Insurance Amount Prediction

Problem Statement: Client needs to know the insurance charge for their customer. For this, he had historical data of many past customers.

We need to predict the Insurance charges, based on the available information.

Data – Insurance_pre
Goal – To predict Charges

Observation: While observing the given dataset, I came to know that it has Total of 6 fields such as **Age, Sex, BMI, Children, Smoker and Charges**.

We had Total of **1338** such observation and from this we need to train our ML Model to predict the charges.

By seeing the data, I concluded that this problem statement comes under **Machine Learning – Supervised – Regression**.

Data Processing: The column **Sex and Smoker** needs to be converted into **numerical** values as they are **categorical** data.

For that we are using **pd.get_dummies** and converting our data into full of numerical values.

Model Creation : Now we are going to use ML-Regression Algorithms to find the best model for given dataset which is evaluated by **R^2 Value**.

1. Multiple Linear Regression

R^2 = 0.797450452

2. Support Vector Machine

Model#	kernel	C	R^2
1	poly	1	-0.072926997
2	poly	10	-0.078770072
3	poly	100	-0.079136804
4	poly	1000	-0.024478463
5	rbf	1	-0.103530634
6	rbf	10	-0.091659416
7	rbf	100	-0.106389209
8	rbf	1000	-0.096753951
9	sigmoid	1	-0.10613596
10	sigmoid	10	-0.108285109
11	sigmoid	100	-0.144857317
12	sigmoid	1000	-2.326437975

- In **Support Vector Machine Algorithm** R^2 is **negative** for all **hyper tuning parameters**, so SVM **does not suit well** for this problem statement.

3. Decision Tree

Model#	criterion	splitter	max_features	R^2
1	squared_error	best	sqrt	0.627425564
2	squared_error	best	log2	0.698080108
3	squared_error	random	sqrt	0.639282615
4	squared_error	random	log2	0.699365777
5	friedman_mse	best	sqrt	0.73375904
6	friedman_mse	best	log2	0.720762413
7	friedman_mse	random	sqrt	0.6926567
8	friedman_mse	random	log2	0.694067761
9	absolute_error	best	sqrt	0.729010744
10	absolute_error	best	log2	0.685595568
11	absolute_error	random	sqrt	0.674795388
12	absolute_error	random	log2	0.753798685
13	poisson	best	sqrt	0.702157974
14	poisson	best	log2	0.66888455
15	poisson	random	sqrt	0.715431546
16	poisson	random	log2	0.689239673

So, in **Decision Tree Algorithm** best model came for

Criterion = ***absolute_error***
Splitter = ***random***
max_features = ***log2***

R^2 = 0.753798685

4. Random Forest

Model#	n_estimators	criterion	max_features	R^2
1	50	squared_error	sqrt	0.875322584
2	50	squared_error	log2	0.8790465
3	50	absolute_error	sqrt	0.884108322
4	50	absolute_error	log2	0.885877713
5	50	friedman_mse	sqrt	0.887203962
6	50	friedman_mse	log2	0.881415754
7	50	poisson	sqrt	0.885918294
8	50	poisson	log2	0.889329018
9	75	squared_error	sqrt	0.880828435
10	75	squared_error	log2	0.882018442
11	75	absolute_error	sqrt	0.884961716
12	75	absolute_error	log2	0.887432131
13	75	friedman_mse	sqrt	0.884565162
14	75	friedman_mse	log2	0.8867061
15	75	poisson	sqrt	0.886394695
16	75	poisson	log2	0.883845672
17	100	squared_error	sqrt	0.885381923
18	100	squared_error	log2	0.885991645
19	100	absolute_error	sqrt	0.885933142
20	100	absolute_error	log2	0.892212557
21	100	friedman_mse	sqrt	0.887614771
22	100	friedman_mse	log2	0.885047923
23	100	poisson	sqrt	0.883827224
24	100	poisson	log2	0.886060996

So, in **Random Forest Algorithm** best model came for

Criterion = ***absolute_error***
n_estimatore = *100*
max_features = ***log2***

R^2 = 0.892212557

Choosing Best Model :

As Random Forest Algorithm (*Criterion = **absolute_error,** n_estimatore = **100**, max_features = **log2**)* R^2 closer to 1 compared to other Model, we are finalizing this and creating model for the same.

Deployment :

Using **pickle** we deployed our model and predicted the Insurance charge by giving various parameters.

```
In [1]: import pickle

In [5]: Load_Model = pickle.load(open("Final_Model_Insurance.sav", 'rb'))
        result = Load_Model.predict([[60,1,34,3,1]])
        print(result)

        [25563.52]

        C:\Users\mugilans\AppData\Local\anaconda3\Lib\site-packages\sklearn\base.py:464: Use
        es, but RandomForestRegressor was fitted with feature names
          warnings.warn(
```

Conclusion :

Best model is created by **Random Forest Algorithm** (*Criterion = **absolute_error,** n_estimatore = **100** , max_features = **log2**)*