*Future Sales Prediction with Machine Learning*

Problem Statement :-

The problem at hand is to develop a predictive model that uses historical sales data to forecast future sales for a retail company. The goal is to create a tool that enables the company to optimize inventory management and make informed business decisions based on data-driven sales prediction. The key challenges include dealing with large datasets, handling missing data, selecting appropriate features, and choosing the right machine learning algorithms for accurate predictions.

Machine Learning Model used :-

A decision tree is a powerful and interpretable machine learning algorithm used for both classification and regression tasks. It recursively splits the dataset into subsets based on the most significant attribute at each node. The goal is to create a tree-like structure where each internal node represents a decision based on a particular feature, and each leaf node represents the predicted outcome.

In the context of regression, like in your code, decision trees predict a continuous target variable. The splitting process involves selecting the feature and the split point that minimizes the variance of the target variable within the subsets. The result is a tree that can make predictions for new data by traversing the branches from the root to a leaf node.

Decision trees are advantageous for their simplicity, interpretability, and ability to handle non-linear relationships in data. However, they are prone to overfitting, especially when deep trees are constructed. Techniques like pruning or using ensemble methods (e.g., Random Forests) can help mitigate this issue and enhance the model's generalization performance.

Code :-

*import pandas as pd*

*import numpy as np*

*import seaborn as sns*

*import matplotlib.pyplot as plt*

*from sklearn.model\_selection import train\_test\_split*

*from sklearn.tree import DecisionTreeRegressor,plot\_tree*

*data = pd.read\_csv('Sales.csv')*

*print("Description of columns : \n")*

*print(data.describe()) #decription of each column*

*print("\nNo.of Null Columns :\n",data.isnull().sum()) #count of null values in columns*

*data = data.dropna() #to remove null data*

*print("\nHeatmap to show correlation between attributes :-")*

*correlation=data.corr(method='pearson')*

*sns.heatmap(correlation, cmap="coolwarm",annot=True)*

*plt.show() #To show correlation between attributes*

*x=data[["TV","Radio","Newspaper"]]*

*y=data["Sales"]*

*xtr,xte,ytr,yte=train\_test\_split(x,y,test\_size=0.2,random\_state=42)*

*model = DecisionTreeRegressor()*

*model.fit(xtr,ytr)*

*plt.figure(figsize=(20, 10), dpi=600)*

*plot\_tree(model, filled=True, feature\_names=list(xtr.columns))*

*plt.savefig("tree\_diagram.png", dpi=600)*

*plt.show()*

*features = np.array([[200,40,100]])*

*us=model.predict(features)*

*print("Advertising cost on TV  : $",features[0][0],)*

*print("Advertising cost on Radio : $",features[0][1])*

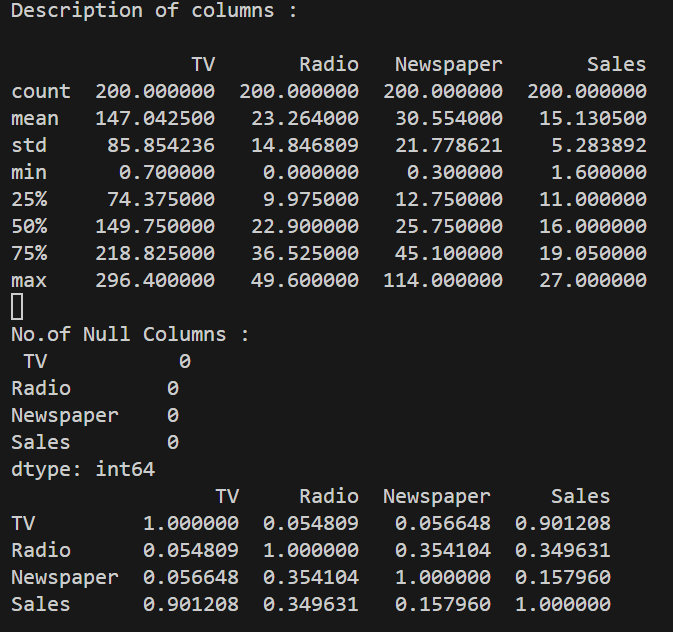
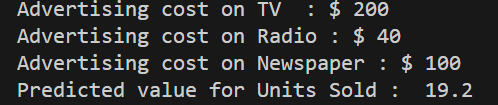
*print("Advertising cost on Newspaper : $",features[0][2])*

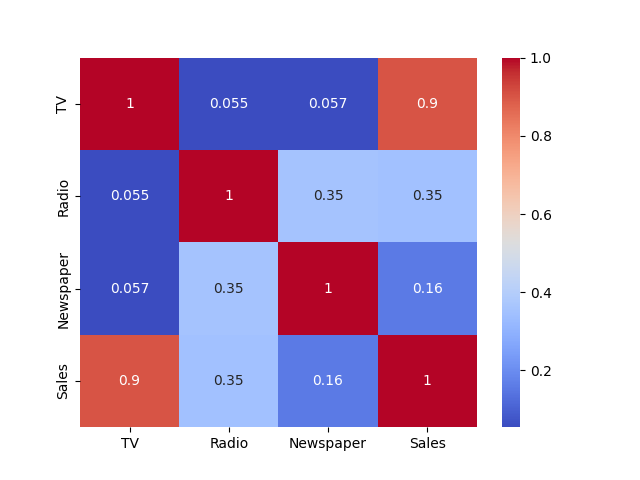
*print("Predicted Sales value for Units Sold : ",us[0])*

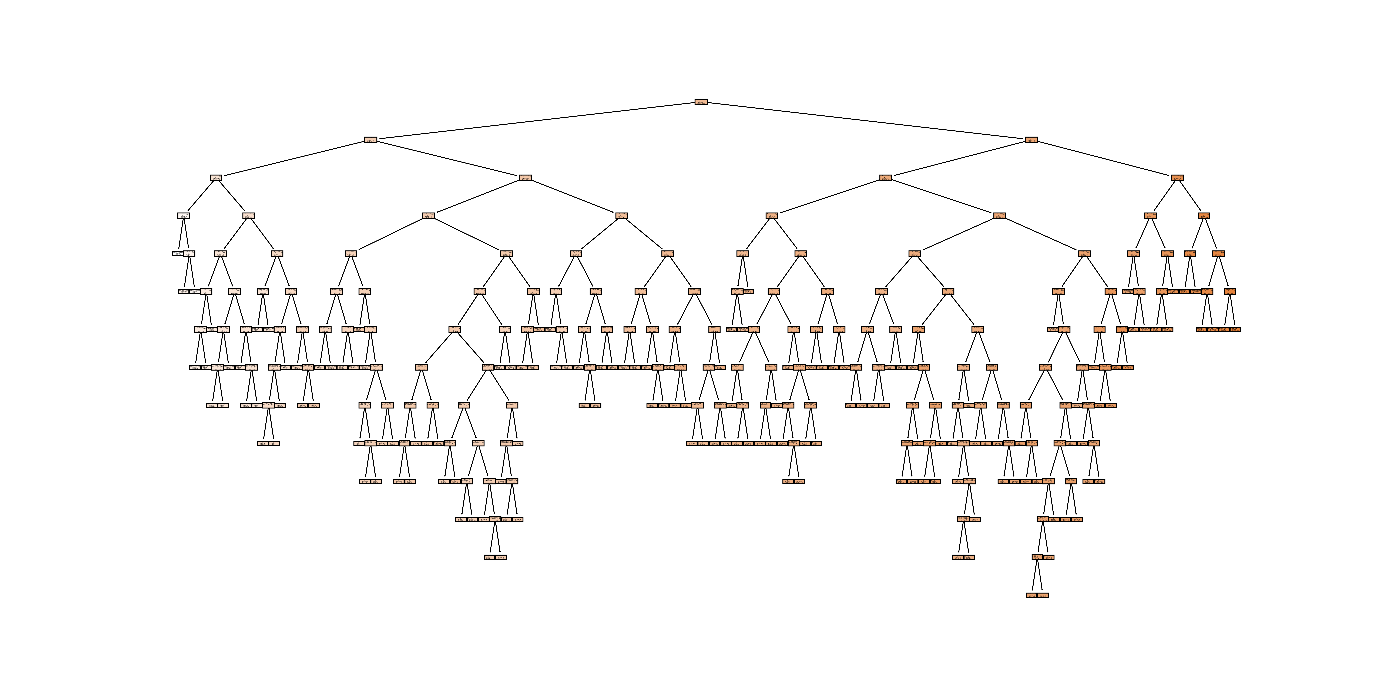
Explanation :-

This code reads sales data from a CSV file, preprocesses it by handling null values, and visualizes the correlation between advertising channels (TV, Radio, Newspaper) and sales using a heatmap. It then splits the data into training and testing sets, constructs a decision tree regression model, and visualizes the model as a tree diagram, saving it as 'tree\_diagram.png'. Finally, it performs a sample prediction using the trained model and prints the predicted units sold based on given advertising costs for TV, Radio, and Newspaper.

Output:-





Decision Tree Diagram