Experiment No 4

* 1. **Aim/Purpose of the Experiment**

Model building using Linear Regression.

* 1. **Learning Outcomes**

Knowledge of the Data cleaning, modelling using linear regression, and different libraries in python.

* 1. **Prerequisites**

Basic knowledge of programming. Coordinate geometry (Straight line), Matplotlib, seaborn, etc.

* 1. **Materials/Equipment/Apparatus / Devices/Software required**

Jupyter Notebook.

* 1. **Introduction and Theory**

Linear regression is a fundamental statistical method used for modeling the relationship between a dependent variable and one or more independent variables. It's often employed in predictive analysis and understanding the association between variables.

* Dependent variable (Y): This is the variable that you want to predict or explain. It's also known as the response variable.
* Independent variable(s) (X): These are the variables that are used to predict the dependent variable. They are also referred to as predictor variables or features.
* Linear relationship: Linear regression assumes that there is a linear relationship between the independent variables and the dependent variable. This means that the change in the dependent variable is directly proportional to a change in the independent variable(s), with a constant rate of change.
* Simple linear regression: When there is only one independent variable, it's called simple linear regression. The relationship between the independent and dependent variables is modeled using a straight line equation: Y = β₀ + β₁X + ε, where β₀ is the intercept, β₁ is the slope coefficient, X is the independent variable, and ε represents the error term.
* Multiple linear regression: When there are multiple independent variables, it's called multiple linear regression. The relationship between the independent and dependent variables is modeled using the equation: Y = β₀ + β₁X₁ + β₂X₂ + ... + βᵣXᵣ + ε, where X₁, X₂, ..., Xᵣ are the independent variables, β₀ is the intercept, β₁, β₂, ..., βᵣ are the coefficients, and ε represents the error term.

**Simple Linear Regression**

## Step 1: Reading and Understanding the Data

Let's start with the following steps:

1. Importing data using the pandas library
2. Understanding the structure of the data

# Supress Warnings

import warnings

warnings.filterwarnings('ignore')

# Import the numpy and pandas package

import numpy as np

import pandas as pd

# Read the given CSV file, and view some sample records

advertising = pd.read\_csv("advertising.csv")

advertising.head()

advertising.shape

advertising.info()

advertising.describe()

## Step 2: Visualising the Data

Let's now visualise our data using seaborn. We'll first make a pairplot of all the variables present to visualise which variables are most correlated to Sales

import matplotlib.pyplot as plt

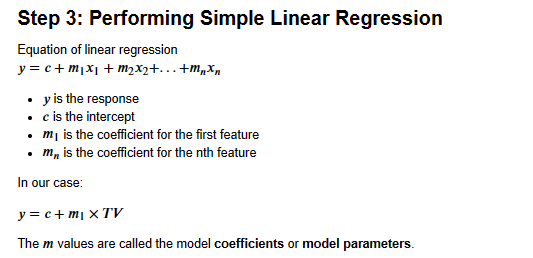
import seaborn as sns

sns.pairplot(advertising, x\_vars=['TV', 'Newspaper', 'Radio'], y\_vars='Sales',size=4, aspect=1, kind='scatter')

plt.show()

sns.heatmap(advertising.corr(), cmap="YlGnBu", annot = True)

plt.show()



Generic Steps in model building using statsmodels

We first assign the feature variable, TV, in this case, to the variable X and the response variable, Sales, to the variable y.

X = advertising['TV']

y = advertising['Sales']

Train-Test Split

You now need to split our variable into training and testing sets. You'll perform this by importing train\_test\_split from the sklearn.model\_selection library. It is usually a good practice to keep 70% of the data in your train dataset and the rest 30% in your test dataset

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

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, train\_size = 0.7, test\_size = 0.3, random\_state = 100)

# Let's now take a look at the train dataset

X\_train.head()

y\_train.head()

#### Building a Linear Model

You first need to import the statsmodel.api library using which you'll perform the linear regression.

import statsmodels.api as sm

By default, the statsmodels library fits a line on the dataset which passes through the origin. But in order to have an intercept, you need to manually use the add\_constant attribute of statsmodels. And once you've added the constant to your X\_train dataset, you can go ahead and fit a regression line using the OLS (Ordinary Least Squares) attribute of statsmodels as shown below

# Add a constant to get an intercept

X\_train\_sm = sm.add\_constant(X\_train)

# Fit the resgression line using 'OLS'

lr = sm.OLS(y\_train, X\_train\_sm).fit()

# Print the parameters, i.e. the intercept and the slope of the regression line fitted

lr.params

# Performing a summary operation lists out all the different parameters of the regression line fitted

print(lr.summary())

* 1. **Operating Procedure**
* Open Jupyter note book
* Take a new python file
* Type the code
* Run it
* Take inputs from user
* Observe the results
* Verify the results manually
* Store the note book file
  1. **Precautions and/or Troubleshooting**

**Precautions:**

* Save Your Work: Regularly save your Jupyter Notebook to avoid losing your work. You can save your notebook by clicking on the save icon or using the keyboard shortcut Ctrl + S (or Cmd + S on Mac).
* Restart Kernel: If you encounter unexpected behavior or errors, try restarting the kernel. This clears all the variables and imported modules, essentially resetting the notebook's state. You can restart the kernel by going to the "Kernel" menu and selecting "Restart."
* Clear Outputs: To reduce clutter and confusion, consider clearing the outputs of code cells that are no longer relevant. You can do this by selecting "Clear Outputs" from the "Edit" menu.
* Readability: Keep your code and comments clear and well-organized to make it easier to understand and maintain. Use markdown cells for explanations, headings, and documentation.
* Check Dependencies: If you're using external libraries or packages, ensure they are properly installed in your Jupyter environment. You can check the installed packages by running !pip list or !conda list in a code cell.
* Kernel Selection: Make sure you're using the correct kernel for your notebook. The kernel determines the programming language and environment in which your code runs. You can change the kernel by clicking on "Kernel" > "Change kernel" in the menu.
* Resource Usage: Be mindful of the resources your notebook is using, especially if you're working with large datasets or running intensive computations. Check system monitor tools to ensure you're not exhausting memory or CPU resources.

**Troubleshooting:**

* Syntax Errors: Check for syntax errors in your code. Python is sensitive to indentation and syntax, so ensure your code is properly formatted.
* Variable Scope: Be aware of variable scope issues, especially if you're reusing variable names or working with nested functions.
* Library Installation: If you encounter Module Not Found Error or similar errors, ensure that the required libraries are installed in your Jupyter environment. You can install libraries using !pip install <library> or !conda install <library> in a code cell.
* Kernel Crashes: If the kernel crashes frequently, consider reducing the complexity of your code or optimizing resource usage. Large datasets or intensive computations can sometimes overwhelm the kernel.
* Browser Issues: If you experience rendering or responsiveness issues in the notebook interface, try clearing your browser cache or using a different browser.
* Documentation: Consult the official Jupyter documentation and community forums for additional troubleshooting tips and solutions to common problems.
  1. **Observations**

Observe the results obtained in each step.

* 1. **Calculations & Analysis**

Calculations should be given for model output.

* 1. **Result & Interpretation**

Result should be printed and pasted in laboratory copy found from Jupyter note book.

* 1. **Follow-up Questions**
* What is linear regression, and how does it work?
* What are the assumptions of a linear regression model?
* What are outliers? How do you detect and treat them? How do you deal with outliers in a linear regression model?
* What is the difference between simple and multiple linear regression?
* What is multicollinearity and how does it affect linear regression analysis?
* What is the difference between linear regression and non-linear regression?
* Can you explain the concept of overfitting in linear regression?
* What are the limitations of linear regression?
* What is the curse of dimensionality?
  1. **Extension and Follow-up Activities (if applicable)**

NA

* 1. **Assessments**
  2. **Suggested reading**

NA