# SALES PREDICTION USING PYTHON

We'll use the provided Advertising.csv file, which likely contains data on advertising expenditures across various platforms and the resulting sales. Here's the general plan for our project:

1. **Data Loading and Exploration**:
   * Load the data from the CSV file.
   * Explore the data to understand its structure and contents.
2. **Data Preprocessing**:
   * Check for missing values and handle them if any.
   * Perform any necessary feature engineering.
3. **Exploratory Data Analysis (EDA)**:
   * Visualize the data to understand relationships between variables.
   * Summarize key statistics.
4. **Model Building**:
   * Split the data into training and testing sets.
   * Train a machine learning model (e.g., Linear Regression) to predict sales based on advertising expenditures.
5. **Model Evaluation**:
   * Evaluate the model's performance using appropriate metrics (e.g., Mean Squared Error, R-squared).
6. **Predictions**:
   * Use the trained model to make predictions on new data.

Let's start by loading the data and performing some initial exploration.

python

Copy code

import pandas as pd

# Load the data

file\_path = '/mnt/data/Advertising.csv'

data = pd.read\_csv(file\_path)

# Display the first few rows of the dataset

data.head()

This will give us an overview of the dataset. After this, we'll proceed with data preprocessing and exploration. Let's run this first step.

The dataset has the following columns:

* Unnamed: 0: This seems to be an index column.
* TV: Amount spent on TV advertising.
* Radio: Amount spent on radio advertising.
* Newspaper: Amount spent on newspaper advertising.
* Sales: Sales figures.

Let's move to the next steps of data preprocessing and exploratory data analysis (EDA).

**Data Preprocessing**

1. **Remove Unnecessary Columns**: The Unnamed: 0 column can be removed.
2. **Check for Missing Values**: Ensure there are no missing values in the dataset.

**Exploratory Data Analysis (EDA)**

1. **Summary Statistics**: Get summary statistics for each column.
2. **Visualizations**:
   * Histogram of each feature.
   * Scatter plots to see relationships between advertising expenditure and sales.

Let's start with the preprocessing steps:

python

Copy code

# Remove the Unnamed: 0 column

data = data.drop(columns=['Unnamed: 0'])

# Check for missing values

missing\_values = data.isnull().sum()

# Display the summary statistics

summary\_stats = data.describe()

missing\_values, summary\_stats

Let's run this code to clean the data and explore it further. ​​

The data is clean with no missing values. Here are the summary statistics:

* **TV Advertising**:
  + Mean: 147.04
  + Std Dev: 85.85
  + Min: 0.7
  + Max: 296.4
* **Radio Advertising**:
  + Mean: 23.26
  + Std Dev: 14.85
  + Min: 0.0
  + Max: 49.6
* **Newspaper Advertising**:
  + Mean: 30.55
  + Std Dev: 21.78
  + Min: 0.3
  + Max: 114.0
* **Sales**:
  + Mean: 14.02
  + Std Dev: 5.22
  + Min: 1.6
  + Max: 27.0

**Next Steps: Exploratory Data Analysis (EDA)**

Let's visualize the data to understand the relationships between the advertising expenditures and sales:

1. **Histograms**: To see the distribution of each variable.
2. **Scatter Plots**: To examine relationships between advertising expenditures and sales.

Let's create these visualizations.

python

Copy code

import matplotlib.pyplot as plt

import seaborn as sns

# Set up the matplotlib figure

plt.figure(figsize=(16, 8))

# Histograms

plt.subplot(2, 2, 1)

sns.histplot(data['TV'], kde=True)

plt.title('TV Advertising Distribution')

plt.subplot(2, 2, 2)

sns.histplot(data['Radio'], kde=True)

plt.title('Radio Advertising Distribution')

plt.subplot(2, 2, 3)

sns.histplot(data['Newspaper'], kde=True)

plt.title('Newspaper Advertising Distribution')

plt.subplot(2, 2, 4)

sns.histplot(data['Sales'], kde=True)

plt.title('Sales Distribution')

plt.tight\_layout()

plt.show()

# Scatter plots

plt.figure(figsize=(16, 8))

plt.subplot(1, 3, 1)

sns.scatterplot(x='TV', y='Sales', data=data)

plt.title('Sales vs TV Advertising')

plt.subplot(1, 3, 2)

sns.scatterplot(x='Radio', y='Sales', data=data)

plt.title('Sales vs Radio Advertising')

plt.subplot(1, 3, 3)

sns.scatterplot(x='Newspaper', y='Sales', data=data)

plt.title('Sales vs Newspaper Advertising')

plt.tight\_layout()

plt.show()

Let's run these visualizations to better understand our data. ​​

The visualizations provide valuable insights into the data:

* **Histograms**:
  + TV, Radio, and Newspaper advertising expenditures are fairly spread out, with a slight right skew.
  + Sales distribution is slightly right-skewed, with most values between 5 and 25.
* **Scatter Plots**:
  + **Sales vs TV Advertising**: Positive correlation; higher TV spending generally results in higher sales.
  + **Sales vs Radio Advertising**: Positive correlation, though with more spread.
  + **Sales vs Newspaper Advertising**: Less clear correlation compared to TV and Radio.

**Next Steps: Model Building**

We'll build a machine learning model to predict sales based on advertising expenditures. Let's start with a simple linear regression model.

1. **Split the Data**: Divide the data into training and testing sets.
2. **Train the Model**: Use the training data to fit the model.
3. **Evaluate the Model**: Assess the model's performance on the test data.

Let's proceed with splitting the data and training the model.

python

Copy code

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

# Split the data into training and testing sets

X = data[['TV', 'Radio', 'Newspaper']]

y = data['Sales']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Train the model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Make predictions

y\_pred = model.predict(X\_test)

# Evaluate the model

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

mse, r2

Let's run this code to train and evaluate our linear regression model. ​​

It seems like I can’t do more advanced data analysis right now. Please try again later.

In the meantime, you can run the following code in your Google Colab notebook to train and evaluate the linear regression model:

python

Copy code

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

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

# Split the data into training and testing sets

X = data[['TV', 'Radio', 'Newspaper']]

y = data['Sales']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Train the model

model = LinearRegression()

model.fit(X\_train, y\_train)

# Make predictions

y\_pred = model.predict(X\_test)

# Evaluate the model

mse = mean\_squared\_error(y\_test, y\_pred)

r2 = r2\_score(y\_test, y\_pred)

print(f'Mean Squared Error: {mse}')

print(f'R-squared: {r2}')

A group of blue graphs

Description automatically generated with medium confidence

A group of blue dots

Description automatically generated

1 to 5 of 5 entriesFilter

| **index** | **Unnamed: 0** | **TV** | **Radio** | **Newspaper** | **Sales** |
| --- | --- | --- | --- | --- | --- |
| **0** | 1 | 230.1 | 37.8 | 69.2 | 22.1 |
| **1** | 2 | 44.5 | 39.3 | 45.1 | 10.4 |
| **2** | 3 | 17.2 | 45.9 | 69.3 | 9.3 |
| **3** | 4 | 151.5 | 41.3 | 58.5 | 18.5 |
| **4** | 5 | 180.8 | 10.8 | 58.4 | 12.9 |

Show 102550100 per page

Like what you see? Visit the [data table notebook](https://colab.research.google.com/notebooks/data_table.ipynb) to learn more about interactive tables.

#### Distributions

A graph of blue bars

Description automatically generated with medium confidence

A graph of blue lines

Description automatically generated with medium confidence

A graph of a number of blue rectangular objects

Description automatically generated with medium confidence

A graph of a bar

Description automatically generated

#### 2-d distributions

A graph with blue dots

Description automatically generated

A graph with blue dots

Description automatically generated

A graph with blue dots

Description automatically generated

A graph with blue dots

Description automatically generated

#### Time series

A line graph with numbers and a green line

Description automatically generated

A green line graph

Description automatically generated

A line graph with a green line

Description automatically generated

A line graph with a green line

Description automatically generated

#### Values