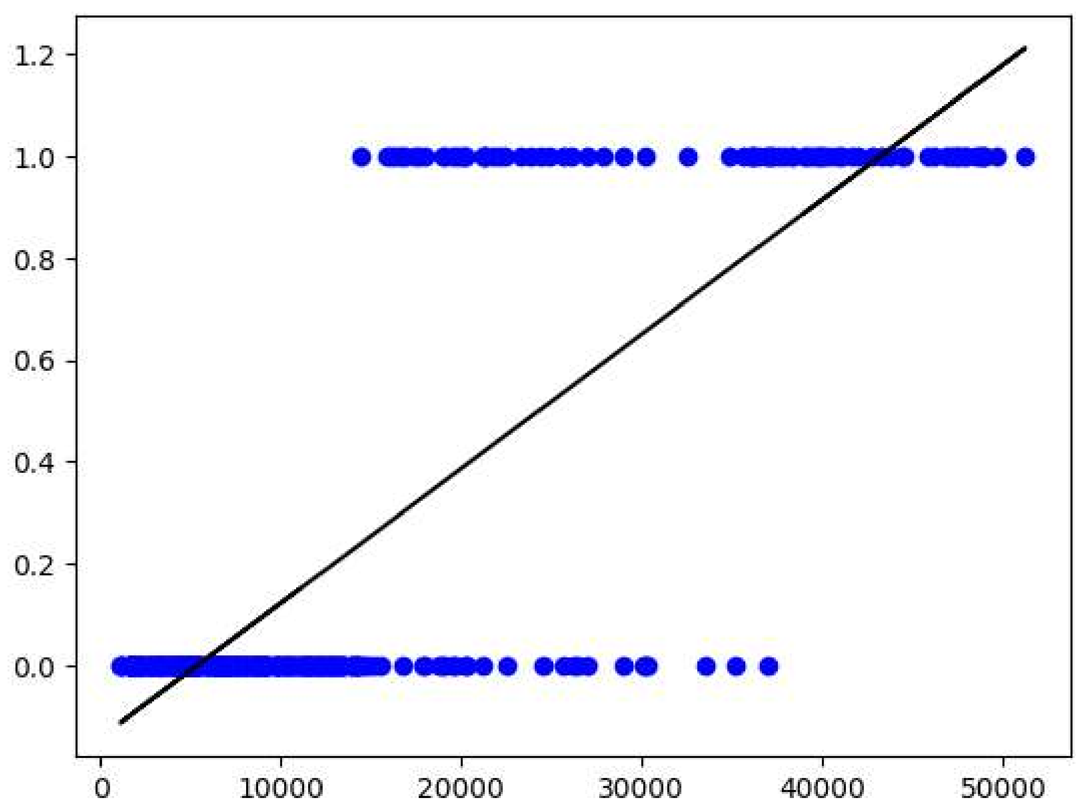
In [10]:
```
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.3)
```

In [11]:
```
lr=LinearRegression()
```

In [12]:
```
lr.fit(x_train,y_train)
```

Out[12]:
```
▾ LinearRegression
LinearRegression()
```

In [13]:
```
y_pred=lr.predict(x_test)
plt.scatter(x_test,y_test,color='b')
plt.plot(x_test,y_pred,color='k')
plt.show()
```



In [14]:
```
lr.score(x_test,y_test)
```
Out[14]:
```
0.6501413584452724
```

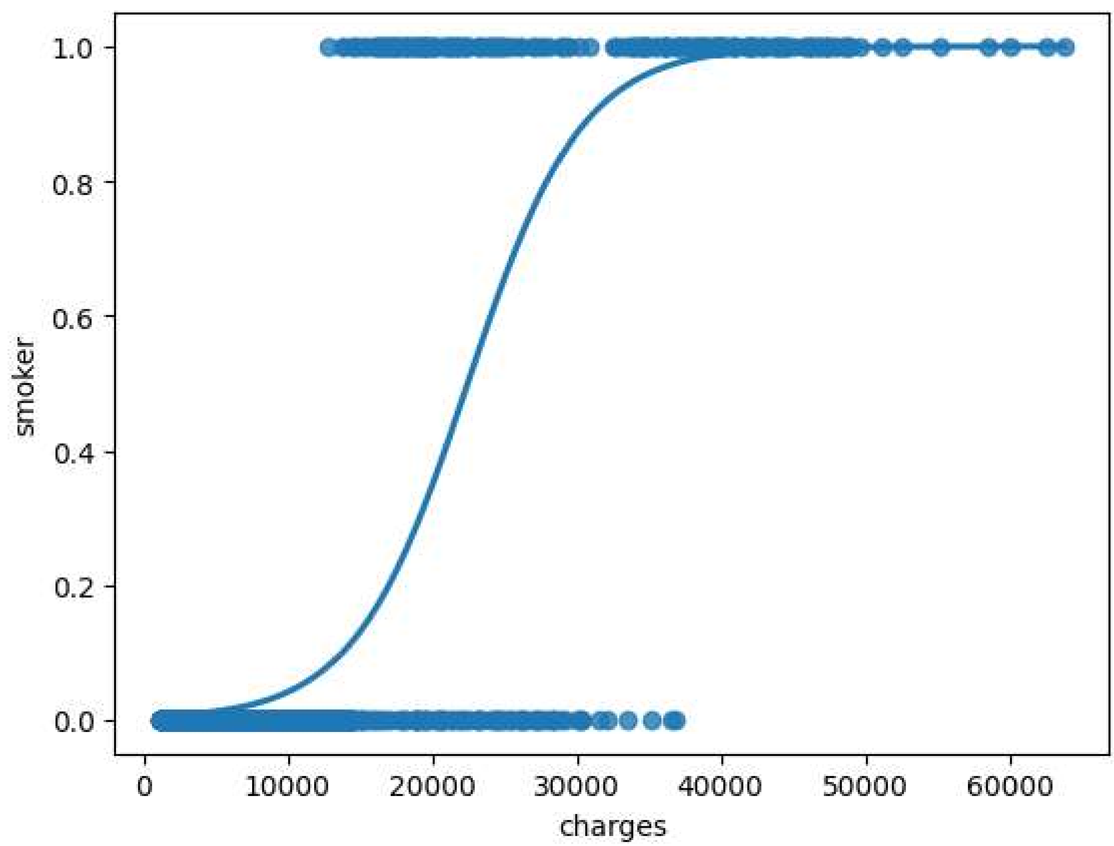
Logistic Regression:

In [15]:
```
from sklearn.linear_model import LogisticRegression
```

In [16]:
```
lr1=LogisticRegression(max_iter=10000)
```

In [17]:
```
lr1.fit(x_train,y_train)
```
Out[17]:
```
▾ LogisticRegression
LogisticRegression(max_iter=10000)
```

In [18]:
```
sns.regplot(x=x,y=y,data=df,logistic=True,ci=None)
```
Out[18]:
```
<Axes: xlabel='charges', ylabel='smoker'>
```



In [19]:
```
lr1.score(x_test,y_test)
```
Out[19]:
```
0.9154228855721394
```

Decision Tree:

In [20]:
```
from sklearn.tree import DecisionTreeClassifier
```

In [21]:
```
dt=DecisionTreeClassifier()
```

In [22]:
```
dt.fit(x_train,y_train)
```

Out[22]:
```
▾ DecisionTreeClassifier
DecisionTreeClassifier()
```

In [23]:
```
dt.score(x_test,y_test)
```

Out[23]:
```
0.9228855721393034
```

Random Forest

In [24]:
```
from sklearn.ensemble import RandomForestClassifier
```

In [25]:
```
rf=RandomForestClassifier()
```

In [26]:
```
rf.fit(x_test,y_test)
```

Out[26]:
```
▾ RandomForestClassifier
RandomForestClassifier()
```

In [27]:
```
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 [28]:
```
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[28]:
```
▸ GridSearchCV
▸ estimator: RandomForestClassifier
    ▸ RandomForestClassifier
```

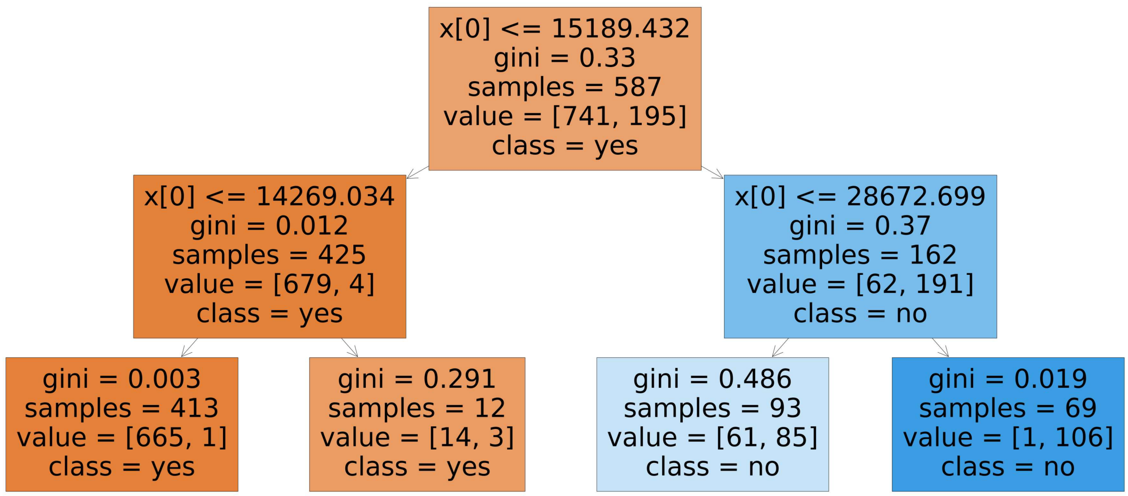
In [29]:
```
grid_search.best_score_
```

Out[29]:
```
0.9241452991452992
```

In [30]:
```
rf_best=grid_search.best_estimator_
print(rf_best)
```

```
RandomForestClassifier(max_depth=2, min_samples_leaf=10, n_estimators=50)
```

In [31]:
```
from sklearn.tree import plot_tree
plt.figure(figsize=(70,30))
plot_tree(rf_best.estimators_[6],class_names=["yes","no"],filled=True);
```



In [32]:
```
rf.score(x_test,y_test)
```

Out[32]:
```
1.0
```

Conclusion

*Based on the accuracy scores of the above implemented models,we can conclude that **Random Forest** is best model for the given dataset.*