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 must develop a model which will predict the insurance charges.

1.) Identify your problem statement

Here the problem statement is to predict the insurance charges based on the input data given by the client. Since the i/p is in numerical formal we can use **MACHINE LEARNING** for providing the solution. Also, since the requirement is clear and we have both the i/p and o/p data handy this will come under **SUPERVISED LEARNING**. Further the o/p is numerical and hence we would go ahead with **REGRESSION**

2.) Tell basic info about the dataset (Total number of rows, columns)

The dataset that has been provided:

- Column→ 6
  - 4- numerical
  - 2 categorical
- Rows→ 1000+

3.) Mention the pre-processing method if you're doing any (like converting string to number – nominal data)

Since the data is NOMINAL, we should be using **ONE-HOT ENCODING** to update to numerical format for our Python code to handle it

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.

Used all the 4 algorithms as below:

- Multiple linear regression
- SVM
- Decision Tree
- Random forest

5.) **All the research values (r2_score of the models) should be documented. (You can make tabulation or screenshot of the results.)**

MLR	
R_score	**0.7894**

SVM				
C values	**Linear-R_score**	**poly-R_score**	**rbf-R_score**	**sigmoid-R_score**
0.01	-0.0888	-0.0895	-0.0896	-0.0895
0.1	-0.0809	-0.0883	-0.089	-0.0882
1	-0.0101	-0.0756	-0.0833	-0.0754
10	0.4624	0.03871	-0.0322	0.0393
100	**0.6288**	0.6179	0.32	0.5276

Decision Tree		
Criterion	**Splitter**	**R_score**
squared_error	best	0.6892
friedman_mse	best	0.6861
absolute_error	best	0.6903
poisson	best	0.6732
squared_error	random	0.7055
friedman_mse	random	0.6814
absolute_error	random	**0.7394**
poisson	random	0.6324

Random forest		
n_estimators	**Criterion**	**R_score**
100	squared_error	0.8519
100	friedman_mse	0.8537
100	absolute_error	0.8521
100	poisson	0.8389
50	squared_error	**0.8556**
50	friedman_mse	0.8518
50	absolute_error	0.8533
50	poisson	0.8397

6**.) Mention your final model, justify why u have chosen the same.**

The final chosen model would be **Random forest** with (n_estimators=50 and Criterion= squared_error) since it has the highest r_score value of **0.8556** so far