Assignment-Regression Algorithm

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.

1.) Identify your problem statement

From the above statement we know that we need to find "insurance charges". We have both input and output data so it fall on "**Supervised Learning**". Since it has numerical output it is "**regression**".

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

Number of row = 1338

Number of columns = 6

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

We are converting 2 columns (sex and smoker).

Sex_female = 0, Sex_Male = 1.

Smoker_Yes = 1, Smoker_No = 0

Code:

"dataset = pd.get_dummies (dataset,dtype=int, drop_first=True)

indep = dataset[['age', 'sex_male', 'bmi', 'children', 'smoker_yes']]

dep = dataset[['charges']]"

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.

I used 4 Algorithm**:**

Multiple Linear Regression

SVM

Decision Tree

Random Forest

(Didn't use Simple Linear Regression because in problem statement we have several parameter)

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

Multiple Linear Regression:

r_score = 0.7865108093853883

SVM:

KERNAL	OUTPUT
RBF	-0. 0937345406747172
Sigmoid	-0. 08670160045380881
Poly	-0. 08639813522126305
Precomputered	Since it is not a square matrix no o/p
Linear	- 0.024891809622290983

Decision Tree:

CRITERION	SPLITTER = BEST	SPLITTER=RANDOM
Squared_error	0. 677656364706567	0. 6988690691501929
Friedman_mse	0. 6833280081032633	0. 6828740919695602
Absulate_error	0. 6650749266821918	0. 7440511916347233
Poisson	0. 6914266835249563	0. 6817991629384057

Random Forest:

r_score = 0.8935091438450422

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

I finalized "**Random Forest**" as best model because comparing to all the other algorithm Random Forest provided the good r_score value