Comparison of Model Performance across Different Algorithms and Parameters

The primary objective of this experiment was to identify the best-performing model for predicting insurance charges based on various parameters such as age, BMI, number of children, sex, and smoking status. The dataset used for this analysis was **insurance_pre.csv.**

Data Preprocessing:

In this dataset, the features **sex** and **smoker** are **nominal variables** as they can't compare and represent distinct categories without any order or ranking.

Therefore, convert text data in Sex and Smoker → numeric (0 or 1)
so that sex: male → 1, female → 0 and smoker: yes → 1, no → 0

Selected Hyperparameters:

In regression models, numerous hyperparameters influence performance and generalization. However, for this experiment, only three key hyperparameters were selected to simplify the analysis and focus on their individual impact. This approach allows for a clearer understanding of how each chosen parameter affects the model's accuracy

	age	bmi	children	charges	sex_male	smoker_yes
0	19	27.900	0	16884.92400	0	1
1	18	33.770	1	1725.55230	1	0
2	28	33.000	3	4449.46200	1	0
3	33	22.705	0	21984.47061	1	0
4	32	28.880	0	3866.85520	1	0

Top_5

The dataset consists of **1338 rows and 6 columns**, representing 1338 individual records. It contains **no missing or null values**, ensuring data completeness.
Among the features, there are **2 categorical variables** and **4 numerical variable**

Problem Identification for the Insurance Charge Prediction System

Stage 1: - In the domain selection stage, Machine Learning are applied as the all data are in numerical forms as shown in the table Top 5

Stage 2: - Supervised Learning is used because requirements are well defined, and Inputs and outputs are present as shown in the above Table Top_5

Stage 3: - There are numerical continuous values in the target variable, therefore Regression approach under supervised learning is used

R2_Score is used as evaluation metric for all model and recorded in the tabular form as it follows

Ada Boost:

n_estimators:- The maximum number of estimators at which boosting is terminated. In case of perfect fit

Learning_rate:- Weight applied to each regressor at each boosting iteration. A higher learning rate increases the contribution of each regression

Loss:- The loss function to use when updating the weights after each boosting iteration.

Model number	**n_estimators**	**Learning_rate**	**loss**	**R_Score**
1	100	1.0	linear	0.8447
2	100	0.1	exponential	0.8328
3	100	0.01	linear	0.8781
4	100	1.0	exponential	0.5385
5	100	0.1	linear	0.8607
6	100	0.01	exponential	0.8786
7	50	1.0	linear	0.8447
8	50	0.1	exponential	0.8666
9	50	0.01	linear	0.8780
10	50	1.0	exponential	0.6292
11	50	0.1	linear	0.8673
12	50	0.01	exponential	0.8813
13	10	1.0	linear	0.8447
14	10	0.1	exponential	0.8840
15	10	0.01	linear	0.8817
16	10	1.0	exponential	0.8266
17	10	0.1	linear	**0.8858**
18	10	0.01	exponential	0.8849

XGBoost:

n_estimators:- The number of trees in the ensemble, often increased until no further improvements are seen

eta:- The learning rate used to weight each model, often set to small values such as 0.3, 0.1, 0.01, or smaller

max_depth:- The maximum depth of each tree, often values are between 1 and 10.

Model	**n_estimators**	**eta**	**max_depth**	**R_Score**
1	100	0.1	3	0.8883
2	100	0.01	7	0.7597
3	100	0.001	3	0.1586
4	100	0.1	7	0.8375
5	100	0.01	3	0.7664
6	100	0.001	7	0.1585
7	50	0.1	3	**0.8929**
8	50	0.01	7	0.5548
9	50	0.001	3	0.0832
10	50	0.1	7	0.8489
11	50	0.01	3	0.5585
12	50	0.001	7	0.0829
13	10	0.1	3	0.7772
14	10	0.01	7	0.1590
15	10	0.001	3	0.0172
16	10	0.1	7	0.7688
17	10	0.01	3	0.1593
18	10	0.001	7	0.0170

LGBoost / LightGBM :

n_estimators:- The maximum number of estimators at which boosting is terminated. In case of perfect fit

Learning_rate :- Weight applied to each regressor at each boosting iteration. A higher learning rate increases the contribution of each regression

boosting_type:- gbdt', Gradient Boosting Decision Tree. 'dart', Dropouts meet Multiple Additive Regression Trees. 'rf', Random Forest.

Model number	n_estimators	Learning_rate	Boosting_ Type	R_Score
1	100	0.1	gbdt	0.8660
2	100	0.01	dart	0.1548
3	100	0.001	gbdt	0.1639
4	100	0.1	dart	0.8602
5	100	0.01	gbdt	0.7743
6	100	0.001	dart	-0.4211
7	50	0.1	gbdt	**0.8766**
8	50	0.01	dart	0.0276
9	50	0.001	gbdt	0.0861
10	50	0.1	dart	0.7974
11	50	0.01	gbdt	0.5692
12	50	0.001	dart	0.0276
13	10	0.1	gbdt	0.7857
14	10	0.01	dart	0.1495
15	10	0.001	gbdt	0.0178
16	10	0.1	dart	0.7551
17	10	0.01	gbdt	0.1646
18	10	0.001	dart	0.0160

Among all the experiments conducted using Ada Boost, XGBoost and LGBoost algorithms with their corresponding hyperparameter configurations, **XGBoost** model emerged as the best-performing model.

The **XGBoost** model achieved an accuracy of **89.29%** with the hyperparameters **n_estimators** = **50**, **eta = 0.1** and **max_depth** = **3**. This model was chosen as the final model because it outperformed all other models in terms of accuracy

Reference:

https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.AdaBoostRegressor.html

https://xgboost.readthedocs.io/en/stable/python/sklearn_estimator.html

https://machinelearningmastery.com/xgboost-for-regression/

https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMRegressor.html