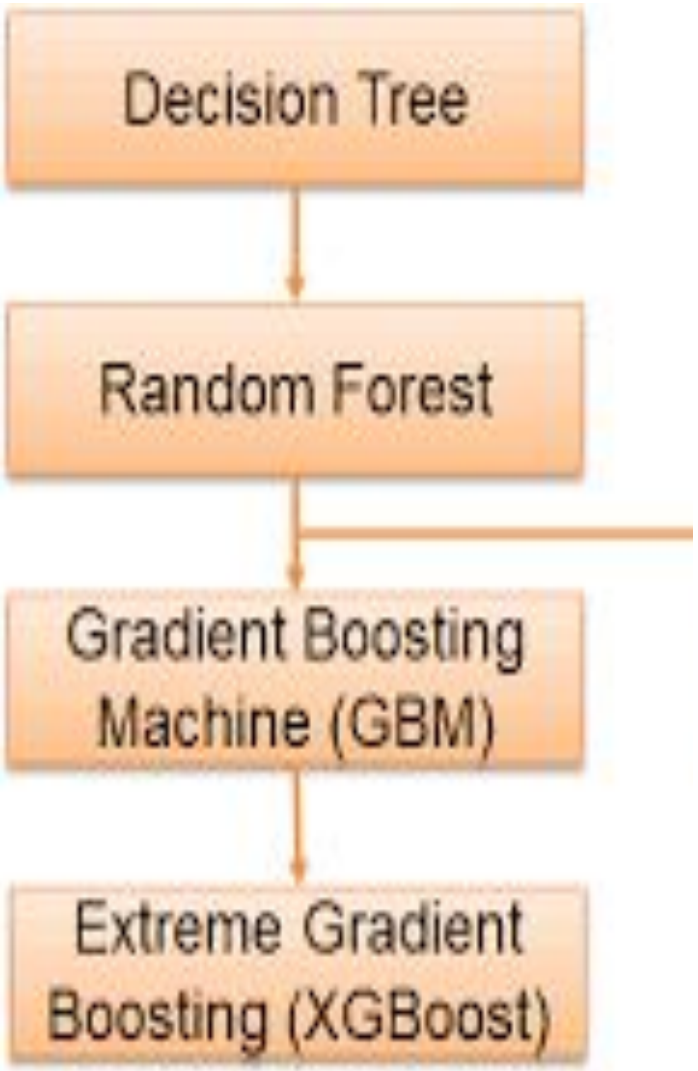
Car Price Prediction Using Machine Learning

MSDSF23M037
Talha Hussain

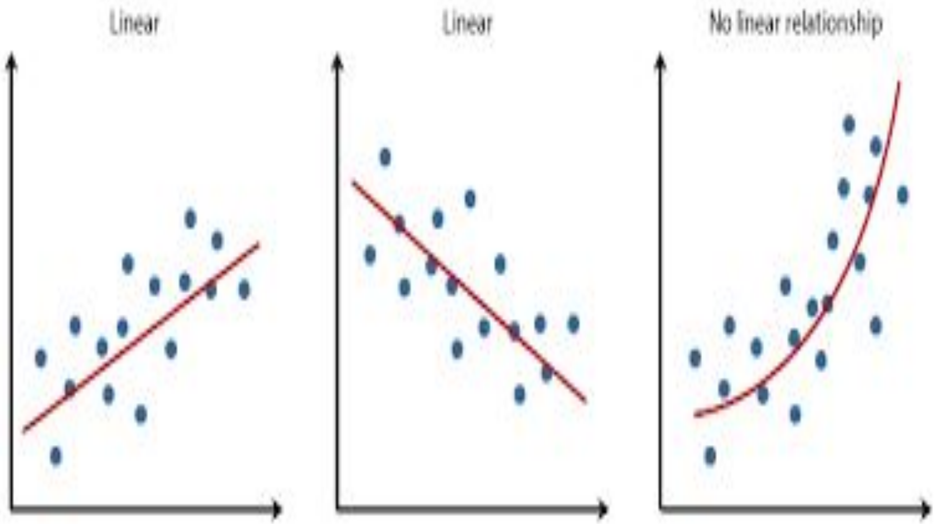
Names of Algorithums

- 1-Linear Regression
- 2-Random Forest Regressor
- 3-Gradient Boosting Regressor
- 4-XGBoost Regressor
- 5-Decision tree



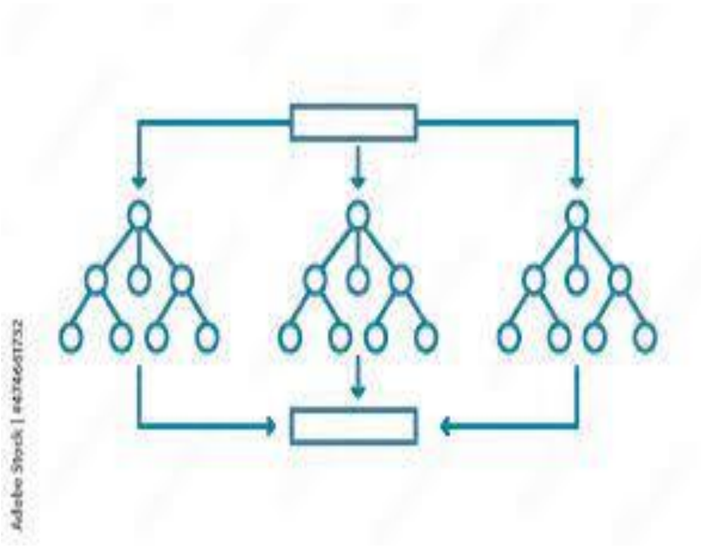
1-Linear Regression

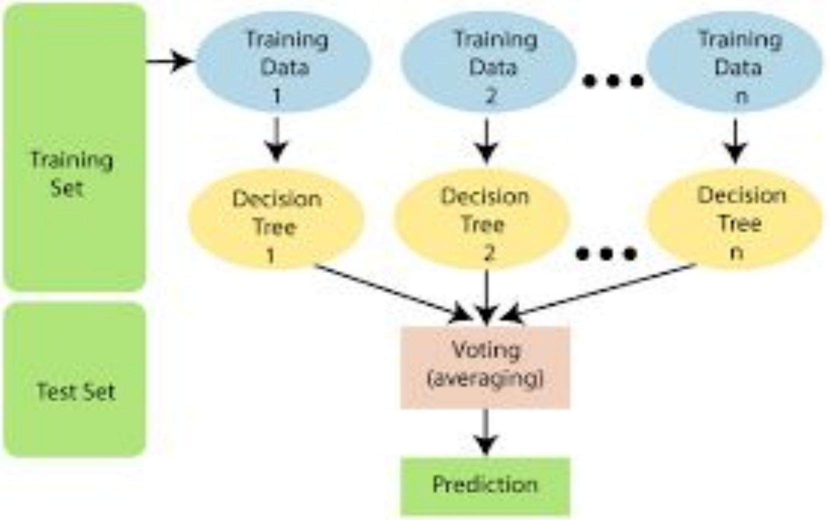
- Linear regression is a simple and interpretable regression algorithm that models the relationship between the input features and the target variable as a linear equation.
- **How it works:** It assumes a linear relationship between the input features and the target variable and tries to find the best-fitting line through the data points.



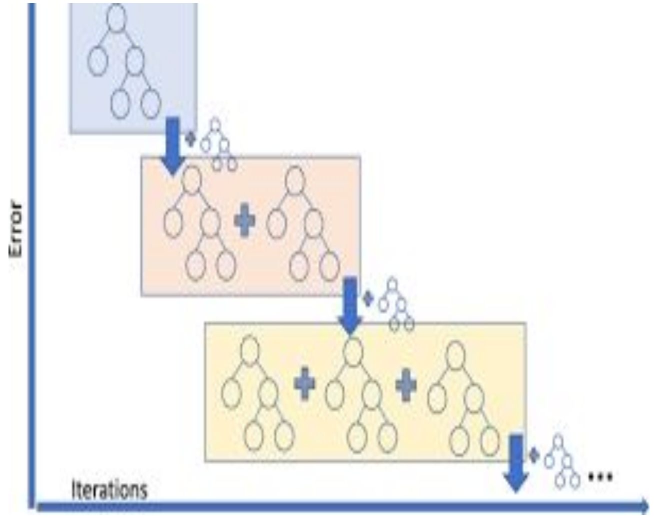
2-Random Forest Regressor

- Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the average prediction of the individual trees for regression tasks.
- **How it works:** It builds a collection of decision trees and combines their predictions to improve overall accuracy and generalization.

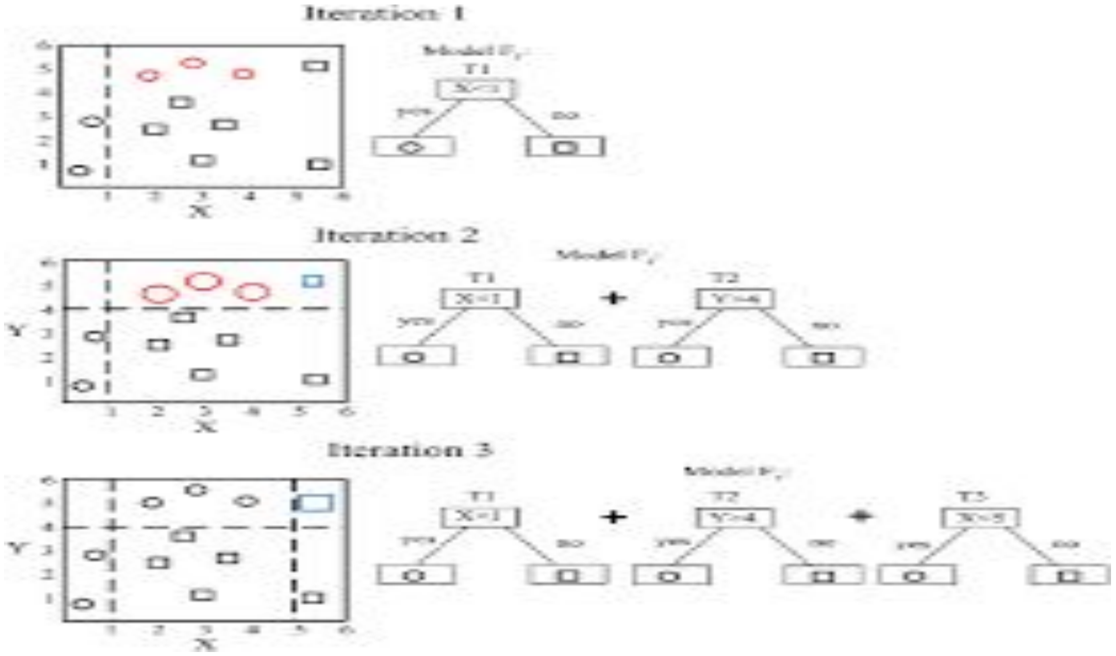




3-Gradient Boosting Regressor



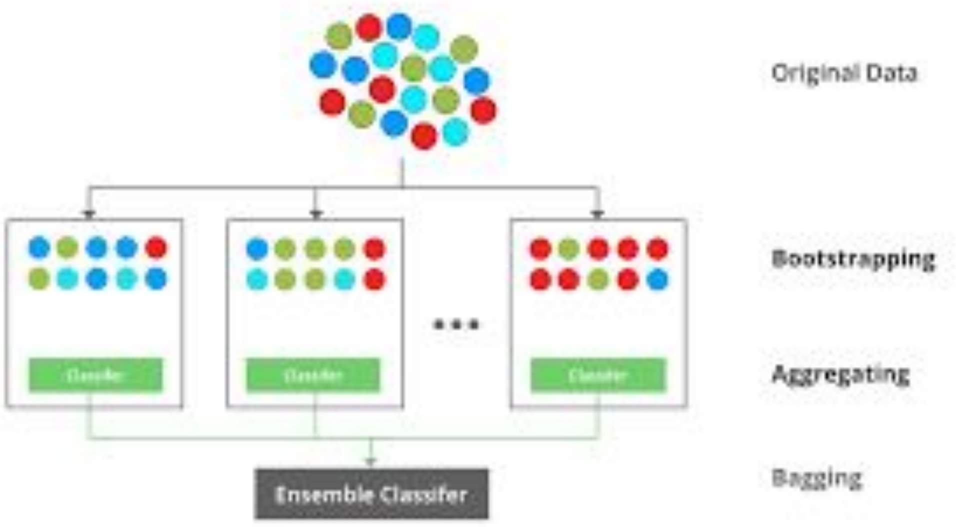
- Gradient Boosting is another ensemble learning technique that builds a series of weak learners (usually decision trees) sequentially, with each tree trying to correct the errors of the previous one.
- **How it works:** It iteratively fits new trees to the residuals (errors) of the existing model, gradually improving the overall model's performance.



4-XGBoost Regressor

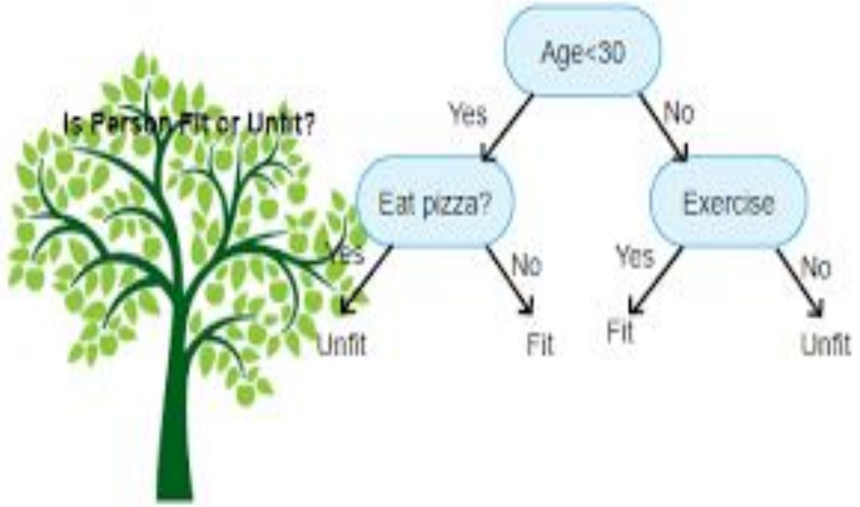
XGBoost (Extreme Gradient Boosting) is a scalable and efficient implementation of gradient boosting. It is known for its speed and performance.

How it works: Similar to traditional gradient boosting, XGBoost builds a series of decision trees sequentially, optimizing a regularized objective function to improve both accuracy and prevent overfitting.



5-Decision tree

- A decision tree model is a predictive algorithm that uses a tree-like structure to make decisions based on input features.
- **How it works:** It recursively splits data into subsets, making binary decisions at each node, and assigns outcomes or predictions at the tree's leaves. The model is interpretable and widely used for classification and regression tasks in machine learning.



Steps

- Display The Dataset
- Shape of Our Dataset
- Get Information About Our Dataset .
- Check Null Values In The Dataset
- Get Overall Statistics About The Dataset
- Data Preprocessing
- Outlier Removal
- Encoding the Categorical Columns
- Store Feature Matrix In X and Response(Target) In Vector Y
- Splitting The Dataset
- Import The models
- Training the dataset
- Tunning hyper parameters
- Prediction on Test Data
- Evaluating the Algorithm
- Error analysis

Dataset

Get Information About Our Dataset

Display the dataset

Shape of Our Dataset

1. Display Top 5 Rows of The Dataset

```
data.head()
```

	Car_Name	Year	Selling_Price	Present_Price	Kms_Driven	Fuel_Type	Seller_Type	Transmission	Owner
0	ritz	2014	3.35	5.59	27000	Petrol	Dealer	Manual	0
1	sx4	2013	4.75	9.54	43000	Diesel	Dealer	Manual	0
2	ciaz	2017	7.25	9.85	6900	Petrol	Dealer	Manual	0
3	wagon r	2011	2.85	4.15	5200	Petrol	Dealer	Manual	0
4	swift	2014	4.60	6.87	42450	Diesel	Dealer	Manual	0

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 301 entries, 0 to 300
Data columns (total 9 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Car_Name       301 non-null    object
 1   Year           301 non-null    int64
 2   Selling_Price  301 non-null    float64
 3   Present_Price  301 non-null    float64
 4   Kms_Driven     301 non-null    int64
 5   Fuel_Type      301 non-null    object
 6   Seller_Type    301 non-null    object
 7   Transmission   301 non-null    object
 8   Owner          301 non-null    int64
dtypes: float64(2), int64(3), object(4)
memory usage: 21.3+ KB
```

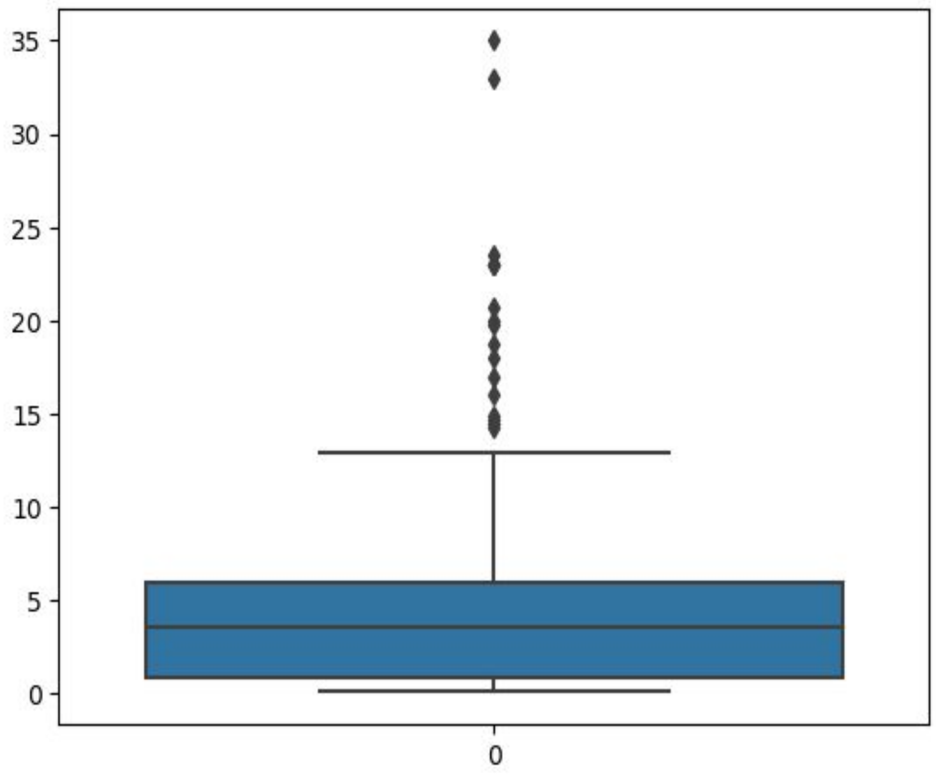
Dataset

- Check Null Values In The Dataset
- Get Overall Statistics About The Dataset
- Data Preprocessing

```
data.describe()
```

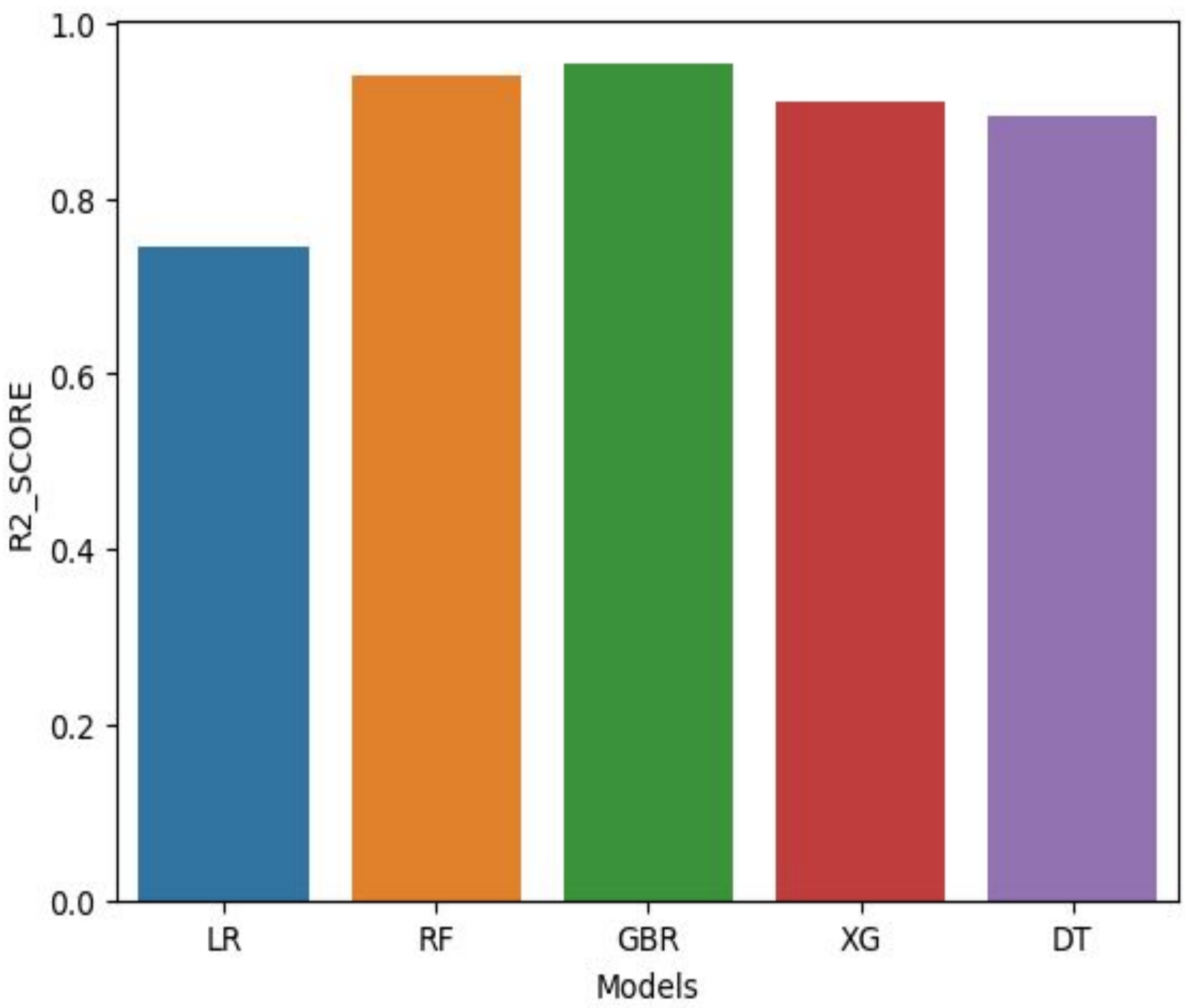
	Year	Selling_Price	Present_Price	Kms_Driven	Owner
count	301.000000	301.000000	301.000000	301.000000	301.000000
mean	2013.627907	4.661296	7.628472	36947.205980	0.043189
std	2.891554	5.082812	8.644115	38886.883882	0.247915
min	2003.000000	0.100000	0.320000	500.000000	0.000000
25%	2012.000000	0.900000	1.200000	15000.000000	0.000000
50%	2014.000000	3.600000	6.400000	32000.000000	0.000000
75%	2016.000000	6.000000	9.900000	48767.000000	0.000000
max	2018.000000	35.000000	92.600000	500000.000000	3.000000

- Outlier Removal
- Encoding the Categorical Columns
- Store Feature Matrix In X and Response(Target) In Vector Y



Dataset

- Splitting The Dataset
- Import The models
- Training the dataset
- Tunning hyper parameters
- Prediction on Test Data
- Evaluating the Algorithm



Results

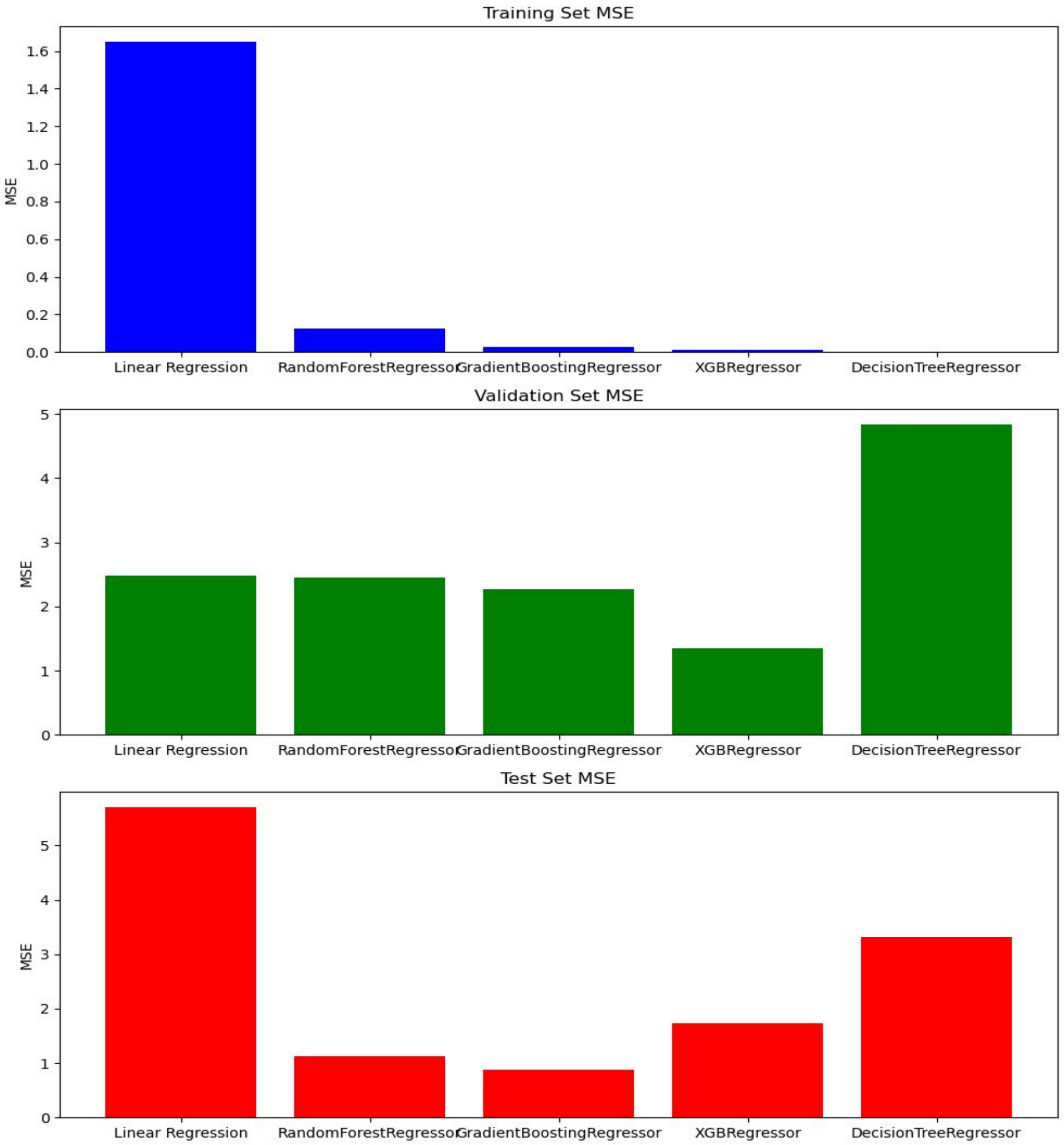
```
Linear Regression MSE on Training Set: 1.6474
Linear Regression MSE on Validation Set: 2.4866
Linear Regression MSE on Test Set: 5.6974

RandomForestRegressor MSE on Training Set: 0.1247
RandomForestRegressor MSE on Validation Set: 2.4474
RandomForestRegressor MSE on Test Set: 1.1347

GradientBoostingRegressor MSE on Training Set: 0.0250
GradientBoostingRegressor MSE on Validation Set: 2.2705
GradientBoostingRegressor MSE on Test Set: 0.8843

XGBRegressor MSE on Training Set: 0.0108
XGBRegressor MSE on Validation Set: 1.3471
XGBRegressor MSE on Test Set: 1.7333

DecisionTreeRegressor MSE on Training Set: 0.0000
DecisionTreeRegressor MSE on Validation Set: 4.8327
DecisionTreeRegressor MSE on Test Set: 3.3231
```



Error analysis

- **Linear Regression:**High training set MSE compared to validation and test sets suggests potential overfitting or complexity insufficient to capture the underlying patterns.
- **RandomForestRegressor:**Low training set MSE, but higher validation and test set MSE indicate some overfitting. However, it's not as severe as in linear regression.
- **GradientBoostingRegressor:**Low training, validation, and test set MSE indicate good generalization. This model seems well-tuned and balanced.
- **XGBRegressor:**Similar to GradientBoostingRegressor, XGBRegressor shows low MSE across all sets, indicating good generalization.
- **DecisionTreeRegressor:**MSE of 0 on the training set suggests overfitting. The high MSE on the validation and test sets indicates poor generalization.

Recommendations:

- The RandomForestRegressor, GradientBoostingRegressor, and XGBRegressor seem to perform well without significant signs of overfitting or underfitting.
- For the DecisionTreeRegressor, reducing its complexity or using techniques like pruning may help mitigate overfitting.
- Fine-tuning hyperparameters or using more advanced models could improve the performance of Linear Regression.
- It's essential to strike a balance between model complexity and generalization for optimal performance on unseen data. Cross-validation and hyperparameter tuning are common techniques to address overfitting and underfitting.

Thanks