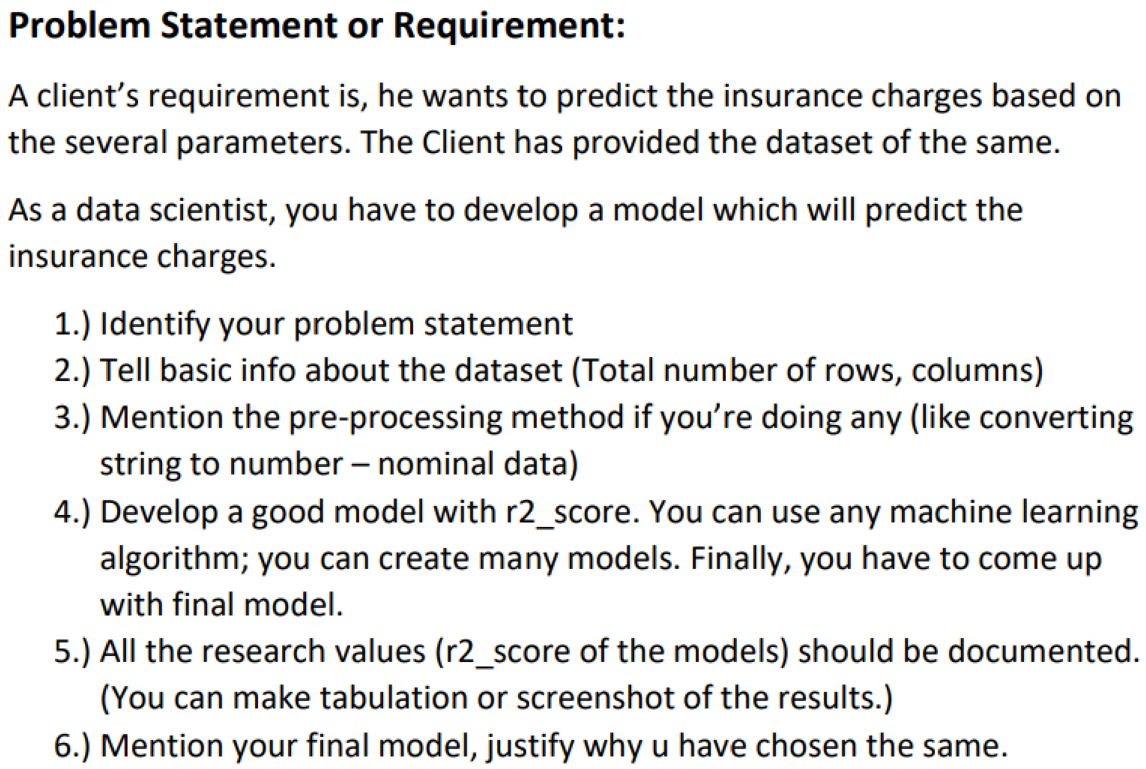


Problem Statement or Requirement:

A client's requirement is, he wants to predict the insurance charges based on the several parameters. The Client has provided the dataset of the same.

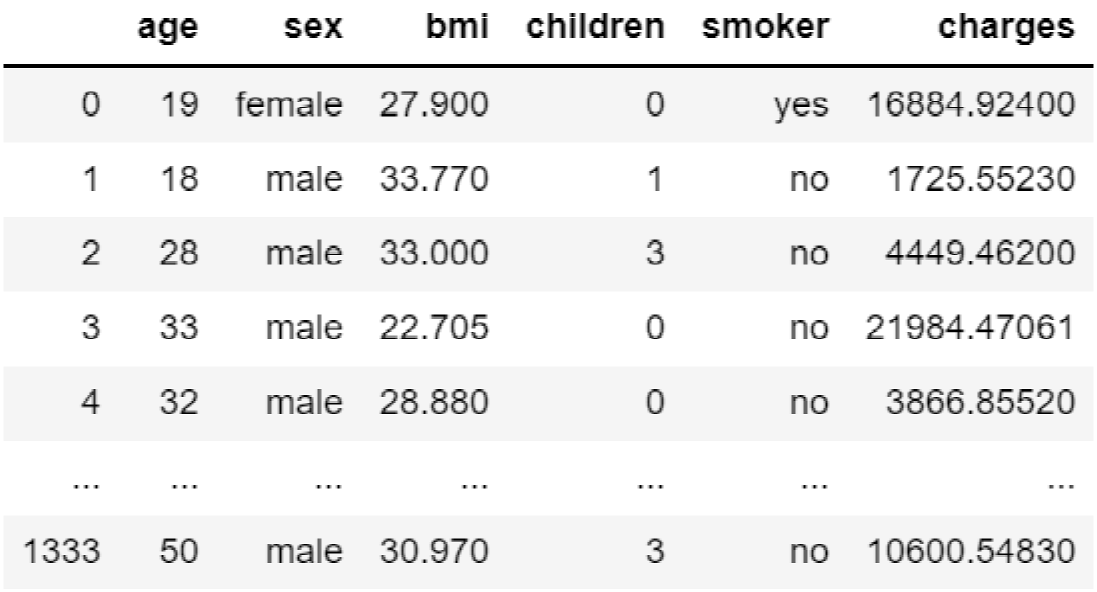
As a data scientist, you have to develop a model which will predict the insurance charges.

1.) Identify your problem statement
2.) Tell basic info about the dataset (Total number of rows, columns)
3.) Mention the pre-processing method if you're doing any (like converting string to number – nominal data)
4.) Develop a good model with r2_score. You can use any machine learning algorithm; you can create many models. Finally, you have to come up with final model.
5.) All the research values (r2_score of the models) should be documented. (You can make tabulation or screenshot of the results.)
6.) Mention your final model, justify why u have chosen the same.

Domain selection:

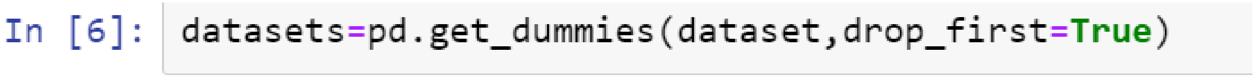
We can solve the above Problem statement using **Machine Learning - Supervised Learning - Regression - Nominal**

Dataset Collection:



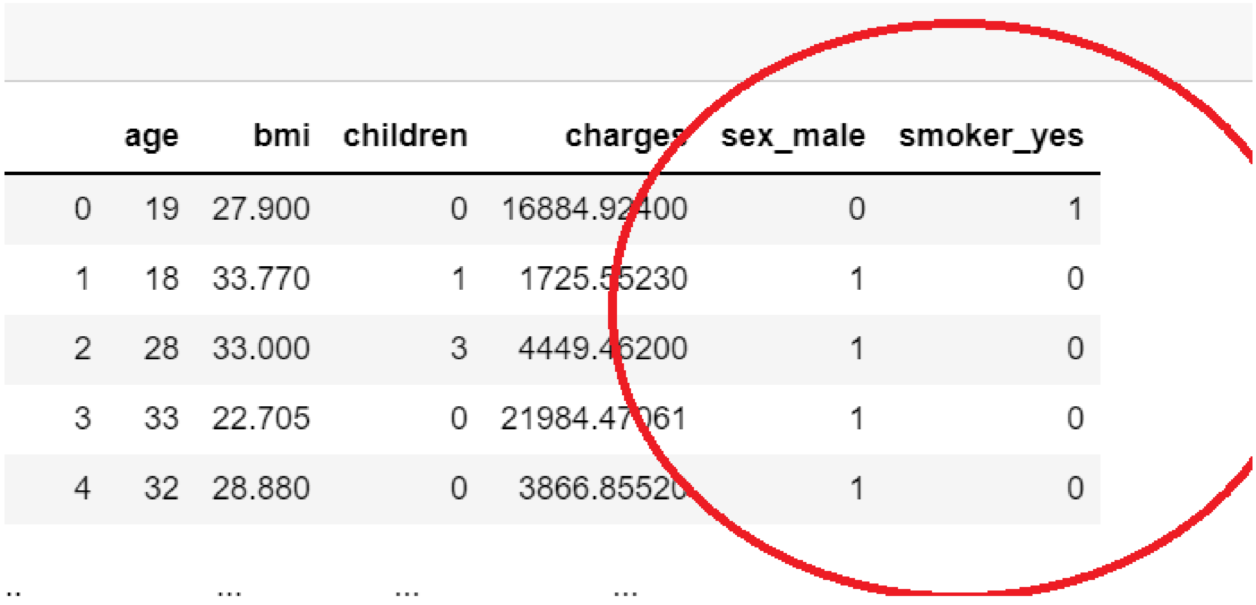
	age	sex	bmi	children	smoker	charges
0	19	female	27.900	0	yes	16884.92400
1	18	male	33.770	1	no	1725.55230
2	28	male	33.000	3	no	4449.46200
3	33	male	22.705	0	no	21984.47061
4	32	male	28.880	0	no	3866.85520
...	...	...	...	...	...	...
1333	50	male	30.970	3	no	10600.54830

In the above problem dataset, we have two nominal columns sex and smoker So we use **One hot encoding** so that we can change that to a numerical value. In python, we can do that by the special library called **Pandas** using **get_dummies** function as shown below



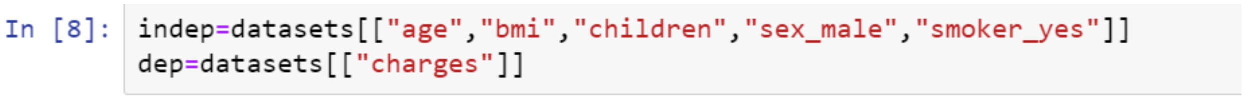
```
In [6]: datasets=pd.get_dummies(dataset,drop_first=True)
```

After completing One hot encoding, our dataset will be like,



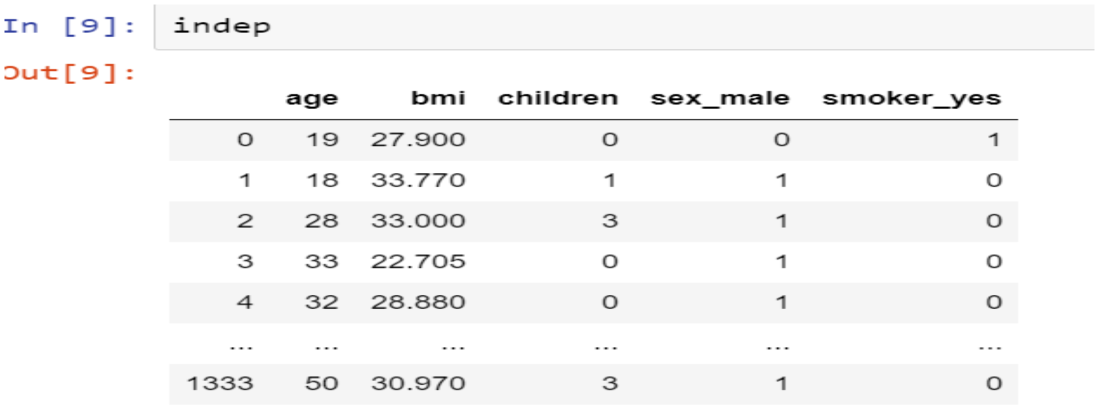
	age	bmi	children	charges	sex_male	smoker_yes
0	19	27.900	0	16884.92400	0	1
1	18	33.770	1	1725.55230	1	0
2	28	33.000	3	4449.46200	1	0
3	33	22.705	0	21984.47061	1	0
4	32	28.880	0	3866.85520	1	0
..		...	...	...		

Splitting Independent and Dependent variables:



```
In [8]: indep=datasets[["age","bmi","children","sex_male","smoker_yes"]]
        dep=datasets[["charges"]]
```

Independent variable:

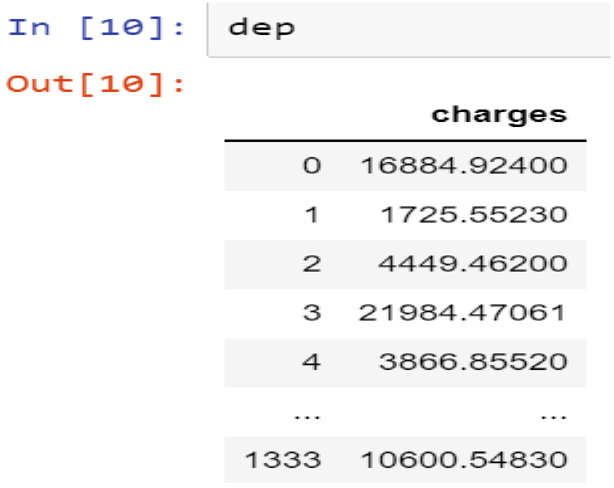


```
In [9]: indep
```

Out[9]:

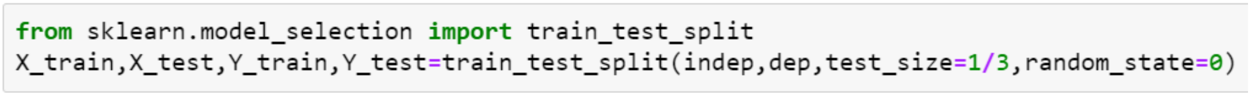
	age	bmi	children	sex_male	smoker_yes
0	19	27.900	0	0	1
1	18	33.770	1	1	0
2	28	33.000	3	1	0
3	33	22.705	0	1	0
4	32	28.880	0	1	0
...	...	...	...	...	...
1333	50	30.970	3	1	0

Dependent variable:



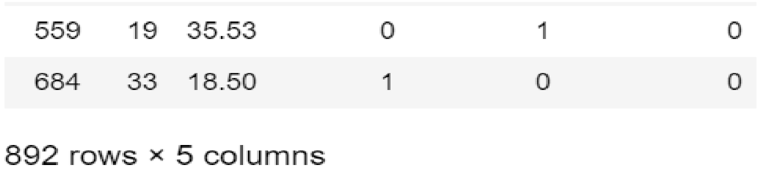
```
In [10]: dep
Out[10]:
              charges
        0  16884.92400
        1   1725.55230
        2   4449.46200
        3  21984.47061
        4   3866.85520
      ...          ...
     1333  10600.54830
```

Splitting Training and Test set:

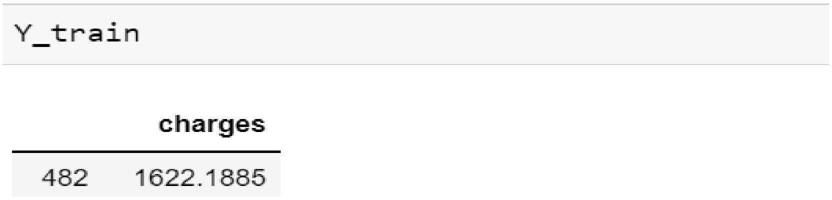


```
from sklearn.model_selection import train_test_split
X_train,X_test,Y_train,Y_test=train_test_split(indep,dep,test_size=1/3,random_state=0)
```

Giving 2/3 to Training set and 1/3 to Test set



```
559  19  35.53  0  1  0
684  33  18.50  1  0  0

892 rows × 5 columns
```

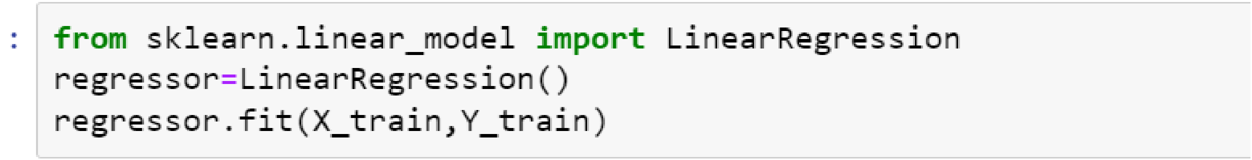


```
Y_train

       charges
482  1622.1885
```

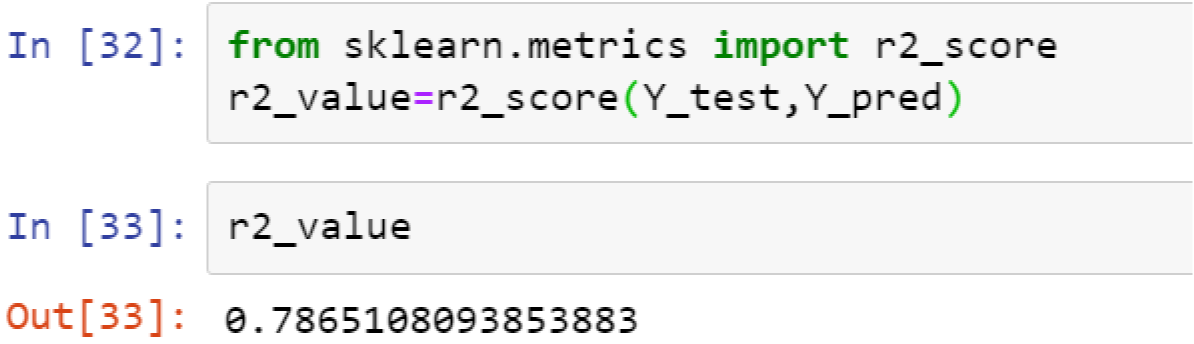
Model Creation and Evaluation metrics:

We create the model using regression algorithms and test it with test data and find the evaluation metrics to check the correctness of the model.

Multiple Linear Regression:



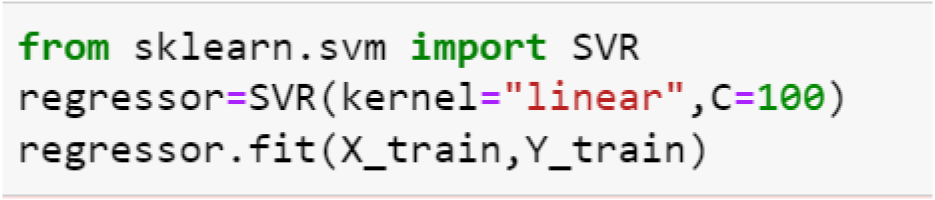
```
: from sklearn.linear_model import LinearRegression
  regressor=LinearRegression()
  regressor.fit(X_train,Y_train)
```



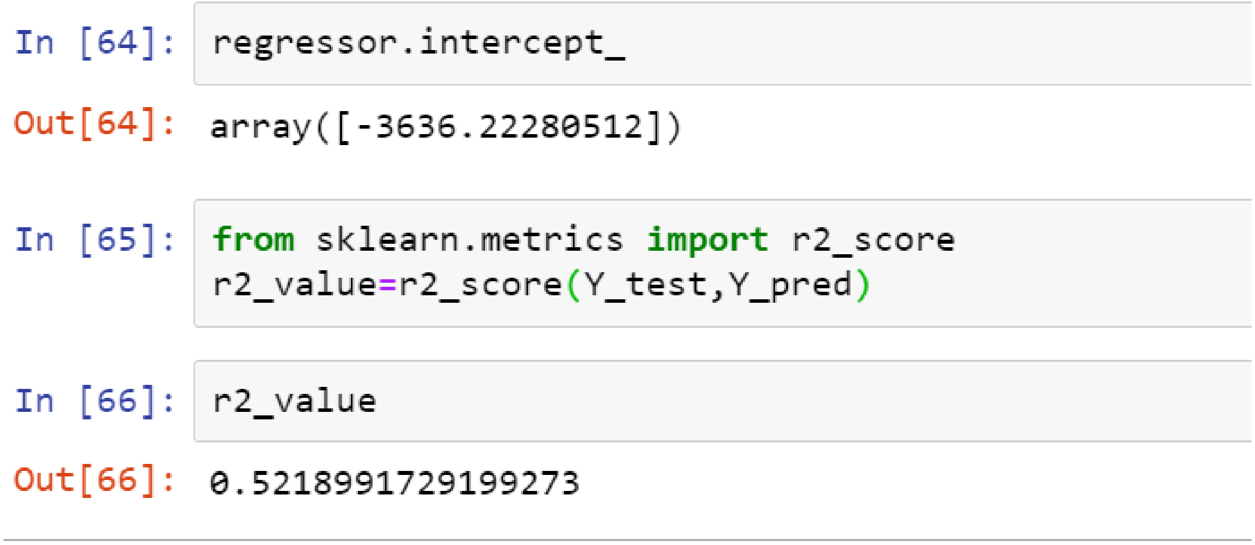
```
In [32]: from sklearn.metrics import r2_score
         r2_value=r2_score(Y_test,Y_pred)

In [33]: r2_value

Out[33]: 0.7865108093853883
```

Here we used Multiple linear regression algorithm and solved the problem statement and we got **78% efficiency**

SupportVectorMachine:



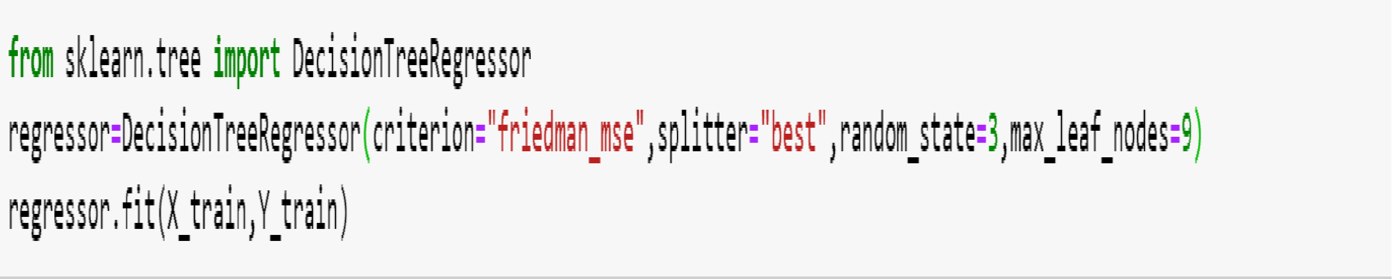
```
from sklearn.svm import SVR
regressor=SVR(kernel="linear",C=100)
regressor.fit(X_train,Y_train)
```



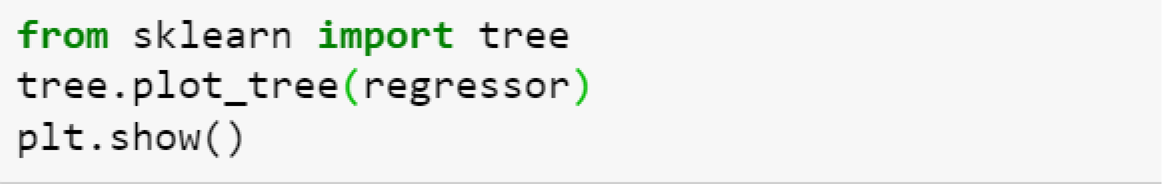
```
In [64]: regressor.intercept_

Out[64]: array([-3636.22280512])

In [65]: from sklearn.metrics import r2_score
         r2_value=r2_score(Y_test,Y_pred)

In [66]: r2_value

Out[66]: 0.5218991729199273
```

Here we used Support Vector Machine algorithm and solved the problem statement and we got **52% efficiency**

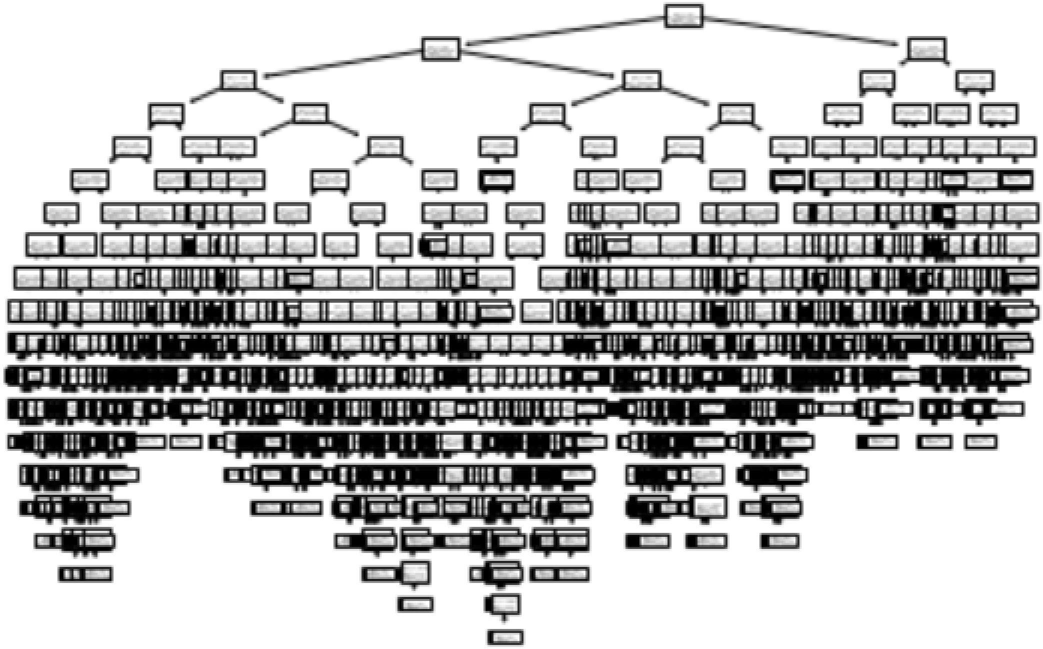
Decision tree:

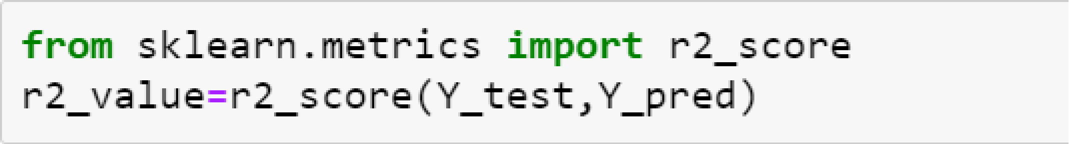


```
from sklearn.tree import DecisionTreeRegressor
regressor=DecisionTreeRegressor(criterion="friedman_mse",splitter="best",random_state=3,max_leaf_nodes=9)
regressor.fit(X_train,Y_train)
```

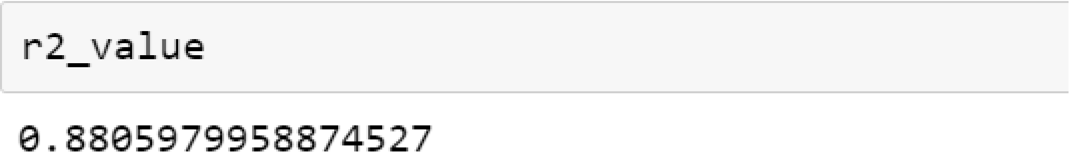


```
from sklearn import tree
tree.plot_tree(regressor)
plt.show()
```





```
from sklearn.metrics import r2_score
r2_value=r2_score(Y_test,Y_pred)
```



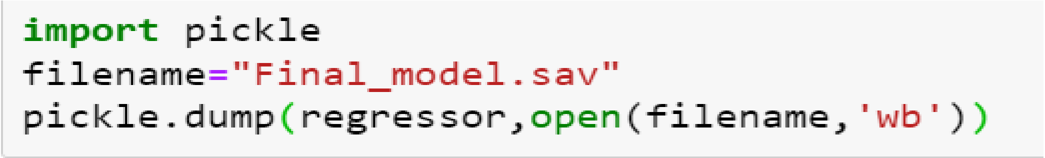
```
r2_value
```

```
0.8805979958874527
```

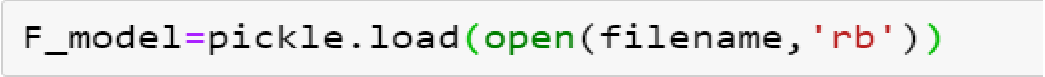
Here we used Decision Tree algorithm and solved the problem statement and we got **88% efficiency**

So, finally the Decision **tree** algorithm showed us **88%** and it is the best model to save and get to the next deployment processes.

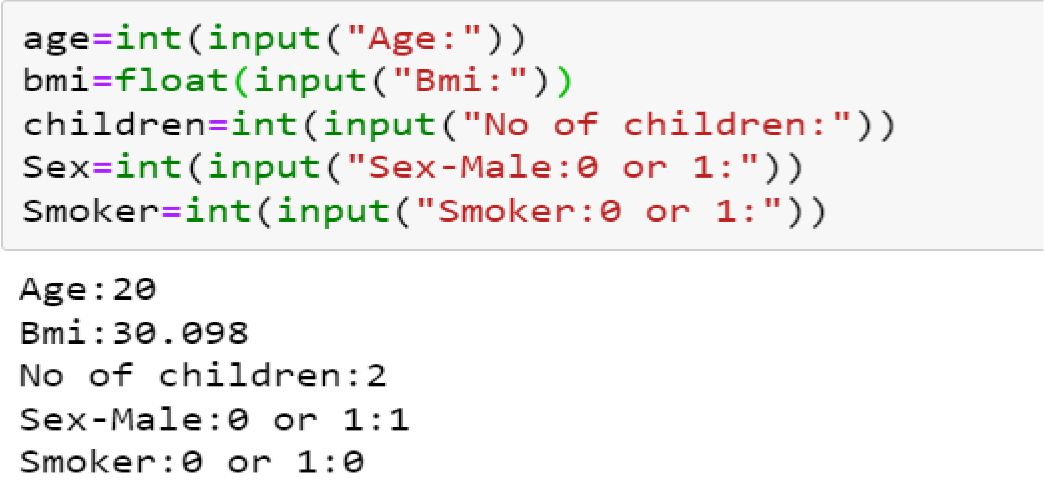
Saving the Best model and deployment:



```
import pickle
filename="Final_model.sav"
pickle.dump(regressor,open(filename,'wb'))
```

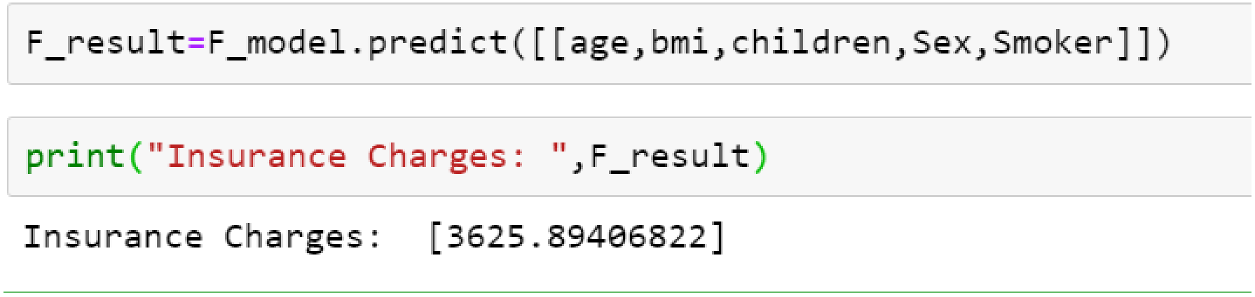


```
F_model=pickle.load(open(filename,'rb'))
```



```
age=int(input("Age:"))
bmi=float(input("Bmi:"))
children=int(input("No of children:"))
Sex=int(input("Sex-Male:0 or 1:"))
Smoker=int(input("Smoker:0 or 1:"))
```

```
Age:20
Bmi:30.098
No of children:2
Sex-Male:0 or 1:1
Smoker:0 or 1:0
```



```
F_result=F_model.predict([[age,bmi,children,Sex,Smoker]])
```

```
print("Insurance Charges: ",F_result)
```

```
Insurance Charges:  [3625.89406822]
```