Future Sales Prediction

Problem Statement:

The objective of this project is to develop a predictive model that forecasts future sales based on historical data. The system aims to assist in understanding and predicting sales patterns, thereby aiding in inventory management, resource allocation, and decision-making for the business.

Design Thinking Process:

- **Empathize:**
  - ➢ Understanding the stakeholders' needs and concerns regarding sales forecasting.
- **Define**:
  - ➢ Defining the problem, setting objectives, and identifying key metrics for successful sales prediction.
- **Ideate:**
  - ➢ Generating potential solutions and strategies for effective sales forecasting.
- **Prototype:**
  - ➢ Developing and testing different models and approaches for sales prediction.
- **Test:**
  - ➢ Evaluating the models' performance and iterating based on the feedback received.

Phases of Development:

- **Data Collection:**
  - Gathering historical sales data, including parameters such as date, product information, sales quantity, and other relevant variables.
- **Data Preprocessing:**
  - Cleaning the dataset, handling missing values, dealing with outliers, and transforming data into a usable format for analysis.
- **Model Development:**
  - Building and training time series forecasting models using suitable algorithms.
- **Model Evaluation:**
  - Assessing the models' accuracy, fine-tuning parameters, and validating predictions.
- **Deployment:**
  - Integrating the best-performing model into a user-friendly interface or system for real-time sales forecasting.

Dataset Description:

The dataset comprises historical sales records containing information such as TV, Newspaper, Radio, Sales. It covers regular sales patterns and fluctuations.

- **TV:**
  - This column represents the advertising budget allocated to TV advertising for a specific product or campaign. The budget may be measured in monetary units (e.g., dollars).
- **Newspaper:**
  - This column represents the advertising budget allocated to newspaper advertising. It includes the amount spent on newspaper advertisements for the same product or campaign.

- **Radio:**
    - This column represents the advertising budget allocated to radio advertising. It includes the amount spent on radio advertisements.
- **Sales:**
    - The "Sales" column provides information on the product's sales performance. It typically measures the number of units sold, revenue generated, or any other relevant sales metric.

Data Preprocessing :

- Handle missing values.
- Check for outliers and consider their treatment, if necessary.
- Normalize numerical features to ensure all features are on a consistent scale.
- Encode categorical variables.
- Split the dataset into training and validation sets to evaluate the model's performance.

Model Training Process:

- **Feature Selection:**
    - Identifying the most influential features for the model.
- **Model Selection:**
    - Choosing appropriate Machine Learning algorithm based on the dataset's characteristics to predict the future sales.
- **Training the Model:**
    - Utilizing the training data to fit the model using train_test_split.
- **Standardizaiton:**
    - Standardize the features using StandardScaler().

- **Validation:**
  - Assessing the model's performance on validation data using metrics such as Mean Squared Error (MSE), R-Squared Error (R2).

Choice of Time Series Forecasting Algorithm and Evaluation Metrics:

Machine Learning Algorithm Selection:

For this project, we have chosen to use Linear Regression as the primary machine learning algorithm to predict future sales. Linear Regression is a suitable choice given the characteristics of the dataset, which includes features like "TV," "Newspaper," and "Radio." Linear Regression models the relationship between these advertising investments and sales by assuming a linear connection. This model is interpretable, computationally efficient, and can provide insights into how each advertising channel influences sales.

Evaluation Metrics:

- **Mean Absolute Error (MAE):**
  - It measures the average absolute difference between predicted and actual sales values.
- **Mean Squared Error (MSE):**
  - It quantifies the average of the squared differences between predicted and actual values.

- **R-squared (R2):**
    - This metric assesses the proportion of variance in sales that is predictable by the model.

Model Coefficients and Intercept :

- **Model Coefficients :**
    - It represent the impact of each feature (advertising channel) on sales.
    - A positive coefficient indicates an increase in the feature leads to higher sales.
    - A negative coefficient suggests the opposite.

- **Model Intercept :**
    - It is the expected sales when all features are zero.
    - It provides a baseline for sales prediction.
    - Understanding these values guides resource allocation and marketing decisions.

Project Code :

```
# Import all the necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler
```

```
# Read the dataset
df = pd.read_csv("Sales.csv")
# Display the first ten rows
df.head(10)
# Display the last rows
df.tail()
# Describe the dataset
df.describe()
# Get the shape of the dataset
df.shape
# Get the size of the dataset
df.size
# Get information about the dataset
df.info()
# Check for null values
df.isnull()
# Total null values
df.isnull().sum().sum()

# Get the columns
df.columns
# Visualize outliers using box plots
fig, axis = plt.subplots(3, figsize=(5, 5))
plt1 = sns.boxplot(df['TV'], ax=axis[0])
plt2 = sns.boxplot(df['Radio'], ax=axis[1])
plt3 = sns.boxplot(df['Newspaper'], ax=axis[2])
plt.tight_layout()
```

```
# Assign variables for feature engineering
x1 = df["TV"]
x2 = df["Newspaper"]
x3 = df["Radio"]
y = df["Sales"]
# Plotting the original features
plt.scatter(x1, y)
plt.scatter(x2, y)
plt.scatter(x3, y)
# Standardize the features using StandardScaler
scaler = StandardScaler()
x1 = np.array(x1).reshape(-1, 1)
x2 = np.array(x2).reshape(-1, 1)
scaler_x1 = StandardScaler().fit(x1)
scaler_x2 = StandardScaler().fit(x2)
x1_scaled = scaler_x1.transform(x1)
x2_scaled = scaler_x2.transform(x2)
# Feature Engineering
# 1. Interaction Features
df['TV_Radio'] = df['TV'] * df['Radio']
df['TV_Newspaper'] = df['TV'] * df['Newspaper']
df['Radio_Newspaper'] = df['Radio'] * df['Newspaper']
# 2. Polynomial Features
df['TV^2'] = df['TV']**2
df['Radio^2'] = df['Radio']**2
df['Newspaper^2'] = df['Newspaper']**2
# 3. Log Transformation
```

```
df['TV_log'] = np.log(df['TV'])
df['Radio_log'] = np.log(df['Radio'])
df['Newspaper_log'] = np.log(df['Newspaper'])
#linear regression
from sklearn.linear_model import LinearRegression
#Train and test the model
from sklearn.model_selection import train_test_split
# Split the data into training and testing sets
X = np.column_stack((x1, x2, x3))  # Use the features you want to include
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Create and fit a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)
# Make predictions on the test set
y_pred = model.predict(X_test)
# Evaluate the model
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
# Print the model's performance metrics
print("Mean Squared Error:", mse)
print("R-squared (R2) Score:", r2)
 #Model's coefficients and intercept
print("Coefficients:", model.coef_)
print("Intercept:", model.intercept_)
```

Conclusion:

In this project, we utilized Linear Regression to predict future sales based on advertising investments in "TV," "Newspaper," and "Radio." The model's coefficients revealed the influence of each channel on sales, aiding resource allocation decisions. The intercept provided a sales baseline. With MSE and R-Squared Error as evaluation metrics, we assessed model performance. This project empowers data-driven marketing strategies, offering insights for optimizing advertising budgets and maximizing sales outcomes.