**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

Predict the Insurance charges

Three stages

* + Machine Learning
  + Supervised Learning
  + Regression

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

* Total no. of rows = 1338
* Total no. of column = 6

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

* Converted two rows of strings into nominal data, such as ‘sex’ and ‘smoker’ rows

1. 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.

Final model is **Random Forest** **Algorithm**

|  |  |  |  |
| --- | --- | --- | --- |
| **Criterion** | **Max Features** | **N\_estimators** | **R\_score** |
| absolute\_error | sqrt | 100 | 0.867 |

1. All the research values (r2\_score of the models) should be documented. (You can make tabulation or screenshot of the results.)
2. **Multiple Linear Regression (R score value = 0.789)**
3. **Support Vector Machine**

**SVM** R score (Kernel=rbf, c= 3000) = 0.864

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Hyper Parameter** | **Linear (r score)** | **RBF non-linear (r score)** | **Poly (r score)** | **Sigmond (r score)** |
| c=10 | 0.462 | -0.032 | 0.038 | 0.039 |
| c=100 | 0.628 | 0.319 | 0.616 | 0.526 |
| c=500 | 0.763 | 0.661 | 0.828 | 0.442 |
| c=1000 | 0.764 | 0.81 | 0.854 | 0.212 |
| c=2000 | 0.743 | 0.854 | 0.858 | -0.621 |
| c=3000 | 0.741 | 0.864 | 0.858 | -2.143 |

1. **Decision Tree**

R score (criterion = “squared\_error”, max\_features=”sqrt”, splitter=”random”) = 0.773

|  |  |  |  |
| --- | --- | --- | --- |
| **Criterion** | **Splitter** | **Max Features** | **R\_score** |
| squared\_error | best | log2 | 0.761 |
| squared\_error | best | sqrt | 0.769 |
| squared\_error | random | sqrt | 0.752 |
| squared\_error | random | log2 | 0.773 |
| friedman\_mse | best | sqrt | 0.734 |
| friedman\_mse | best | log2 | 0.657 |
| friedman\_mse | random | sqrt | 0.712 |
| friedman\_mse | random | log2 | 0.655 |
| absolute\_error | best | log2 | 0.746 |
| absolute\_error | best | sqrt | 0.653 |
| absolute\_error | random | log2 | 0.678 |
| absolute\_error | random | sqrt | 0.669 |
| poisson | best | sqrt | 0.732 |
| poisson | best | log2 | 0.714 |
| poisson | random | log2 | 0.66 |
| poisson | random | sqrt | 0.664 |

1. **Random Forest**

R\_score(criterion = “absolute\_error”, max\_features=”sqrt”, n\_estimators=100) = 0.867

|  |  |  |  |
| --- | --- | --- | --- |
| **Criterion** | **Max Features** | **N\_estimators** | **R\_score** |
| squared\_error | sqrt | 10 | 0.863 |
| squared\_error | sqrt | 100 | 0.865 |
| squared\_error | log2 | 10 | 0.851 |
| squared\_error | log2 | 100 | 0.863 |
| friedman\_mse | sqrt | 10 | 0.847 |
| friedman\_mse | sqrt | 100 | 0.862 |
| friedman\_mse | log2 | 10 | 0.851 |
| friedman\_mse | log2 | 100 | 0.863 |
| absolute\_error | sqrt | 10 | 0.848 |
| absolute\_error | sqrt | 100 | 0.867 |
| absolute\_error | log2 | 10 | 0.859 |
| absolute\_error | log2 | 100 | 0.867 |
| poisson | sqrt | 10 | 0.858 |
| poisson | sqrt | 100 | 0.862 |
| poisson | log2 | 10 | 0.844 |
| poisson | log2 | 100 | 0.864 |

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

I chose the **Random Forest Algorithm** because a sore of 0 represents a poor model and score of 1 signifies a good model. The Random Forest R score value is **0.867**, which is the closest to 1 when compared to other algorithms.