Decision Tree Regression

Name: Sahasa Kumar Yadavalli **ID:** 23079491

Git hub link: https://github.com/Sahas-data/Machine_Learning.git

---

1. Introduction

Decision Tree Regression (DTR) is a supervised learning algorithm used for prediction of continuous numerical values. It forms a tree structure in such a way that every node is a decision on the basis of a feature, and the edges are the possible outcomes. The leaf nodes of the tree contain the predicted values of the target variable. The model splits the dataset into subsets based on feature values, incrementally passing through different feature thresholds in an effort to minimize the prediction error at each step. Decision Tree Regression is so popular because it is easy to implement, intuitive, and user-friendly.

Strength of the algorithm is that it can model non-linear relationships among input attributes and the target variable. Since the model is tree-based, it becomes easy to visualize it, making the algorithm highly interpretive and useful for feature interpretation of how input features influence prediction. It also has no requirement to scale features, which renders the algorithm robust for data preprocessing.

In this tutorial, we will explore in detail how Decision Tree Regression works, its key concepts, and how to use it. We will also cover model evaluation techniques, advantages and disadvantages of the approach, and applications in various industries. Finally, we will explore how to improve the accuracy of Decision Tree Regression models.

---

2. How Decision Tree Regression Works

Decision Tree Regression (DTR) is a type of algorithm that recursively partitions the data set into subsets based on feature values in order to construct a model to predict continuous output values. This is how it does it:

1. Start at the Root: The algorithm begins with bisecting the dataset at the root node along the feature that best separates the data. This is achieved by reducing prediction error with respect to a measure such as Mean Squared Error (MSE) for regression.

2. Recursive Partitioning: The data is divided further at each node based on the feature and threshold that best splits the data. This splitting is recursively done until a stopping criterion is met. In each level, all splits are considered by the algorithm and it selects the one that minimizes the error in predictions the most.

3. Predict Output in Leaf Node: Once the dataset is divided into subsets, the predicted output for the target variable is the leaf node. This is typically the mean of the target variable for that subset**.**

4. Stop When a Stopping Criterion is Reached: The tree construction process is halted when certain conditions are met, for instance, a specified maximum depth, a minimum number of samples in a leaf, or a specified maximum number of leaf nodes**.**

By partitioning the data into smaller units, Decision Tree Regression constructs a model that is capable of handling intricate relationships within the data.

---

3. Key Concepts and Formulas

3.1 Splitting Criteria (Mean Squared Error - MSE):

In Decision Tree Regression, splitting rule used to partition the data is typically Mean Squared Error (MSE). MSE is an approximation of the variance of the target attribute in a data subset. It aims to find splits that result in minimum MSE such that subsets are as uniform as possible as far as the target attribute is concerned. The formula for MSE is:

$$\boldsymbol{MSE} = (\boldsymbol{1/n}) * \boldsymbol{\Sigma} (\boldsymbol{y\_i} - \hat{y})\text{^}\boldsymbol{2}$$

Where:

- yi is the actual value of the target variable for each data point.

- $\hat{y}$ is the predicted value (mean of the subset).

- n is the number of data points in the subset.

With every step in decision tree growth, the algorithm computes all the splits, and the one that decreases MSE the most is chosen. By this, the tree models the data better and makes predictions accordingly**.**

3.2 Recursive Partitioning

Recursive partitioning is the process by which a decision tree splits the data at each node. The tree selects the feature and threshold that best reduce the MSE for every possible split. This is iterated recursively, with the tree splitting the data as long as a stopping criterion (e.g., maximum depth or minimum samples per leaf) is not fulfilled. The leaf nodes are the predicted values, typically the mean of the target variable in the subset. This recursive process helps construct a tree structure that is indicative of complex relationships in the data.

---

4. Implementation of Code

In this case, we used Scikit-learn's Decision Tree Regressor model to estimate the housing price in California. We first imported the data by using fetch_california_housing(). The data was split into training and testing sets, 80% of which was trained and 20% tested. StandardScaler was used to normalize the features to improve the performance of the model. The model was then fit on the scaled training data via the fit() method.

We made predictions on test data after training and calculated some performance measures. They are the Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and R-squared (R^2) score. We also plotted the Learning Curve to observe the performance of the model for varying training data sizes. The learning curve tells us about the performance of the model in terms of the training size and whether overfitting or underfitting occurs.

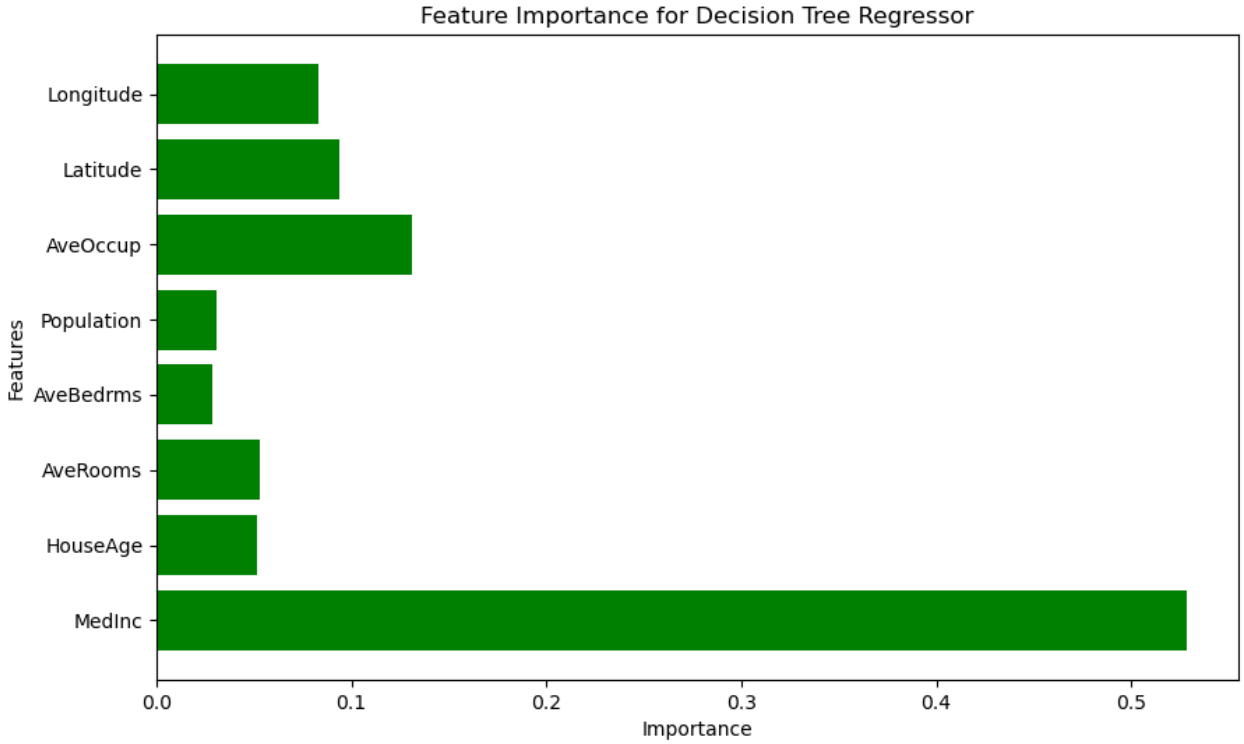
---

5. Model Evaluation and Performance

The performance of the model can be evaluated on a number of metrics, each providing valuable information:

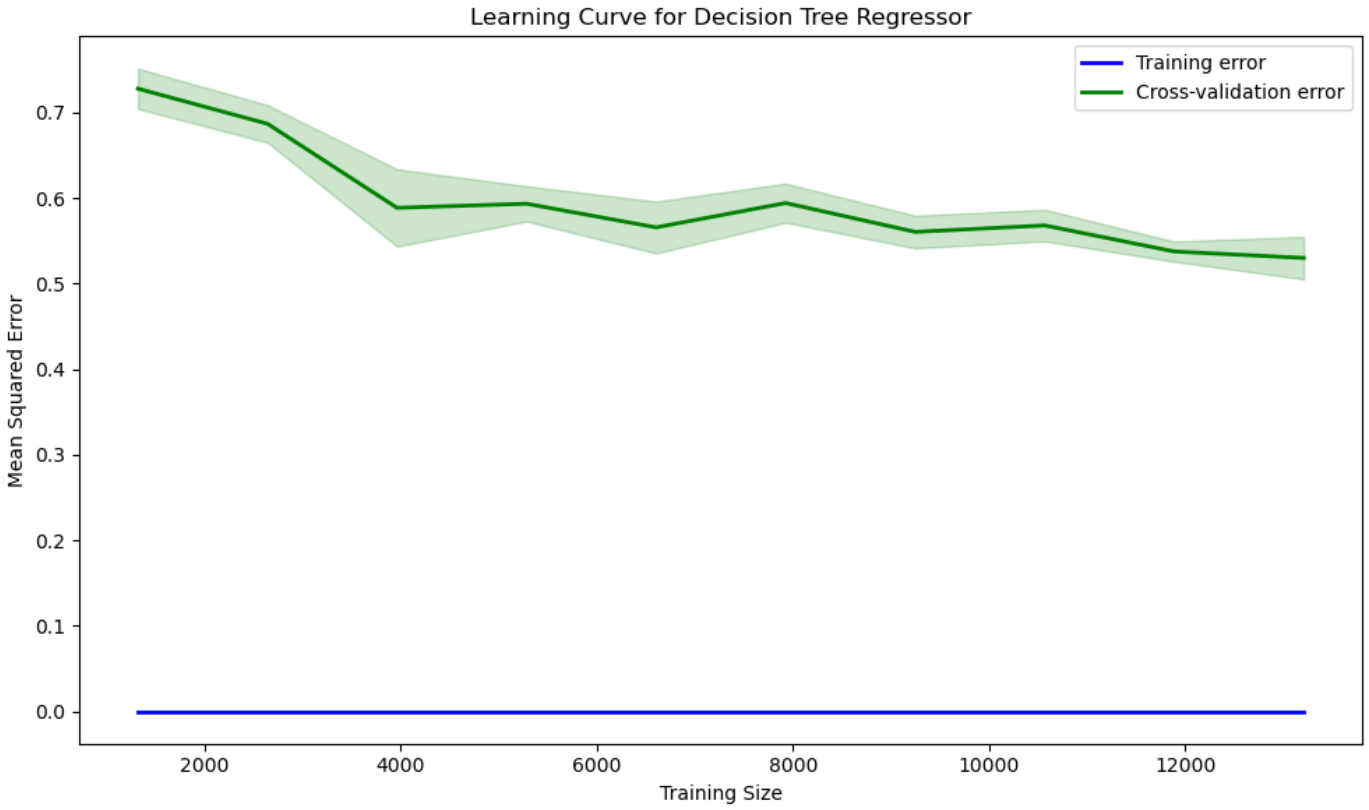
Mean Squared Error (MSE) of 0.49 informs us that predicted values are very near actual values, indicating a good fit of the model.

Root Mean Squared Error (RMSE) of 0.70, while adequate, informs us that the average prediction error of the model is about 0.70 units, which is acceptable but can be improved.



R-squared (R^2) of 0.62 indicates that 62% of the variance of the target variable (house prices) is explained by the model. However, the remaining 38% is not explained, indicating that there is room for model

improvement.



By these measures, the model is a good fit for the simple regression problem, though there is some space for improvement. Overfitting is apparent because training error is 0.0, while cross-validation error is considerably higher. To set things straight, we could prune the tree or limit its depth. Some more feature engineering and experimenting with more sophisticated models like Random Forest or Gradient Boosting can lead to better generalization and higher performance**.**

---

6. Advantages, Pros & Cons, and Comparison with Other ML Algorithms

Advantages:

- Interpretability : Decision Trees are simple to interpret and visualize. The tree structure gives clear perspectives on decision-making, making them a popular choice for domains where model interpretability is paramount.

- Non-linearity : Decision Trees can model non-linear relationships between data, which is a significant advantage compared to linear models that have to be hand-designed to address non-linear issues.

- No feature scaling needed : Unlike techniques like Support Vector Machines or k-NN, Decision Trees do not require normalization or feature standardization, so they are easier to utilize on raw data.

Disadvantages:

- Overfitting : Decision Trees are prone to overfitting, especially with complex data. Without appropriate pruning or termination points, they can learn the noise and not be able to generalize to unseen data.

- Instability : Low variability in the training set will lead to high variability in the tree structure, and hence Decision Trees are highly data-sensitive.

- Bias towards features with higher levels : Decision Trees are prone to being biased towards features with higher numbers of levels or different values and thus might give biased results.

Comparison with Other Algorithms:

- **Vs. Linear Regression :** Decision Trees don't need explicit feature engineering like Linear Regression in order to capture non-linear relationships.

- **Vs. Random Forest :** Random Forest, by the averaging of a number of trees, reduces overfitting and is less susceptible to variation from a single Decision Tree.

- **Vs. K-Nearest Neighbours :** While noisy data could disturb K-NN, Decision Trees are less obscure and possess clearer decision-making criteria.

---

7. Applications

Decision Tree Regression is widely applied across various industries to predict continuous values from input features.

- Real Estate : In real estate, Decision Trees are used to predict house prices based on features like square footage, location, number of rooms, and other property features.

- Healthcare : Decision Trees are utilized in healthcare to determine the health treatment cost based on the patient information that comprises their age, their history of medical records, and their particular treatment they require.

- Finance : Financial institutions like banks use Decision Tree Regression to forecast stock prices, predict financial performance, or predict future trends based on historical data and market indicators.

- Marketing : In marketing, Decision Trees are used to predict the customer lifetime value (CLV) from customer behaviour, purchasing habits, and demographic factors. This helps companies make informed decisions regarding marketing planning and resource allocation.

These industries benefit from Decision Tree Regression's capacity to identify non-linear relationships and from its interpretability in decision-making.

---

8. How to Improve Accuracy

To enhance the performance of Decision Tree Regression, several techniques can be employed:

1. **Pruning :** Pruning is one of the strongest techniques to prevent overfitting. Pruning ensures the tree generalizes well to unseen data by removing branches that don't contribute significantly to the model's predictive power. This can be achieved either by limiting the depth of the tree or by using minimum samples per leaf.

2. **Cross-Validation :** Using cross-validation, i.e., K-fold cross-validation, helps to assess the model's performance better. It ensures that the model's accuracy is checked on different subsets of the dataset to prevent the possibility of overfitting to a single training set.

3. **Ensemble Methods :** Models such as Random Forest or Gradient Boosting combine multiple decision trees to improve predictive power. These algorithms reduce variance (Random Forest) or bias (Gradient Boosting), leading to more stable models.

4. **Feature Engineering :** Incorporating additional relevant features or encoding existing ones can significantly impact the model's predictive ability. Feature selection techniques can be applied to identify the strongest predictors of the target variable as well.

5. **Hyperparameter Tuning :** The model can be optimized more by tuning hyperparameters such as max depth, min samples per split, etc. This can lead to a more accurate and efficient model. Grid search or randomized search can be used for finding the optimal combination of hyperparameters.

Collectively, these methods can be employed to render Decision Tree Regression more robust and accurate for regression issues.

---

9. Conclusion

Decision Tree Regression is a widely used and robust algorithm that provides interpretable models for continuous value prediction. It recursively partitions data based on feature values, and every leaf node maps to a predicted value. One of the strongest features of Decision Tree Regression is its ability to model non-linear relationships between features and the target variable, thereby being applicable to a broad array of regression problems.

The algorithm is simple and easy to understand, and the users are able to simply chart out the decision-making process and observe what kind of patterns are happening in the data. This makes it a great tool in healthcare, finance, and marketing industries where transparency is paramount.

However, Decision Tree Regression is prone to overfitting, especially with deep trees, where the model can learn noise or irrelevant patterns in the training data. This can be countered by techniques like pruning, which removes unnecessary branches, and cross-validation to check the performance of the model. Additionally, ensemble techniques like Random Forest or Gradient Boosting can be employed to avoid overfitting and improve accuracy by combining multiple trees.

In short, Decision Tree Regression is an essential and effective regression model with good interpretability and flexibility. Provided that the right tuning and optimization techniques, such as pruning, cross-validation, and ensemble, are used, the accuracy of Decision Tree models can greatly be improved, and they can be a valuable tool in machine learning processes across industries**.**

---

10. References

1. **Breiman, L., et al. (1986).** ***Classification and Regression Trees.***
2. **Scikit-learn Documentation: https://scikit-learn.org/stable/modules/tree.html**
3. **Quinlan, J. R. (1986).** ***Induction of Decision Trees.***
4. **Hastie, T., Tibshirani, R., & Friedman, J. (2009).** ***The Elements of Statistical Learning: Data Mining, Inference, and Prediction.***
