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.

Data set Analysis:

1.It contains 6 columns : Age, Sex, bmi, children, smoker and charges

2. Dataset contains 1338 rows

3. I could see 2 nominal columns : sex and smoker which has to be converted with the help of one hot encoding

4. Domain Selection : Machine Learning

5. Learning Selection : Supervised Learning

6. Method : Regression

Since the data involves multiple inputs, here we can't use Simple Linear Regression. So lets start with other available algorithms.

1.MLR – Multiple Linear Regression

R2_score value : 0.7894

```
[45]: from sklearn.linear_model import LinearRegression
      regressor = LinearRegression()
      regressor.fit(X_train, y_train)

[45]: LinearRegression
      LinearRegression()

[47]: weight = regressor.coef_
      weight

[47]: array([[  257.8006705 ,   321.06004271,   469.58113407,   -41.74825718,
              23418.6671912 ]])

[49]: bias = regressor.intercept_
      bias

[49]: array([-12057.244846])

[51]: y_predict = regressor.predict(X_test)

[53]: from sklearn.metrics import r2_score
      r_score = r2_score(y_test, y_predict)
      r_score

[53]: 0.7894790349867009
```

2. Support Vector Machine – SVM:

R2 score value : -0.0765

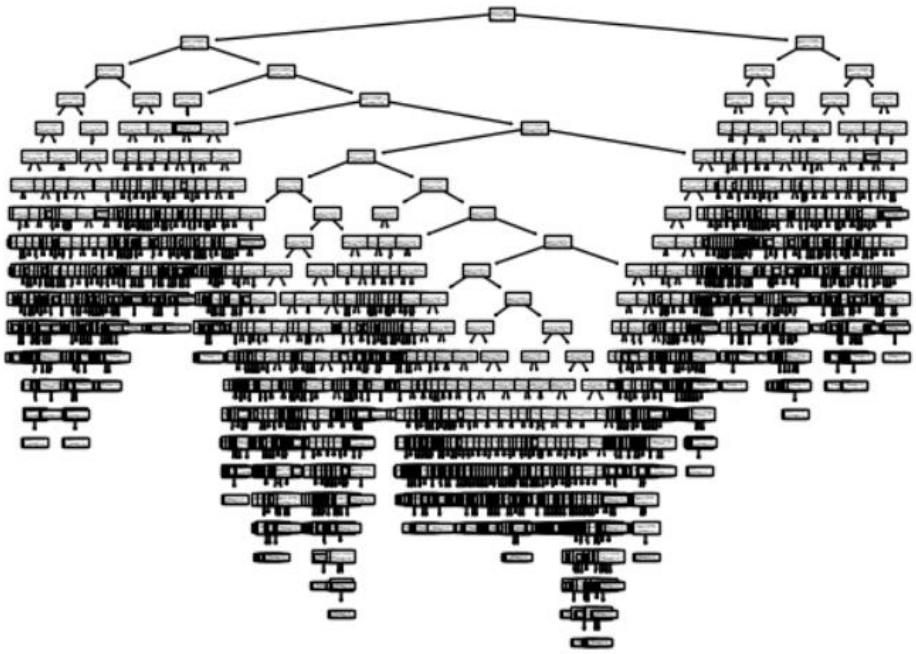
3. Decision Tree:

R2 score value (without parameter tuning) : 0.6882844096334517

With Parameter turning:

criterion='squared_error', splitter='best'	0.6950003678214174
criterion='friedman_mse', splitter='best'	0.6819201895420186
criterion='absolute_error', splitter='best'	0.69172581764679
criterion='poisson', splitter='best'	0.7318777565761565
criterion='poisson', splitter='best'	0.6925022991797849

So we can take that one as the best score of this model



```
[61]: y_predict = regressor.predict(X_test)

[63]: from sklearn.metrics import r2_score
      r_score = r2_score(y_test, y_predict)
      r_score

[63]: 0.6925022991797849
```

4. Random Forest:

n_estimators=50, criterion='squared_error', random_state=0	**0.8498329315421834**
n_estimators=100, criterion='squared_error', random_state=0	**0.8538307913484513**
n_estimators=50, criterion='absolute_error', random_state=0	**0.8526655993519747**
n_estimators=100, criterion='absolute_error', random_state=0	**0.8520093621081837**
n_estimators=50, criterion='friedman_mse', random_state=0	**0.8500716139332296**
n_estimators=100, criterion='friedman_mse', random_state=0	**0.8540518935149612**
n_estimators=50, criterion='poisson', random_state=0	**0.8491075958392151**
n_estimators=100, criterion='poisson', random_state=0	**0.8526334258892607**

[9]:
```
dataset.columns
```

[9]:
```
Index(['age', 'bmi', 'children', 'charges', 'sex_male', 'smoker_yes'], dtype='object')
```

[11]:
```
independent = dataset[['age', 'bmi', 'children', 'sex_male', 'smoker_yes']]
```

[13]:
```
dependent = dataset[['charges']]
```

[15]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(independent, dependent, test_size=0.30, random_state=0)
```

[83]:
```
from sklearn.ensemble import RandomForestRegressor
regressor = RandomForestRegressor(n_estimators=100, criterion='poisson', random_state=0)
regressor.fit(X_train, y_train)
```

```
D:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:1473: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  return fit_method(estimator, *args, **kwargs)
```

[83]:
```
RandomForestRegressor
RandomForestRegressor(criterion='poisson', random_state=0)
```

[85]:
```
y_predict = regressor.predict(X_test)
```

[87]:
```
from sklearn.metrics import r2_score
r_score = r2_score(y_test, y_predict)
r_score
```

[87]:
```
0.8526334258892607
```