**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.

Solution:

1. Customer wants to know the insurance charges for the policy based on the policy holder age,bmi,gender,howmany children they have, do they have smoking habits.
2. Input file has 1339 rows and 6 rows. By using this we can create dataset with independent as input and dependent as output.
3. We can come with Train and Test data by train\_test\_split function from sklearn
4. Once we separate training data, we can feed the data to train the model by .fit function and out put prediction also by .predict function.
5. R2\_score value will be decided the model confirmation.

**Stage1**

According to the problem statement, client wants to know the insurance charges, It shows that it is a numerical value🡪 **Machine Learning**

**Stage2**

Customer has given the proper input and output 🡪**Supervised**

**Stage3**

Out put deals with the numbers 🡪 **Regression**

**Phase 1:**

Data collection 🡪 Customer has given the input file as data to be processed

Data Preprocessing 🡪

Input/Output split 🡪 independent—Input , dependent –Output split can be done.

Train and Test data🡪 X\_train, X\_test,y\_train,y\_test. X -Input , y - Output.

Train set 🡪 X\_train and y\_train – This data uses to feed the model to learn.

Model creation🡪 Once the model completes the training data, we can conclude the model is been created.

Test set 🡪 Created model will be used to predict the output, based on the X\_test value.

Metrics 🡪 R2\_Scoce will be decided either we can use the model or data by comparing the given output and the model predicted output

Save the model 🡪 If the R2\_score value is near to 1, then we can save the model and this can be recommended to customer.

**Phase2:**

Deployment 🡪 They saved model will be sent to the deployment team ..

The below are the R2\_score values of each algorithms od used

|  |  |  |  |
| --- | --- | --- | --- |
| Multi Linear Regression | | | |
| **S.No** | | **Variable** | **r2\_Score** |
| 1 | Default Value | | 0.789479035 |

Decision Tree- Result, based on the variable changing

**SVM**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **kernel** | **gamma** | **r2\_score** |
| 1 | rbf | scale | -0.08851297 |
| 2 | linear | scale | -0.10013929 |
| 3 | poly | scale | -0.07245894 |
| 4 | sigmoid | scale | -0.08992539 |
| 5 | precomputed | scale |  |
| 6 | linear | auto | -0.10013929 |
| 7 | poly | auto | 0.862393504 |
| 8 | rbf | auto | -0.08938837 |
| 9 | sigmoid | auto | -0.089709 |

**Decision tree**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **criterion** | **splitter** | **max\_features** | **r2\_score** |
| 1 | squared\_error | best | None | 0.708043 |
| 2 | squared\_error | random | None | 0.714421 |
| 3 | squared\_error | best | sqrt | 0.68426 |
| 4 | squared\_error | best | log2 | 0.703994 |
| 5 | squared\_error | random | sqrt | 0.70157 |
| 6 | squared\_error | random | log2 | 0.68944 |
| 7 | friedman\_mse | best | None | 0.693447 |
| 8 | friedman\_mse | random | None | 0.691527 |
| 9 | friedman\_mse | best | sqrt | 0.704165 |
| 10 | friedman\_mse | best | log2 | 0.727821 |
| 11 | friedman\_mse | random | sqrt | 0.659596 |
| 12 | friedman\_mse | random | log2 | 0.678854 |
| 13 | absolute\_error | best | None | 0.683054 |
| 14 | absolute\_error | random | None | 0.710325 |
| 15 | absolute\_error | best | sqrt | 0.704294 |
| 16 | absolute\_error | best | log2 | 0.684966 |
| 17 | absolute\_error | random | sqrt | 0.663971 |
| 18 | absolute\_error | random | log2 | 0.724183 |
| 19 | poisson | best | None | 0.731049 |
| 20 | poisson | random | None | 0.74565 |
| 21 | poisson | best | sqrt | 0.710254 |
| 22 | poisson | best | log2 | 0.702438 |
| 23 | poisson | random | sqrt | 0.704142 |
| 24 | poisson | random | log2 | 0.674255 |

**Random Forest- ensemble**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **criterion** | **n\_estimators** | **max\_features** | **r2\_score** |
| 1 | squared\_error | 100 | 1.0 | 0.857606 |
| 2 | squared\_error | 50 | 1.0 | 0.85498 |
| 3 | squared\_error | 500 | 1.0 | 0.854148 |
| 4 | squared\_error | 100 | sqrt | 0.870775 |
| 5 | squared\_error | 100 | log2 | 0.871047 |
| 6 | squared\_error | 100 | None | 0.857112 |
| 7 | absolute\_error | 100 | 1.0 | 0.852135 |
| 8 | absolute\_error | 50 | 1.0 | 0.846532 |
| 9 | absolute\_error | 500 | 1.0 | 0.855985 |
| 10 | absolute\_error | 100 | sqrt | 0.870441 |
| 11 | absolute\_error | 100 | log2 | 0.874648 |
| 12 | absolute\_error | 100 | None | 0.853913 |
| 13 | friedman\_mse | 100 | 1.0 | 0.859613 |
| 14 | friedman\_mse | 50 | 1.0 | 0.850746 |
| 15 | friedman\_mse | 500 | 1.0 | 0.856645 |
| 16 | friedman\_mse | 100 | sqrt | 0.87125 |
| 17 | friedman\_mse | 100 | log2 | 0.869813 |
| 18 | friedman\_mse | 100 | None | 0.856566 |
| 19 | poisson | 100 | 1.0 | 0.860327 |
| 20 | poisson | 50 | 1.0 | 0.85714 |
| 21 | poisson | 500 | 1.0 | 0.855603 |
| 22 | poisson | 100 | sqrt | 0.868114 |
| 23 | poisson | 100 | log2 | 0.868221 |
| 24 | poisson | 100 | None | 0.860909 |

Random Forest: ensemble

n\_estimators=100,criterion='absolute\_error',max\_features='log2'

The model is near to 1 among the other regression, so we can this model to use the customer.