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

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. Kindly create Repository in the name Regression Assignment. Upload all the ipynb and final document in the pdf Communication is important (How you are representing the document.)

Solution:

1. The client wants a machine learning model to predict insurance charges based on input parameters. This is a **supervised learning** problem, specifically a **regression** problem.

2. The dataset has 1338 rows and 6 columns. The provides dataset columns are age, sex, bmi, children, smoker and charges.
3. The sex and smoker columns contain categorical data. Therefore One-hot encoder is applied to convert categorical data to numerical data.
4.

Model Name	**R2 score**
Multiple Linear Regression	0.797450452408694
Multiple Linear Regression	0.881633285044482
Decision Tree	0.8893369867925229
Random Forest Regression	0.898679055705128

5. SVR ,Decision Tree and Random Forest Tabulation are attached below

6. The **Random Forest model** achieved the highest R^2 score of **0.898**. It handles non-linearity data better than other models.

Kernel	Gamma	C	Epsilon	Cache Size	R2 Score
linear	scale	1000	0.001	300	0.75012
linear	scale	5000	0.1	200	0.74865
linear	scale	5000	0.1	300	0.74865
linear	scale	5000	0.001	200	0.74865
linear	scale	5000	0.001	300	0.74865
linear	auto	10	0.001	300	0.5007
linear	auto	100	0.1	200	0.64241
linear	auto	100	0.1	300	0.64241
linear	auto	100	0.001	200	0.64242
linear	auto	100	0.001	300	0.64242
linear	auto	1000	0.1	200	0.75012
linear	auto	1000	0.1	300	0.75012
linear	auto	1000	0.001	200	0.75012
linear	auto	1000	0.001	300	0.75012
linear	auto	5000	0.1	200	0.74865
linear	auto	5000	0.1	300	0.74865
linear	auto	5000	0.001	200	0.74865
linear	auto	5000	0.001	300	0.74865
poly	scale	1000	0.1	200	0.86049
poly	scale	1000	0.1	300	0.86049
poly	scale	1000	0.001	200	0.86049
poly	scale	1000	0.001	300	0.86049
poly	scale	5000	0.1	200	0.86051
poly	scale	5000	0.1	300	0.86051
poly	scale	5000	0.001	200	0.86051
poly	scale	5000	0.001	300	0.86051
poly	auto	100	0.001	200	0.65781
poly	auto	100	0.001	300	0.65781
poly	auto	1000	0.1	200	0.86049
poly	auto	1000	0.1	300	0.86049
poly	auto	1000	0.001	200	0.86049
poly	auto	1000	0.001	300	0.86049
poly	auto	5000	0.1	200	0.86051
poly	auto	5000	0.1	300	0.86051
poly	auto	5000	0.001	200	0.86051
poly	auto	5000	0.001	300	0.86051
rbf	scale	1000	0.1	200	0.82522
rbf	scale	1000	0.1	300	0.82522
rbf	scale	1000	0.001	200	0.82522
rbf	scale	1000	0.001	300	0.82522
rbf	scale	5000	0.1	200	0.88163

Criterion	Splitter	Min Samp	Min Samp	Max Leaf	R2 Score
squared_	best	2	1	None	0.68964
squared_	best	2	1	10	0.88207
squared_	best	2	1	20	0.87295
squared_	best	2	2	None	0.81984
squared_	best	2	2	10	0.88107
squared_	best	2	2	20	0.87884
squared_	best	5	1	None	0.77253
squared_	best	5	1	10	0.88207
squared_	best	5	1	20	0.87295
squared_	best	5	2	None	0.82623
squared_	best	5	2	10	0.88107
squared_	best	5	2	20	0.87884
squared_	random	2	1	None	0.75459
squared_	random	2	1	10	0.8547
squared_	random	2	1	20	0.87908
squared_	random	2	2	None	0.83808
squared_	random	2	2	10	0.8547
squared_	random	2	2	20	0.87908
squared_	random	5	1	None	0.83464
squared_	random	5	1	10	0.8547
squared_	random	5	1	20	0.87908
squared_	random	5	2	None	0.83826
squared_	random	5	2	10	0.8547
squared_	random	5	2	20	0.87908
absolute	best	2	1	None	0.69522
absolute	best	2	1	10	0.87312
absolute	best	2	1	20	0.88934
absolute	best	2	2	None	0.8331
absolute	best	2	2	10	0.87312
absolute	best	2	2	20	0.88974
absolute	best	5	1	None	0.81906
absolute	best	5	1	10	0.87312
absolute	best	5	1	20	0.88934
absolute	best	5	2	None	0.85089
absolute	best	5	2	10	0.87312
absolute	best	5	2	20	0.88974
absolute	random	2	1	None	0.74405
absolute	random	2	1	10	0.82178
absolute	random	2	1	20	0.87989
absolute	random	2	2	None	0.84639
absolute	random	2	2	10	0.82178
absolute	random	2	2	20	0.87989
absolute	random	5	1	None	0.85718
absolute	random	5	1	10	0.82178
absolute	random	5	1	20	0.87899
absolute	random	5	2	None	0.83809
absolute	random	5	2	10	0.82178
absolute	random	5	2	20	0.87899
poisson	best	2	1	None	0.64793

1	Trees	Criterion	Max Dep	Max Feal	Min Impu	Random	R2 Score
2	50	squared_error		sqrt	0	42	0.88321
3	50	squared_error		sqrt	0	0	0.879
4	50	squared_error		sqrt	0.01	42	0.88328
5	50	squared_error		sqrt	0.01	0	0.87716
6	50	squared_error		log2	0	42	0.88321
7	50	squared_error		log2	0	0	0.879
8	50	squared_error		log2	0.01	42	0.88328
9	50	squared_error		log2	0.01	0	0.87716
10	50	squared_error		sqrt	0	42	0.8927
11	50	squared_error		sqrt	0	0	0.88974
12	50	squared_error		sqrt	0.01	42	0.8927
13	50	squared_error		sqrt	0.01	0	0.88974
14	50	squared_error		log2	0	42	0.8927
15	50	squared_error		log2	0	0	0.88974
16	50	squared_error		log2	0.01	42	0.8927
17	50	squared_error		log2	0.01	0	0.88974
18	50	absolute_error		sqrt	0	42	0.89218
19	50	absolute_error		sqrt	0	0	0.88417
20	50	absolute_error		sqrt	0.01	42	0.88939
21	50	absolute_error		sqrt	0.01	0	0.88861
22	50	absolute_error		log2	0	42	0.89218
23	50	absolute_error		log2	0	0	0.88417
24	50	absolute_error		log2	0.01	42	0.88939
25	50	absolute_error		log2	0.01	0	0.88861
26	50	absolute_error		sqrt	0	42	0.89536
27	50	absolute_error		sqrt	0	0	0.89031
28	50	absolute_error		sqrt	0.01	42	0.89868
29	50	absolute_error		sqrt	0.01	0	0.88956
30	50	absolute_error		log2	0	42	0.89536
31	50	absolute_error		log2	0.01	0	0.88956
32	50	poisson		sqrt	0	42	0.88261
33	50	poisson		sqrt	0	0	0.87713
34	50	poisson		sqrt	0.01	42	0.88432
35	50	poisson		sqrt	0.01	0	0.88527
36	50	poisson		log2	0	42	0.88261
37	50	poisson		log2	0	0	0.87713
38	50	poisson		log2	0.01	42	0.88432
39	50	poisson		log2	0.01	0	0.88527
40	50	poisson		sqrt	0	42	0.88828
41	50	poisson		sqrt	0	0	0.89059
42	50	poisson		sqrt	0.01	42	0.88968
43	50	poisson		sqrt	0.01	0	0.89444
44	50	poisson		log2	0	42	0.88828
45	50	poisson		log2	0	0	0.89059
46	50	poisson		log2	0.01	42	0.88968
47	50	poisson		log2	0.01	0	0.89444
48	100	squared_error		sqrt	0	42	0.88719
49	100	squared_error		sqrt	0	0	0.88068
50	100	squared_error		sqrt	0.01	42	0.88774