**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

Need to Predict the insurance charges

Supervised Learning

ML

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

* Total number of Rows: 1338
* Total number of column’s =6

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

* This is a nominal Data – (One Hot encoding)

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.

Best model is Random Forest

Result: 0.8498329315421834

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)

SVMR

|  |  |  |
| --- | --- | --- |
| Model | C value | RScore |
| SVR- rbf |  | -0.088377326 |
| SVR- Iinear |  | -1.436870821 |
| SVR- Poly |  | -12266.20451 |
| SVR-Sigmoid |  | -0.073441808 |
| SVR- rbf | 1 | -0.088377326 |
|
| SVR- rbf | 10 | -0.080410236 |
|
| SVR- rbf | 100 | -0.021253538 |
|
| SVR- rbf | 1000 | -0.015877384 |
|

Decision Tree

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S.no | Criterion | Max Features | Splitter | R Score |
| 1 | poisson |  | best | 0.715504484 |
| 2 | poisson |  | random | 0.711582515 |
| 3 | poisson | None | random | 0.619351748 |
| 4 | poisson | None | best | 0.734089855 |
| 5 | poisson | sqrt | random | 0.697415061 |
| 6 | poisson | sqrt | best | 0.711913732 |
| 7 | poisson | log2 | random | 0.708121012 |
| 8 | poisson | log2 | best | 0.676034273 |
| 9 | absolute\_error |  | best | 0.682280328 |
| 10 | absolute\_error |  | random | 0.717483788 |
| 11 | absolute\_error | None | random | 0.761834759 |
| 12 | absolute\_error | None | best | 0.679974584 |
| 13 | absolute\_error | sqrt | random | 0.702861357 |
| 14 | absolute\_error | sqrt | best | 0.681477717 |
| 15 | absolute\_error | log2 | random | 0.711412805 |
| 16 | absolute\_error | log2 | best | 0.662393503 |
| 17 | friedman\_mse |  | random | 0.705501975 |
| 18 | friedman\_mse |  | best | 0.694382853 |
| 19 | friedman\_mse | None | random | 0.705392443 |
| 20 | friedman\_mse | None | best | 0.693428456 |
| 21 | friedman\_mse | sqrt | random | 0.609821611 |
| 22 | friedman\_mse | sqrt | best | 0.615524914 |
| 23 | friedman\_mse | log2 | random | 0.642921911 |
| 24 | friedman\_mse | log2 | best | 0.734099976 |
| 25 | squared\_error |  | best | 0.689065988 |
| 26 | squared\_error |  | random | 0.725223506 |
| 27 | squared\_error | None | random | 0.747104994 |
| 28 | squared\_error | None | best | 0.711858848 |
| 29 | squared\_error | sqrt | random | 0.616286529 |
| 30 | squared\_error | sqrt | best | 0.707447573 |
| 31 | squared\_error | log2 | random | 0.608402404 |
| 32 | squared\_error | log2 | best | 0.722784432 |